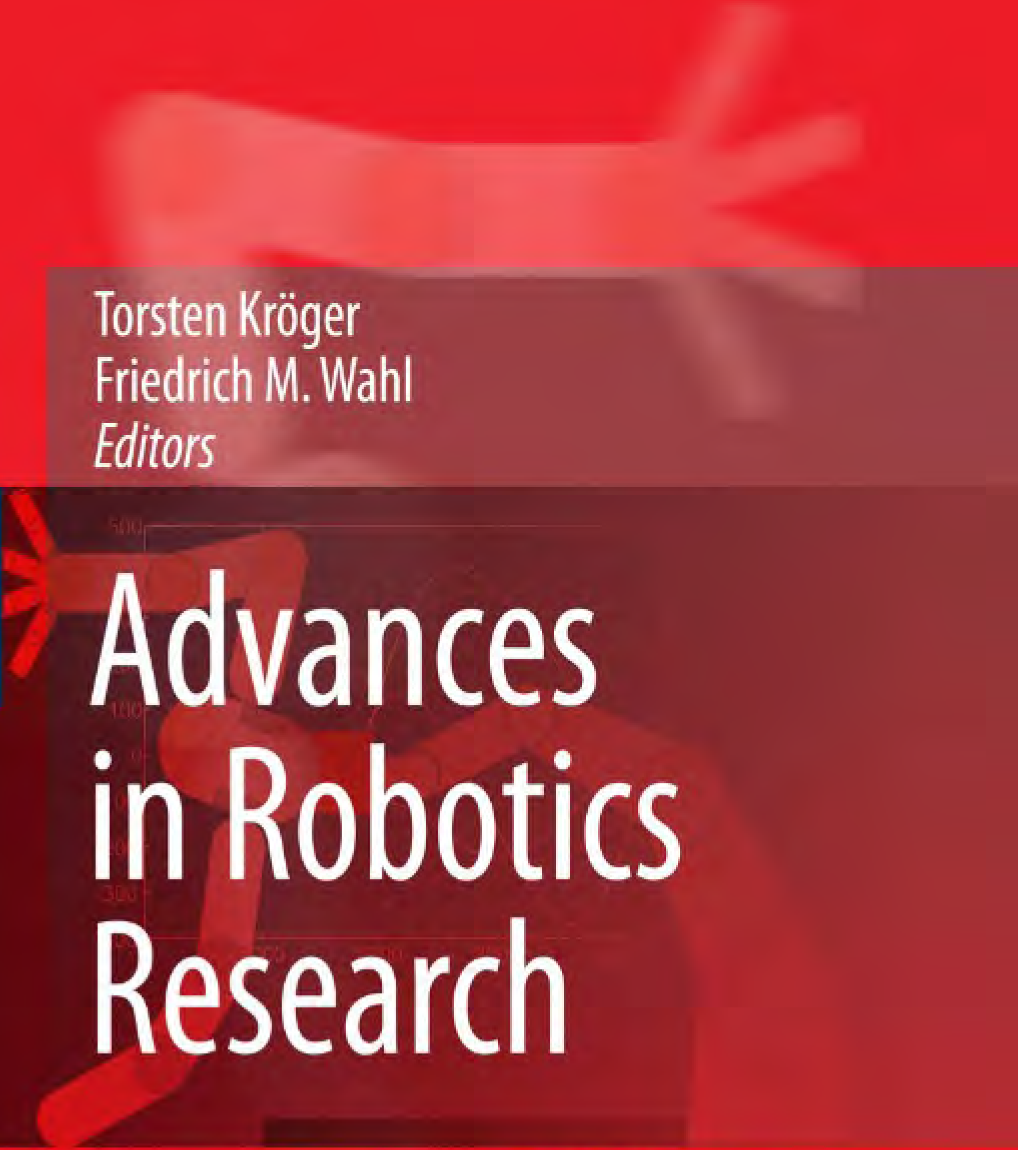


Torsten Kröger
Friedrich M. Wahl
Editors

Advances in Robotics Research

Theory, Implementation, Application

Springer

Advances in Robotics Research

Torsten Kröger and Friedrich M. Wahl (Eds.)

Advances in Robotics Research

Theory, Implementation, Application

Springer

Dr.-Ing. Torsten Kröger
Institut für Robotik und Prozessinformatik
Technische Universität Braunschweig
Mühlenpfordtstraße 23
38106 Braunschweig
Germany
E-mail: t.kroeger@tu-bs.de
http://www.rob.cs.tu-bs.de

Prof. Dr.-Ing. Friedrich M. Wahl
Institut für Robotik und Prozessinformatik
Technische Universität Braunschweig
Mühlenpfordtstraße 23
38106 Braunschweig
Germany
E-mail: f.wahl@tu-bs.de
http://www.rob.cs.tu-bs.de

ISBN 978-3-642-01212-9 e-ISBN 978-3-642-01213-6

DOI 10.1007/978-3-642-01213-6

Library of Congress Control Number: Applied for

© 2009 Springer-Verlag Berlin Heidelberg

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typesetting: Scientific Publishing Services Pvt. Ltd., Chennai, India.

Cover Design: eStudioCalamar S.L.

Printed in acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

Preface

This book is a collection of scientific papers presented at the German Workshop on Robotics — a convention of researchers from academia and industry working on mathematical and algorithmic foundations of robotics, on the design and analysis of robotic systems as well as on robotic applications. As a new event of the Deutsche Gesellschaft für Robotik (DGR, German Robotics Society), the workshop took place at the Technische Universität Carolo-Wilhelmina zu Braunschweig on June 9-10, 2009.

Covering some of the most important ongoing robotics research topics, this volume contains 31 carefully selected and discussed contributions. All of them were presented at the workshop that was attended by 80 researchers representing a wide range of research areas within robotics. The papers are organized in ten scientific tracks: Kinematic and Dynamic Modeling, Motion Generation, Sensor Integration, Robot Vision, Robot Programming, Humanoid Robots, Grasping, Medical Robotics, Autonomous Helicopters, and Robot Applications. Two invited talks by Antonio Bicchi and Atsuo Takanishi presented surveys of research activities in the fields of human-robot interaction and humanoid robotics.

The Program Committee was comprised of Karsten Berns, Oliver Brock, Wolfram Burgard, Martin Buss, Thomas Christaller, Rüdiger Dillmann, Bernd Finkemeyer, Martin Hägele, Bodo Heimann, Dominik Henrich, Gerd Hirzinger, Alois Knoll, Helge-Björn Kuntze, Gisbert Lawitzky, Jürgen Roßmann, Roland Siegwart, Markus Vincze, and Heinz Wörn. After an extensive review and discussion process, the committee met at February 17, 2009, and composed the scientific program from a pool of 49 submissions.

Organizing scientific conventions with a high level of originality cannot be performed by individuals alone. One always has to intercommunicate, to discuss, to exchange knowledge and experiences — to work together. Without the help of many people, the organization of the meeting would not have been possible. This includes all Program Committee members as well as all technical reviewers. A special word of thanks goes to Ilona Engel for her great and diligent support during all stages of the organization and, in particular, for the work she did during the workshop. Ralf Westphal organized the Web site in an excellent way, and he was responsible for the registration procedure. Regarding design and layout of all handouts, Simon Winkelbach was our design specialist for all issues concerning the workshop. The greatest word of thanks is — of course — due to all authors and participants of the German Workshop on Robotics. Finally, we would like to acknowledge the financial support of the Technische Universität Carolo-Wilhelmina zu Braunschweig and the KUKA Roboter GmbH.

Braunschweig
June 2009

Torsten Kröger
Friedrich M. Wahl

Contents

Kinematic and Dynamic Modelling

Joint Dominance Coefficients: A Sensitivity-Based Measure for Ranking Robotic Degrees of Freedom 1
Klaas Klasing, Dirk Wollherr, Martin Buss

Learning Kinematics from Direct Self-Observation Using Nearest-Neighbor Methods 11
Hannes Schulz, Lionel Ott, Jürgen Sturm, Wolfram Burgard

Guidelines for Low Mass and Low Inertia Dynamic Balancing of Mechanisms and Robotics 21
Volkert van der Wijk, Just L. Herder

Motion Generation

Probability-Based Robot Search Paths 31
Jan Deiterding, Dominik Henrich

Mapping and Navigation of Mobile Robots in Natural Environments 43
Jürgen Roßmann, Petra Krahwinkler, Arno Bücken

Sensor-Based Online Planning of Time-Optimized Paths in Dynamic Environments 53
Thorsten Gecks, Dominik Henrich

Sensor Integration

Analysis of Strain Transfer to FBG's for Sensorized Telerobotic End-Effector Applications 65
Dean J. Callaghan, Mark M. McGrath, Ginu Rajan, Eugene Coyle, Yuliya Semenova, Gerald Farrell

Intuitive Collision Avoidance of Robots Using Charge Generated Virtual Force Fields 77
Alexander Winkler, Jozef Suchý

6D Pose Uncertainty in Robotic Perception 89
Wendelin Feiten, Pradeep Atwal, Robert Eidenberger, Thilo Grundmann

Robot Vision

3D Shape Detection for Mobile Robot Learning 99
Andreas Richtsfeld, Markus Vincze

3D Collision Detection for Industrial Robots and Unknown Obstacles Using Multiple Depth Images 111
Markus Fischer, Dominik Henrich

Reducing Motion Artifacts in Mobile Vision Systems via Dynamic Filtering of Image Sequences 123
Christoph Walter, Felix Penzlin, Norbert Elkmann

Robot Programming

A Software Architecture for Model-Based Programming of Robot Systems 135
Michael Geisinger, Simon Barner, Martin Wojtczyk, Alois Knoll

Intuitive Robot Programming of Spatial Control Loops with Linear Movements 147
Katharina Soller, Dominik Henrich

Model-Based Programming "by Demonstration"–Fast Setup of Robot Systems (ProDemo) 159
Jürgen Roßmann, Henning Ruf, Christian Schlette

Humanoid Robots

New Concept for Wrist Design of the Humanoid Robot ARMAR 169
Albert Albers, Jens Ottnad, Christian Sander

Biological Motivated Control Architecture and Mechatronics for a Human-Like Robot 179
Thomas Wahl, Sebastian Blank, Tobias Luksch, Karsten Berns

Using the Torso to Compensate for Non-Minimum Phase Behaviour in ZMP Bipedal Walking 191
Houman Dallali, Martin Brown, Bram Vanderborght

Grasping

Object-Specific Grasp Maps for Use in Planning Manipulation Actions 203
Franziska Zacharias, Christoph Borst, Gerd Hirzinger

Vision Controlled Grasping by Means of an Intelligent Robot Hand 215
Giulio Milighetti, Moritz Ritter, Helge-Björn Kuntze

Learning an Object-Grasp Relation for Silhouette-Based Grasp Planning 227
Nicolas Gorges, Heinz Wörn

Efficient Parallel Random Sample Matching for Pose Estimation, Localization, and Related Problems 239
Daniel Kubus, René Iser, Simon Winkelbach, Friedrich M. Wahl

Medical Robotics

Automated Robot Assisted Fracture Reduction 251
Ralf Westphal, Simon Winkelbach, Thomas Gösling, Markus Oszwald, Tobias Hüfner, Christian Krettek, Friedrich M. Wahl

Path Planning for Robot-Guided Endoscopes in Deformable Environments 263
Markus Rilk, Friedrich M. Wahl, Klaus W.G. Eichhorn, Ingo Wagner, Friedrich Bootz

Autonomous Helicopters

An Unmanned Helicopter for Autonomous Flights in Urban Terrain 275
Florian Adolf, Franz Andert, Sven Lorenz, Lukas Goormann, Jörg Dittrich

Interaction of Altitude Control and Waypoint Navigation of a 4 Rotor Helicopter 287
Tim Puls, Hannes Winkelmann, Sönke Eilers, Matthias Brucke, Andreas Hein

Cooperative Autonomous Helicopters for Load Transportation and Environment Perception 299
Konstantin Kondak, Markus Bernard, Fernando Caballero, Ivan Maza, Anibal Ollero

Applications

Robot on the Leash—An Intuitive Inexpensive Interface for Robots Using the Nintendo Wii Remote 311
Sven Olufs, Markus Vincze

Robot Basketball: Ball Dribbling—A Modified Juggling Task 323
Georg Bätz, Marion Sobotka, Dirk Wollherr, Martin Buss

Research on Exoskeletons at the TU Berlin 335
Christian Fleischer, Konstantin Kondak, Andreas Wege, Ingo Kossyk

Hard- and Software Architecture of a Small Autonomous Underwater Vehicle for Environmental Monitoring Tasks 347
Christoph Osterloh, Marek Litza, Erik Maehle

Author Index 357

List of Contributors

Florian Adolf
German Aerospace Center (DLR), Institute of Flight Systems,
Lilienthalplatz 7, 38108 Braunschweig, Germany,
e-mail: `florian.adolf@dlr.de`

Albert Albers
Institute of Product Development Karlsruhe, University of Karlsruhe,
Kaiserstraße 10, 76131 Karlsruhe, Germany,
e-mail: `albers@ipek.uka.de`

Franz Andert
German Aerospace Center (DLR), Institute of Flight Systems,
Lilienthalplatz 7, 38108 Braunschweig, Germany,
e-mail: `franz.andert@dlr.de`

Pradeep Atwal
Institute for Applied Mathematics, Bonn University,
Regina-Pacis-Weg 3, 53115 Bonn, Germany,
e-mail: `pradeep.atwal@uni-bonn.de`

Georg Bätz
Institute of Automatic Control Engineering, Technische Universität München,
Theresienstraße 90, 80290 Munich, Germany,
e-mail: `georg.baetz@tum.de`

Simon Barner
Institut für Informatik VI, Technische Universität München,
Boltzmannstraße 3, 85748 Garching bei München, Germany,
e-mail: `barner@in.tum.de`

Markus Bernard
Institut für Technische Informatik und Mikroelektronik,
Technische Universität Berlin, Einsteinufer 17, 10587 Berlin, Germany,
e-mail: `bernard@cs.tu-berlin.de`

Karsten Berns
Robotics Research Lab at the Department of Computer Sciences,
Kaiserslautern University of Technology,
P.O. Box 3049, 67653 Kaiserslautern, Germany,
e-mail: `berns@informatik.uni-kl.de`

Sebastian Blank
Robotics Research Lab at the Department of Computer Sciences,
Kaiserslautern University of Technology,
P.O. Box 3049, 67653 Kaiserslautern, Germany,
e-mail: `blank@informatik.uni-kl.de`

Friedrich Bootz
Klinik und Poliklinik für Hals-Nasen-Ohrenheilkunde/Chirurgie,
Universitätsklinikum Bonn, Sigmund-Freud-Straße 25, 53105 Bonn, Germany,
e-mail: `friedrich.bootz@ukb.uni-bonn.de`

Christoph Borst
German Aerospace Center (DLR), Institute of Robotics and Mechatronics,
Münchner Straße 20, 82234 Weßling, Germany,
e-mail: `christoph.borst@dlr.de`

Martin Brown
The University of Manchester, Control Systems Centre,
School of Electrical and Electronic Engineering, M60 1QD, United Kingdom,
e-mail: `martin.brown@manchester.ac.uk`

Matthias Brucke
OFFIS, University of Oldenburg, Escherweg 2,
26121 Oldenburg, Germany,
e-mail: `matthias.brucke@offis.de`

Arno Bücken
Institute of Man-Machine-Interaction, RWTH Aachen,
Ahornstraße 55, 52074 Aachen, Germany,
e-mail: `buecken@mmi.rwth-aachen.de`

Wolfram Burgard
Department of Computer Science, University of Freiburg,
Georges-Köhler-Allee 79, 79110 Freiburg, Germany,
e-mail: `burgard@informatik.uni-freiburg.de`

Martin Buss
Institute of Automatic Control Engineering, Technische Universität München,
Theresienstraße 90, 80290 Munich, Germany,
e-mail: `mb@tum.de`

Fernando Caballero
Robotics, Vision and Control Group, University of Seville,
Camino de los Descubrimientos, 41092 Seville, Spain,
e-mail: `caba@cartuja.us.es`

Dean J. Callaghan
School of Manufacturing and Design Engineering, Faculty of Engineering,
Dublin Institute of Technology, Bolton Street, Dublin 1, Ireland,
e-mail: `dean.callaghan@dit.ie`

Eugene Coyle
School of Electrical Engineering Systems, Faculty of Engineering,
Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland,
e-mail: `eugene.coyle@dit.ie`

Houman Dallali
The University of Manchester, Control Systems Centre,
School of Electrical and Electronic Engineering, M60 1QD, United Kingdom,
e-mail: `houman.dallali@postgrad.manchester.ac.uk`

Jan Deiterding
Lehrstuhl für angewandte Informatik III, Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth, Germany,
e-mail: `jan.deiterding@uni-bayreuth.de`

Jörg Dittrich
German Aerospace Center (DLR), Institute of Flight Systems,
Lilienthalplatz 7, 38108 Braunschweig, Germany,
e-mail: `joerg.dittrich@dlr.de`

Klaus W.G. Eichhorn
Klinik und Poliklinik für Hals-Nasen-Ohrenheilkunde/Chirurgie,
Universitätsklinikum Bonn, Sigmund-Freud-Straße 25, 53105 Bonn, Germany,
e-mail: `klaus.eichhorn@ukb.uni-bonn.de`

Robert Eidenberger
Department of Computational Perception, Johannes Kepler University Linz,
Altenbergerstraße 69, 4040 Linz, Austria,
e-mail: `robert.eidenberger.ext@siemens.com`

Sönke Eilers
OFFIS, University of Oldenburg, Escherweg 2,
26121 Oldenburg, Germany,
e-mail: `soenke.eilers@offis.de`

Norbert Elkmann
Fraunhofer Institute for Factory Operation and Automation,
Sandtorstraße 22, 39106 Magdeburg, Germany,
e-mail: `norbert.elkmann@iff.fraunhofer.de`

Gerald Farrell
School of Electronic and Communications Engineering, Faculty of Engineering,
Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland,
e-mail: `gerald.farrell@dit.ie`

Wendelin Feiten
Department of Intelligent Autonomous Systems, Information and Communication,
Siemens AG, Otto-Hahn-Ring 6, 81739 Munich, Germany,
e-mail: `wendelin.feiten@siemens.com`

Markus Fischer
Lehrstuhl für angewandte Informatik III, Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth, Germany,
e-mail: `markus.fischer@uni-bayreuth.de`

Christian Fleischer
Institut für Technische Informatik und Mikroelektronik,
Technische Universität Berlin, Einsteinufer 17, 10587 Berlin, Germany,
e-mail: `fleischer@cs.tu-berlin.de`

Thorsten Gecks
Lehrstuhl für angewandte Informatik III, Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth, Germany,
e-mail: `thorsten.gecks@uni-bayreuth.de`

Michael Geisinger
Institut für Informatik VI, Technische Universität München,
Boltzmannstraße 3, 85748 Garching bei München, Germany,
e-mail: `geisinge@in.tum.de`

Lukas Goormann
German Aerospace Center (DLR), Institute of Flight Systems,
Lilienthalplatz 7, 38108 Braunschweig, Germany,
e-mail: `lukas.goormann@dlr.de`

Nicolas Gorges
Institute for Process Control and Robotics, University of Karlsruhe,
Engler-Bunte-Ring 8, 76131 Karlsruhe, Germany,
e-mail: `gorges@ira.uka.de`

Thomas Gösling
Unfallchirurgische Klinik, Medizinische Hochschule Hannover,
Carl-Neuberg-Straße 1, 30625 Hannover, Germany,
e-mail: `goesling.thomas@mh-hannover.de`

Thilo Grundmann
Department of Intelligent Autonomous Systems, Information and Communication,
Siemens AG, Otto-Hahn-Ring 6, 81739 Munich, Germany,
e-mail: `thilo.grundmann.ext@siemens.com`

Andreas Hein
OFFIS, University of Oldenburg, Escherweg 2,
26121 Oldenburg, Germany,
e-mail: `andreas.hein@offis.de`

Dominik Henrich
Lehrstuhl für angewandte Informatik III, Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth, Germany,
e-mail: `dominik.henrich@uni-bayreuth.de`

Just L. Herder
Department of Biomechanical Engineering, Faculty of Mechanical, Maritime and
Materials Engineering, Delft University of Technology,
Mekelweg 2, 2628 CD Delft, The Netherlands,
e-mail: `j.l.herder@tudelft.nl`

Gerd Hirzinger
German Aerospace Center (DLR), Institute of Robotics and Mechatronics,
Münchner Straße 20, 82234 Weßling, Germany,
e-mail: `gerd.hirzinger@dlr.de`

Tobias Hüfner
Unfallchirurgische Klinik, Medizinische Hochschule Hannover,
Carl-Neuberg-Straße 1, 30625 Hannover, Germany,
e-mail: `huefner.tobias@mh-hannover.de`

René Iser
Institut für Robotik und Prozessinformatik, Technische Universität
Carolo-Wilhelmina zu Braunschweig, Mühlenpfordtstraße 23, 38106
Braunschweig, Germany,
e-mail: `r.iser@tu-bs.de`

Klaas Klasing
Institute of Automatic Control Engineering, Technische Universität München,
Theresienstraße 90, 80290 Munich, Germany,
e-mail: `kk@tum.de`

Alois Knoll
Institut für Informatik VI, Technische Universität München, Boltzmannstraße 3,
85748 Garching bei München, Germany,
e-mail: `knoll@in.tum.de`

Konstantin Kondak
Institut für Technische Informatik und Mikroelektronik,
Technische Universität Berlin, Einsteinufer 17, 10587 Berlin, Germany,
e-mail: `kondak@cs.tu-berlin.de`

Ingo Kossyk
Institut für Technische Informatik und Mikroelektronik,
Technische Universität Berlin, Einsteinufer 17, 10587 Berlin, Germany,
e-mail: `kossyki@cs.tu-berlin.de`

Petra Krahwinkler
Institute of Man-Machine-Interaction, RWTH Aachen,
Ahornstraße 55, 52074 Aachen, Germany,
e-mail: `krahwinkler@mmi.rwth-aachen.de`

Christian Krettek
Unfallchirurgische Klinik, Medizinische Hochschule Hannover,
Carl-Neuberg-Straße 1, 30625 Hannover, Germany,
e-mail: `krettek.christian@mh-hannover.de`

Daniel Kubus
Institut für Robotik und Prozessinformatik,
Technische Universität Carolo-Wilhelmina zu Braunschweig,
Mühlenpfordtstraße 23, 38106 Braunschweig, Germany,
e-mail: `d.kubus@tu-bs.de`

Helge-Björn Kuntze
Fraunhofer-Institute for Information and Data Processing (IITB),
Fraunhoferstraße 1, 76131 Karlsruhe, Germany,
e-mail: `kuntze@iitb.fraunhofer.de`

Marek Litza
Institute of Computer Engineering, University of Lübeck,
Ratzeburger Allee 160, 23538 Lübeck, Germany,
e-mail: `litza@iti.uni-luebeck.de`

Sven Lorenz
German Aerospace Center (DLR), Institute of Flight Systems,
Lilienthalplatz 7, 38108 Braunschweig, Germany,
e-mail: `sven.lorenz@dlr.de`

Tobias Luksch
Robotics Research Lab at the Department of Computer Sciences,
Kaiserslautern University of Technology, P.O. Box 3049,
67653 Kaiserslautern, Germany,
e-mail: `luksch@informatik.uni-kl.de`

Erik Maehle
Institute of Computer Engineering, University of Lübeck,
Ratzeburger Allee 160, 23538 Lübeck, Germany,
e-mail: `maehle@iti.uni-luebeck.de`

Ivan Maza
Robotics, Vision and Control Group, University of Seville,
Camino de los Descubrimientos, 41092 Seville, Spain,
e-mail: `imaza@cartuja.us.es`

Mark M. McGrath
School of Manufacturing and Design Engineering, Faculty of Engineering,
Dublin Institute of Technology, Bolton Street, Dublin 1, Ireland,
e-mail: `mark.mcgrath@dit.ie`

Giulio Milighetti
Fraunhofer-Institute for Information and Data Processing (IITB),
Fraunhoferstraße 1, 76131 Karlsruhe, Germany,
e-mail: `milighetti@iitb.fraunhofer.de`

Anibal Ollero
Robotics, Vision and Control Group, University of Seville,
Camino de los Descubrimientos, 41092 Seville, Spain,
e-mail: `aollero@cartuja.us.es`

Sven Olufs
Automation and Control Institute, Technische Universität Wien,
Gußhausstraße 27–29 / E376, 1040 Wien, Austria,
e-mail: `olufs@acin.tuwien.ac.at`

Christoph Osterloh
Institute of Computer Engineering, University of Lübeck,
Ratzeburger Allee 160, 23538 Lübeck, Germany,
e-mail: `osterloh@iti.uni-luebeck.de`

Markus Oszwald
Unfallchirurgische Klinik, Medizinische Hochschule Hannover,
Carl-Neuberg-Straße 1, 30625 Hannover, Germany,
e-mail: `oszwald.markus@mh-hannover.de`

Lionel Ott
Department of Computer Science, University of Freiburg,
Georges-Köhler-Allee 79, 79110 Freiburg, Germany,
e-mail: `ottli@informatik.uni-freiburg.de`

Jens Ottnad
Institute of Product Development Karlsruhe, University of Karlsruhe,
Kaiserstraße 10, 76131 Karlsruhe, Germany,
e-mail: `ottnad@ipek.uka.de`

Felix Penzlin
Fraunhofer Institute for Factory Operation and Automation,
Sandtorstraße 22, 39106 Magdeburg, Germany,
e-mail: `felix.penzlin@iff.fraunhofer.de`

Tim Puls
OFFIS, University of Oldenburg,
Escherweg 2, 26121 Oldenburg, Germany,
e-mail: `tim.puls@offis.de`

Ginu Rajan
Photonics Research Centre, Faculty of Engineering,
Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland,
e-mail: `ginu.rajan@dit.ie`

Andreas Richtsfeld
Automation and Control Institute, Technische Universität Wien,
Gußhausstraße 27–29 / E376, 1040 Wien, Austria,
e-mail: `ari@acin.tuwien.ac.at`

Markus Rilk
Institut für Robotik und Prozessinformatik,
Technische Universität Carolo-Wilhelmina zu Braunschweig,
Mühlenpfordtstraße 23, 38106 Braunschweig, Germany,
e-mail: `m.rilk@tu-bs.de`

Moritz Ritter
Fraunhofer-Institute for Information and Data Processing (IITB),
Fraunhoferstraße 1, 76131 Karlsruhe, Germany,
e-mail: `ritter@iitb.fraunhofer.de`

Jürgen Roßmann
Institute of Man-Machine-Interaction, RWTH Aachen,
Ahornstraße 55, 52074 Aachen, Germany,
e-mail: `rossmann@mmi.rwth-aachen.de`

Henning Ruf
Institute of Man-Machine-Interaction, RWTH Aachen, Ahornstraße 55, 52074
Aachen, Germany, e-mail: `ruf@mmi.rwth-aachen.de`

Christian Sander
Institute of Product Development Karlsruhe, University of Karlsruhe,
Kaiserstraße 10, 76131 Karlsruhe, Germany,
e-mail: `sander@ipek.uka.de`

Christian Schlette
Institute of Man-Machine-Interaction, RWTH Aachen, Ahornstraße 55,
52074 Aachen, Germany,
e-mail: `schlette@mmi.rwth-aachen.de`

Hannes Schulz
Department of Computer Science, University of Freiburg,
Georges-Köhler-Allee 79, 79110 Freiburg, Germany,
e-mail: `schulzha@informatik.uni-freiburg.de`

Yuliya Semenova
Photonics Research Centre, Faculty of Engineering,
Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland,
e-mail: `yuliya.semenova@dit.ie`

Marion Sobotka
Institute of Automatic Control Engineering,
Technische Universität München,
Theresienstraße 90, 80290 Munich, Germany,
e-mail: `marion.sobotka@tum.de`

Katharina Soller
Lehrstuhl für angewandte Informatik III, Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth, Germany,
e-mail: `katharina.soller@uni-bayreuth.de`

Jürgen Sturm
Department of Computer Science, University of Freiburg,
Georges-Köhler-Allee 79, 79110 Freiburg, Germany,
e-mail: `sturm@informatik.uni-freiburg.de`

Jozef Suchý
Chemnitz University of Technology, Faculty of Electrical Engineering and
Information Technology, Robot Systems,
Reichenhainer Straße 70, 09126 Chemnitz, Germany,
e-mail: `jozef.suchy@etit.tu-chemnitz.de`

Bram Vanderborght
Italian Institute of Technology, Robotics, Brain and Cognitive Sciences Department,
Via Morego 30, 16163 Genova, Italy,
e-mail: `bram.vanderborght@iit.it`

Markus Vincze
Automation and Control Institute, Technische Universität Wien,
Gußhausstraße 27–29 / E376, 1040 Wien, Austria,
e-mail: `vincze@acin.tuwien.ac.at`

Ingo Wagner
Klinik und Poliklinik für Hals-Nasen-Ohrenheilkunde/Chirurgie,
Universitätsklinikum Bonn, Sigmund-Freud-Straße 25, 53105 Bonn, Germany,
e-mail: `ingo.wagner@ukb.uni-bonn.de`

Friedrich M. Wahl
Institut für Robotik und Prozessinformatik,
Technische Universität Carolo-Wilhelmina zu Braunschweig,
Mühlenpfordtstraße 23, 38106 Braunschweig, Germany,
e-mail: `f.wahl@tu-bs.de`

Thomas Wahl
Robotics Research Lab at the Department of Computer Sciences,
Kaiserslautern University of Technology, P.O. Box 3049,
67653 Kaiserslautern, Germany,
e-mail: `t_wahl@informatik.uni-kl.de`

Christoph Walter
Fraunhofer Institute for Factory Operation and Automation,
Sandtorstraße 22, 39106 Magdeburg, Germany,
e-mail: `christoph.walter@iff.fraunhofer.de`

Andreas Wege
Institut für Technische Informatik und Mikroelektronik,
Technische Universität Berlin, Einsteinufer 17, 10587 Berlin, Germany,
e-mail: `awege@cs.tu-berlin.de`

Ralf Westphal
Institut für Robotik und Prozessinformatik,
Technische Universität Carolo-Wilhelmina zu Braunschweig,
Mühlenpfordtstraße 23, 38106 Braunschweig, Germany,
e-mail: `ralf.westphal@tu-bs.de`

Volkert van der Wijk
Department of Biomechanical Engineering, Faculty of Mechanical,
Maritime and Materials Engineering, Delft University of Technology,
Mekelweg 2, 2628 CD Delft, The Netherlands,
e-mail: `v.vanderwijk@kineticart.nl`

Simon Winkelbach
Institut für Robotik und Prozessinformatik,
Technische Universität Carolo-Wilhelmina zu Braunschweig,
Mühlenpfordtstraße 23, 38106 Braunschweig, Germany,
e-mail: `s.winkelbach@tu-bs.de`

Hannes Winkelmann
OFFIS, University of Oldenburg,
Escherweg 2, 26121 Oldenburg, Germany,
e-mail: `hannes.winkelmann@offis.de`

Alexander Winkler
Chemnitz University of Technology, Faculty of Electrical Engineering and Information Technology, Robot Systems, Reichenhainer Straße 70,
09126 Chemnitz, Germany,
e-mail: `alexander.winkler@e-technik.tu-chemnitz.de`

Martin Wojtczyk
Institut für Informatik VI, Technische Universität München,
Boltzmannstraße 3, 85748 Garching bei München, Germany,
e-mail: `wojtczyk@in.tum.de`

Dirk Wollherr
Institute of Automatic Control Engineering, Technische Universität München,
Theresienstraße 90, 80290 Munich, Germany,
e-mail: `dw@tum.de`

Heinz Wörn
Institute for Process Control and Robotics, University of Karlsruhe,
Engler-Bunte-Ring 8, 76131 Karlsruhe, Germany,
e-mail: `woern@ira.uka.de`

Franziska Zacharias
German Aerospace Center (DLR), Institute of Robotics and Mechatronics,
Münchner Straße 20, 82234 Weßling, Germany,
e-mail: `franziska.zacharias@dlr.de`

Joint Dominance Coefficients: A Sensitivity-Based Measure for Ranking Robotic Degrees of Freedom

Klaas Klasing, Dirk Wollherr, and Martin Buss

Abstract. Many robotics applications require a weighting scheme for individual degrees of freedom in a kinematic linkage. Such schemes are used for example in path and motion planning algorithms to penalize large end-effector movements or scale distance computations for the retrieval of nearest neighbors. Most often, the weights are manually picked and heuristically adjusted for specific linkages. In this paper we propose *joint dominance coefficients* as a universal tool for estimating the influence of each degree of freedom of a robot on the overall robot displacement. The measure is easy to compute, converges quickly and can be applied to any kind of parameterized kinematic linkage, including tree-structured and closed kinematic chains. A mathematical derivation is provided along with application examples for various robotic linkages. The results show that the method accurately and reliably yields the desired weights.

1 Introduction

Most robotics applications involve kinematic linkages with several degrees of freedom (DoF). These DoF arise from revolute or prismatic joints that connect the robot links. The overall geometry of the robot is usually parameterized by the DoF of the base link (movement of the robot base) and the joint values (angles for revolute joints, distances for prismatic joints), which describe the *robot configuration*. A central question for many applications is how the overall robot displacement depends on the degrees of freedom of the robot. Such a dependency measure is useful for example in the context of path and motion planning applications, where DoF that cause larger displacements must be penalized by corresponding weights.

While for prismatic DoF there is a linear relationship between parameter change and robot displacement, for revolute DoF the overall displacement is a nonlinear

---

Klaas Klasing · Dirk Wollherr · Martin Buss
Institute of Automatic Control Engineering, Technische Universität München,
Theresienstraße 90, 80290 Munich, Germany,
e-mail: `{kk,dw,mb}@tum.de`

function of the robot configuration. As an example, consider a 20-link serial kinematic chain with equally long links connected by revolute joints. When the chain is fully extended, an angular change in the first joint has a much larger influence on overall robot displacement than the same angular change applied to the 19^{th} joint in the chain. However, for a configuration in which the robot is 'folded' to its minimal expansion, for both the first and the 19^{th} joint an angular change has the same effect on overall displacement. The influence of each DoF on robot displacement is therefore configuration dependent. The question addressed in this paper is whether there is a practically feasible way of assessing the influence of each DoF over all possible robot configurations.

The measure presented in the following provides an automated numeric procedure for assigning a weight to each degree of freedom that correctly captures the overall influence of the respective DoF on the robot displacement. To the best of our knowledge there exists no method that is able to generically derive meaningful coefficients similar to those obtained by our approach.

The remainder of this paper is structured as follows: Section 2 briefly reviews existing related methods. Section 3 presents prerequisites, a derivation of *joint dominance coefficients* as well as an efficient way to calculate the coefficients in practice. Section 4 demonstrates the application of the method to a number of different kinematic linkages. A discussion of the results and the applicability of the method is given in Section 5.

2 Related Work

The problem of weighting the links in a kinematic linkage is equivalent to the problem of scaling an underlying distance metric. In the field mechanism design finding suitable distance metrics for specific linkage topologies is an active research topic [1, 2]. In the context of sampling-based path planning suitable distance metrics are relevant for effective sampling [3], nearest neighbor searching [4] and local planning [5]. All of these works are related in that they try to provide a consistent and meaningful notion of distance within the topology defined by the linkage. In contrast, this paper is not concerned with deriving new metrics for specific linkages. Instead it aims to provide a universal numeric measure for each DoF in a linkage, which can then be used to adjust a given metric.

The proposed coefficients can also be used for the analysis of kinematic linkages. In this respect they are loosely related to *swept volume* [6, 7], which represents a powerful tool for analyzing the reachable workspace of a kinematic linkage. However, for the simple purpose of finding suitable weights for specifying DoF displacement influence, swept volume methods are somewhat of an overkill. While weights similar to the coefficients proposed in this paper could be derived from swept volume methods for many linkages, these methods are computationally expensive and require detailed geometric considerations. In contrast, the proposed joint dominance coefficients only need a parametrization of the forward kinematics as well as a number of representative points on the linkage. Although only moderately sized tree-structured linkages are examined in this paper, the sampling-based nature of joint dominance coefficients makes them perfectly suitable for closed-loop parametrizations, such as [8], as well as systems with many DoF. In both cases analytical derivation of similar coefficients from swept volume quickly becomes infeasible because of the complexity increase.

Joint dominance coefficients are also loosely related to the individual Lipschitz constants that can be used to bound robot displacement for iterative collision checking [5]. While our approach provides an average value per DoF, the Lipschitz constants represent an upper limit for displacement on a per-DoF basis. To our knowledge there exists no automated generic procedure for deriving these constants; in fact in most path planning applications that utilize iterative collision checking, the DoF scaling is usually hand-tuned until certain displacement constants seem to be met.

3 Methodology

In this section the concept of joint dominance coefficients is derived and a method for efficient calculation of the coefficients is presented.

3.1 *Distance Metrics*

In the following we will assume that a robot $\mathscr{A}(q)$ has n degrees of freedom that are parameterized by a vector $q \in \mathbb{R}^n$, $q = [q_1 \ \dots \ q_n]$. The geometry of the robot $\mathscr{A}$ is defined over either $\mathbb{R}^2$ or $\mathbb{R}^3$. The set of all possible robot configurations makes up the so-called *configuration space* $\mathscr{C}$, or simply C-space [9, 5]. Each DoF is assumed to be bounded by joint limits $[q_{i,min}, q_{i,max}]$.

A central prerequisite for all research methods that utilize the *configuration space* of a robot is a notion of the distance between two configurations q and q'. For many applications a simple Euclidean distance in joint space

$$\rho_E(q,q') = \|q - q'\|_2 \tag{1}$$

works sufficiently well for the purposes of nearest neighbor searching, uniform sampling of configurations etc. However, if the linkage contains many revolute joints, the metric does not respect the topology of the configuration space[1] and for many pairs of configuration (q,q') does not reflect the actual displacement of the robot in the world.

A much better measure is provided by the so-called *robot displacement metric* [5]

$$\rho_D(q,q') = \max_{a \in \mathscr{A}} \{ \|a(q) - a(q')\| \} \tag{2}$$

---

[1] For $n+1$ links attached by n revolute joints the configuration space has the topology of an n-dimensional torus.

that yields the maximum distance that any point on the robot $\mathscr{A}$ has been displaced. Unfortunately, $\rho_D(q,q')$ cannot be efficiently computed for generic kinematic linkages in practice, however, as we shall see in the following, it is possible to find efficient approximations.

3.2 *Derivation of the Joint Dominance Coefficient Vector*

To distinguish between the 'importance' of different degrees of freedom, a so-called *joint dominance coefficient* for each DoF/joint is proposed. This coefficient reflects how much a given degree of freedom influences the overall average displacement of the robot.

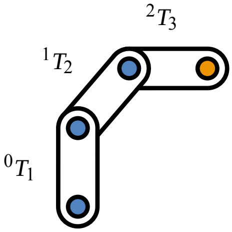


Fig. 1 Several representative points on a kinematic linkage can be used to approximate the robot displacement metric.

For simplicity of derivation consider the serial kinematic linkage depicted in Figure 1. Let the position of the end effector (in orange) with respect to frame 3 be denoted by ^3p_e. The values of the rotational joints[2] will be denoted by q_1, q_2, and q_3 and the corresponding homogeneous transformation matrices $T_1(q_1)$, $T_2(q_3)$, and $T_3(q_3)$. Then, the position of the end effector in world coordinates is given by

$$^0p_e = {}^0T_1 \cdot {}^1T_2 \cdot {}^2T_3 \cdot {}^3p_e = f(q). \tag{3}$$

The local sensitivity of the end effector position $f(q)$ with respect to the value of the ith joint q_i is – for a given configuration $q^* = [q_1^*, \ldots, q_{i-1}^*, q_i, q_{i+1}^*, \ldots, q_n^*]$ – given by

$$\left.\frac{\partial f(q)}{\partial q_i}\right|_{q^*} := f'(q_i)\big|_{q^*}. \tag{4}$$

Note that the sensitivity derivative is a vectorial function of q_i. Taking the Euclidean norm of this vector to obtain its magnitude and integrating over the joint limits of the respective joint q_i yields

$$\sigma_i(q^*) = \int_{q_{i,min}}^{q_{i,max}} \left\| f'(q_i)\big|_{q^*} \right\|_2 dq_i, \tag{5}$$

---

[2] All derivation steps hold for prismatic joints as well as for more complicated tree-structured linkages, too.

which is a scalar measure that indicates how much q_i influence end effector displacement over its joint range. Two further adjustments are necessary to obtain a meaningful coefficient for the entire configuration space: Firstly, sensitivity is a local measure that may drastically differ for different configurations. Since the goal is to have one coefficient per joint for the entire configuration space, the overall measure is aggregated by randomly sampling n_s configurations q_j^*, where $j = 1,\dots,n_s$. The average influence of joint q_i on the end effector position is then given by

$$\sigma_i = \frac{1}{n_s} \sum_{j=1}^{n_s} \int_{q_{i,min}}^{q_{i,max}} \left\| f'(q_i)\big|_{q_j^*} \right\|_2 dq_i. \tag{6}$$

Secondly, a good displacement metric should not only take into account the end effector, but essentially the movement of any link on the robot. The *robot displacement metric* from (2) would be an ideal measure for this but is hard to calculate in practice. We therefore propose to use a collection of n_p representative points $p_1,\dots,p_{n_p}$ on the robot for expressing the overall displacement. Obviously, the position of each point is calculated from a sequence of transformation matrices and can be represented by corresponding functions $f_1(q),\dots,f_{n_p}(q)$. Such points would usually be chosen at the robot base, at the origin of each coordinate frame and at each end effector. For the simple robot from Figure 1 all highlighted points would be chosen as representative points.

Then, the overall displacement between two configurations q and q' is expressed by the new metric

$$\rho_P(q,q') = \sum_{i=1}^{n_p} \left\| f_i(q) - f_i(q') \right\|_2 . \tag{7}$$

Therefore, the overall average influence of a given joint on the robot displacement from (7), which we call *joint dominance coefficient*, is given by

$$\sigma_i = \frac{1}{n_s} \sum_{j=1}^{n_p} \sum_{k=1}^{n_s} \int_{q_{i,min}}^{q_{i,max}} \left\| f'(q_i)\big|_{q_k^*} \right\|_2 dq_i. \tag{8}$$

When calculated for each joint, the resulting vector $\sigma = [\sigma_1 \cdots \sigma_n]$ of joint dominance coefficients should be normalized to obtain

$$\sigma^* = \frac{1}{\|\sigma\|_2}\sigma = \frac{1}{\|\sigma\|_2} \begin{bmatrix} \sigma_1 \\ \vdots \\ \sigma_n \end{bmatrix} . \tag{9}$$

This *joint dominance coefficient vector* indicates what percentage of the overall average displacement of a robot is caused by each degree of freedom.

3.3 Calculation of the Joint Dominance Coefficient Vector

In a practical implementation, the calculation of the partial derivatives $f'(q_i)$ as well as the analytical evaluation of the integral would require a symbolic math engine. Although the derivatives exhibit a certain structure and can be broken down into a finite series of *sin* and *cos* expressions that stem from the transformation matrices, arbitrary kinematic linkages may still cause arbitrarily complex symbolic terms that need to be integrated. For efficient calculation of σ it is therefore desirable to have a numeric approximation of (8). Such an approximation can be achieved in two steps: Firstly, the derivative is substituted by a difference quotient, and secondly the integration is replaced by a summation. To this end each degree of freedom is gridded at a certain resolution and n_r samples $q_{i,1},\dots,q_{i,n_r}$ are placed evenly between $q_{i,min}$ and $q_{i,max}$, i.e. $q_{i,1} = q_{i,min}$ and $q_{i,n_r} = q_{i,max}$. For notational feasibility, let $q(i,k,l)$ denote the kth randomly generated configuration vector for which the ith entry (corresponding to the ith DoF) has been set to $q_{i,l}$. Then the resulting term for the approximated joint dominance coefficient is

$$\hat{\sigma}_i = \frac{1}{n_s} \sum_{j=1}^{n_p} \sum_{k=1}^{n_s} \sum_{l=1}^{n_r-1} \left\| f_j(q(i,k,l)) - f_j(q(i,k,l+1)) \right\|_2 . \tag{10}$$

Since the vector is normalized in the end and n_s is the same for all degrees of freedom the factor $\frac{1}{n_s}$ can be dropped. After rearranging the order of summation, one obtains

$$\hat{\sigma}_i = \sum_{j=1}^{n_s} \sum_{k=1}^{n_r-1} \rho_P\left(q(i,j,k), q(i,j,k+1)\right), \tag{11}$$

where ρ is the displacement metric from (7).

To summarize, $\hat{\sigma}^*$ is calculated by generating n_s random configurations, computing – for each configuration – the displacement ρ_D between each two consecutive configurations spaced at n_r even steps in $[q_{i,min}, q_{i,max}]$ for the ith degree of freedom, adding all displacements for each i and finally normalizing the vector. Note that the estimation has a runtime complexity of $O(n \cdot n_s \cdot n_r)$ and thus is linear in the number of DoF.

3.4 Remarks

Several remarks are in order for the chosen way of calculating the coefficient . Firstly, the algorithm relies on random sampling to aggregate the local sensitivity-based measures to an overall measure for the entire configuration space. Random sampling was preferred over sampling on a grid, because the number of samples on a grid increases exponentially with the number of DoF. In fact this is the same problem that led researchers to turn to sampling-based path planning as a means for solving high-dimensional planning problems. Consequently, much the same requirements apply for the sampling sequence. Since we try to achieve a fairly uniform covering of the configuration space with a finite number of n_s samples, the sampling sequence should be *dense* [5]. For our purposes we used uniform random sampling in the hypercube bounded by the individual $[q_{i,min}, q_{i,max}]$. A deterministic (quasi-random) alternative would be the use of *Halton points* [10]. Since the number of samples is known beforehand it is also possible to use *Hammersley points* [11] or *lattice sampling* [12].

A second remark concerns the convergence of the computation. A detailed proof is omitted due to space constraints but can be constructed along the following lines: The integral in the inner term of (8),

$$\int_{q_{i,min}}^{q_{i,max}} \left\| f'(q_i)\big|_{q_k^*} \right\|_2 dq_i, \tag{12}$$

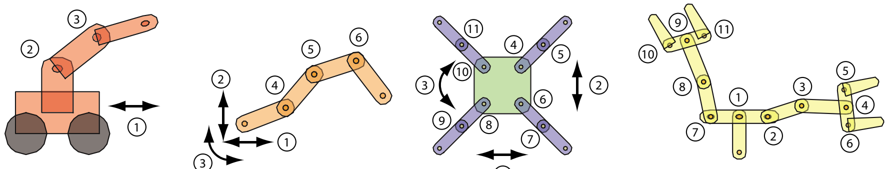
has been replaced by its Riemann sum in (10), which implies asymptotic convergence as n_r grows. The convergence of the overall coefficient with growing n_s then depends on the continuous differentiability of the configuration space manifold, and the denseness of the sampling sequence. An empirical study of the convergence is presented in the following section, which allows for specifying recommended values for n_r and n_s.

4 Application to Kinematic Linkages

To evaluate the meaningfulness of the proposed coefficients, the method was applied to the four kinematic linkages depicted in Figure 2. All of them represent robots in a

Table 1 Admissable DoF ranges for each robot.

Robot	DoF	$q_{i,min}$	$q_{i,max}$
simple3	1	-15 m	15 m
	2	-100°	100°
	3	-90°	90°
snake6	1,2	-21.5 m	21.5 m
	3	-180°	180°
	4,5,6	-90°	90°
crab11	1,2	-25 m	25 m
	3	-180°	180°
	4,6,8,10	-55°	55°
	5,7,9,11	-20°	20°
tree11	1,3,4,5,6,8,9,10,11	-20°	20°
	2,7	-45°	45°



(a) simple3 (b) snake6 (c) crab11 (d) tree11

Fig. 2 Four robots used for evaluation: (a) A simple robot with 3 DoF (b) A snake robot with 6 DoF (c) A crab-like robot with 11 DoF (d) A tree-like robot with 11 DoF.

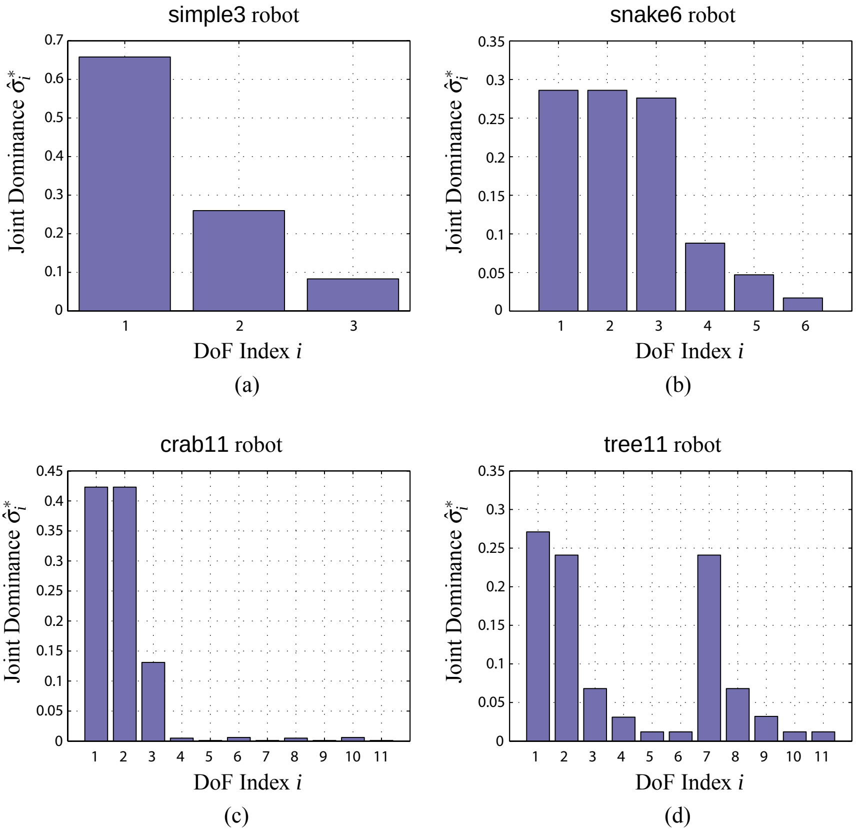


Fig. 3 Joint dominance coefficients of the four robots. Translational and rotational DoF of the base link have the strongest influence (a), (b), and (c). For each DoF the coefficient nicely reflects the position in the kinematic tree and the available joint range.

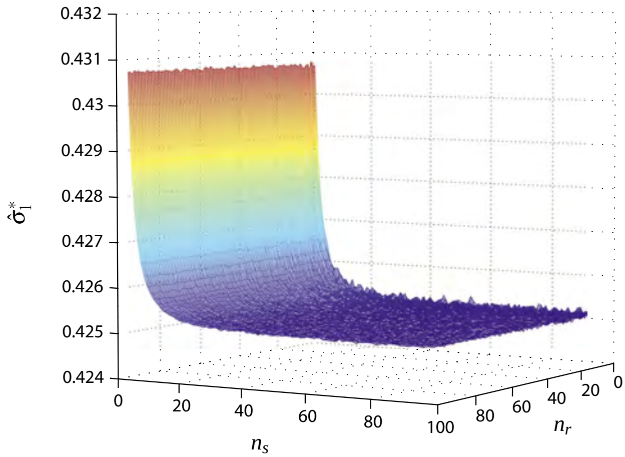


Fig. 4 Convergence of the first joint dominance coefficient of the `crab11` robot.

2D world[3]. Their respective degrees of freedom are marked by the numbered circles. The `simple3` robot in Figure 2(a) can move in the horizontal direction and actuate its rotational joints. Both the `snake6` robot and the `crab11` robot can translate and rotate freely in the plane and actuate all indicated degrees of freedom. The `tree11` robot has a fixed base, but can actuate any of its revolute joints.

Table 1 lists the admissable range of each DoF for all four robots. The translational DoF were constrained to the ranges usually used in planning scenarios, i.e. the size of the workcell the robot would move in.

The joint dominance coefficient vectors of the four robots are shown in Figures 3(a)-(d). The coefficients were calculated with $n_r = 20$ and $n_s = 100$. Quite clearly, the values nicely reflect the overall influence of each DoF. The translational DoF of the base dominate because they cause the largest displacement, regardless of the remaining configuration. After that, the rotation of the base has the greatest influence, followed by the hierarchy of the individual DoF in the kinematic tree.

Figure 4 shows the value of the first joint dominance coefficient $\hat{\sigma}_1^*$ of the `crab11` robot plotted over different values of n_r and n_s[4]. It can be seen that the value converges quite quickly in n_r and decently fast in n_s. For $n_r \geq 10$ and $n_s \geq 50$ the value remains invariant in the first three significant digits. Individual simulations over n_r and n_s for the other robots showed the exact same convergence behavior.

---

[3] Equivalent results would be obtained for a 3D world, as the displacement is measured by the sum of Euclidean distances and the joints are assumed to have only one revolute or prismatic DoF each.

[4] Since the algorithm uses random sampling, each value was averaged over 20 runs. The variance was observed to converge proportionally with the mean.

5 Conclusion

In this paper we have presented *Joint Dominance Coefficients*, a novel numeric procedure for estimating the influence of each DoF in a kinematic linkage on the overall displacement. The algorithm makes use of random sampling in the configuration space to assess the displacement caused by a change in each DoF over its range. The method was observed to converge quickly and yield meaningful weights for various tree-structured kinematic linkages.

The presented applications mainly stem from our involvement with sampling-based path planning. In fact the proposed method was developed to overcome the repeatedly encountered problem of manually having to pick suitable weights for a given linkage; however, we expect that it can be useful far beyond the domain in which we have examined it. Future research could aim at extending the method to the mentioned Lipschitz constants or to providing coefficients for dynamic constraints, i.e. maximum joint velocities.

Acknowledgements. This work is supported in part within the DFG excellence initiative research cluster *Cognition for Technical Systems – CoTeSys*, see also `www.cotesys.org`.

References

1. Angeles, J.: Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms (Mechanical Engineering Series). Springer, New York (2006)
2. Eberharter, J., Ravani, B.: Local metrics for rigid body displacements. Journal of Mechanical Design 126, 805–812 (2004)
3. Kuffner, J.: Effective sampling and distance metrics for 3d rigid body path planning. In: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA (2004)
4. Yershova, A., LaValle, S.M.: Efficient nearest neighbor searching for motion planning. In: Proceedings of the IEEE International Conference on Robotics and Automation (2002)
5. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
6. Abdel-Malek, K., Blackmore, D., Joy, K.: Swept volumes: Foundations, perspectives, and applications. International Journal of Shape Modeling (2002)
7. Weld, J.D., Leu, M.C.: Geometric representation of swept volumes with application to polyhedral objects. Int. J. Rob. Res. 9(5), 105–117 (1990)
8. Tang, X., Thomas, S.L., Amato, N.M.: Planning with reachable distances: Fast enforcement of closure constraints. In: Proceedings of the IEEE ICRA (2007)
9. Latombe, J.-C.: Robot Motion Planning. Kluwer, Norwell (1991)
10. Halton, J.H.: On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. Numerische Mathematik 2, 84–90 (1960)
11. Hammersley, J.M.: Monte-Carlo methods for solving multivariable problems. Annals of the New York Academy of Science 86, 844–874 (1960)
12. Matousek, J.: Geometric Discrepancy. Springer, Berlin (1999)

Learning Kinematics from Direct Self-Observation Using Nearest-Neighbor Methods

Hannes Schulz, Lionel Ott, Jürgen Sturm, and Wolfram Burgard

Abstract. Commonly, the inverse kinematic function of robotic manipulators is derived analytically from the robot model. However, there are cases in which a model is not a priori available. In this paper, we propose an approach that enables an autonomous robot to estimate the inverse kinematic function on-the-fly directly from self-observation and without a given kinematic model. The robot executes randomly sampled joint configurations and observes the resulting world positions. To approximate the inverse kinematic function, we propose to use instance-based learning techniques such as Nearest Neighbor and Linear Weighted Regression. After learning, the robot can take advantage of the learned model to build roadmaps for motion planning. A further advantage of our approach is that the environment can implicitly be represented by the sample configurations. We analyze properties of this approach and present results obtained from experiments on a real 6-DOF robot and from simulation. We show that our approach allows us to accurately control robots with unknown kinematic models of various complexity and joint types.

1 Introduction

Robotic manipulators typically require a Cartesian controller that maps the three-dimensional world space coordinates to joint configurations. The equations for the inverse kinematics can analytically be derived in advance when a robot model is available. Careful parameter calibration then ensures high accuracy in positioning tasks.

In the emerging field of home robotics, robots are likely to be assembled without surveillance of an engineer, meaning that no accurate construction model is available at design time. Further, home robots need to operate for extended periods of

---

Hannes Schulz · Lionel Ott · Jürgen Sturm · Wolfram Burgard
Department of Computer Science, University of Freiburg,
Georges-Köhler-Allee 79, 79110 Freiburg, Germany,
e-mail: `{schulzha,ottli,sturm,burgard}@informatik.uni-freiburg.de`

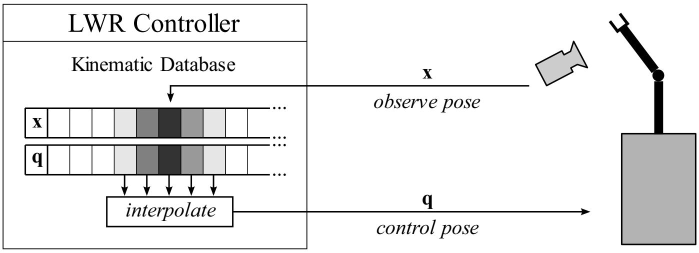


Fig. 1 Schematic overview of the proposed controller. Configurations are sent to the robot, the world-position is visually observed and recorded. Configurations for novel world positions are obtained by locally interpolating between known configurations.

time, during which they might be subject to damage, inaccurate repairs or other unforeseen modifications. This requires the development of new approaches to robust self-adaptation of the robot's internal kinematic models.

In this paper we introduce methods that can be used to approximate the inverse kinematic function without an a priori model. The robot visually observes the effects of its motor-commands on the position of its end-effector. The robot can then use these observations to estimate the kinematic function using nearest-neighbor methods (Figure 1). The remainder of this paper is organized as follows: Section 2 relates our paper to other approaches on kinematic learning. In Section 3 we introduce our approach on using nearest-neighbor methods for inverse kinematics estimation and planning. In Section 4, we analyze the performance and robustness in experiments on a real robot and in simulation.

2 Related Work

The problem of learning kinematics of robots has been investigated heavily in the past. In classical engineering, a typical solution starts out with a parameterized forward model, whose parameters are iteratively optimized in order to minimize the prediction error. However, only few approaches exist that do not rely on such a priori models of the robot.

Dearden and Demiris [2] present a method which estimates the parameters for the motion and observation model of a recursive Bayesian filter from visual observations. Whereas this approach has been applied successfully to a gripper with two separately controlled fingers, the sheer size of the parameter space prevents us from the direct application to highly complex robots. Sturm *et al.*[9] presented an alternative method that also uses Bayesian networks to estimate the robot model. In this work, the networks are learned by visually observing the individual parts of the robot while performing random movements. The authors estimate the global robot structure by learning the actuator models and finding the network topology that best explains the data in terms of data likelihood. The resulting model can then be used to predict the end-effector pose, so that gradient descent strategies can be applied for solving inverse kinematics. Our method presented in this paper, being simpler in nature, requires only a single marker on the end-effector instead of a marker for every rigid body part.

In the approach presented by Cheah *et al.*[1], the robot learns, instead of the end-effector position, only the Jacobian, which describes the first derivative of the robot's kinematics and dynamics, using the visually observed world position of the end-effector, the joint velocities and configurations. The Jacobian can then be applied for calculating the inverse kinematics. The approach is evaluated on 2-DOF (degrees of freedom) robots in a 2D workspace only. In our experiments we demonstrate, that our approach also works in the full 3D space and also with up to 6-DOF.

In the area of visual servoing (see the work by Malis for an overview [8]) the robotic arm is typically controlled locally by observing the effect of motor commands on the end-effector in the image space. As visual servoing is a pure online-technique that relies on visual feedback, it cannot be used for (offline) planning. We could however apply visual servoing to improve the accuracy of our method after a plan has been generated.

To sum up, compared to previous approaches our method does not rely on prior knowledge of the robot model, it scales better even with an increasing number of DOF, and allows for global planning. Additionally, our approach is able to implicitly represent the environment and to handle self-collisions.

3 The Model-Free Approach

We collect data by sending a configuration request $\mathbf{q} = (q_1, q_2, \ldots, q_n)$ to a robot with n joints and observing the resulting end-effector position $\mathbf{x} = (x_1, x_2, x_3)$ in 3D space. After m trials, this results in a kinematic database

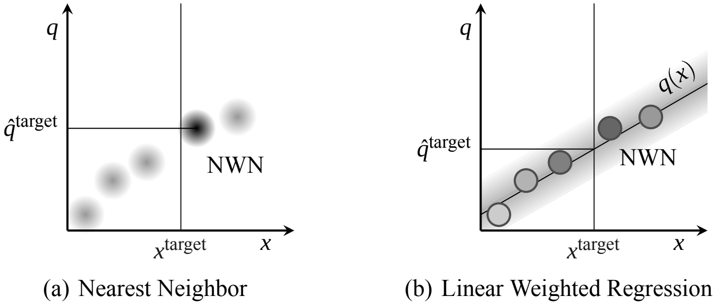
$$S = \left(\langle \mathbf{q}^1, \mathbf{x}^1 \rangle, \langle \mathbf{q}^2, \mathbf{x}^2 \rangle, \ldots, \langle \mathbf{q}^m, \mathbf{x}^m \rangle\right).$$

If a collision is encountered, the corresponding sample is rejected. We describe two online learning methods for estimating the inverse kinematics from the acquired data. This allows us to generate a valid configuration $\mathbf{q}$ for a given target position $\mathbf{x}$. We furthermore present an approach for planning based on the kinematic database.

3.1 Nearest Neighbor Method

Using a probabilistic formalization, we assume that the collected samples in S are drawn from a joint probability distribution $p(\mathbf{q}, \mathbf{x})$ defined by the unknown underlying mechanical model. Estimating a configuration $\hat{\mathbf{q}}^{\text{target}}$ that is supposed to yield $\mathbf{x}^{\text{target}}$ then corresponds to finding a configuration that maximizes the conditional probability

$$\mathbf{q}^{\text{target}} = \arg\max_{\mathbf{q}} p(\mathbf{q} | \mathbf{x}^{\text{target}}).$$



(a) Nearest Neighbor (b) Linear Weighted Regression

Fig. 2 Visualization of our approach for a 1-DOF robot. (a) Shaded areas: Collected samples, NWN: Nearest world-neighbor of x^{target}, Horizontal line: Joint position $\hat{q}^{\text{target}}$ associated with NWN. (b) Circles: Subset of collected samples which are close to NWN in configuration space, Shaded area: $p(q,x)$, with an assumed linear relationship between x and q. Evaluation of fitted linear function $q(x)$ at the desired world position x^{target} yields desired configuration $\hat{q}^{\text{target}}$. Lighter shading of samples indicates higher world space distance to x^{target} and thus less influence on the slope of $q(x)$.

As a first approximation, we search for the closest world-neighbor to the target location $\mathbf{x}^{\text{target}}$ in our database, and assign the associated joint value to $\hat{\mathbf{q}}^{\text{target}}$:

$$\hat{\mathbf{q}}^{\text{target}} = \arg\min_{\mathbf{q}^i \in S} \left\| \mathbf{x}^{\text{target}} - \mathbf{x}^i \right\|_2 \tag{1}$$

This idea is also visualized for a 1-DOF robot in Figure 2a.

There are two obvious disadvantages of this method: First, it is sensitive to the noise in the observations. Second, the accuracy of the method is directly related to the sample density in S. Interestingly, it does not rely on the dimensionality of $\mathbf{q}$ or the effective complexity of the mapping between configuration space and world space; in particular, the database lookups can be implemented very efficiently for example by using KD-trees [4]. Also, in the noise-free case, the nearest neighbor (NN) method will converge to the true function in the limit.

3.2 Linear Weighted Regression

To improve the NN method one can interpolate between the k-nearest neighbors in configuration space. However, this interpolation is not straightforward since the configurations resulting in a single world position are ambiguous, as a robot may have multiple ways to approach it. We solve this problem by interpolating only between configurations that are "close" in configuration space. We start by selecting the nearest neighbor $\langle \mathbf{q}^w, \mathbf{x}^w \rangle$ to $\mathbf{x}^{\text{target}}$ in world space and from there select the set of k nearest neighbors in configuration space:

$$\bar{S} = \mathop{\arg\min}_{\bar{S} \subset S, |\bar{S}| = k} \sum_{\langle \mathbf{q}, \mathbf{x} \rangle \in S} \| \mathbf{q} - \mathbf{q}^{w} \| \tag{2}$$

This locally selected subset $\bar{S}$ of kinematic samples can now be used for interpolation, because all samples in $\bar{S}$ are neighbors in configuration space.

Note that rotational joints, for example, move the end-effector on a circular path. Even though it is possible to interpolate between the joint angles, the information how a joint affects the position of the end-effector is part of the model, which is not available. We assume that the actuators are locally linear, allowing us to fit a linear model to the samples in $\bar{S}$ such that for a 1-DOF robot whe have

$$\forall \langle \mathbf{q}, \mathbf{x} \rangle \in \bar{S} : q = w_3 x_3 + w_2 x_2 + w_1 x_1 + w_0, \tag{3}$$

where $\mathbf{w} = (w_3, w_2, w_1, w_0)$ is the weight vector defining a plane.

Since the data contains observation noise and the assumption of local linearity is only an approximation, the linear model will not fit perfectly. Therefore, we estimate $\widehat{\mathbf{w}}$ from the least-squares solution

$$\widehat{\mathbf{w}} = \mathop{\arg\min}_{\mathbf{w}} \sum_{s^i \in \bar{S}} \left(q^i - (w_3 x_3^i + w_2 x_2^i + w_1 x_1^i + w_0) \right)^2 = \mathop{\arg\min}_{\mathbf{w}} \| X\mathbf{w} - Q \|_2^2, \tag{4}$$

where $X \in \mathbb{R}^{N \times 4}$ with $N = |\bar{S}|$ is the number of joint neighbors used. $Q \in \mathbb{R}^{N \times 1}$ are the corresponding joint values and $\mathbf{w} \in \mathbb{R}^{4 \times 1}$ are the weights we want to estimate. Using SVD [5], Equation (4) can be solved efficiently. To reduce the effects of non-linearity, we weight the error introduced by a sample and thus its influence on the slope of the function by the inverse of its squared world space distance to $\mathbf{x}^{\text{target}}$. Given the weights $\widehat{\mathbf{w}}$, we can determine the configuration $\hat{q}^{\text{target}}$ for a novel world position $\mathbf{x}^{\text{target}}$ as

$$\mathop{\arg\max}_{q} p(q|\mathbf{x}) \approx \hat{q}^{\text{target}} = \widehat{w}_3 \mathbf{x}_3^{\text{target}} + \widehat{w}_2 \mathbf{x}_2^{\text{target}} + \widehat{w}_1 \mathbf{x}_1^{\text{target}} + \widehat{w}_0$$

When this method is extended to multiple joints, the weight vector $\mathbf{w}$ needs to to be replaced by a weight matrix W. This would treat the individual joints as being independent during the maximization of $p(\mathbf{q}|\mathbf{x}^{\text{target}})$. Of course, this is not the case, choosing the value of one joint limits the possible values of subsequent joints. Actually, finding the configuration with highest likelihood of leading to a pose $\mathbf{x}^{\text{target}}$ involves finding the global maximum, i.e., computing $\arg\max_{q_1 \ldots q_n} p(q_1, \ldots, q_n|\mathbf{x}^{\text{target}})$. We suggest to find this maximum greedily by factorizing this probability distribution according to the chain rule as

$$\begin{aligned}
\mathbf{q}^{\text{target}} &= \mathop{\arg\max}_{q_1 \cdots q_n} p(q_1, \ldots, q_n|\mathbf{x}^{\text{target}}) \\
&= \mathop{\arg\max}_{q_1 \cdots q_n} p(q_1|\mathbf{x}^{\text{target}}) p(q_2|\mathbf{x}^{\text{target}}, q_1) \cdots p(q_n|\mathbf{x}^{\text{target}}, q_1, \ldots, q_{n-1})
\end{aligned}$$

and then maximizing the factors one by one, extending the linear least-squares model from Equation (4) for each joint:

$$\hat{\mathbf{q}}^{\text{target}} = \begin{pmatrix} \arg\max_{q_1} p(q_1|\mathbf{x}^{\text{target}}) \\ \arg\max_{q_2} p(q_2|\mathbf{x}^{\text{target}}, q_1) \\ \dots \\ \arg\max_{q_n} p(q_n|\mathbf{x}^{\text{target}}, q_1, \dots, q_{n-1}) \end{pmatrix} \tag{5}$$

In this way, the linearization of the kinematic model for each joint takes into account all previous joints. Note that the factorization in Equation (5) is valid even for an arbitrary order of the joints.

3.3 Path Planning

With the methods described above, we can infer joint configurations that move the robot arm to previously un-encountered target positions. The data stored in the kinematic database S implicitly encodes information about the robot itself and the surroundings, as it contains only samples from accessible, obstacle-free regions both in configuration space and world space. In the following, we present a planning method based on probabilistic roadmaps [7]. Planning is performed directly in the kinematic database, by using a cost metric combining world space and configuration space metrics.

From our kinematic database, we create a undirected graph with the nodes corresponding to the samples in the kinematic database. Two nodes $s^i, s^j \in S$ are assumed to be connected in the graph, when they are close in world space, i.e., $\left\|\mathbf{x}^i - \mathbf{x}^j\right\|_2 < d_{max}$.

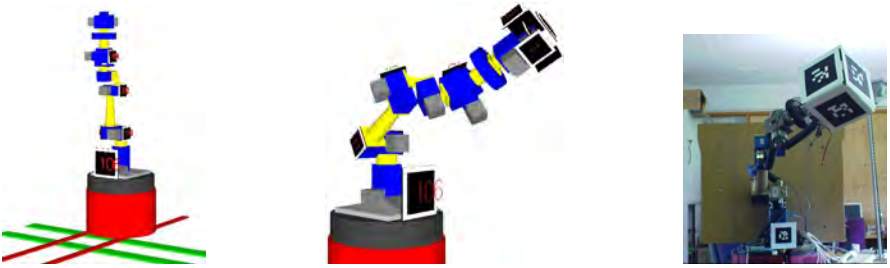
The planning problem then reduces to finding the shortest path $\langle \pi(0), \dots, \pi(n) \rangle$ through the graph [6]. To avoid long initial paths and to get as close as possible to the target, we add a starting node at the current configuration $\mathbf{q}^{\text{current}}$ of the robot, and an ending node close to $\mathbf{x}^{\text{target}}$:

$$s^{\pi(0)} = \underset{s^i \in S}{\arg\min} \left\|\mathbf{q}^i - \mathbf{q}^{\text{current}}\right\|_2, \qquad s^{\pi(n)} = \underset{s^i \in S}{\arg\min} \left\|\mathbf{x}^i - \mathbf{x}^{\text{target}}\right\|_2.$$

We find the shortest path using the A* algorithm. Our cost function takes into account differences in world space and in configuration space:

$$\sum_{i=0}^{n-1} \left(\alpha \left\|\mathbf{x}^{\pi(i)} - \mathbf{x}^{\pi(i+1)}\right\|_2 + \beta \left\|\mathbf{q}^{\pi(i)} - \mathbf{q}^{\pi(i+1)}\right\|_2^2 \right),$$

where $\alpha = 1/d_{max}$, and β can be set likewise to 1 divided by the maximum joint distance. The squared joint-distance causes the planner to prefer many small steps through the roadmap instead of one large one. This increases the resolution of the controlled path and simultaneously penalizes shortcuts through potential obstacles that appear as unpopulated regions in the roadmap.



(a) (b) (c)

Fig. 3 Robots used for testing. (a) Robots with rotational and prismatic (“rails”) joints, (b) Simulated 6-DOF robot, and (c) Real-world 6-DOF counterpart.

4 Results

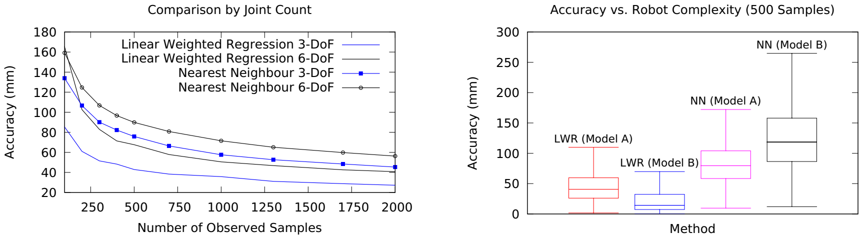
We first investigate the performance of both methods for a varying number of joints. Then, the influence of different joint types is evaluated. Finally we applied the methods to both a simulated version of our 6-DOF robot and the real robot composed of rotatory Schunk PowerCube modules.

4.1 Positioning Accuracy

We compared the accuracy of the a simulated robot with 3 versus 6 active joints. The results are displayed in Figure 4a. For an arm length of 100 cm operating in a hemisphere. The Nearest Neighbor method achieves an accuracy of 9.0 cm (3-DOF) and 10.7 cm (6-DOF) with 300 samples, while Linear Weighted Regression obtains an accuracy of 5.2 cm and 8.3 cm, respectively. Note that the decrease in accuracy of Nearest Neighbor can only be attributed to the increasing workspace volume. Linear Weighted Regression, however, additionally suffers from the increasing dimensionality in configuration space, nevertheless it outperforms Nearest Neighbor in both cases.

Another property we investigated was the ability to cope with different types of joints. For this purpose, we constructed two robots. The first robot (similar to Figure 3b) has five rotational joints in the arm. For the second robot (Figure 3a) we replaced three rotational joints with prismatic joints.

Analyzing the accuracy of both methods, we find that Linear Weighted Regression outperforms Nearest Neighbor on both robots (Figure 4b). As expected, Nearest Neighbor performance drops when the workspace volume increases. The accuracy of Linear Weighted Regression increases since prismatic joints conform to the linearity assumption and are thus better suited for interpolation.



(a) (b)

Fig. 4 (a) Accuracy of our methods on 3 and 6-DOF robots with arm length of 1 m. (b) Prismatic joints (conform to linearity assumption) increase accuracy of LWR, although workspace volume increases. Contrarily, Nearest Neighbor (NN) performance decreases. Model A contains only rotational joints, while Model B contains prismatic joints as well.

4.2 Real Robot Evaluation

The following experiments aim at evaluating the general performance of the approaches with a complex robot, with six rotational joints. Our robot "Zora", depicted in Figure 3c, consists of a B21 base and a robotic arm composed of Schunk Powercubes on top with a total length of 100 cm.

In order to observe the 3D position of the end-effector, we placed a 20cm × 20cm × 20cm AR-Toolkit [3] marker cube on it (Figure 3c). We inferred the end-effector position from the 6D pose of the observed markers, with an average observation accuracy of 7.4 cm. Alternatively, motion capturing systems or a robot-mounted camera observing its position with respect to the environment could be used.

The database S was filled with 250 samples by sending a joint-configuration to the robot and observing the resulting world position of the end-effector. The recording took approximately two hours. These joint-configurations were generated by choosing a random value for each joint.

We evaluated our methods in simulation by determining the accuracy of our simulated robot with respect to the samples gathered on the real robot for noise-free ground truth comparison.

With Nearest Neighbor, we achieved a positioning accuracy of 8.2 cm ($\sigma =$ 6.0 cm), whereas we determined the accuracy of Linear Weighted Regression to be 5.7 cm ($\sigma = 5.8$ cm). The reported values were obtained on an arm with a length of 100 cm.

4.3 Path Planning

We qualitatively tested our planning system described in Section 3.3. To this end, we constructed a simple test environment containing a 2-DOF planar robot and two obstacles. The robot had to move to random positions in its workspace. We then

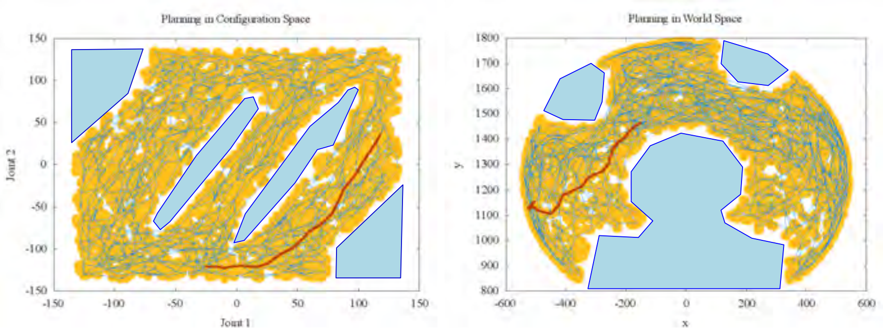


Fig. 5 Planning in configuration space (left) and world space (right). Our method is able to find a path short in both spaces. Blue regions indicate areas inaccessible to the robot (self-collisions, obstacles). Yellow: recorded samples. Blue lines: roadmap of Planner. Red lines: sample trajectory.

analyzed the resulting trajectories. The 2D visualization of world space and the corresponding configuration space with a sample trajectory is displayed in Figure 5.

Although the trajectory does not run close to obstacles in world space, the corresponding configuration space trajectory is avoiding such an obstacle by following a curved path. Moreover, the path is curved more than necessary in configuration space to allow an almost straight path in world space. It therefore seems that our approach can help to circumvent problems that would occur if planning was only performed in configuration space.

5 Conclusions

In this paper, we presented a general and efficient approach to learn the inverse kinematic function for robots with unknown mechanical design. In contrast to classical approaches, which require the knowlege of the model to determine the kinematic function, we collect data consisting of visually observed end-effector positions and the corresponding robot configurations. We then locally approximate the inverse kinematic function. Further, we showed how a planner can construct a probabilistic roadmap directly from the kinematic database. In experiments carried out with simulated and real robots with varying number of DOFs and joint types, we obtained accurate positioning results.

In future work, we would like to extend the world space representation to full 6D, i.e., control both the translational and rotational components. For the planning, we want to additionally include those kinematic samples that were previously rejected due to (self-)collisions. Thanks to the simplicity and efficiency of our approach, we plan to realize a real-time implementation on an embedded device in near future.

References

1. Cheah, C.C., Liu, C., Slotine, J.J.E.: Adaptive jacobian tracking control of robots based on visual task-space information. In: Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 3498–3503 (2005)
2. Dearden, A., Demiris, Y.: Learning Forward Models for Robots. In: International Joint Conference on Artificial Intelligence, vol. 19, p. 1440. Lawrence Erlbaum Associates LTD, Mahwah (2005)
3. Fiala, M.: ARTag, a Fiducial Marker System Using Digital Techniques. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005, vol. 2 (2005)
4. Freidman, J.H., Bentley, J.L., Finkel, R.A.: An algorithm for finding best matches in logarithmic expected time. ACM Trans. Math. Softw. 3(3), 209–226 (1977)
5. Golub, G., Kahan, W.: Calculating the singular values and pseudo-inverse of a matrix. Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis 2(2), 205–224 (1965)
6. Latombe, J.: Robot Motion Planning. Kluwer Academic Publishers, Norwell (1991)
7. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
8. Malis, E.: Survey of vision-based robot control. ENSIETA European Naval Ship Design Short Course, Brest, France (2002)
9. Sturm, J., Plagemann, C., Burgard, W.: Unsupervised body scheme learning through self-perception. In: IEEE International Conference on Robotics and Automation, 2008. ICRA 2008, pp. 3328–3333 (2008)

Guidelines for Low Mass and Low Inertia Dynamic Balancing of Mechanisms and Robotics

Volkert van der Wijk and Just L. Herder

Abstract. Dynamic balance of machines is important when, for example, high precision in combination with low cycle times is necessary. One of the major problems with dynamic balancing is that considerable mass and considerable inertia need to be added to the mechanism. So far, only a few studies have been carried out into the comparison of various dynamic balancing principles in order to reduce these additions. Based on the findings of these studies, this paper aims to formulate guidelines for the design of dynamically balanced mechanisms with low mass and low inertia additions. Furthermore, the influence of limited design space on the resulting mass and inertia is investigated.

1 Introduction

Whenever mechanisms and robots have to move at high speeds, dynamic motion of the machine elements induces vibrations (shaking forces and shaking moments) to their base and surroundings, causing noise, wear and fatigue [6], discomfort [3] and inaccuracy [12]. Dynamically balanced mechanisms (i.e. mechanisms of which both linear momentum and angular momentum are constant) however do not induce vibrations. Therefore, high accuracy in combination with low cycle times is possible. Further advantages include that in factories, machines do not influence one another and floors do not need a special construction to withstand machine's dynamic loads. For moving objects and vehicles (at the ground or in space), dynamic balance is important to maintain position, orientation and stability [5, 1].

One of the major problems with dynamic balancing is that a considerable amount of mass and a considerable amount of inertia need to be added to the mechanism [4, 12], while generally also the complexity of the mechanism is increased. As an

---

Volkert van der Wijk · Just L. Herder
Department of Biomechanical Engineering, Faculty of Mechanical,
Maritime and Materials Engineering, Delft University of Technology,
Mekelweg 2, 2628 CD Delft, The Netherlands,
e-mail: `v.vanderwijk@kineticart.nl`, `j.l.herder@tudelft.nl`

example, Wu and Gosselin in 2005 [14] were the first to completely dynamically balance a 6-Degree-of-Freedom (DoF) spatial parallel mechanism. However to balance a payload of 50 grams, 15 counter-masses (CM) and 9 counter-inertias (CI, counter-rotating elements), totalling 4.5 kg, had to be added, yielding a ratio of balancer elements' total mass over payload of 90.

It is likely that this disadvantage is an important reason for the limited research and application interest in dynamic balancing. In (planar) machines, dynamic balancing is more common than in (spatial) robotics. To reduce the increase of mass and inertia, and hence complexity, to an acceptable level, dynamic balancing usually is done partly. The most common approaches include to only force balance some elements [6, 7], to dynamically balance some specific frequencies [2], and constrained optimization of design parameters [9]. However, to reach the accuracy in combination with the production speed that future mechanisms and robotics will need to have, complete dynamic balancing (i.e. eliminating all shaking forces and shaking moments) with a low addition of mass and inertia is necessary.

Only few studies and numerical experiments have been done concerned with the comparison of general balancing principles regarding their addition of mass and addition of inertia [4, 12, 13, 11]. These studies argue that, for the investigated balancing principles, dynamic balancing tends to increase the mass and inertia considerably. In the case that both low mass and low inertia are of concern, the balancing principle of 'duplicate mechanisms' (Fig. 1a), where three axial and mirror copies of the initial mechanism are added and altogether result in a dynamically balanced system, proved to be the most advantageous balancing principle. It also turned out that the use of 'Counter-Rotary Counter-Masses' (Fig. 1b), where the CM for force balancing also is used as CI for moment balancing, is more advantageous than the common practice of applying separate CI's for moment balancing (Fig. 1c).

The objective of this article is to formulate general guidelines for the development of dynamically balanced machinery that have a minimum of additional mass and a minimum of additional inertia, based on the literature cited above. In addition, the influence of the design space, i.e. the space that is available for the balancing elements, is taken into account, since in practise this often has a considerable influence on the performance.

There is a difference between the inertia of an element and the inertia of the mechanism. This paper deals with the minimization of the inertia of the mechanism, which is represented by the reduced inertia as defined in [8]. This is the inertia of the mechanism reduced to the input parameters, in fact the inertia that an actuator feels when driving the system.

The discussion of low mass and low inertia dynamic balancing is divided in three different sections. First the influence of the balancing principles and their application, i.e. the resulting balancing architecture, on the mass and inertia additions is discussed. The influence of the balancing parameters, i.e. the masses and dimensions of the elements, is treated in the second section. The third section investigates the influence of the design space on the mass and inertia additions in dynamically balanced mechanisms. After these three sections general guidelines for low mass and low inertia dynamic balancing are formulated and listed.

2 Influence of Balancing Architecture

The architecture of the elements that are added to the mechanism for dynamic balancing, influences the addition of mass and the addition of inertia. The architecture of these balancing elements depends both on the choice of balancing principle with which the mechanism is balanced and how the chosen balancing principle is applied. In [12] and [13], from literature concerning planar and spatial, serial and parallel mechanisms, in total three fundamental and generally applicable balancing principles were found, which are shown in Fig. 1. Figure 1a shows the principle of duplicate mechanisms (DM). Although for moment balancing a single mirror copy would be sufficient, three horizontal and vertical mirror copies of the initial mechanism are necessary for the full elimination of the shaking forces in both horizontal and vertical direction. Figure 1c shows the principle of Separate Counter-Inertias (SCI), where a counter-mass (CM) (m^*) is used for force balancing only, while moment balancing is accomplished with a separate counter-inertia (CI) (I^*_{CI}). Figure 1b shows the principle of Counter-Rotary Counter-Masses (CRCM) where the moment is balanced by using the inertia of the CM (I^*) itself.

The influence of the architecture of these principles on the addition of mass and inertia depends on the contribution of the balancing elements to the dynamic balance [13]. For the SCI-principle, the CM does only contribute to the force-balance, however the inertia of the CM increases the inertia of the mechanism and since the inertia is balanced by the CI, the inertia of the CI increases also. For the CRCM principle, the CM contributes to both the force and moment balance. The inertia of the CM is directly used to balance the inertia of the mechanism.

The influence of the element contribution to the dynamic balance is also shown in [10], where a double pendulum is balanced with CRCMs as shown in Fig. 2. There are various ways to apply the CRCM principle to a double pendulum, for instance by simply stacking two CRCM-balanced links with the configuration of Fig. 1b.

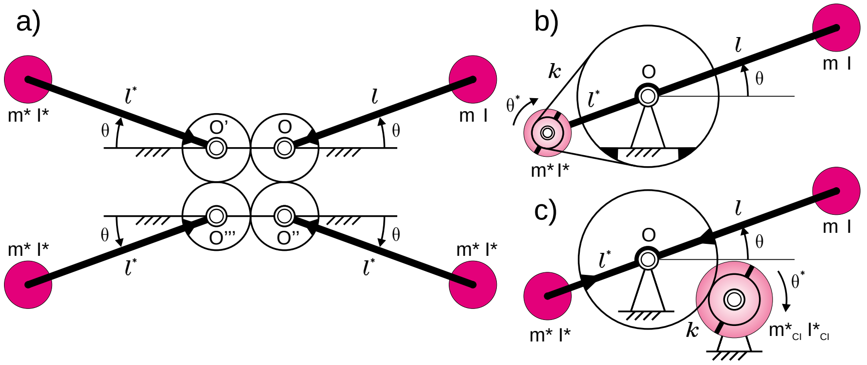


Fig. 1 General balancing principles, balancing by using (a) Duplicate Mechanisms (DM), (b) Counter-Rotary Counter-Masses (CRCM) and (c) Counter-Masses and Separate Counter-Inertias (SCI) (source [13]).

However the configuration of Fig. 2 showed to be most advantageous for low inertia addition, since the inertia I_2^* of the CRCM of link 2 is balancing the moment of link 2 for any motion of the linkage, while it does not influence the inertia I_1^* of the CRCM at link 1.

Regarding the addition of mass, both the CRCM and SCI principle have the same problem. Although the CM m_2^* contributes to the dynamic balance of link 2, it negatively influences CM m_1^*. In fact CM m_2^* itself must also be balanced, which is done by CM m_1^* and is an important source of mass addition. The reason for this is that CM m_2^* is balancing link 2 about a moving point, while if link 2 would be balanced about a fixed point directly, the problem would be eliminated.

An example of this is shown in [11], where a double pendulum is balanced with only one CRCM (as a balanced pantograph). The problem in this case is that, in order to balance both the force and moment of each link for any motion, the mass distribution is not constant, even though the CoM remains stationary at the fixed point *O*. This means that the inertia of the mechanism depends on the position and velocity of the mechanism. Hence it is not possible to balance the moment of this mechanism solely with mechanical elements (passive balancing). In [11] this problem is solved by actively counter-rotating the CRCM with an additional actuator (active balancing). The results indeed show that this approach is more advantageous than the passive approach with CRCMs.

The contribution of the balancing elements to the dynamic balance is not the reason why the DM principle turned out to be the most advantageous for low mass and low inertia dynamic balancing. Since one mirror copy is sufficient for balancing the moment of the mechanism, the two other copies only contribute to the force balance, however they increase the inertia of the mechanism. The advantage of the DM principle will become clear in the next section.

In summary, an architecture in which all balancing elements contribute to both the force and moment balance and where the mechanism is force balanced about a

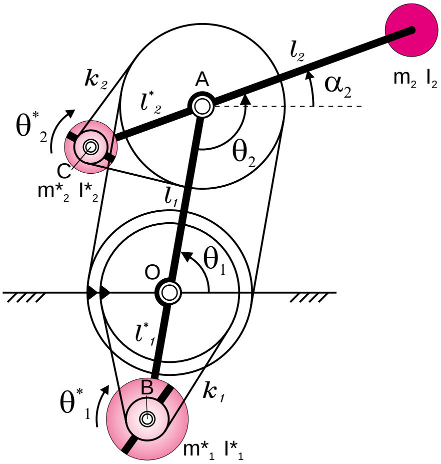


Fig. 2 CRCM principle applied to a double pendulum in a low inertia configuration. Because the CRCM of link 2 at point *C* is driven by a chain connection with a gear that is mounted on the base at *O*, the inertia I_2^* of this CRCM does not influence the inertia I_1^* of the CRCM at link 1 (source [10]).

stationary point directly is most advantageous for dynamic balancing with low mass and low inertia addition. This applies to any mechanism, from single-DoF planar to multi-DoF spatial. Expanding the idea of combining the function of force balancing and the function of moment balancing into the same element to the architecture of the mechanism itself, it could be stated that for low mass and a low inertia dynamically balanced mechanisms all the mechanism links should be, besides kinematically relevant, also dynamically relevant by contributing to the dynamic balance.

3 Influence of Balancing Parameters

From the studies in [11], [12] and [13] it can be concluded that the main parameters of the balancing elements of concern for the addition of mass and addition of inertia are the position of the CM with respect to its center-of-rotation (CoR) and the transmission ratio of the counter-rotation with the rotation of the mechanism. The former parameter determines the mass of the CM and the mechanism inertia to be balanced while the latter determines the inertia of the CI.

Two other parameters that influence the mass and inertia addition are the mass distribution (inertia) of the CM that is not used for moment balancing and the mass of the CI that is not used for force balancing. While the former influences the mechanism inertia to be balanced, the latter influences the total mass of the system. Of a CM that is also used as CI, the mass distribution is determined solely by the main parameters.

Table 1 Equations for force balance, moment balance, mechanism inertia and total mass for the CRCM and the SCI principle of the mechanisms in Fig. 1 (source [13]).

	CRCM principle	SCI principle
Force Balance:	$ml = m^* l^*$	$ml = m^* l^*$
Dynamic Balance:	$I + ml^2 + m^* l^{*2} + kI^* = 0$	$I + ml^2 + I^* + m^* l^{*2} + kI^*_{cr} = 0$
Mechanism Inertia:	$I_\theta^{red} = I + ml^2 + m^* l^{*2} + k^2 I^*$	$I_\theta^{red} = I + ml^2 + I^* + m^* l^{*2} + k^2 I^*_{cr}$
Total Mass:	$m_{tot} = m + m^*$	$m_{tot} = m + m^* + m^*_{cr}$

In [13] the influence of these parameters becomes most clear by using the mechanisms of Fig. 1 as an example. The equations for the force balance, the moment balance, the mechanism inertia and the total mass of the mechanisms are given in Table 1. Substituting the force balance and moment balance equations in the equations of the total mass and mechanism inertia results for the CRCM principle in

$$m_{tot} = m\left(1 + \frac{l}{l^*}\right) \tag{1}$$

$$I_\theta^{red} = I + ml^2 + mll^* + k\left(I + ml^2 + mll^*\right) \tag{2}$$

and for the SCI principle in

$$m_{tot} = m\left(1+\frac{l}{l^*}\right)+m^*_{CI} \tag{3}$$

$$I^{red}_{\theta} = I+ml^2+mll^*+I^*+k\left(I+ml^2+mll^*+I^*\right) \tag{4}$$

For the DM principle the equations for the mechanism inertia and total mass are

$$m_{tot} = 4m \tag{5}$$

$$I^{red}_{\theta} = 4(I+ml^2) \tag{6}$$

In this example the position of the CM is represented by l^* and the transmission ratio by k. For the SCR principle it is visible that besides the influence of l^* and k, the mass of the CI m^*_{CI} influences the total mass and the mass distribution of the CM I^* influences the inertia of the mechanism, while these influences do not exist for the CRCM principle since the inertia of the CM is used as CI. The equations for the DM principle do not have any balancing parameters. The only possibility is to copy the initial mechanism three times.

For both the CRCM as the SCI principle holds that a large transmission ratio results into a large mechanism inertia. According to the equation of the moment balance in Table 1, a large transmission ratio results into a small inertia of the CI. This implies that for a low addition of mechanism inertia, the inertia of the CI should be large. The explanation for this is the quadratic appearance of the transmission ratio in the equations of the mechanism inertia.

This last observation is the reason why the DM principle proved to be the most advantageous for the combination of low mass and low inertia dynamic balancing. The transmission ratio of the DM principle is relatively low (-1) and the inertia of the counter rotating elements is large, since their inertias depend on the length of the links which have a relatively large size. Generally, links can be characterized by having a relatively large inertia with respect to their mass.

A large l^* results for both the CRCM as the SCI principle into a low total mass but also into a large inertia of the mechanism. This means that there is a trade off between the addition of mass and the addition of inertia.

The comparing studies of [11], [12] and [13] were limited to 2-dof mechanisms. With multi-DoF planar or spatial mechanism, where DoFs influence one another, optimal values for the mechanism inertia and the total mass are likely to exist. Whenever the parameter values for the minimum inertia are different than for the minimum mass, a trade off remains. This still needs further investigation.

4 Influence of Design Space

The previous section showed that for low mass addition, CMs should be placed far away from their CoRs and for a low inertia addition the inertia of the CIs should be large, which by keeping the mass low means a large size. However in practise the dimensions of the balancing elements are limited by the design space. The comparative studies did not take this into consideration yet. Two interesting questions however arise. What is the optimal design of the mechanism within a specific design space and in relation with the other machine components, what is the optimal design space?

Figure 3a shows the CRCM principle of Fig. 1b, however here the CRCM is designed as a ring to obtain a large inertia. Also the transmission ratio is kept small. If the workspace of this manipulator by full rotation is the circumscribed circle about the mechanism, it is not possible to place other machine components within this space since they would interfere with the manipulator. However, it is possible to place the balancing elements in this area. By choosing the design space to be equal to the workspace, no concessions need to be done for the design of other machine parts.

If the target of the configuration of Fig. 3a is to have a low inertia, l^* and I^* have to be large. However by increasing the link length l^* the size of the CRCM decreases and hence the inertia of the CRCM decreases. Increasing the size of the CRCM results into a decreasing l^*. This means that by taking the design space into account, an optimum for the addition of inertia to the mechanism will exist.

Figure 3b shows a situation in which the design space is outside the workspace and might depend on, for instance, the location of the other machine components or the available space behind the machine in the factory. The target is to design the most advantageous mechanism for low mass and low inertia dynamic balancing. In Fig. 3b the configuration is shown that has the largest CRCM for a large inertia. Although both increasing and decreasing the length l^* makes the CRCM become smaller, it does not mean that the lowest mechanism inertia is found in this position.

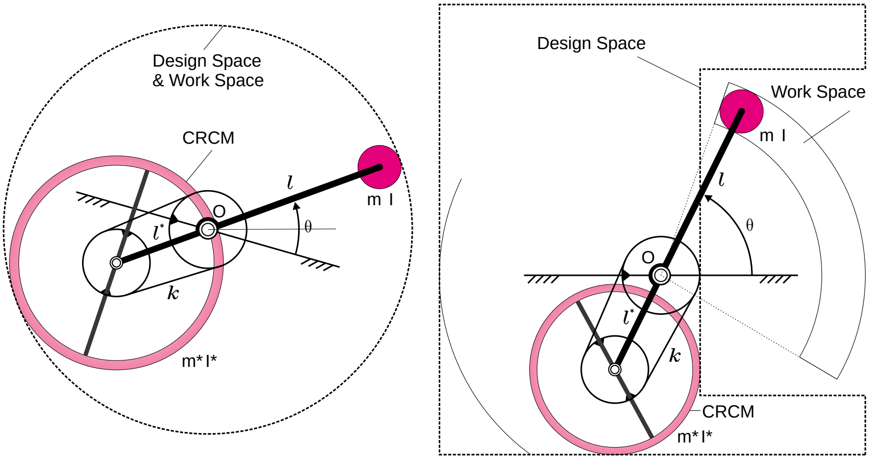


Fig. 3 Design limitations due to the available space for the dynamic balancing elements. (a) Design space equal to workspace; (b) Design space outside the workspace.

The ring-shaped CRCM of Fig. 3 requires a large empty space. In fact the CRCM can attain any shape as long as its CoM is at the CoR of the CRCM. The CRCM can for in instance be a link as shown in Fig. 4b with transmission ratio $k = -1$. The DM principle already showed that counter-rotating links (CRL) are advantageous. Although the length of this CRL is larger than the diameter of the ring shaped CRCM, the design space that is necessary for the balancing elements is much smaller. With this solution it is possible to place other machine components nearby.

Figures 3 and 4 showed some approaches, that are also suitable for multi-DoF planar and spatial mechanisms, of integrating dynamic balancing into machines. The best solution however, still depends on the wishes of the designer since a trade off between mass addition and inertia addition must be made. To assist the designer with this choice, [12] introduces the Mass-Inertia factor μ which is defined as

$$\mu = w_M \cdot \hat{m} + \sum_j w_j \cdot \hat{I}_j \tag{7}$$

and weights the relative addition of mass and the relative addition of inertia with to be chosen weight factors w_M and w_j respectively. $\hat{m}$ and $\hat{I}_j$ are the mass ratio and the inertia ratio of input parameter j which are calculated with $\hat{m} = \frac{m_{tot}}{m_{tot}^o}$ and $\hat{I}_j = \frac{I_j^{red}}{I_j^{red,o}}$. m_{tot}^o and $I_j^{red,o}$ are the total mass and the mechanism inertia per input parameter before balancing, respectively, and m_{tot} and I_j^{red} are the total mass and mechanism inertia after balancing, respectively. For the lowest Mass-Inertia factor, the balanced mechanism is optimal for low mass and low inertia. An optimization of the balanced mechanism for low mass and low inertia including also the design space may be most useful when the complete mechanism is taken into account, including all other machine parts.

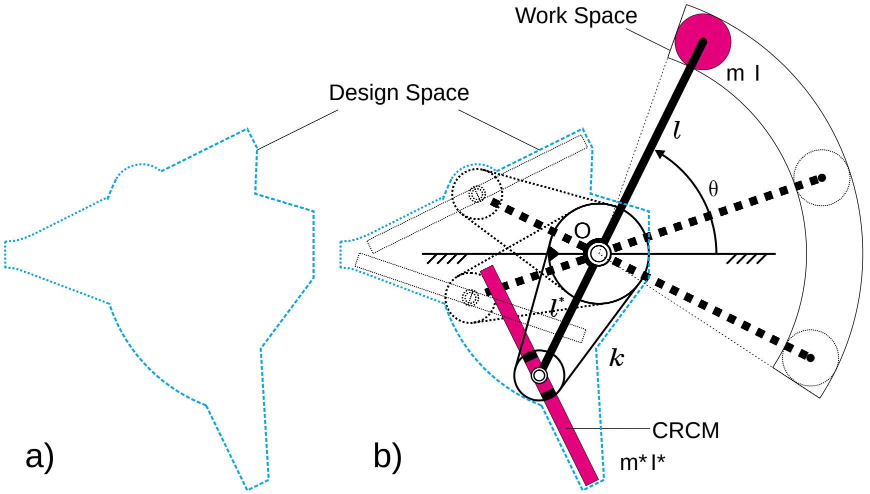


Fig. 4 Optimizing the mass and inertia addition together with the design space.

5 Guidelines for Low-Mass and Low-Inertia Dynamic Balancing

Minimal addition of mass and minimal addition of inertia

- For a minimal addition of mass and inertia, all elements of the mechanism including links, counter-masses, and counter-rotary counter-masses have to contribute to both the force balance and the moment balance of the mechanism for any motion of the mechanism. For existing mechanisms this applies to the additional elements while for the design of new mechanisms it applies to all elements. Mechanism links have to be positioned such that they counter-rotate with other links which means that dynamic balancing already starts within the kinematics of the unbalanced mechanism.
- For minimal addition of mass and inertia, the design space of the mechanism that is available for the position and the motion of balancing elements has to be maximal.
- For minimal addition of mass and inertia, the use of separate counter-rotations must be omitted.

Minimal addition of mass

- For minimal addition of mass, counter-masses and counter-rotary counter-masses have to be placed at maximum distance from their center of rotation.
- For minimal addition of mass, the balancing elements (counter-masses, counter-rotary counter-masses, counter-rotating links) have to be positioned such that they do not need to be balanced by other counter-masses.

Minimal addition of inertia

- For minimal addition of inertia, counter-masses and counter-rotary counter-masses have to be placed at minimum distance from their center of rotation.
- For minimal addition of inertia, counter-masses that are fixed to their link and elements that do not contribute to the moment balance must have minimal inertia.
- For minimal addition of inertia, geared counter-rotating elements themselves must have maximal inertia.
- For minimal addition of inertia, the gear- and transmission ratios of the counter-rotating elements have to be minimal.

Minimal addition of mass or inertia

- A trade off between the addition of mass and the addition of inertia must be made. The designer has to decide the relative importance of each for the intended purpose.

6 Conclusion

Based on the results of some comparative studies into several balancing principles and by taking into account the limitations of the design space, general guidelines for designing dynamically balanced mechanisms that have a low mass and a low inertia were formulated. Although a trade off between the addition of mass and the addition of inertia seems unavoidable, by optimization of the balancing parameters and the design space choices can be made judiciously.

References

1. Brown, G.W.: Suspension system for supporting and conveying equipment, such as a camera, patent number: US-4710819 (1987)
2. Chiou, S.-T., Tzou, J.-C.: On the balancers of any frequency term of the shaking moment of spatial mechanisms. Journal of Mechanical Engineering Science 211(part C), 37–48 (1997)
3. Ishida, K., Matsuda, T.: Performance characteristics and working comfortableness of forest workers of a new non-vibrating chain saw utilizing perfectly balanced rotation-reciprocation device. In: Proceedings of the Fifth World Congress of Theory of Machines and Mechanisms, ASME, pp. 951–954 (1979)
4. Kochev, I.S.: General theory of complete shaking moment balancing of planar linkages: A critical review. Mechanism and Machine Theory 35, 1501–1514 (2000)
5. Van der Linde, R.Q.: Design, analysis and control of a low power joint for walking robots, by phasic activation of mckibben muscles. IEEE Trans. Robotics and Automation 15(4), 599–604 (1999)
6. Lowen, G.G., Berkof, R.S.: Survey of investigations into the balancing of linkages. Journal of Mechanisms 3, 221–231 (1968)
7. Lowen, G.G., Tepper, F.R., Berkof, R.S.: Balancing of linkages - an update. Mechanism and Machine Theory 18(3), 213–220 (1983)
8. VDI2149: Blatt 1: Getriebedynamik-starrkörper mechanismen (dynamics of mechanisms-rigid body mechanisms). Verein Deutscher Ingenieure - Richtlinien (1999)
9. Verschuure, M., Demeulenaere, Swevers, J., De Schutter, J.: Counterweight balancing for vibration reduction of elastically mounted machine frames: A second-order cone programming approach. Journal of Mechanical Design 130(022302), 1–11 (2008)
10. Van der Wijk, V., Herder, J.L.: Double pendulum balanced by counter-rotary counter-masses as useful element for synthesis of dynamically balanced mechanisms. In: Proceedings of DETC 2008, ASME, DETC2008-49402 (2008)
11. Van der Wijk, V., Herder, J.L.: Dynamic balancing of mechanisms by using an actively driven counter-rotary counter-mass for low mass and low inertia. In: Proceedings of the Second International Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators, Montpellier, pp. 241–251 (2008)
12. Van der Wijk, V., Herder, J.L., Demeulenaere, B.: Comparison of various dynamic balancing principles regarding additional mass and additional inertia. Journal of Mechanisms and Robotics (accepted, 2009)
13. Van der Wijk, V., Herder, J.L., Demeulenaere, B., Gosselin, C.M.: Comparative analysis and optimization for low-mass and low-inertia dynamic balancing of a 1-DOF rotatable link balanced by a counter-rotary counter-mass. Journal of Mechanisms and Robotics (submitted) (2008)
14. Wu, Y., Gosselin, C.M.: Design of reactionless 3-dof and 6-dof parallel manipulators using parallelepiped mechanisms. IEEE Transactions on Robotics 21(5), 821–833 (2005)

Probability-Based Robot Search Paths

Jan Deiterding and Dominik Henrich

Abstract. The aim of this work is to show that robots can be improved using knowledge present or gained in previous executions of the task. Here, this knowledge is used to create search paths tailored to the problem at hand. We describe two approaches to create such paths automatically which enable robots to find an unknown position faster than standard paths. The solution presented here is not limited to the robotic domain, but can also be used for other purposes such as searching for injured persons after accidents. Our approach is evaluated through simulations and we show that these paths perform adequately well or even better than standard paths.

1 Introduction

Industrial robots are able to perform complex tasks without symptoms of fatigue, exhibiting highest precision and speed. However, these tasks are nearly always executed in a fixed environment; that is the precision is gained by ensuring that all objects are placed in exactly the same position every time. All parts need to have the same dimension, position, orientation, etc. Only by employing external sensors such as vision or force/torque sensors, we can enable a robot to deal with imprecisions and variations occurring in the objects and the environment. The price for this flexibility is that sensor based motions are slow compared to pre-computed motions. Especially when searching objects, the time required is significant.

In [4] we have classified changes that can occur between two executions of the same robot program by two characteristics: The origin of the change and the robot's reaction to it (Table 1). Here we differentiate four types of change: (a) An *indeterminacy* is something we are not aware of at this moment, but once we have learned about it, it will remain constant for a prolonged period of time. (b) *Variations* on the other hand occur every time the robot performs the task at hand. (c) *Faults and errors* happen when a sudden, unforeseen change in the workspace occurs. (d) A *drift* is similar to variations but not caused by the task itself, but by gradual changes within the workspace, e.g. the settings of machines and tools change over time.

Jan Deiterding · Dominik Henrich
Lehrstuhl für angewandte Informatik III, Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth, Germany,
e-mail: `{jan.deiterding,dominik.henrich}@uni-bayreuth.de`

In this paper, we are interested in ways of dealing with variations. A search motion must be performed to determine the variation in the current execution of the task. The central idea of this paper is to re-use knowledge gained in previous searches to create search paths tailored to the task and thus shorten the time span required for the search.

The rest of this paper is organized as follows: In the next section, we give a short overview of related work concerning this topic. Based on this, we describe how a search path can be optimized with regard to a given probability density. Additionally, we introduce a modification to our sorting algorithm to further optimize the path for probability densities with multiple maxima. Then we show the validity of our approach in simulations and compare the results with standard search paths in a two-dimensional environment.

Table 1 Classification of changes that can occur between multiple executions of the same program [4].

		Origin of change	
		Caused by the task	*Caused by abrasion*
Reaction to change	*One-step learning*	Indeterminacies	Faults and errors
	Continuous learning	Variations	Drifts

2 Related Work

The topic of robots performing some kind of search is very broad, so we will only refer to work dealing with searches in industrial environments and results which can be transferred to this domain.

A good overview is given in [8]. Despite the fact that sensor data processing has made significant progress allowing for relatively fast processing capabilities, standard search motions which only use minimal sensory information are still commonly used in industrial applications. A reason for this is that especially small and medium companies lack experts skilled in sensor data processing. Also of interest are the works of [10] and [6] which cover robot motion planning based on sensor data and probability densities.

If a search cannot be avoided, usually cameras are used that supervise the search area for the given variation. While this approach is straightforward and has the advantage that the localization can be made while the robot performs some other task, this is only applicable if the search area can be monitored at all. A typical example for a task where this is impossible is the assembly of a gear box in a car. Tolerances are extremely small and the search area is occluded by other parts of the vehicle so camera supervision is impossible and local sensors must be used.

Sharma [7] incorporates stochastic models into gross motion planning and defines a stochastic assembly process that yields increased performance.

An important area of research outside the industrial domain is *complete coverage paths* in mobile robotics. Here a robot must cover an area e.g. to search for injured people after an accident. A good overview of this area of research is given by [12]. Also of interest is [11], which uses a genetic algorithm approach and knowledge gained in previous executions to optimize the path of a mobile robot.

In summary, efficient search strategies are one of the central problems of robotics. While there are many specific solutions, e.g. [1] and [5], these are nearly always tailored towards specific tasks and the results can rarely be transferred to other areas. Here, we take a more general approach to search motions for industrial applications and outline the requirements for optimized search strategies. In industrial applications, the search area is precisely defined and does not change over multiple executions. The ideas presented in this paper are independent from the type of sensor used, the only requirement is that it provides a binary decision whether the goal of the search has been found or not. We only deal with the search itself, not with any actions that have to be taken by the robot afterwards, e.g., actually inserting a peg into a hole. Examples how this can be achieved using sensors are given in [2] and [3].

3 Search Paths Based on Probability Densities

In this section, we present the concept of a search path generated along a given probability density. The idea is that the robot stores successful positions from previous executions and creates a probability density from this knowledge. This can be achieved by employing the methods described in [9]. The path is not fixed and may change with every update of the probability density.

3.1 Robot Search Paths

A search is a motion that covers a specified area in order to locate an object whose exact position p_g is unknown. Exactly one object is searched for at a time. Here, we set the following preconditions:

1. The search area can be m-dimensional, but its boundary in every dimension must be a straight line. The area can be divided into a set S of cells describing discrete (hyper) cubes with fixed edge length Δc. When the robot moves to a cell, the whole area covered by that cube is probed. The decision whether p_g is found is binary, so there are no hints guiding us towards the goal. We assume that it takes a constant time span to check if the goal is located in a cell.
2. Any movement between two cells is allowed. There is no need for a neighbouring connection between two cells. No cell lying between the current and the next is tested when moving there.
3. A valid search path $P = (p_0,\dots,p_{|S|})$ must visit each cell of the whole area at least once. We include the possibility that the search fails: $p_g \notin S$. A search path is then an ordered sequence of all cells in S.

4. The distance between two cells $c_i, c_j \in S$ is relevant when planning the path. There is a positive cost function $d(c_i,c_j)$ describing the time and effort to move from cell c_i to c_j. Two neighbouring cells have unit distance.
5. There is a probability density φ describing the chance that the object lies within any given cell. This density may be continuous.

Additionally, a change of direction in the search path may slow down the motion in order to perform the turn along the path. We disregard this factor here.

Conditions 1 to 4 describe the general requirements imposed on a search path. Condition 5 is a new addition describing the knowledge we have about the location of the object that we are searching. This allows us to begin the search in the most probable cell and descend along the density instead of employing a predetermined path.

There are three criteria along which we compare different search paths to each other: Their total (maximum) length l and the expected number of cells visited $E_C(P)$ as well as the expected length of the path $E_L(P)$ for the given probability density $\varphi(p_i)$ and path P:

$$E_C(P) = \sum_{i=1}^{|P|} \varphi(p_i) \cdot i \qquad\qquad E_L(P) = \sum_{i=1}^{|P|} \varphi(p_i) \cdot d(p_{i-1}, p_i)$$

A developer faced with the task of designing a search path should consider two aspects. On one hand, it may be useful to limit the total length to its minimal value, so the path is not exceedingly long. On the other hand, if the (average) search time is crucial, it may make more sense to create a search path with higher total length but lower expected values. The decision, which of the two expected values is more important, depends on the type of search: If the movement between two cells is relatively fast compared to the time it takes to check a cell, the number of cells visited is significant. In case of slow motions, e.g. controlled movements along surfaces, the expected length is of more importance.

There are two de facto standard search paths for searches in two-dimensional environments: A zigzag path and a spiral path. Both paths can be easily extended to more than two dimensions. The zigzag path is usually chosen if the probability density is uniform, so there is no need to start in a specific cell. The spiral path is usually chosen when φ is unimodal, e.g. Gaussian, with mean in the middle of S. In this case the search starts in the most likely position and gradually descends along φ. Note, that both paths are optimal regarding their length; no cell is visited twice.

3.2 Optimizing Search Paths

Now, we are interested in finding search paths that optimize the expected values for a given probability density. A search path which is minimal in this sense will find p_g as soon as possible.

To create a search path P with lower expected values than a standard path for the given dimension of S, one has to approach cells with high probability first while neglecting cells with low probability until the end of the search. In case the expected length of the path is of importance, it should be attempted to minimize huge jumps across S as much as possible. The downside is that the distance of two consecutive cells in P now may be much higher than 1. So this search path may not be minimal with respect to the total length.

In a *sorting* strategy to generate an optimized search path, the cells of S are ordered like this: The beginning of the path is the most probable cell, so

$$p_0 = \{c_i \mid c_i \in S \wedge \forall c_j \in S : \varphi(c_i) \geq \varphi(c_j)\}$$

The remaining cells of the path are chosen by a recursive definition: We always choose the next cell according to its probability in relation to its distance d to the current cell, so

$$p_{k+1} = \{c_i \mid c_i \in S \setminus \{p_0,\dots,p_k\} \wedge \forall c_j \in S \setminus \{p_0,\dots,p_k\} : \frac{\varphi(c_i)}{d(p_k,c_i)^n} > \frac{\varphi(c_j)}{d(p_k,c_j)^n}\}$$

The impact of the distance when choosing the next cell is controlled by the exponent n, which must not be negative. The choice of this parameter depends on the type of application and must be chosen by the developer. The lower the value of n, the lesser the impact on the distance in the selection process. So cells with a high distance to the current cell may be selected as well. The higher n is, the more the selection process favours cells which lie close to the current cell. Note, that if n is set to zero, the cells are simply ordered along their respective probability. This will minimize the expected number of cells visited, but result in an extremely long total path, as it will cover great distances to move from one cell to the next (Figure 1, left). Vice versa, if n is set to infinity, the path always moves to neighbouring cells (Figure 1, right). Technically, we cannot get stuck in dead ends, because of Condition 2. But it is possible that we must move to a cell far away from the current one, because there are no neighbouring cells left. This strategy is not heuristic but always computes the best path for the given probability density and choice of n. It may be possible that more than one path exists with the same expected value.

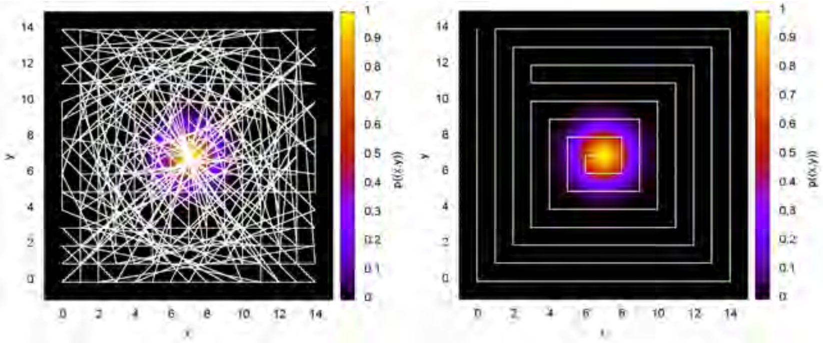


Fig. 1 Impact of the distance when sorting cells. Both paths start at the center. Left: The relative distance between the cells has no impact at all. Right: The relative distance between the cells is of infinite impact.

An example is shown in Figure 1 on the right for a Gaussian distribution. All spiral paths that start in the center will have the same expected value regardless of the fact which neighbouring cell is visited first. The strategy presented here only computes one of these paths. Which one this will be depends on the ordering of cells with the same probability and relative distance to the current cell.

3.3 Search Space Subdivision

This ordering works very well if φ has only one maximum or if the impact of the distance is chosen so that only cells close to the current cell are selected. But, if there are two or more maxima with a significant distance relative to each other (Figure 2, left), the path generated by this approach will oscillate between these maxima. Because of this, we extend our approach and introduce a *divide-and-conquer* strategy to split S into separate regions. We will repeat this process until the generated regions only contain one or zero maxima and then generate separate, optimized paths for each region with the sorting strategy. Finally, we connect all subpaths once more with the sorting strategy.

The algorithm in pseudo code looks like this:

```
(00) createSmartPath(bool top, list<path> paths) {
(01)        if(findMaxima() == 0) {
(02)               createZigZagPath();
(03)               addThisPathToList(paths);}
(04)        else if(findMaxima() == 1) {
(05)               orderCellsByProbability();
(06)               addThisPathToList(paths);}
(07)        else {
(08)               subspaces = splitSearchSpace();
(09)               foreach(subspace)
(10)                      createSmartPath(false, paths);}
(11)        if(top)
(12)               finalpath = orderPathsByProbability(paths)}
```

The function call in Line 0 takes two parameters: The Boolean parameter `top` describes if this is the topmost function call and the list `paths` is empty initially. Successive calls of this function will add subpaths to this list. In Lines 1 to 6, either a standard zigzag path or a probability ordered path is fitted into the given region, if there is none or only one maximum in that area. The result is added to `paths`. Otherwise the area is split into separate regions (Line 8). A simple approach is to perform a horizontal and a vertical cut through the geometric center of S. Other approaches would be to employ Voronoi diagrams between the maxima for a more fitting split. If the area is divided into separate regions, the same function is called to determine a path for this region (Lines 9 and 10). As a result, we have calculated a subpath for every region of the whole area. Now these paths are connected to each other by using the sorting strategy (Lines 11 and 12).

This extension yields the advantage that the path will not oscillate between two or more maxima, but remains in the immediate surrounding of one maximum. The downside is that – regardless of the value of n – there will always be a significant distance we have to bridge between two regions. This will worsen the total length of the path.

4 Experimental Results

In this section, we describe simulation results to show how optimized search paths compare to standard search paths for various probability densities. We have limited the simulations to a two-dimensional workspace. In this case, the paths are already complex compared to a one-dimensional search, but can still be visualized.

4.1 Simulation Setup

We have set up a two-dimensional squared workspace with an edge length of 15 cm. The position we are trying to find is a hole with a diameter of 1.5 cm. We have set the size of the cells in S to $\Delta c = 1\ \text{cm}^2$. This gives us 225 cells in the search area. So, there are 225! possible search paths, which is already too much for a brute-force computation.

We have created three probability densities:

1. A Gaussian density φ_g with mean $p_m = (7,7)$ cm in the middle of S and $\sigma = 1$ cm.
2. A mixed Gaussian density φ_m consisting of four maxima at $p_1 = (3, 3)$ cm, $p_2 = (11, 3)$ cm, $p_3 = (3, 11)$ cm, $p_4 = (11, 11)$ cm and $\sigma = 1$ cm each (Figure 2, left). A typical example for such a density is a peg-in-hole task on a square plate where the hole is not centered and the plate may be rotated by 90°, 180° or 270°.
3. An off-centered density φ_c with a maximum along the third quadrant in a circle around p_m with radius $r = 5$ cm (Figure 2, right). A typical example for such a density is a peg-in-hole task where the plate may be rotated by any value between 0° and 90°.

We have not used a uniform density, because no knowledge is present in such a density. To that effect, the expected value of all search paths is identical. The only difference will be in the total length. Because of that it is sufficient to use a standard search path.

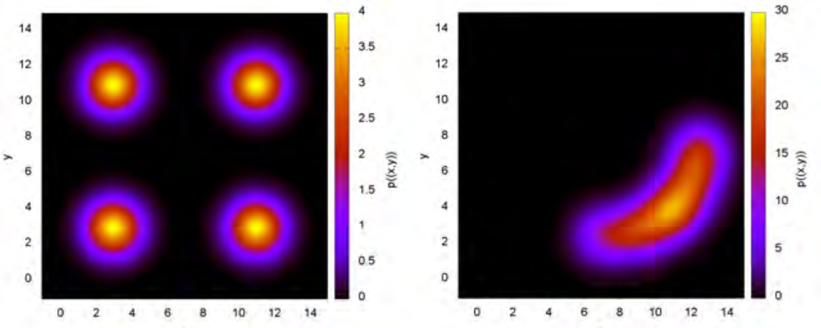


Fig. 2 Multi-modal probability density (left) and density where the maximum is along a quarter circle in the third quadrant of S (right) used in the simulations.

4.2 Comparison to a Standard Search Path

In two experiments, we used these three densities and generated search paths for each one. The first time, we only used the sorting strategy and the second time the divide-and-conquer strategy. For each density, we generated paths with varying values for n in the range [0; 10] with increments of 0.1. We then compared the generated paths to a standard spiral path starting in the center and measured the ratio by which the total length and the expected values differ from this standard path. Figure 1 shows two paths for the Gaussian probability density with a low and a high value for n. Since there is only one maximum present, both strategies generate the same paths. Figures 3 and 4 show generated paths for a value of $n = 5.6$ for the second and the third probability density.

We can see that the sorting strategy generates paths that follow the underlying density and prefer neighbouring cells. Nevertheless, these paths can lead to dead ends. In this case substantial jumps have to be made to reach unvisited cells. The

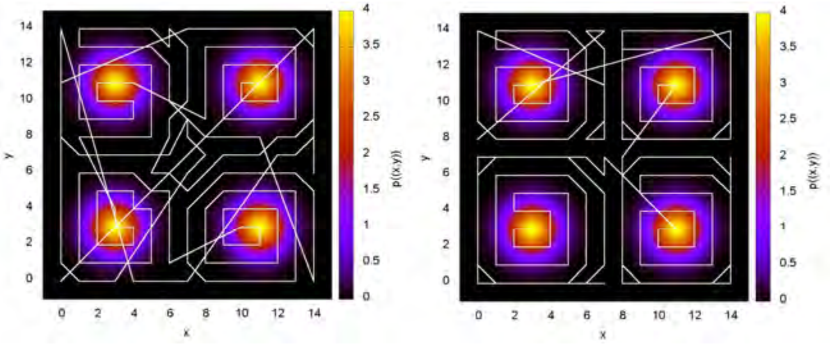


Fig. 3 Generated paths for the multi modal probability density outlined in Fig.2, left. Left: The path generated with the sorting strategy starts at (11,11). Right: The path generated with the divide-and-conquer strategy starts at (3,3).

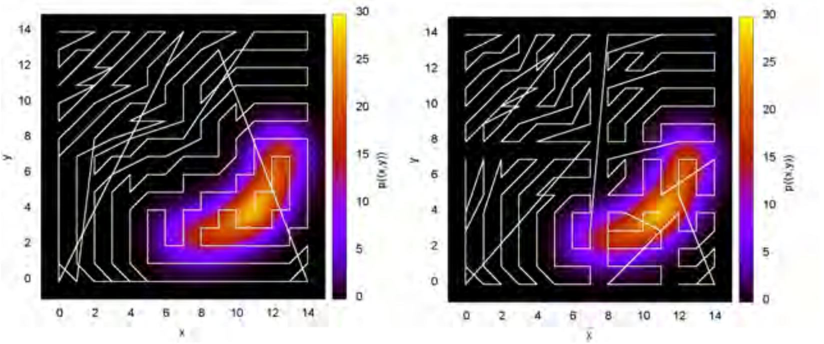


Fig. 4 Generated paths for the quarter circle probability density outlined in Fig. 2, right. Both paths start at (13,4). Left: Path generated with the sorting strategy. Right: Path generated with the divide-and-conquer strategy.

divide-and-conquer strategy reduces these jumps by separating S into distinct regions. Although here jumps may be necessary as well, these are limited to some extent.

Now we compare both strategies to a spiral path. Figure 5 shows the ratio by how much the generated paths relate to the spiral path for different values of n. We compare all paths by the expected number of cells visited before p_g is found, the length of this expected path and the total path length. All results are set into relation to a spiral path starting at the center of S. A value larger than one means, that the generated path performs superior to the spiral path. Vice versa, a value below one means that the path performs worse than the spiral path.

Once more, the results of the divide-and-conquer strategy for the Gaussian density are not shown, because this strategy generates the same paths as the sorting strategy. In all cases, the total path length (blue and brown lines) is worse than that of the spiral path, which is already optimal. But, with higher values of n, the paths draw near the optimal length. For low values of n, the divide-and-conquer strategy produces slightly better results, because in this case, all jumps are limited to the current region.

For low values of n, the sorting strategy generates paths that test significantly fewer cells than the spiral path (red line). But, because these cells are far apart, the expected length covered (green line) can be worse than that of the spiral path.

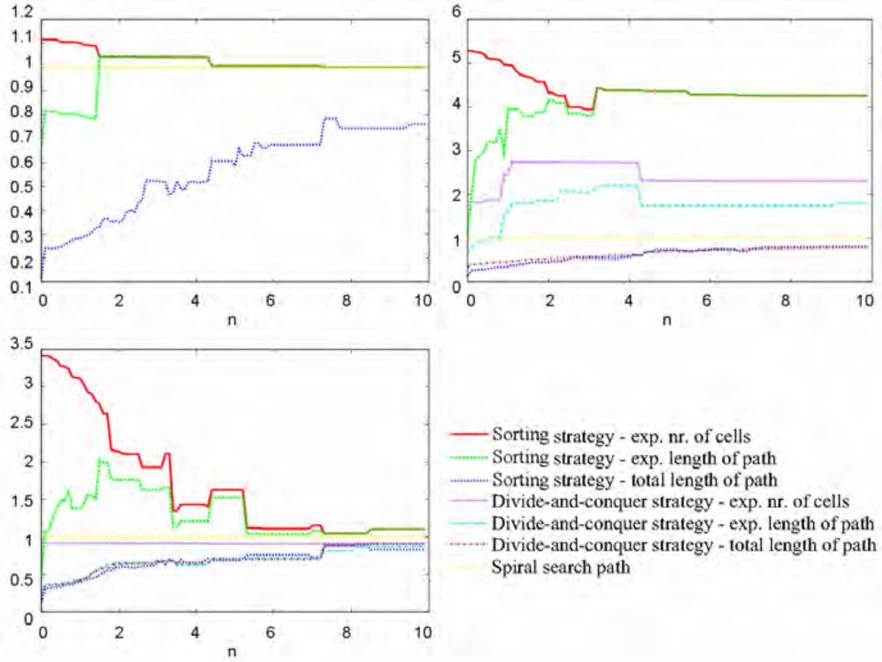


Fig. 5 Comparison of the generated paths to a standard spiral path. Top left: For a Gaussian density φ_g. Bottom left: For a mixed Gaussian density φ_m. Top right: For an off-centered density φ_c.

With higher values for *n*, the two lines converge, because now neighbouring cells are preferred, similar to the spiral path. The higher *n* gets, the lower the advantage of the sorting strategy, because of the dominance of neighbouring cells in the selection process.

The divide-and-conquer strategy showed no significant improvement in comparison to the sorting strategy. In case of the off-centered density, the expected number of cells and the expected length of the path (purple and turquoise lines) are higher than the spiral path but lower than the sorting strategy. The overall path length is only slightly better than the sorting strategy and only for low values of *n*. In case of the mixed Gaussian density, the expected number of cells and the expected path length are even worse than for the spiral path, because the splitting algorithm cuts the area and only moves to the next subsection, when the current subsection is completely covered. Because there are many cells with very low probability in every subsection, this lowers the expected values significantly. It may be possible to achieve better results with a more sophisticated splitting algorithm, but this exceeds the scope of this paper.

4.3 Random Probability Distributions

In the next step, we have generated random probability densities in order to evaluate our strategies for a broader set of probability densities. We have created these densities by randomly placing *k* Gaussians uniformly in the search area. For every value of *k*, we have created 100 different probability densities. Then, the sorting strategy was applied to each density with various settings for *n*. We compared the generated paths to the standard spiral path and computed the average for each combination of *k* and *n*. The results are shown in Figure 6.

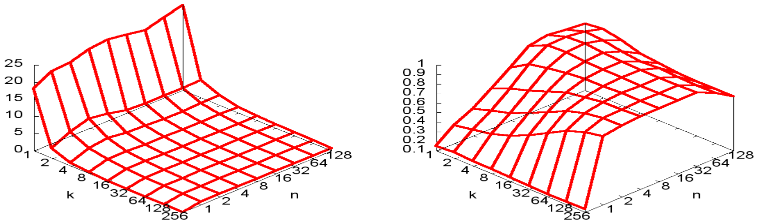


Fig. 6 Expected length of path (left) and total length of path (right) in relation to a standard spiral path for a varying values of *k* and *n*.

We can see that the generated paths perform better than the spiral path for low values of *k* regardless of the choice of *n* in terms of the expected length of the path (Figure 6, left). The more Gaussians are combined the more the overall probability density converges to a uniform density. In this case neither the optimized paths nor the spiral paths are superior because there is no information present in the probability density at all. When we take a look at the overall length of the search path (Figure 6, right), once more we can see that paths generated with a low impact of the relative distance between two cells are significantly longer than the spiral path. With increasing *n*, the optimized paths are nearly as short as the spiral path. There are two noteworthy aspects: The ratio increases faster for high values of *k*. This is because the Gaussians lie closer to each other. But for high values of *k* and *n*, the ratio decreases. This is because now there are so many Gaussians in the overall density that the path tends to get stuck in corners and large jumps have to be made to approach the next free cell increasing the total length.

5 Conclusion

The aim of this work is to show that search paths based on probability densities are capable of locating the position in question faster than standard search paths. The central idea is to search in areas with high probability of success first in order to maximize the expected value.

We have described the general requirements for path planning and three ways to rate search paths. While standard paths are optimal with respect to the total length, optimized paths can improve these in terms of the average time the search takes. We have shown in simulations that, for Gaussian probability densities, the optimized paths perform almost as well as standard paths and better if there is more information present about the search area. The strategies presented in this paper are no heuristics, but always compute the best path for a given probability density and choice of the impact of the distance between two consecutive cells.

The advantage of our approach is that standard search paths can be seen as special solutions to the more general approach taken here. The algorithms to create optimized paths can be incorporated into the programming environment and no additional knowledge is required by the developer. The update of the probability density describing the search area and the path planner itself can be completely hidden from the developer.

The next step is to test reasonable splitting algorithms for the divide-and-conquer strategy to further improve the expected values. Various approaches to subdivide the search area, such as Voronoi diagrams, can be taken and compared to each other. So far, we have separated the search area into hypercubes. An interesting approach may be to use a hexagonal grid, providing us with more neighbours for a local path planner.

References

1. Cheng, P., Cappelleri, D., Gavrea, B., Kumar, V.: Planning and Control of Meso-scale Manipulation Tasks with Uncertainties. Robotics: Science and Systems (2007)
2. Chhatpar, S.R., Branicky, M.S.: Search Strategies for Peg-in-Hole Assemblies with Position Uncertainty (2001), `http://dora.cwru.edu/msb/pubs/iros2001b.pdf`
3. Chhatpar, S.R.: Localization for Robotic Assemblies with Position Uncertainty (2005)

4. Deiterding, J., Henrich, D.: Automatic optimization of adaptive robot manipulators. In: 2007 IEEE International Conference on Intelligent Robots and Systems, San Diego/USA (2007)
5. Gage, D.W.: Randomized search strategies with imperfect sensors. In: Proceedings of SPIE Mobile Robots VIII, pp. 270–279 (1993)
6. Khatib, M.: Sensor-based motion control for mobile robots. Maher Khatib Sensor-based motion control for mobile robots PHD thesis,LAAS-CNRS (December 1996)
7. Sharma, R., LaValle, S., Hutchinson, S.: Optimizing robot motion strategies for assembly with stochastic models. IEEE Trans. on Robotics and Automation 12(2), 160–174 (1996)
8. Siciliano, B., Khatib, O. (eds.): Springer Handbook of Robotics. Springer, Heidelberg (2008)
9. Parzen, E.: On estimation of a probability density function and mode. Ann. Math. Stat. 33, 1065–1076 (1962)
10. Tenenbein, A., Weldon, J.-L.: Probability Distributions and Search Schemes. Information Storage and Retrieval 10(7-8), 237–242 (1974)
11. Hu, Y., Yang Simon, X.: A knowledge based genetic algorithm for path planning of a mobile robot. In: 2004 IEEE International Conference on Robotics and Automation, pp. 4350–4355 (2004)
12. Zelinsky, A., Jarvis, R.A., Byrne, J.C., Yuta, S.: Planning Paths of Complete Coverage of an Unstructured Environment by a Mobile Robot. In: Proceedings of International Conference on Advanced Robotics, Tokyo Japan (November 1993)

Mapping and Navigation of Mobile Robots in Natural Environments

Jürgen Roßmann, Petra Krahwinkler, and Arno Bücken

Abstract. With support of the North-Rhine Westphalian Ministry of Environment algorithms are being developed to construct "tree maps" by delineating single trees from remote sensing data like LIDAR, ortho photos and satellite imagery. These tree maps are in turn used as a basis for a new approach to forest working vehicle localization and navigation to overcome GPS limitations under a forest canopy. Experiments already showed that the availability of the tree map in connection with statistical sensor fusion strategies greatly improves the accuracy for navigation of autonomous vehicles in the forest.

1 Forest Navigation

Navigation is not only needed "in the streets and on other planets" but also as a part of new logistic approaches in forestry applications. Modern work machines like wood harvesters can automatically cut trees, remove branches and cut the logs into stems with the desired lengths but an expert is still needed to plan a thinning project and to mark the trees to fell by hand. The harvesting and thinning processes can be made more efficient if the forest ranger can employ IT to mark the trees using geographic coordinates instead of colored crosses sprayed on the stems. In order to realize a new approach to tree marking based on augmented reality the forest machines have to be able to locate itself accurately. The new localization techniques also greatly support automatic navigation to the next tree to fell. Signal absorption in the forest canopy leads to poor results from GPS sensors in a forest with errors up to 50m and more. Furthermore the canopy leads to intermitant problems where not enough signals can be picked up for several seconds. Another problem is that even if a detailed terrain model is available, other factors as understory and ground humidity influence the movements of an autonomous robot in the forest. Commonly, wheel spin and slipping further induce errors into a position estimation based on forward projection and odometry.

Jürgen Roßmann · Petra Krahwinkler · Arno Bücken
Institute of Man-Machine-Interaction, RWTH Aachen,
Ahornstraße 55, 52074 Aachen, Germany,
e-mail: `{rossmann,krahwinkler,buecken}@mmi.rwth-aachen.de`

Other approaches as described in [7] and [8] use differential GPS (DGPS) sensors as their main source of information on the position. In [8] a DGPS sensor is combined with laser scanners and a Kalman Filter approach. However our experiments also with a high precision DGPS sensor showed that also these sensors do not provide the capability – and accuracy – to position the working machine with respect to a single tree. Also high-precision DGPS sensors suffer from signal loss under the canopy as they also require strong satellite signals to calculate a reliable position.

2 Map Building

Current car navigation systems use their knowledge – the map-information – to correct the position estimation determined by a GPS receiver. If the GPS position fix is within a certain range around a street, it will be moved towards the closest point on the "virtual street" and displayed at this position. In order to use a similar strategy in the forest we first have to think about a map representation of the trees and a semi-automatic way to generate an appropriate map.

2.1 Data Acquisition

All examples in this paper are based on data recorded in summer 2007 using a Riegl LMS-Q560i scanner. The data shows about 450km² in the area "Schmallenberg" which is located close to Winterberg, Germany. The resolution of the LIDAR data was specified to be between 4 and 6 points per square-meter in general and 12 points in some selected areas. The Riegl scanner delivers full waveform data which was processed into one first echo point and one last echo point per ray. The point-cloud of the first echoes was directly transferred into a grid-based representation with a cell-size of 0.4m x 0.4m, the so-called Digital Surface Model (DSM), while the last echo data was first (semiautomatically) filtered in order to eliminate plants, buildings and other disturbances before it was also converted into a grid-based representation. Due to impenetrability of a dense canopy or of buildings the resulting Digital Terrain Model (DTM) shows gaps. To fill the gaps it was interpolated which delivers a representation called a Filled Digital Terrain Model (FDTM). In a last step a Differential Model (DM, also known as normalized digital surface model nDSM or canopy height model CHM) was calculated as the difference between DSM and FDTM. The DM gives the height of the crown surface of a tree above the ground. This is the most important LIDAR data set for single tree delineation.

2.2 Data Processing

The LIDAR DM can be seen in two ways: As a three-dimensional model of the canopy or as a two-dimensional height-map (Figure 1). The latter view facilitates the use of a watershed algorithm for single tree delineation [1]. With a standard watershed-algorithm the z-axis of the three dimensional data is only used to generate gradients and calculate affiliations, resulting in a set of areas, each annotated with its size. So the size of the region would be the only criterion to decide whether a region represents a tree or only a branch of a tree.

In contrast to this, the use of full 3D-information allows to consider the volume of a peak pointing out of the canopy – instead of only the area of a cutting surface. We decided to better approximate the peak-volumes by extending the watershed-algorithm to work on three dimensional data. For the illustration of this advanced approach, we will use a sectional drawing through a three dimensional DM. Figure 2a shows some trees and the sectional drawing above them. To make it easier to imagine rainfall and water-flow the sectional drawing is turned upside down in the subsequent images with the most significant points – the maximum heights in the original data that may represent tree-tops – as local minima of the graph. Figure 2b illustrates the idea of a standard watershed algorithm. Water is poured over the area uniformly. The water-flow is simulated and the amount of arriving water is measured at all local minima.

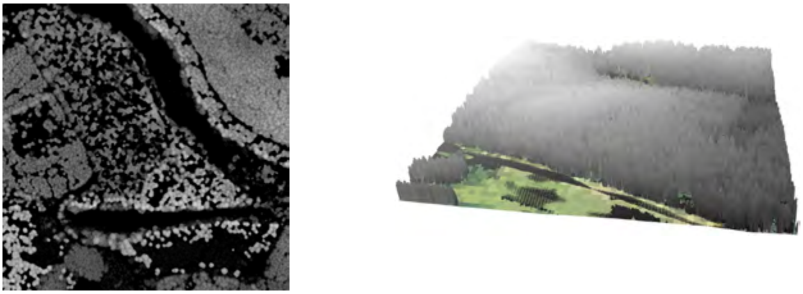


Fig. 1 The Source Data: Differential Model in Height Map-Representation and in 3D - Visualisation.

To get the volumetric information, we “fill” the DM with water. Then, in each cycle, we puncture the point having the “highest water-pressure” acting on it and measure the amount of water streaming out of this opening. (Figure 2c) The result is a value which is always higher or equal to the real volume of the peak. The interesting feature is that the result is far away from the real volume for the most extreme points (for the most likely treetops) but very close to the real volume for the critical peaks that are hard to decide. This way, the tallest trees in a surrounding are preferred during the generation - which is usually the best solution.

For each opening that receives a volume higher than a user-specified threshold, a tree is generated in the map using the x and y position of the puncture - which is always the highest point within its peak –, the z value taken from the FDTM. The tree is annotated with its height which can be derived from the DM. Figure 2d shows a situation where only one peak is left. The remaining volume is below the threshold so there will be no additional tree generated at this position.

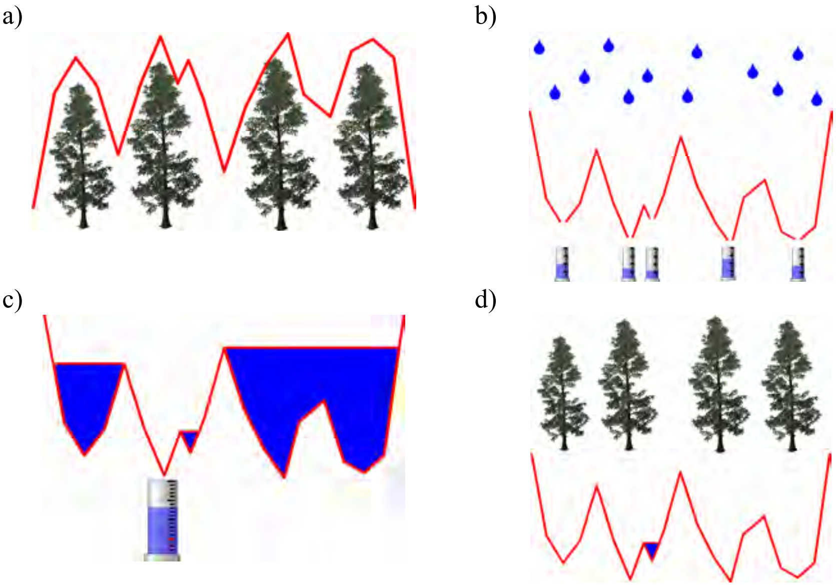


Fig. 2 Single Tree Detection in LIDAR data. a) Trees and LIDAR Information, b) Watershed-Algorithm, c) Volumetric Algorithm – First Cycle, d) Last Cycle.

In contrast to the known 2D approach, the volumetric approach adds another dimension to the data used for calculation and makes it easier to decide whether a peak is a tree or just a branch of a tree. This is especially valuable for the available data because the z-axis of the grid-based DM-model has a resolution of 1cm compared to the 0.4m-resolution of the x- and y-axis.

In areas where no LIDAR data is available, image data can be used to extract tree positions. A LIDAR DM and a greyscale photo are quite similar, the main difference is the fact that treetops in the spectral image may have the same brightness but represent different heights. This lack is not important when we generate a map of tree-positions which is enough for localisation purposes. It turns out that the volumetric algorithm performs well on LIDAR and spectral data.

By using the volumetric algorithm, the percentage of correctly detected treetops could be increased significantly compared to the watershed-algorithm.

2.3 Data Processing

The described algorithms were implemented in the Verosim 4D -GIS. The volumetric algorithm only needs one threshold so it can easily be controlled interactively and adjusted to different forest conditions. The threshold value for this algorithm depends on age and density of the forestry unit. The result of changing the threshold is displayed in real-time to help the user finding the correct value for each forest unit.

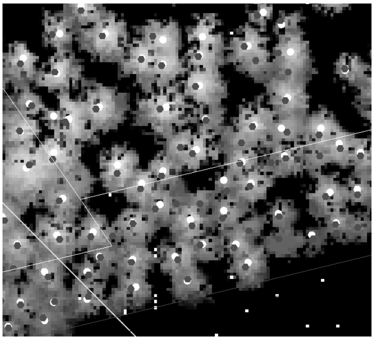


Fig. 3 Results of the volumetric Algorithm in comparison to trees measured in the field.

To test the precision and reliability of the volumetric algorithm the position of all trees at a test-site were collected in the field. Figure 4 shows these ground-truth trees in red while the automatically generated trees are marked in blue (spruce) and brown (beech). The algorithm proves to deliver results up to a detection rate of 95% in coniferous units which are ready to harvest. A comparison between the volumetric algorithm and the 2d watershed can be found in [4].

The volumetric approach generates **a database of single trees** calculated from remote sensing data. Along with the trees and their geo-coordinates the height and the diameter at breast-height are determined. This data can be used to generate a tree-map with a mean error between 0.5 and 1.5m which is still below the mean tree distance of about 2.5m.

3 Localization

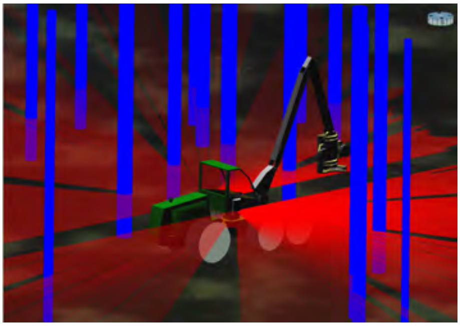
In order to test the algorithms described below, a virtual model was created based on the calculated tree map. It contained the forest as well as a wood harvester, which was operated as an autonomous robot. Furthermore the harvester was equipped with virtual laser scanners to retrieve the required information about the surrounding.

The first tests were conducted in a virtual environment. In order to take the evaluation of the algorithm a step further a data logging part was added to the algorithm. It allows recording sensor data on a real machine and provide it for playback.

3.1 "Visual GPS"

Based on the tree map, which was generated from remote sensing data, and the tree group, which was detected by one or more laser scanners, a matching algorithm can be run.

Fig. 4 Laser scanner beams and extracted tree group.



Therefore a particle filter algorithm as described in [5] was implemented. A single tree as a landmark cannot be associated with its corresponding tree in the map, however patterns of tree positions can be matched. As an initial guess the position of an inaccurate GPS sensor is used and the particles are distributed in an area with the GPS position at its center. As shown in figure 2, a square area was chosen to guarantee even particle distribution and short calculation time. Each particle represents a hypothesis for the position of the vehicle and is tested for its probability to represent the searched position.

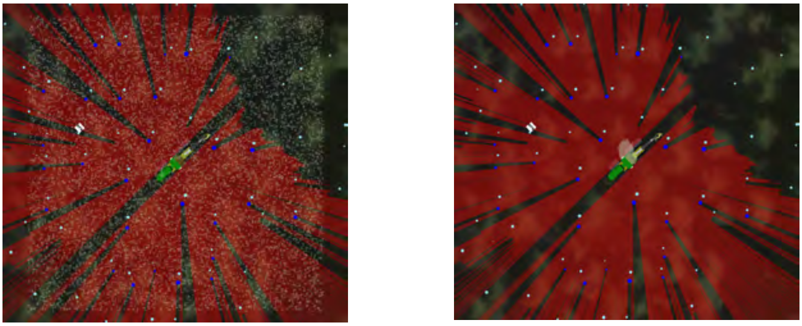


Fig. 5 Particle initialization and Particle concentration after resampling.

As not only the location of a vehicle in the forest is unknown but also its orientation, the heading has to be estimated along with the position.. Therefore the probability measure used in the propagation step was enhanced. Instead of embracing only the distances of the trees, their relative position is used considering the heading φ_t of the current particle:

$$w_t^{[m]} \propto \sum_{tree\,group} \min_{tree\,candidates} \left| \begin{array}{l} \left|(x_{tree,rot} - x_R)\right| - \left|(x_{candidate} - x_t)\right| + \\ \left|(y_{tree,rot} - y_R)\right| - \left|(y_{candidate} - y_t)\right| \end{array} \right|$$

$$\begin{aligned} with \quad & x_{tree,rot} = x_{tree} \cdot \cos(\varphi_t) + y_{tree} \cdot \sin(\varphi_t) \\ and \quad & y_{tree,rot} = -x_{tree} \cdot \sin(\varphi_t) + y_{tree} \cdot \cos(\varphi_t) \end{aligned}$$

The rotation variant approach directly calculates the heading of the vehicle but the sensibility towards rotation, which results from the new probability measure, leads to a higher number of particles which have to be used during the initialization step.

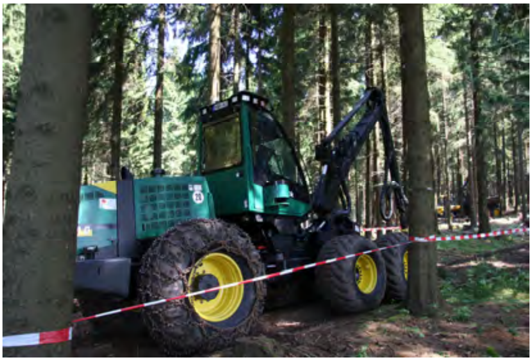


Fig. 6 The modified wood-harvester.

3.1.1 Local Search

To overcome the high calculation time the number of particles has to be reduced. Therefore the initial position is estimated by using an ordinary GPS sensor or the last known position. Although the GPS measurement is faulty in the forest it can limit the search to a restricted area. Machines most often start at the edge of a forest stand, at a forest road or an opening. At these spots the canopy usually is transparent and GPS sensors work with higher precision. Therefore they provide a good initialization for the algorithm.

In the following steps the previous position can be used instead of or in combination with the output of the GPS sensor for determining the search area. The previous position provides a better initial pose estimation than the GPS sensor and therefore gives the opportunity to further decrease the search area. On the contrary, the GPS position provides important information during the startup phase and in can be used to solve the “kidnapped robot”-problem.

To reduce the number of trees for which the distance has to be calculated, trees with a distance from the initial pose estimation $\mathbf{x}_{GPS}$ below the sum of the estimation of the maximal position error and the maximal distance of the trees in the scanned tree group from the reference position are extracted from the tree map.

$$\left|\mathbf{x}_{candidate} - \mathbf{x}_{GPS}\right| < r_{error} + \max_{tree\,group} \left|\mathbf{x}_{tree} - \mathbf{x}_{R}\right|$$

4 Results

The simple criterion presented here proved to be reliable in the vast majority of cases. Problems can occur when the tree group contains trees, which are not part of the tree map (“false positive”). This can happen due to missing trees in the tree map or faulty tree cognition in the local laser scanner measurement. In the first case the understory might not have been detected in the airborne laser scanner data. In the second case other objects like the harvesters aggregate might have been mistaken to be a tree or the laser beams might have hit the ground and a tree is detected in the irregular measurement due to the structure of the ground. However the used algorithm to identify trees in the laser scanner measurements was adapted to prevent the detection of false positives.

The case of trees not detected in the local laser scanner measurements but contained in the tree map ("false negative") does not lead to problems in the pose estimation step. The reason for this is that the algorithm searches for a corresponding tree for each unit in the tree group. For a false positive no corresponding tree can be found whereas a false negative is simply not considered. However if the size of the tree group is too small the estimation errors grow. The minimum number of trees depends on the search area radius. At a size below 15 trees the number of faulty position increases rapidly as more similar patterns can be found.

To evaluate the accuracy of the system on a real wood harvester a surveyors office was instructed to measure the vehicles position at 7 distinct locations. At each position the sensor input data was written to file for several seconds. This data was evaluated and for each location more than 45 pose estimations were calculated. The differences of the calculated and the measured positions are shown in figure 7

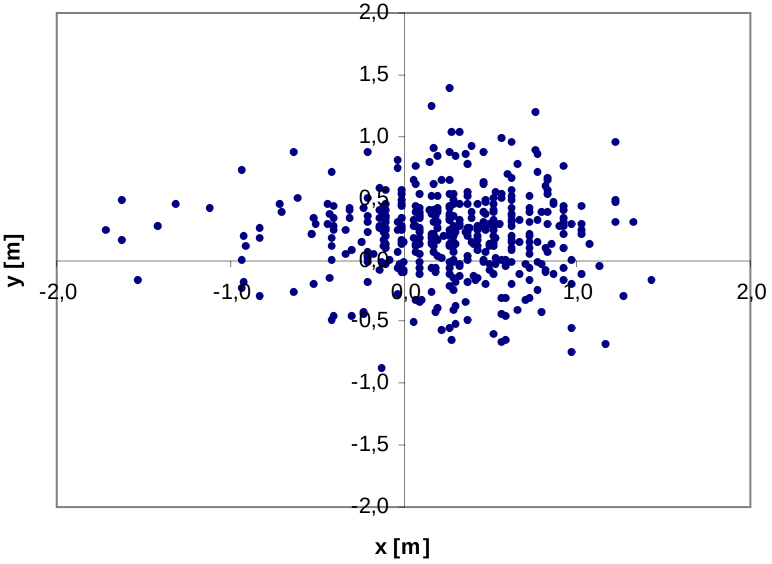


Fig. 7 The modified wood-harvester.

The mean error amounts to approximately 0.55m.

Furthermore the algorithm was tested in action on non-flat surface. Although the laser scsanner beams hit the ground several times, the algorithm worked reliably as long as enough trees could be found and matched.

5 Conclusion

The presented algorithm – as a method which combines a new single tree delineation approach with advanced robot localization and sensor fusion techniques – proved to be an excellent basis for the development of forest work machines into "autonomously working robot". Practical tests – e.g. of the localization features in preparation of the KWF fair 2008 – were very successful and proved that the desired accuracy to navigate to single trees in the map can be achieved. Furthermore, the applications showed that the required sensors and the computational power can be integrated into standard forest work machines in a professional manner without "voiding the guarantees".

Currently, the sensor fusion idea of the algorithm is taken further to be able to update the map based on the data generated by the machines at work. The ideas are somewhat related to known SLAM approaches, again combining forest specific knowledge with autonomous robot know how.

References

1. Diedershagen, O., Koch, B., Weinacker, H., Schütt, C.: Combining LIDAR- and GIS-Data for the Extraction of Forest Inventory Parameters (2003)
2. Rossmann, J., Schluse, M., Bücken, A., Jung, T., Krahwinkler, P.: Der Virtuelle Wald in NRW. AFZ Der Wald – Allgemeine Forst Zeitschrift für Wald und Forstwirtschaft 18, 966–971 (2007)
3. Rossmann, J., Schluse, M., Bücken, A., Krahwinkler, P.: Using Airborne Laser-Scanner-Data in Forestry Management: a Novel Approach to Single Tree Delineation. In: Proceedings of the ISPRS Workshop Laser Scanning 2007, and SilviLaser 2007, Espoo, vol. XXXVI, Part 3, pp. 350–354 (2007)
4. Rossmann, J., Bücken, A.: Using 3D-Laser-Scanners and Image Recognition for Volume-Based Single-Tree-Delineation and Parameterization for 3D-GIS-Applications, 3d-Geoinfo 07, Delft (2007)
5. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics, Massachusetts Institute of Technology, Cambridge (2005)
6. Jung, T., Rossmann, J.: Realisierung von Simulatoren für Forstmaschinen für den Einsatz in der Maschinenführerausbildung mit einem universellen 3D-Simulations system, 10. IFF Wissenschaftstage (2007)
7. Georgsson, F., Hellström, T., Johansson, T., Prorok, K., Ringdahl, O., Sandström, U.: Development of an Autonomous Path Tracking Forest Machine – a status report. In: Springer, field and service robotics: results of the 5th international conference edition, pp. 603–614. Springer, Heidelberg (2006)

8. Miettinen, M., Öhman, M., Visala, A., Forsman, P.: Simultaneous Localization and Mapping for Forest Harvesters. In: IEEE International Conference on Robotics and Automation, Roma, pp. 517–522 (2007)
9. Jutila, J., Kannas, K., Visala, A.: Tree Measurement in Forest by 2D Laser Scanning. In: Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, Jacksonville, pp. 491–496 (2007)
10. Rossmann, J., Krahwinkler, P., Bücken, A.: Arbeitsmaschinen als autonome Roboter im Forst: Virtuelle Prototypen, Verfahren und Anwendungen, 7. In: Paderborner Workshop, Augmented & Virtual Reality in der Produktentstehung, Paderborn (2008)

Sensor-Based Online Planning of Time-Optimized Paths in Dynamic Environments

Thorsten Gecks and Dominik Henrich

Abstract. Dynamic environments, in which robots and for example humans share a common workspace impose a variety of requirements on path planning algorithms, including real-time capabilities and collision tests based on sensor input. We propose a randomized-roadmap-based path planning algorithm that limits the number of collision tests and distance calculations to a volume achievable in real-time, while still being able to achieve high path clearance and statistical completeness given an unlimited number of planning cycles. It does so by exploiting the knowledge about static portions of the environment using a static, collision-checked roadmap and by interleaving planning and execution. Image-based distance measurements are induced by the graph search algorithm and interpolated to allow planning of time-optimized paths on the roadmap with a reduced number of distance measurements.

1 Introduction

Robots are emerging from their stereotyped role as dumb, mechanical tools in separated environments and becoming user-friendly, cooperative workmates. Thus, sensible reactions to dynamic environments are becoming a predominant issue. Safety and intuitive usability during user interaction are important aspects, as well as efficiency and availability on the process side. A building block for these kinds of robots is the ability to detect and cope with objects in the robot's workspace in a reactive fashion.

Planning in the presence of dynamic obstacles is a challenging problem, especially if these obstacles are unknown in advance, meaning that the information about their location and geometry has to be acquired by sensors. For robots with a high degree of freedom, it is hard to explicitly translate workspace obstacles into configuration space in order to reconstruct the free/occupied configuration space as a whole. In consequence, collision tests or distance measurements to obstacles are time-consuming for each robot configuration.

Thorsten Gecks · Dominik Henrich
Lehrstuhl für angewandte Informatik III, Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth, Germany,
e-mail: `{thorsten.gecks,dominik.henrich}@uni-bayreuth.de`

The demand for a high-clearance planning method is generated from two sides: a psychological component and a safety component. Humans do not feel comfortable when faced with a speedy, heavy-duty robot arm in close proximity. This becomes even more relevant, when the worker is performing a different task requiring his full attention and, due to distraction, no immediate reaction to erroneous robot movements is possible. The safety component requires the robot to reduce its speed in the vicinity of dynamic obstacles so it could react safely to unforeseen obstacle movements within the remaining time to collision. This distance-based robot speed regulation generally favors high-clearance paths for time-optimized path planning.

2 State of the Art

Distance-optimized or maximum-clearance path planning has long been a field of research that has produced a variety of methods. Principally, these methods explore a local portion or the global, complete workspace or configuration space of the robot, computing a path from the start to the goal, while maintaining a great distance to all obstacles all along its way.

The most reactive and real-time capable algorithms are the family of potential field methods [7]. Simulated potential fields allow the calculation of repulsive forces that can be translated into robot movements. While several extensions exist, this method is generally susceptible to local minima that prevent the robot from reaching the goal. Those extensions commonly rely on a specific modelling and/or a complete knowledge of obstacles in the robot configuration space [1].

Another family of methods relies on geometric information about obstacles (either in workspace or configuration space), that is easy to decompose in that it relies on simple geometric primitives (i.e. B-Reps with planar surfaces). Based on a decomposition of the free space, it is possible by searching the connectivity graph to find a path from the start to the goal with maximized clearance [11]. This technique can be combined with the previously mentioned potential fields if each decomposed cell contains a local potential field that repels the robot away from the cell boundaries, while still allowing it to react to dynamic obstacles [8].

A combination of artificial forces and roadmap-based planning was presented in [2, 10]. The elastic strip method initially uses an arbitrary planner to produce an optimal path from the start to the goal. Upon execution, the path is deformed by any moving dynamic obstacles due to artificial repulsive forces associated with those obstacles. While this approach is real-time capable and complete, the path can become highly inefficient if obstacles deform it from the initial state to such an extent that re-planning becomes necessary. Also, paths that were originally obstructed can become viable if obstacles move, but are not considered after the initial path planning.

For the class of probabilistic roadmap planners, several algorithms have been proposed to either sample configurations on the medial axis of the free space [6] or to retract the path to the medial axis in a post-processing step after planning through a standard probabilistic roadmap [3]. Both methods rely on detailed knowledge of the obstacle surfaces in the planning space.

As we have seen from this section, the approaches presented so far either provide real time reaction to dynamic obstacles but can become caught in local minima (and thus are incomplete) or rely on the exact knowledge of the configuration space or the execution of a high number of collision tests, which is impractical for robots with a high degree of freedom. In the following section we will outline a path planning algorithm that is able to provide real time path planning while generating high-clearance paths in dynamic environments.

3 Real-Time Shortest-Path Planning Algorithm

In this section, we will give a short overview of the basic techniques that we use as a basis for path planning in dynamic environments. Section 4 extends these techniques towards time-optimized path planning. For further details on the following, see [4].

The basic algorithm described below is real-time-capable under the assumption, that the most computationally intensive part of planning is the collision tests (or distance calculation) and not the graph search. In our case, *each* collision test takes 1…5 ms to compute, while the graph search without collision tests on a reasonably-sized roadmap has an average runtime of well below 10 ms. To separate graph searching and collision tests, the graph search algorithm rests upon an abstract layer, which provides the edge costs determined by the collision test. The number of collision tests/distance calculations used is then adaptable and transparent to the graph search algorithm. Thus, the overall runtime can be tailored to the real-time demands imposed of the application.

The proposed path planning algorithm is based on a static randomized roadmap $G(V, E)$, consisting of a set of vertices V and a set of edges E. Edges and vertices are tested in an offline setup step for collision with a known static environment. Online tests refer to the unknown, dynamic objects in the environment, which are detected by sensors, such as multiple surveillance cameras. As these online tests are relatively expensive, their number is reduced by several techniques such as lazy collision checking and interleaving of planning and execution. The following pseudo code outlines the major components.

In the function *planPath* the start (current) v_{curr} and the goal robot configuration v_g are connected to the roadmap G via *connect*(v), which constructs a new vertex v, if it does not already exist. It then adds collision-tested connections from the new vertex to its k neighbors. As long as the target is not reached, planning *searchShortestPath*(v_g, v_{curr}) and execution phases *executePath*(P) are interleaved. The function *searchShortestPath*(…) finds the shortest path from v_{curr} to v_g via the roadmap G using the graph searching algorithm A*, if it exists. If no path is returned by *searchShortestPath*(…) and the goal does not collide with unknown obstacles (detected by *collision*($O_{unknown}$, v_{curr})), the function *addVertex*(V) samples a new node (uniform or non-uniform, see [3]) and connects it to theroadmap.

```
planPath($v_{curr}, v_g$)
    connect($v_{curr}$)
    connect($v_g$)
    while($v_{curr} \neq v_g$)
        $P$ = searchShortestPath($v_{curr}, v_g$)
        if ($P \neq \{\}$)
            executePath($P$)
        else if (not collision($O_{unknown}, v_g$))
            addVertex()

executePath($P$)
    while($v_{curr} \neq v_g$)
        if (testPath($O_{unknown}, v_{curr}, P$)
            setInvalid($e_{curr}$)
            connect($G, v_{curr}$)
            return
        else
            $v_{curr}$ = driveRobot($P, \Delta t$)
```

A valid path is executed in the function *executePath*(P). The function *testPath* ($O_{unknown}$, v_{curr}, P) tests the given path P from the current robot position v_{curr} towards the goal. The depth of the test only depends on the available computing power and the given collision test costs. If a collision occurs, the current edge e_{curr} is invalidated and omitted in the next path planning phase. Non-invalidated edges thus form a dynamic subgraph G_{valid}, which incrementally adapts to the current obstacle situation as path execution proceeds. Invalidated edges are revalidated using various methods described and evaluated in [4]. As a consequence of collision checking before execution, the algorithm is guaranteed to find a collision-free path to the goal and because of the randomized roadmap approach it is statistically complete.

4 Time-Optimized Path Planning

In the context of human-robot cooperation and in general when dynamic obstacles are involved, robot speed must be adapted to the current obstacle distance, thus being able to react to movements of the unknown objects before a collision becomes unavoidable. The planning method described beforehand typically follows the configuration space surfaces of the workspace obstacles, as it searches for the shortest, collision-free path to the goal. In combination with speed regulation this leads to rather long path execution times. Optimizing solely for maximum clearance on the other hand could lead to highly inefficient path planning, producing detours and also resulting in high path execution times, too. Thus, clearance competes with path length. In the following sections we will outline how to solve this conflict and how to meet real-time demands at the same time.

4.1 Basic Planning Algorithm

As a basic requisite for using the A* graph search, it is necessary to define edge costs. Edge costs define vertex costs $f(v)$ via the sum $g(v)$ of edge costs from the start vertex to the current vertex. Together with an estimation of costs to the target $h(v)$, we get: $f(v) = w \cdot g(v) + (1 - w) \cdot h(v)$, the well known cost function of the A* (here we choose $w = 0.5$). For time-optimized planning, edge costs are defined by the time it takes to travel them (Eq. 1).

$$t(v_a, v_b) = \sum_{i=0}^{L-1} \left(\frac{1}{s(d_\xi(v(u_i), v(u_{i+1})))} \left\| v(u_{i+1}) - v(u_i) \right\| \right) \tag{1}$$

In this general (approximating) term, the edge from v_a to v_b is subdivided into L segments ($v(u_i)$, $v(u_{i+1})$), and a travel time is calculated for each one by dividing the segment length by the robot speed $s(d_\xi)$, which is derived from the average distance d_ξ to the workspace obstacles along the segment ($v(u_i)$, $v(u_{i+1})$). The distance-regulated speed may be a linear function (for some $k > 0$) of the form:

$$s(d) = \begin{cases} 0 & \text{if } d < d_0 \\ (d - d_0)k & \text{if } d \in [d_0, d_{\max}] \\ s_{\max} = (d_{\max} - d_0)k & \text{if } d > d_{\max} \end{cases} \tag{2}$$

To address the real-time demands, the presented algorithm needs to reduce costly distance calculations. Thus, the overall number of distance calculations is limited per invocation of the path planning algorithm and unknown distances have to be estimated at the roadmap points and along edges of the roadmap, as the geometry of the configuration space obstacles is only implicitly known through the collision test.

Edges are not subdivided to calculate distances along the edge, as the distances between the subdivision points would have to be estimated again in an infinite recursion, so that there is no obvious reason to do so. The distance function along the edge is thus estimated from the distances determined at its vertices. The following two sections describe the edge distance estimation techniques and the vertex distance estimation from calculated vertices, which is also necessary due to the limitation of distance calculations.

4.2 Distance Estimation for Roadmap Edges

In this section we will simplify all possible distance curves along a roadmap edge to a discrete model, deriving several useful conclusions. The distances in the vertices of the edge from v_a to v_b are given by $d_a = d(v_a)$ and $d_b = d(v_b)$. Given a linear connection in configuration space, the edge can be represented by $v(\lambda) = (1 - \lambda)\, v_a + \lambda\, v_b$. We want to derive the distance curve $d(\lambda)$. The environment is considered to be static, that is, no model of obstacle movements exists, as no model of the obstacles in configuration space exists.

In the proposed model, we subdivide λ into L discrete steps $\Delta\lambda = \lambda / L$. For each step, any point of the robot can move by a certain maximum distance, conservatively approximated by the MAXMOVE method [5]. Thus, the distance to obstacles can increase or decrease maximally by $\Delta d = \text{MAXMOVE}^{-1}(\Delta\lambda(v_b - v_a))$. The distance distribution starts at $(0, d_a)$ and ends at $(1, d_b)$. The resolution of the grid can be adapted so that $(1, d_b)$ can be represented by a discrete grid point (if $d_b \in \mathbf{Q}$).

Within this model, the possible spectrum of distance curves can be represented by sequences of L additions of the vector $(1, \pm 1)$. To simplify the syntax, we will reduce this to a sequence of L ± 1-steps. Two invariants hold true for all possible sequences:

- The number of steps is fixed and given by the resolution of the grid. Specifically, this means that the sum of the occurances of +1 and −1 in the sequence is constant: $\#(+1) + \#(-1) = L$
- As the endpoint $(1, d_b)$ must be reached, the following equation needs to be satisfied: $\#(+1) - \#(-1) = (d_b - d_a) / \Delta d$

Based on these two invariants, it is clear that the number of +1-steps and −1-steps is constant and each sequence is simply a permutation of any other valid sequence. Following this result, there are two limiting distance curves of maximum distance and minimum distances, as seen in Figure 1.

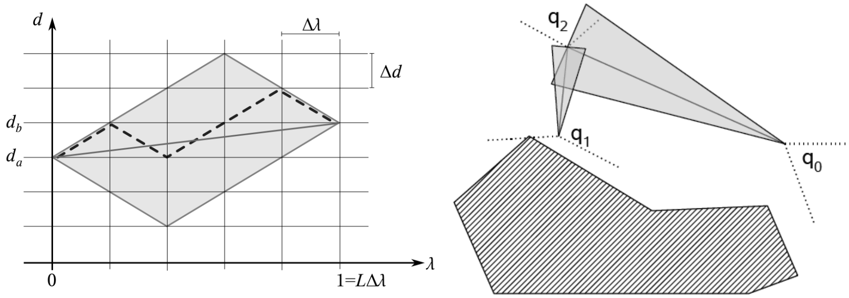


Fig. 1 Left: Model of distance curves. The gray area defines the range of all possible distance curves, the dashed curve represents a discrete example curve. Right: Point distance estimation with estimated distance ranges expanding from calculated distances at point q_0 and q_1 to estimated point q_2.

Thus, at least two distance curve estimations can be derived directly from this model, a *pessimistic edge distance estimation* (the minimum distances curve) and an *optimistic edge distance estimation* (the maximum distances curve). A third estimate that can be derived naturally from this model is the *average edge distance estimation*. In the following, we will derive the equation of the latter estimation.

In the following, n is the number of +1-steps in the sequence and m the number of −1-steps, such that $L = n + m$. The absolute number of all possible curves C_{all} is then given by (selection without repetition, order does not matter):

$$C_{all} = \binom{L}{n} = \frac{L!}{n!(L-n)!} = \frac{L!}{n!m!} \tag{3}$$

For the calculation of the average edge distance estimation, we take a look at the expectation $E_{dist}(i)$ of the distance difference from step i to step i+1:

$$E_{dist}(i) = p_i^+(+\Delta d) + p_i^-(-\Delta d) \tag{4}$$

Where p_i^+ is the probability that a +1-step is at the i-th position and p_i^- the probability for a −1-step. The probability p_i^+ is the number of all curves $C_1(i)$ with a +1 at step i divided by the number of all possible curves C_{all}:

$$p_i^+ = \frac{C_1(i)}{C_{all}} = \frac{\begin{pmatrix} L-1 \\ n-1 \end{pmatrix}}{\begin{pmatrix} L \\ n \end{pmatrix}} = \frac{\frac{(L-1)!}{(n-1)!m!}}{\frac{L!}{n!m!}} = \frac{(L-1)!n!}{L!(n-1)!} = \frac{n}{L} \tag{5}$$

p_i^+ is thus independent of i. With $p_i^- = m/L$ the resulting E_{dist} is independent of i:

$$E_{dist} = \left(\frac{n}{L}\right)(+\Delta d) + \left(\frac{m}{L}\right)(-\Delta d) = \left(\frac{n-m}{L}\right)\Delta d \tag{6}$$

Thus, the average difference vector in each step is $(\Delta\lambda, E_{dist})$. If we sum up all difference vectors from $(0, d_a)$, we get:

$$d(1) = (0, d_a) + (\Delta\lambda, E_{dist})L = (1, (n-m)\Delta d) = (1, d_a + (d_b - d_a))\,, \tag{7}$$

which is the endpoint of all curves. So in effect, the average edge distance estimation is a linear function connecting $(0, d_a)$ and $(1, d_b)$.

4.3 Distance Estimation in Roadmap Vertices

The distance $d(v)$ to unknown obstacles in vertex v is estimated based on the distance of the neighbour vertices v_{pred}. Generally, the distance $d(v)$ may vary in the range of $d(v_{pred}) \pm \mathrm{MAXMOVE}^{-1}(v - v_{pred})$. An estimated distance in a roadmap vertex is thus represented as interval $[l, h]$ describing the range within which the true obstacle distance falls.

If a new distance is measured in a vertex, the intervals of estimated neighbour vertices and their successors in the roadmap need to be adapted. As it would be very costly to update all dependent vertices of a roadmap upon distance calculation of a single vertex, the distance estimation of a vertex is updated in the expansion steps of the graph searching algorithm. This may change edge costs, thus the graph search algorithm D* [9] is used in a slightly modified form, to update all dependent vertices in the search tree efficiently. Intervals are updated by using the distance information (either calculated or an estimation interval) of connected neighbour vertices, that are “more informed” (see also Figure 1, right). First, the distance information of the neighbouring vertex is propagated along the connecting edge using:

$$[l_{new}, h_{new}] = [l_{pred} - d_{MM}, h_{pred} + d_{MM}] \tag{8}$$

With $d_{MM} = \mathrm{MAXMOVE}^{-1}(v - v_{pred})$. For vertices with a calculated distance d_{calc}, the interval reduces to a point $l = h = d_{calc}$.

Second, the interval of the current vertex v_{cur} (which is $[-\infty, +\infty]$ initially) is updated to the smallest common interval:

$$[l_{cur}, h_{cur}] = [\max(l_{cur}, l_{new}), \min(h_{cur}, h_{new})] \tag{9}$$

For edge cost calculation a single distance needs to be selected from the interval. Three different selection functions can be specified similar to the edge distance estimation: an *optimistic vertex estimation* $f_{dist}(v) = h$, a *pessimistic vertex estimation* $f_{dist}(v) = l$ and an *average vertex estimation* $f_{dist}(v) = (l+h)/2$.

5 Discussion

Translated into a speed, the pessimistic edge or vertex estimations may cause the goal to be unreachable, because the distances are estimated below the minimum speed distance before the goal is reached on any possible path through the roadmap. To prevent this, a speed slightly above zero is chosen as minimum speed in the speed function $s(d)$ for path planning. This is safe, as the path is tested to be collision free before execution and invalidated edges are just skipped by the planner. If a calculated vertex distance indicates a collision, all edges connected to that vertex are invalidated.

Besides these basic considerations, the chosen estimation technique influences the generated course of the path. The pessimistic vertex estimation biases the planner to prefer shorter paths in estimated regions of the roadmap because of high costs, while the optimistic estimation reduces path costs in the calculated regions of the roadmap, because of low costs in the estimated regions. This leads to obstacle evasion behaviour in the optimistic case and a goal-oriented behaviour in the pessimistic case. This is confirmed by experimental results (Figure 2e, f). For the edge estimations, the pessimistic edge estimator biases the path towards an extended clearance in comparison to the optimistic edge estimator.

In a dynamic environment, the distances calculated become invalid quickly and thus need to be reset before execution of the planning algorithm each system cycle. With only this limited environment information, the path planner can get trapped in local minima or cycle in endless loops. Parameters influencing this behaviour are the number of vertices and edges and the number of available distance tests per execution of the planner. In an environment with more or less static obstacles, this behaviour is undesirable. If the planner can detect whether obstacles move, it can keep distance calculations already done in previous planning cycles. As safety is assured by the underlying collision testing before path execution, this detection does not need to be exact. The planner may only degrade in performance with respect to time-optimized paths when planning on distances that have changed slightly due to small obstacle movements. The sensor system can thus deliver an underestimation of the workspace dynamics, indicating a static environment upon only small obstacle movements.

6 Experimental Results

In the following sections we present experimental results for a 2D simulated configuration space and in a real world application with a six-degrees-of-freedom industrial robot. The 2D simulation environment operates the robot using the same parameters (including the distance-speed-function $s(d)$, see Eq. 2) and configuration space size used in the real world application. The collision test and distance calculation are slowed down artificially to resemble the behaviour of the real-world sensor-based counterparts. The simulation environment ensures repeatability of the experiments and is well-suited for visualization of the results. The real-world application shows the feasibility of the approach.

6.1 Simulation of a 2-DOF Robot

The simulation experiments were carried out with the shortest-path algorithm outlined in Section 3 and the distance-enhanced algorithm from Section 4 to allow for speedup comparisons. The experiments comprised several standard benchmarks. Some of the setups and results are depicted in Figure 2. All static roadmaps are

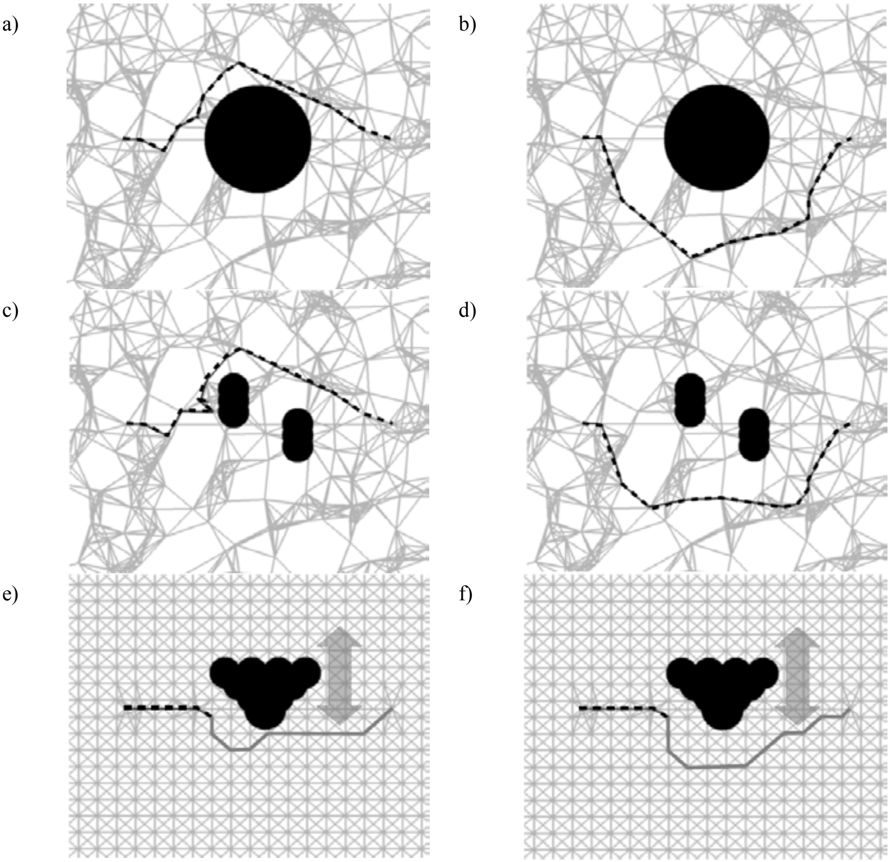


Fig. 2 Robot paths in simulated 2D environments. Executed paths are indicated by dashed lines. For the benchmarks SIMPLE (a,b) and DETOUR (c,d), the shortest path (left) and the time-optimized path (right) are planned in a roadmap with random sampling. The start of the planning task is at the left end of the path, the target at the right. Switchbacks that may be present originate from insufficient planner information about the environment. Images e), f) show a planning step (the robot's progress is denoted by the dashed line and the planned path is indicated by the solid dark gray line) with a moving object (movement indicated by arrow). Both images use a time-optimized planner on a regular grid. e) Uses a pessimistic estimator for roadmap vertices and f) an optimistic estimator. The optimistic estimator biases the graph searching D* to evade the obstacle, while the pessimistic biases it to reduce path lengths to the target.

initialized with either random or regular vertex sampling with two different sizes (500 and 4000 vertices). Table 1 shows true complete path execution times sorted by benchmark. The planning cycle time of the system is < 100 ms. The time-optimizing planner speeds up the execution time from the start to the goal position by up to 63 percent in the DETOUR benchmark.

Table 1 Benchmark results for shortest-path and time-optimizing planning algorithms, Benchmark SIMPLE (1), SIMPLE with a moving object (2), DETOUR (3). Listed are average *path execution* times in seconds. The planning time in each system cycle is less than 100 ms, thus achieving an update rate of >10 Hz on a 2.2 GHz standard workstation.

		Optimistic Edge Distance Estimation		Pessimistic Edge Distance Estimation		Average Edge Distance Estimation	
Bench -mark	Short-est-Path	Optim. Vertex	Pessim. Vertex	Optim. Vertex	Pessim. Vertex	Optim. Vertex	Pessim. Vertex
1 29.	7	13.9	15.7	13.3	15.9	13.9	15.7
2 16.	3	12.4	12.7	12.3	11.6	12.0	12.2
3 31.	4	13.3	12.9	11.4	12.1	12.8	12.8

6.2 Real-World Application with a 6-DOF Industrial Robot

In this application (Figure 3), the robot speed is scaled up to a maximum of ca. 500mm/s based on its clearance. The system operates at 10 Hz frame rate. The roadmap contains 5000 points, each one connected to ca. 15 neighbouring vertices.

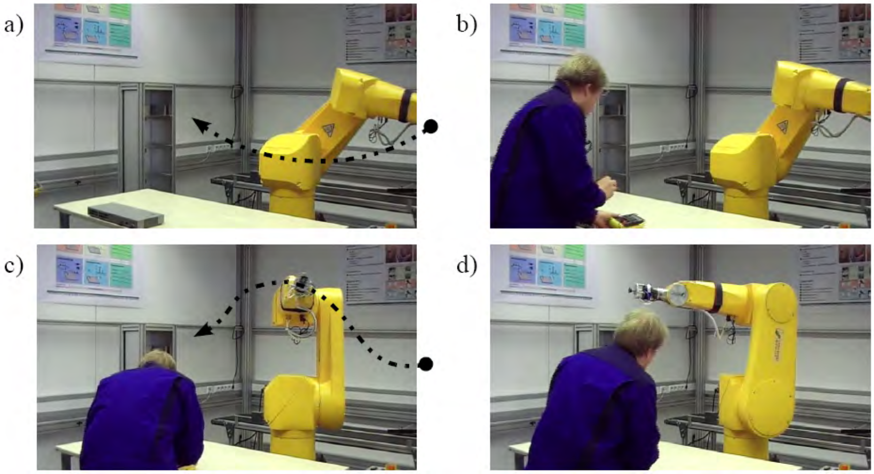


Fig. 3 Robot path in a real-world application. The original robot path is indicated by the dashed line in a). The robot evades a worker entering to perform a maintenance task. The robot executes the path indicated by the dashed line in c).

7 Conclusion

We have presented a time-optimizing path planner for robots with high degree-of-freedom. The planner offers real-time performance by limiting the number of distance calculations and collision tests, while being statistically complete and thus being able to reach the goal even in cluttered environments given an unlimited number of planning cycles. The costs of the collision test are further reduced by offline testing the static environment. The distances are estimated and propagated through the roadmap to benefit from a few calculated distances as much as possible. In the presence of distance-based speed regulation in human-robot coexistence, massive speedups of up to 63 percent compared to the shortest-path solution can be realized.

References

1. Barraquand, J., Latombe, J.-C.: Robot motion planning: A distributed representation approach. International Journal of Robotics Research 10(6), 628–649 (1991)
2. Brock, O., Kathib, O.: Elastic Strips: A framework for integrated planning and execution. In: Corke, P., Trevelyan, J. (eds.) Proceedings of the International Symposium on Experimental Robotics. Lecture Notes in Control and Information Sciences, vol. 250, pp. 328–338. Springer, Heidelberg (1999)
3. Geraerts, R., Overmars, M.H.: Clearance based path optimization for motion planning. In: Proceedings of the International Conference on Robotics and Automation, New Orleans, USA, April 26 – May 1, 2004, vol. 3, pp. 2386–2392 (2004)
4. Gecks, T., Henrich, D.: Path Planning and Execution in Fast-Changing Environments with Known and Unknown Objects. In: Proceedings of the International Conference on Intelligent Robots and Systems, San Diego, USA, pp. 21–26 (2007)
5. Henrich, D., Wörn, H., Wurll, C.: On-line path planning with optimal C-space discretization. In: Proceedings of the International Conference on Intelligent Robots and Systems, Victoria, Canada, October 12-16 (1998)
6. Holleman, C., Kavraki, L.E.: A Framework for Using the Workspace Medial Axis in PRM Planners. In: Proceedings of the International Conference on Robotics and Automation, San Francisco, USA, April 24-28, 2000, pp. 1408–1413 (2000)
7. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. International Journal of Robotics Research 5(1), 90–98 (1986)
8. Lindemann, S.R., LaValle, S.M.: Smoothly blending vector fields for global robot navigation. In: Proceedings of the 44th IEEE Conference on Decision & Control, Sevilla, Spain, December 12-15, 2005, pp. 3353–3559 (2005)
9. Likhachev, M., Fergusson, D., Gordon, G., Stentz, A., Thrun, S.: Anytime dynamic a*: An anytime, replanning algorithm. In: Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS) (2005)
10. Quinlan, S., Kathib, O.: Elastic bands: Connecting path planning and control. In: Proceedings of the International Conference on Robotics and Automation, vol. 2, pp. 802–807 (1993)
11. Schwartz, J.T., Sharir, M.: On the piano movers problem: II. General techniques for computing topological properties of real algebraic manifolds. Advances in Applied Mathematics 1(4), 293–351 (1983)

Analysis of Strain Transfer to FBG's for Sensorized Telerobotic End-Effector Applications

Dean J. Callaghan, Mark M. McGrath, Ginu Rajan, Eugene Coyle,
Yuliya Semenova, and Gerald Farrell

Abstract. Sensorized instruments which cater for the measurement of interaction forces during surgical procedures are not available on current commercial Minimally Invasive Robotic Surgical (MIRS) systems. This paper investigates the effectiveness of advanced optical sensing technology (Fiber Bragg Grating) as surgical end effector strain/force sensors. The effects of adhesive bonding layer thickness and length are specifically addressed owing to their importance for effective strain transfer and ensuring compactness of the resulting sensing arrangement. The strain transfer characteristics of the compound sensing arrangement are evaluated by the examination of shear transfer through the fiber coating and adhesive layers. Detailed analysis of the sensing scheme is facilitated through the use of FEA. Validation of the resulting models is achieved through experimentation carried out on an application-specific evaluation platform. Results show that strain values from an FBG are comparable to that of an electrical strain gauge sensor.

1 Introduction

Current commercially available MIRS systems greatly augment the surgeon's ability to carry out an operating procedure effectively but lack the facility to relay haptic (kinesthetic and tactile) information to the user.

Dean Callaghan · Mark M. McGrath
School of Manufacturing and Design Engineering, Faculty of Engineering,
Dublin Institute of Technology, Bolton Street, Dublin 1, Ireland,
e-mail: {dean.callaghan,mark.mcgrath}@dit.ie

Ginu Rajan · Yuliya Semenova
Photonics Research Centre, Faculty of Engineering, Dublin Institute of Technology,
Kevin Street, Dublin 8, Ireland,
e-mail: {ginu.rajan,yuliya.semenova}@dit.ie

Eugene Coyle
School of Electrical Engineering Systems, Faculty of Engineering,
Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland,
e-mail: eugene.coyle@dit.ie

Gerald Farrell
School of Electronic and Communications Engineering, Faculty of Engineering,
Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland,
e-mail: gerald.farrell@dit.ie

This failure to provide for force measurement at the instrument end-effector restricts the effectiveness of MIRS systems in the detection of interaction, cutting and grasping forces during surgical tasks. A major obstacle in the provision of force feedback in MIRS systems is attributed to the actual measurement of the interaction forces [1].

Research is ongoing into the use of strain/force sensors for the measurement of interaction forces at the instrument-tissue interface. Resistive strain gauge technology has been utilized either in the form of a modular sensor [2, 3] or attached onto the instrument trocar [4-7]. These arrangements only measure interaction and bending forces on the trocar and do not measure grasping and cutting forces. One approach to overcoming this problem is the placement of strain/force sensing transducers either onto the instrument tip or as close as possible to it. Resistive strain gauges have been attached onto the tips of surgical graspers [8-10] and subsequently provide for the measurement of the forces during grasping operations. Sterilization of the arrangement as well as appropriate protection and shielding of the sensors are two of the primary issues associated with the use of resistive strain gauges on instrument tips.

Optical fiber sensors are being used as strain and force sensors in a number of areas including robotic surgery. Moreover, this mode of sensing offers a number of advantages, including, compact dimensions, immunity from electromagnetic interference and multiplexing capabilities.

The NeuroArm neurosurgical robotic system which is designed to be MRI compatible and can be used within a closed magnet bore [11] utilizes optical sensing methods. A three DOF optical force measurement scheme provides a sense of haptic perception to the user. It is unclear from the system description whether or not it has the capacity for measurement of grasping and scissor-cutting forces.

Park *et al* are investigating the feasibility of using FBG sensors in detecting the deflection of surgical needles in an MRI environment [12]. Results indicate that the FBG sensor can be used to estimate the tip deflection, but have also exhibited some hysteresis due to ineffective bonding at the sensor-needle interface.

Work carried out by Greenish *et al* employed “U” shaped strain gauge sensors (Entran ESU-025-1000), in the investigation of the forces generated along a pair of scissor blades, whilst cutting a range of anatomical tissues [13]. Forces on the scissor blades were found to be in the range of 2 N to 30 N for a range of biological tissue types. An evaluation testbed developed by Callaghan *et al* [14] facilitates the measurement of blade forces while cutting a range of synthetic tissue samples. Force values are in the range 2 N to 14 N with maximum blade strain of 121 $\mu\varepsilon$.

This work presents preliminary investigations into the use of FBG’s as force measurement sensors for robotic surgical end-effectors. An investigation was carried out to ascertain the minimum bond length required to ensure uniform strain over the FBG sensing element. Experimental validation of the FE results was carried out on a novel application-specific test rig incorporating a simplified geometrical realization of an actual surgical scissor blade. The rig design allows for the bonding of both an electrical strain gauge (ESG) and a 5 mm FBG sensor, enabling their respective performances to be evaluated for loads up to 30 N. This loading induces strain values of up to 350 με in the blade allowing the effectiveness of strain transfer from blade structure to the FBG sensor to be evaluated.

2 FBG Working Principle

An elementary fiber Bragg grating comprises of a short section of single-mode optical fiber in which the core refractive index is modulated periodically using an intense optical interference pattern [15], typically at UV wavelengths. This periodic index modulated structure enables the light to be coupled from the forward propagating core mode into backward propagating core mode generating a reflection response. The light reflected by periodic variations of the refractive index of the Bragg grating, having a central wavelength λ_G, is given by [16],

$$\lambda_G = 2n_{eff}\Lambda \tag{1}$$

where n_{eff} is the effective refractive index of the core and Λ is the periodicity of the refractive index modulation.

The basic principle of operation of any FBG based sensor system is to monitor the shift in the reflected wavelength due to changes in measurands such as strain and temperature. The wavelength shift, $\Delta\lambda_S$, for the measurement of applied uniform longitudinal strain, $\Delta\varepsilon$, is given as,

$$\Delta\lambda_S = \lambda_G(1-\rho_\alpha)\Delta\varepsilon \tag{2}$$

where ρ_α is the photo elastic coefficient of the fiber given by the formula,

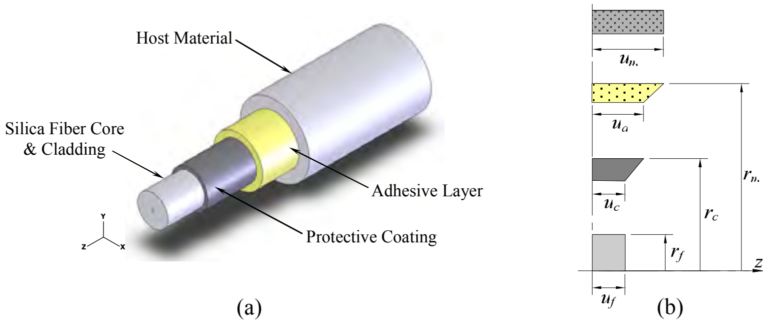
$$\rho_\alpha = \frac{n^2}{2}[\rho_{12} - \upsilon(\rho_{11} - \rho_{12})] \tag{3}$$

where ρ_{11} and ρ_{12} are the components of the fiber optic strain tensor and υ is the Poisson's ratio. For a silica core fiber the value of $(1 - \rho_\alpha)$ is usually 0.78. Thus, by measuring the wavelength shift, using techniques such as those described in [17], changes in temperature or strain can be determined depending on which parameter the FBG sensor is being used to measure.

A single mode optical fiber (SMF28) with an acrylate coating, containing an FBG of peak reflected wavelength of 1550 nm and a peak reflectivity <70%, was used in this experimental work. A 15 mm portion of the coating is removed to allow the 5 mm FBG to be written into the fiber core. A polyimide recoat, with a stiffness value greater than that of the acrylate, had then been applied over the 15 mm portion, providing for enhanced strain transfer. The shift in the FBG reflected wavelength due to strain is measured using an FBG interrogator from Smart Fibers Ltd. (Wx-02). The internal temperature sensor of the interrogator is used to compensate for ambient temperature fluctuations.

2.1 Strain Transfer Theory

Initial theoretical investigations into the strain transfer from a host material to a cylindrical fiber were carried out by Cox [18]. The resulting derived solution is



(a) (b)

Fig. 1 (a) Structure of an optical fiber embedded within a host material (b) Material displacements and radii.

adapted in this work to a four-layer cylindrical model for the purpose of identifying the strain transfer parameters which influence strain transfer between host material and fiber core. Following this, the derived expression is amended enabling the embedded model to be utilized for a surface mounted arrangement.

A bare fiber encapsulated within a protective coating, adhesive layer and the host material is illustrated in Fig. 1. The host material is the only element to which an axial load (z-direction) will be directly applied. The resulting strain will be transferred to the bare fiber as a result of shear strain within the two intermediate layers. The average strain transfer coefficient (ASTC) $\bar{\alpha}$, is defined as the ratio of the average strain over the bonded fiber, $\bar{\varepsilon}_f$, to that of the host material, ε_m, and can be calculated using the following expression,

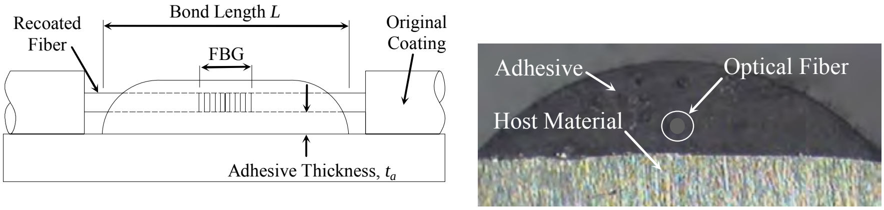
$$\bar{\alpha} = \frac{\bar{\varepsilon}_f}{\varepsilon_m} = 1 - \frac{\sinh(kL)}{kL\cosh(kL)} \tag{4}$$

where L is the length of the FBG sensor and k is the shear lag parameter encapsulating the material and geometric properties of the fiber, coating and adhesive layers and is given by,

$$k^2 = \frac{2}{E_f r_f^{\,2}\left[\frac{1}{G_c}\ln\left(\frac{r_c}{r_f}\right) + \frac{1}{G_a}\ln\left(\frac{r_m}{r_c}\right)\right]} \tag{5}$$

where G_c and G_a are the shear modulii values for the protective coating and adhesive layer respectively and E_f being the Young's modulus of the fiber material.

A surface mounted fiber differs from an embedded fiber in that the host material does not fully encapsulate the coated fiber (Fig. 2(a)). A correction term introduced by [13] which incorporates the adhesive layer thickness t_a into expression (5) ensures its applicability to a surface mounted fiber.



(a) (b)

Fig. 2 (a) Surface bonded FBG sensor (b) View of the optical fiber within the adhesive layer.

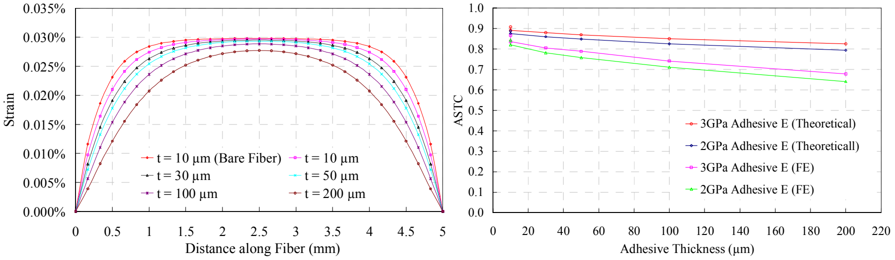
2.2 FE Strain Transfer

In practice the adhesive layer will take up a flattened profile (Fig. 2(b)) which is approximated as an ellipse for the purpose of this FE model. A previous study [19] demonstrated that variation in side width and top thickness of the elliptical profile had negligible effect on the strain transfer from host material to fiber core. It was shown, however, that the adhesive layer thickness, t_a, between the protective coating and the host material surface greatly influenced the average strain in the fiber core. FE models in this work are being used to study the effects of varying the adhesive layer thickness, bond length and stiffness values on the ASTC in a manner representative of a surface mounted FBG.

A 0.03% uniform strain was applied to the host material in the axial (z) direction for each simulation. The following parameters were varied and the strain distribution along the fiber core monitored after each simulation,

- The thickness of the adhesive layer between fiber coating and host material was varied from 10 µm to 200 µm.
- The Young's modulus of the adhesive was set at either 2 GPa or 3 GPa.

The strain distribution along the fiber core for an adhesive Young's modulus of 3 GPa over a range of thicknesses from 10 µm to 200 µm is shown in Fig. 3(a). Trends in the data show that the strain distribution over the bonded 5 mm length is uniform over a region of approximately 1.5 mm for adhesive layer thicknesses between 10 µm and 30 µm. A further reduction in the uniform strain length is observed for layer thicknesses between 50 µm and 200 µm. The average value obtained from each of the hyperbolic strain curves in Fig. 3(a) represents the strain in the fibre core. Shear concentrations through both the adhesive and protective coating layers indicate that 100% uniformity in strain over the FBG length is unattainable by bonding the 5 mm FBG length only. The ratio of the average core strain to that of the host material (ASTC) is plotted in Fig. 3(b) for five different adhesive layer thicknesses and Young's modulus values of 2 and 3 GPa over a 5mm bond length only. Comparing the FE and the theoretical ASTC values it can be seen that strain transfer is most effective when the adhesive layer thickness is smallest (10 µm) and its Young's modulus is greatest (3 GPa). Reducing the stiffness of the adhesive from 3 GPa to 2GPa has the effect of reducing the ASTC by an average of 2%.



(a) (b)

Fig. 3 (a) FE strain distribution along fiber core (b) FE and theoretical ASTC values for 5mm bond length.

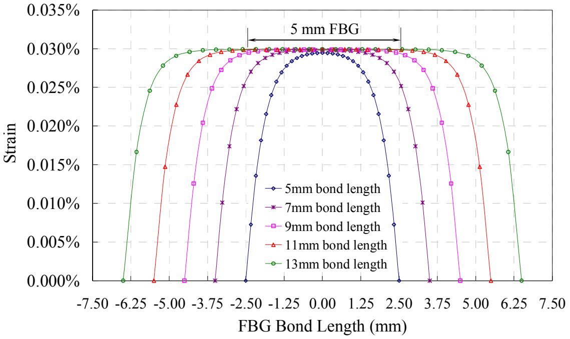
2.3 Influence of Fiber Bond Length

It can be seen from (4) that the ASTC is influenced by the shear lag parameter, *k,* which incorporates material and geometrical properties including adhesive layer thickness. The fiber bond length, *L,* also influences the effectiveness with which the host material strain is transferred to the FBG core. Achieving accurate strain measurement necessitates that a uniform strain distribution is obtained along the 5 mm grating region. This ensures that,

1. The strain sensitivity of the FBG (1.2 pm/με) is valid, as this value is based on the assumption of strain uniformity.
2. No spectral broadening or distortion of the reflected wavelength spectra occurs which can result in measurement errors.

FE analysis was carried out to establish the minimum FBG bond length which ensures strain uniformity over the 5mm grating. Five models were created using properties taken from an FBG employed in our experimental investigations, with the bond length being the only variable. Ideally, the adhesive layer thickness should be maintained as small as possible as outlined in Section 2.2, however achieving values in this range is prohibited by the technique employed for attachment of the fiber to the blade surface. This technique utilizes the dimensions of the

Fig. 4 FE strain distribution for various adhesive bond lengths at a fixed adhesive thickness of 58.5 μm.



original Acrylate coating (250 µm) either side of the recoated portion (133 µm), to maintain a consistent bond thickness of 58.5 µm along the bond length. Results from the simulations showed that for an adhesive layer thickness of 58.5 µm, and a 4 µm thick polyimide coating, a minimum bond length of 11 mm ensures uniform strain distribution along the 5 mm FBG (Fig. 4).

2.4 Sensor Placement

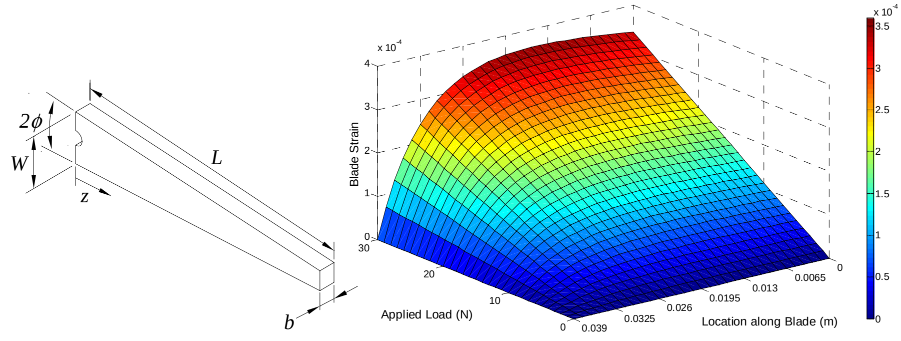
Strain measurements from the FBG sensor can be compared with those obtained from the FE model provided the strain field being measured by the sensor is uniform along the bonded grating length. Assuming that the scissor blade can be approximated as a uniformly tapered cantilever beam the point of maximum strain as well as the strain distribution over the bonded region of the fiber is established using the following expression,

$$\varepsilon_z = \frac{6F(L-z)}{Eb\left(W - 2z\tan\phi\right)^2} \quad (6)$$

where z is the distance from the blade pivot point, F is the applied load, W is the thickness of the blade at its pivot, b is the width of the blade, E is the Young's modulus of the blade material and ϕ is half the angle of the blade (Fig. 5(a)).

The maximum strain was found to occur at a location 14 mm from the blade pivot point (Fig. 5(b)). Minimal strain variation (0.003%) is estimated between 11.5 mm and 16.5 mm on the blade surface onto which the FBG sensor is attached. This small strain field variation will have negligible adverse effect on the reflected FBG signal.

Preliminary investigations into the nature of the strain values for blades of proportionately smaller dimensions revealed sufficiently large deflection and strain values still exist. Analysis of a small blade with $L = 5$ mm, $b = 1.5$ mm and $W = 2$ mm reveals that for loads up to 10 N, maximum strain values of up to 270 µm are achieved.



(a) (b)

Fig. 5 (a) Blade geometry (b) Strain distribution on blade upper surface over a range of applied loads.

3 Experimental Validation

An experimental testing platform has been developed allowing investigation into the effects of adhesive layer thickness and bond length on the transfer of strain from a host material to an FBG fiber. The test rig consists of a simplified blade arrangement which is representative of one blade of a scissor end effector (Fig. 6). Blade dimensions (L = 39 mm, b = 2.286 and W = 7.1 mm) are comparable to that of a pair of straight blade Metzenbaum-Nelson scissors (nopa® instruments AC 110/18). The blade is symmetrical about its pivot point allowing for the simultaneous evaluation of an FBG strain sensor and an ESG. A self temperature-compensated ESG (N11-MA-2-120) with a polyimide backing substrate was used as the reference strain sensor. This established strain measurement technology serves as the best available sensing technique or performance comparison. These ESG readings represent the blade surface strain ε_m at the fiber location, allowing an indication of the ASTC value (4) to be obtained.

Bonding of the FBG sensor to the surface of the blade was achieved using a special fixture incorporating four linear stages. This fixture allows the fiber to be fixed at one location while the blade is aligned in the x, y and z planes prior to the addition of the adhesive layer. After satisfactory alignment, the blade is lowered and the adhesive layer is manually applied over the bond length (11mm). The blade is then raised to its original position where it meets the fiber which is held in position by an applicator containing a 250 μm deep groove. This groove ensures that the fiber is fixed securely to the blade without squeezing the Acrylate coating at the points of contact between applicator and blade (either side of the recoated fiber). This technique is important since squeezing of the Acrylate coating during bonding results in curvature of the recoated fiber and consequently an inconsistent adhesive thickness along the adhesive bond length.

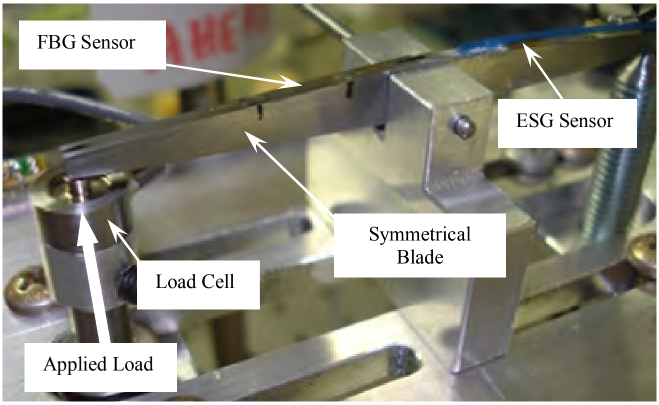


Fig. 6 Experimental test apparatus with surface mounted FBG and ESG strain sensors.

Fig. 7 FBG spectrum at zero load and maximum load.

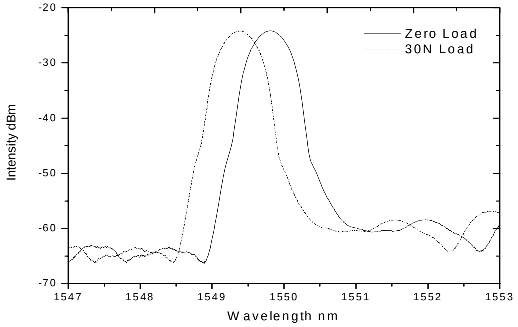
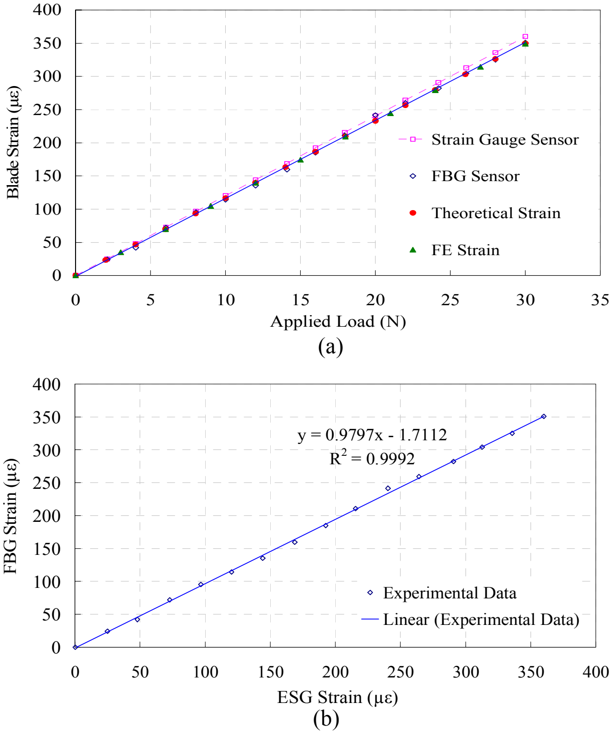


Fig. 8 (a) Experimental and theoretical strain (b) FBG and strain gauge comparison.



(a)

(b)

A range of loads in increments of 2 N between 0 and 30 N were applied to one end of the symmetrical blade resulting in an equal load being applied at the opposite end. This technique induces equal strain fields in each side of the blade. The reflected spectra of the FBG sensor were measured using an optical spectrum analyzer (Agilent 86140B). The measured spectra for zero strain and maximum applied load have the same bandwidth with only a peak shift due to the induced strain being observed (Fig. 7). This confirms that uniform strain is being induced over the grating length, as an appropriate bond length of 11 mm is being used.

The experimental data from the FBG and the ESG over the loading range of the blades are presented in Fig. 8(a). Blade strains obtained using FE and elementary beam theory are also presented. The strain values from the FBG sensor are in good agreement with those from the surface mounted ESG with a maximum of 2.8 % variation in strain being observed over the full range.

The theoretical ASTC is estimated using expression (4) where $\bar{\varepsilon}_f$ is the uniform strain within the fiber core and ε_m is the uniform strain on the blade surface measured by the surface mounted ESG. The FE simulation indicates that an ASTC of 1 is obtainable using an appropriate bond length, adhesive layer thickness and coating thickness. An ASTC value of 0.98 is calculated using FBG and ESG strain values (Fig. 8(b)). This indicates the sensitivity of the FBG is comparable to that of the more established ESG technology signifying its suitability for the proposed measurement application. It is observed that the strain values from the FBG sensor correlate closely with both the analytical and FE values also indicating that a high level of strain transfer from blade to FBG is attained.

4 Conclusions

The aim of the present study is to assess the factors influencing strain transfer between the structure of a simplified scissor blade and the core of an FBG strain sensor. FEA has been employed to improve on the theoretical analysis by creating a more representative model of a surface mounted FBG sensor. Results revealed that for a fixed bond length, a thin adhesive layer with a high Young's Modulus value allowed maximum strain transfer to be achieved along the middle portion of the fiber only. It is essential that the fiber experiences a uniform strain over its full sensing length so that accurate strain readings from an FBG can be obtained. Further FE analysis based on the properties of the fiber revealed that for an adhesive layer thickness of 58.5 µm, a minimum bond length of 11 mm ensures both uniform strain distribution and maximum strain transfer to the FBG core.

A novel evaluation testbed was used which allows FBG and ESG strain readings to be assessed simultaneously. Results show that the sensors are in good agreement with a maximum variation of 2.8% between respective strain readings. The ratio of the strain measured by the FBG to that of the ESG was found to be 0.98. Analysis of the reflected FBG spectrum at zero and maximum load reveals that no errors occurred in the FBG strain measurements as a result of strain non-uniformity along the grating. This indicates that 11 mm is an appropriate bond length for the 5 mm FBG used, allowing strain uniformity and complete strain transfer to be obtained.

It has been shown that an FBG sensor can be used in the measurement of strain on a simplified scissor blade arrangement as effectively as an ESG sensor. Ensuring appropriate bonding layer thicknesses and short fiber bond lengths widens the potential application of FBG sensors in the field of telerobotic surgery where the associated end-effectors have compact dimensions. Ongoing work involves incorporating the strain sensor into the blade structure, the aim of which is to make the FBG an integral part of the sensorized end-effector unit. In conjunction with this, an appropriate temperature compensation technique is being implemented which will improve the accuracy of the strain/force measurement system.

References

1. Tholey, G., Pillarisetti, A., Desai, J.P.: On-site three dimensional force sensing capability in a laparoscopic grasper. Ind. Robot. 31, 509–518 (2004)
2. Kuebler, B., et al.: Development of actuated and sensor integrated forceps for minimally invasive robotic surgery. Int. J. Med. Rob. and Comp. Ass. Surg. 1, 96–107 (2005)
3. Berkelman, P.J., et al.: A Miniature Microsurgical Instrument Tip Force Sensor for Enhanced Force Feedback During Robot-Assisted Manipulation. IEEE Trans. Rob. Auto. 19, 917–922 (2003)
4. Tavakoli, M., et al.: Haptic interaction in robot-assisted endoscopic surgery: a sensorized end-effector. Int. J. Med. Rob. Comp. Ass. Surg. 1, 53–63 (2005)
5. Mayer, H., et al.: A system for robotic heart surgery that learns to tie knots using recurrent neural networks. In: Int. Conf. Intell. Rob. Syst., pp. 543–548 (2006)
6. Prasad, S.K., et al.: A modular 2-DOF force-sensing instrument for laparoscopic surgery. In: Ellis, R.E., Peters, T.M. (eds.) MICCAI 2003. LNCS, vol. 2878, pp. 279–286. Springer, Heidelberg (2003)
7. Tholey, G., Desai, J.P.: A modular, automated laparoscopic grasper with three-dimensional force measurement capability. In: Int. Conf. Rob. Autom., pp. 250–255 (2007)
8. Fischer, G., et al.: Ischemia and Force Sensing Surgical Instruments for Augmenting Available Surgeon Information. Proc. BioRob, 1030-1035 (2006)
9. Wagner, C.R., Howe, R.D.: Force feedback benefit depends on experience in multiple degree of freedom robotic surgery task. IEEE Trans. on Rob. 23, 1235–1240 (2007)
10. Dargahi, J., Najarian, S.: An endoscopic force-position sensor grasper with minimum sensors. Canadian J. Elec. Comp. Eng. 28, 155–161 (2003)
11. Sutherland, G.R., et al.: NeuroArm: an MR compatible robot for microsurgery. Int. Congress Series, vol. 1256, pp. 504–508 (2003)
12. Park, Y.-L., et al.: MRI-compatible Haptics: Feasibility of using optical fiber Bragg grating strain-sensors to detect deflection of needles in an MRI environment. In: Int. Soc. for Magnetic Resonance in Med. (ISMRM) (2008)
13. Greenish, S.: Acquisition and Analysis of Cutting Forces of Surgical Instruments for Haptic Simulation (Thesis) Dep. of Elec. and Comp. Eng. McGill University (1998)
14. Callaghan, D.J., McGrath, M.M., Coyle, E.: A Force Measurement Evaluation Tool for Telerobotic Cutting Applications: Development of an Effective Characterization Platform. Int. J. Math., Phys. Eng. Sci. 1, 144–150 (2007)
15. Hill, K.O., Meltz, G.: Fiber Bragg grating technology fundamentals and overview. J. Lightwave Tech. 15, 1263–1276 (1997)
16. Rao, Y.-J.: In-fibre Bragg grating sensors. Measurement Sci. Tech. 8, 355–375 (1997)
17. Wang, Q., et al.: Low-cost Wavelength Measurement based on a Macrobending Singlemode Fiber. Optics Letters 31, 1785–1787 (2006)
18. Cox, H.L.: The elasticity and strength of paper and other fibrous materials. British J. App. Phys. 3, 72–79 (1952)
19. Wan, K.T.: Investigation of the strain transfer for surface-attached optical fiber strain sensors. Smart Materials and Structures 17, 35–37 (2008)

Intuitive Collision Avoidance of Robots Using Charge Generated Virtual Force Fields

Alexander Winkler and Jozef Suchý

Abstract. This article presents an approach to intuitive collision avoidance of hand-operated robots. Hand-operations are realized by means of zero-force impedance control using force/torque sensor. Impedance control is then combined with the method of artificial potential fields exerting force on the end-effector. This force can be felt by human, who acts on the manipulator and cooperates with the robot. The way to generate potential fields in this work is based on virtual electrical charges placed on the obstacle surfaces. In comparison with other approaches this method is quite universal and can be flexibly used for all forms of obstacles. It may be favourable to implement an additional artificial damping field to prevent hurting obstacles in the case of higher end-effector velocities. It is also possible to use this approach with moving obstacles. In this situation the charges would be placed dependent on sensor information provided e.g. by camera.

1 Introduction

To guarantee physical integrity of human operators, industrial robots usually worked behind safety fences. In the last several years the so called safety controllers were developed and they became more and more acceptable, [10]. The application of safety controllers permits human-robot interaction which means coexistence and collaboration between human and robot in the shared workspace, [7], [8].

One kind of human-robot interaction is hand-operation of robots by means of forces and torques affected by operators on the manipulator arm or end-effector. Forces and torques can be measured by force/torque wrist sensor or by joint torque sensors. Possible applications are assistance tasks where workpieces are handled

---

Alexander Winkler · Jozef Suchý
Chemnitz University of Technology, Faculty of Electrical Engineering and Information Technology, Robot Systems, Reichenhainer Straße 70, 09126 Chemnitz, Germany,
e-mail: `alexander.winkler@e-technik.tu-chemnitz.de`, `jozef.suchy@etit.tu-chemnitz.de`

jointly by human and robot. Furthermore, in this way the robot can learn and record a complex task from operator. Moreover, human-robot cooperation is not restricted just to industrial robots. It can be assigned also to service/domestic and to humanoid robots.

In this context it may by useful to consider obstacles and restrictions within the workspace in an intuitive manner. This means that in the present paper the operator feels physically boundaries when operating the manipulator. One method to realize this functionality is the integration of virtual potential fields in the workspace which will result in vector force fields acting on the robot. Using potential fields is a well known and investigated approach to planning paths of mobile or stationary robots, [3], [9]. However, it makes also sense to apply virtual force fields in real time during human-robot-interaction or robot-robot interaction. In comparison with commonly used methods to generate the artificial potential field [6] in this contribution the approach of virtual charges is preferred, because it is quite universal and can be used for almost all forms and combinations of obstacles. The common effect of using algorithms based on charges is their easy implementation in the robot controller in contrast to high computing time. Though, nowadays the relatively high available computing power makes this aspect more and more irrelevant.

In this paper the application of virtual force fields is focused to hand-operation of robots. For this reason the next section delivers short insight into the field of force guided or force controlled robots based on zero-force impedance control. Section 3 introduces the new approach to real time generation of virtual forces around obstacles and near restricted areas in the workspace. Within this approach due to the presence of wrist force and torque sensor the algorithm lets the operator physically feel the neighborhood of real or virtual obstacles. It is based upon virtual point charges placed on the obstacle surfaces. After implementation of hand-operation via impedance control and generation of the artificial force vector field, some experiments will demonstrate the performance of the algorithm. The presented algorithm may be improved by some additional features, e.g. by adding artificial damping fiel near obstacles [1], described together with other features in section 3. Finally, in section 4 the short conclusion is given.

2 Hand-Operation of Robot Manipulators

Hand-guiding or hand-operation of robots by a human operator acting by means of forces and torques on its end-effector or on the whole manipulator arm can be understood as a special kind of robot force control, namely the zero-force-control. For this purpose the robot is, as a rule, equipped with a wrist force/torque sensor or with joint torque sensors. So, hand-operation can be performed in Cartesian or in joint space. Another method is the estimation of the interaction forces and torques from the joint motor currents in the case when these values are available, [14]. Further approaches are possible in the future, e.g. the application of tactile matrices.

2.1 Hand-Guidance in Operation Space

With a six component force/torque sensor mounted in the robot wrist it would be obvious to control the robot in operation space. Measured interaction forces will result in linear motions of the robot tool, and interaction torques will generate rotations of the tool. The relationship between interaction forces/torques and robot behavior can be described by the general mappings $\mathscr{F}_v$ and $\mathscr{F}_\omega$

$$\mathbf{v} = \mathscr{F}_v(\mathbf{F}) \quad \text{and} \quad \omega = \mathscr{F}_\omega(\mathbf{T}) \tag{1}$$

where **F** and **T** are the vectors of measured interaction forces and torques, respectively. $\mathbf{v}$ is the vector of the linear velocity and ω represents the angular velocity of the tool frame. It will be also possible to compute the Cartesian interaction forces/torques in tool frame from measured values of joint torque sensors. This, however, seems not to be expedient because contact forces on the whole manipulator arm are then taken into consideration.

2.2 Hand-Guidance in Joint Space

A different behavior of the robot in comparison with the operation space approach can be achieved using the joint space approach to force guidance, [13]. For this purpose the interaction joint torques τ have to be measured by joint torque sensors. Alternatively, they can be calculated from measured values of force/torque wrist sensor using the geometric Jacobian matrix **J**:

$$\tau = \mathbf{J}^T \begin{bmatrix} \mathbf{F} \\ \mathbf{T} \end{bmatrix} \tag{2}$$

These joint torques will result in the motion of the particular joint similarly described by

$$\dot{\mathbf{q}} = \mathscr{F}_q(\tau) \,, \tag{3}$$

where $\dot{\mathbf{q}}$ is the vector of joint velocities.

The advantage of hand-operated robot arms in joint space is that the generated motion matches the expected motion of the particular mechanical system represented by its kinematics. Besides, it is eventually possible to pass singularities which divide the task space of the manipulator without any problem, [12].

2.3 Linear Impedance Dynamics of the Hand-Operated Robot

The relationship between interaction forces/torques and the robot motion can be understood as the enforced dynamics of the force controlled robot. Dynamics may be comparatively freely defined by control algorithms. However, some restrictions have to be taken into consideration. The parameters of this kind of dynamics have to match the dynamics of robot joint drives and the dynamics of operator to guarantee the safety.

The common version of robot dynamics during hand-guidance is the dynamics of mass-damper-system for each joint or each Cartesian degree of freedom. These virtual mass-damper-systems can be implemented in the robot controller by

$$\tau_i = M_i\ddot{q}_i + D_i\dot{q}_i \quad \text{or} \quad F_i = m_i\ddot{X}_i + d_i\dot{X}_i \tag{4}$$

for motion generation in joint space or in operation space. In (4) τ_i and F_i represent the interaction joint torques or the Cartesian interaction forces/torques, respectively. Robot joint angles are given by q_i and X_i are the Cartesian coordinates of the tool frame ($\dot{\mathbf{X}} = [v_x\ v_y\ v_z\ \omega_x\ \omega_y\ \omega_z]^T = [\mathbf{v}\ \omega]^T$). Whichever algorithm of hand-operation will be implemented its dynamic behavior can be adjusted by the parameters of virtual masses or mass moments of inertia m_i, M_i and damping coefficient d_i, D_i. Hence, the zero-force controlled robot is realized by special type of well known impedance control, [5].

3 Collision Avoidance Using Virtual Force Fields

Using hand-operation as a method of human-robot collaboration not desired collisions between robot and environment should be avoided. Especially, when contacting stiff surfaces high interaction forces may result in uncoordinated robot motions endangering the operator.

It would be preferable if the human interacting with the robot could feel obstacles during hand-guiding. One idea how to implement it is to make use of virtual forces emitted by the obstacles and boundaries. These forces act against the interaction force and show intuitively restrictions to the operator. The application of artificial potential fields is a well known and investigated approach to path planning of mobile and stationary robots, [3]. In contrast to scalar potential fields the application of vector force fields to human robot interaction induces a lot of new possibilities. They can be also combined with the impedance control of the robot.

3.1 Generation of Virtual Force Fields

Convenient approach to generate the virtual force fields within the robot workspace is based on virtual electric charges, which generate virtual electrostatic field in their neighborhood. The electrostatic force $\mathbf{F}_{12}$ between two charges Q_1 and Q_2 acts according to:

$$\mathbf{F}_{12} = -\frac{1}{4\pi\varepsilon}\frac{Q_1Q_2}{||\mathbf{r}||^2}\frac{\mathbf{r}}{||\mathbf{r}||} \tag{5}$$

In (5) ε represents the electric permeability and $\mathbf{r}$ is the position vector between the charges. The absolute force is reciprocally proportional to the square of the distance between both charges. However, for realization of virtual force fields this particular form of dependence is not obligatory and (5) can be generalized introducing force function $\mathscr{F}$:

$$\mathbf{F}_{12} = \mathscr{F}(||\mathbf{r}||)\,\frac{\mathbf{r}}{||\mathbf{r}||} \tag{6}$$

Hence, the function $\mathscr{F}$ describes the relationship between distance and virtual force.

For the generation of virtual force fields surrounding complex obstacles several charges are necessary. Let the number of charges be n. It will be favorable to place them on the whole surface of obstacle. In contrast to some other approaches taking the form of obstacles directly into consideration when generating the force field, this method is universal and easily programmable. Now, for the calculation of the virtual force $\mathbf{F}_V$ acting against the operator near obstacles the principle of superposition gives

$$\mathbf{F}_V = \sum_{i=1}^{n} \left(\mathscr{F}_i(||\mathbf{p} - \mathbf{e}_i||)\,\frac{\mathbf{p} - \mathbf{e}_i}{||\mathbf{p} - \mathbf{e}_i||} \right), \tag{7}$$

where $\mathbf{p}^T = [p_x\ p_y\ p_z]$ is the current position of the robot end-effector and $\mathbf{e}_i^T = [e_{x_i}\ e_{y_i}\ e_{z_i}]$ describes the location of arbitrary virtual charge with respect to the world frame. Force function $\mathscr{F}_i$ of each particular charge can be chosen individually. Nevertheless, it may be convenient to choose one common function for all charges.

When choosing hyperbolic or exponential force functions the absolute force has to be limited to avoid dangerous situations for the human operator. Furthermore, a maximum action distance between charge and end-effector should be defined. Outside of it the resulting force should be set to zero.

After processing the superposition of all force components the resulting virtual force vector $\mathbf{F}_V$ acts against the real interaction forces $\mathbf{F}_H$, thus giving rise to the resulting force $\mathbf{F}$:

$$\mathbf{F} = \mathbf{F}_H - \mathbf{F}_V \tag{8}$$

It does not seem to be expedient to generate virtual torques in the neighborhood of an obstacle although it might be possible. It can be anticipated that the reaction of the manipulator arm within the virtual force field depends on the mode of hand-guiding. Using hand-guiding in operation space will result in position changes only and the orientation of the tool will be kept constant. On the other side the joint space approach will cause orientation changes, too.

3.2 Hand-Guided Industrial Robot Influenced by Virtual Forces

To demonstrate the approach to intuitive collision avoidance a six axis articulated robot STÄUBLI RX90B was used. It is equipped with a 6-DOF JR3 force/torque wrist sensor to realize force guidance. The robot workspace can be seen in Fig. 1. There are a number of virtual charges placed on the ground floor and on the surface of the storage rack located on the left side. In the here presented case the distance between every two charges is $100mm$. This value was chosen to reduce computation time of the robot program running on the commercial CS7B robot controller. Obviously, the smaller distance would result in the smoother force field. The

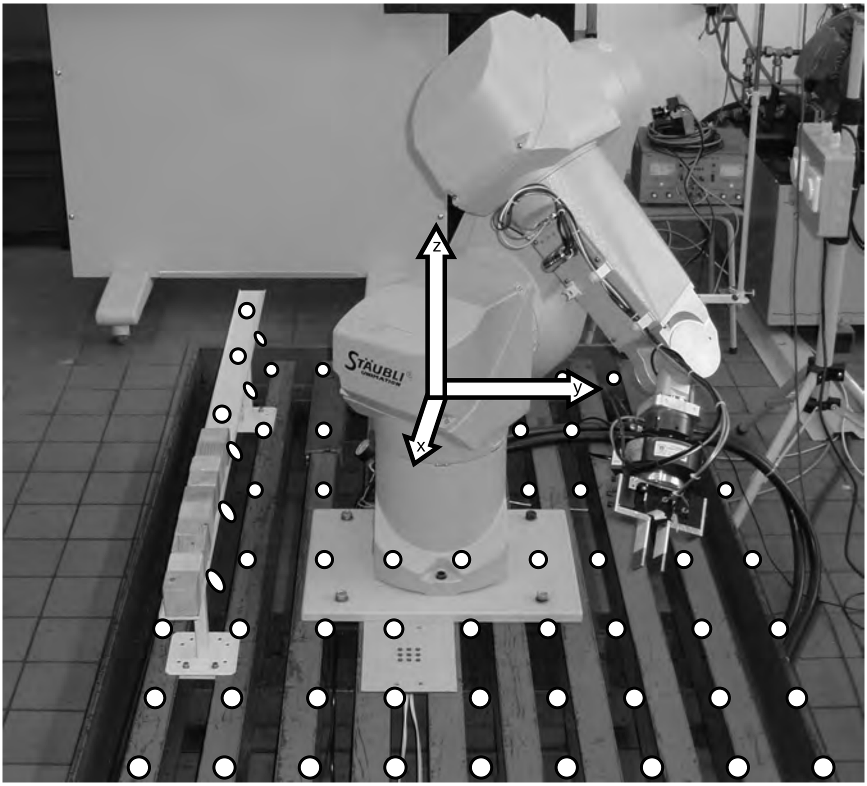


Fig. 1 Several virtual charges placed in the robot workspace.

force function of each charge is set to $\mathscr{F}(r) = 40000Nmm^2 \cdot r^{-2}$, where $r = ||\mathbf{p} - \mathbf{e}_i||$. The force effect of each individual charge is limited by the distance of $150mm$. The resulting force of all virtual charges is saturated to the value of $15N$.

Fig. 2 shows the force field in front of the robot generated by the charges placed on the ground floor located at $z = -400mm$ with respect to robot world frame. For this purpose the absolute value of the virtual force vector on different planes has been plotted. It can be seen that the maximum force value of $15N$ is reached at the distance of approximately $50mm$ between end-effector and charge layer. As can be seen, some valleys appear in the gaps between the charges. This effect may be reduced by increasing the number of charges.

The force in dependence of distance between the end-effector and the border of workspace can be seen in Fig. 3. For this purpose the y-coordinate is fixed to zero. Because of the point charges the force field is somewhat wavy. Nevertheless, this property will not downgrade the functionality of intuitive collision avoidance.

The functioning of this approach can be seen in Fig. 4. A human operates the manipulator by acting on its end-effector. The interaction force vectors are displayed by the dashed arrows. Close to the ground floor some charges are placed which

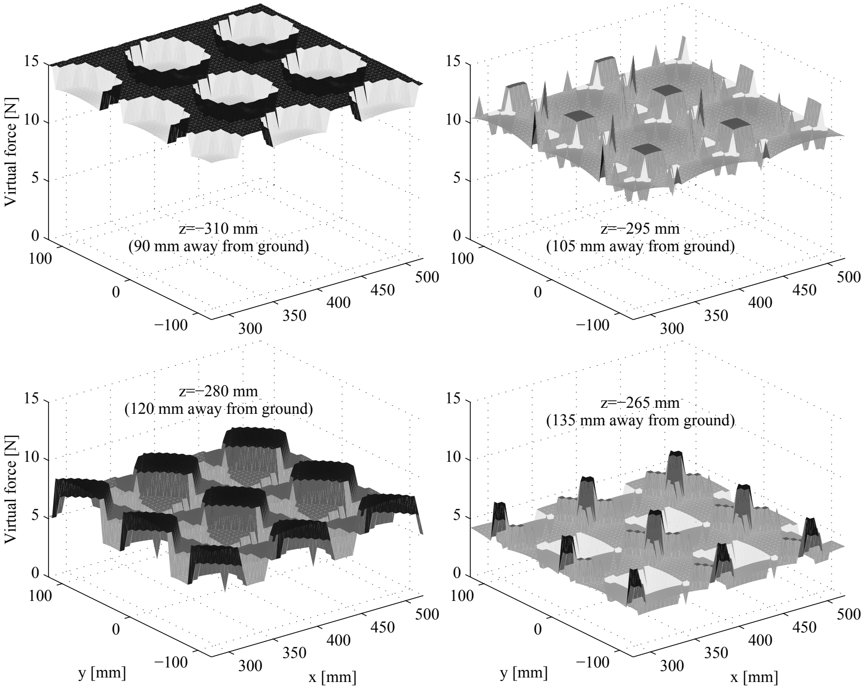


Fig. 2 Force field emitted by the ground floor charges on different distance planes.

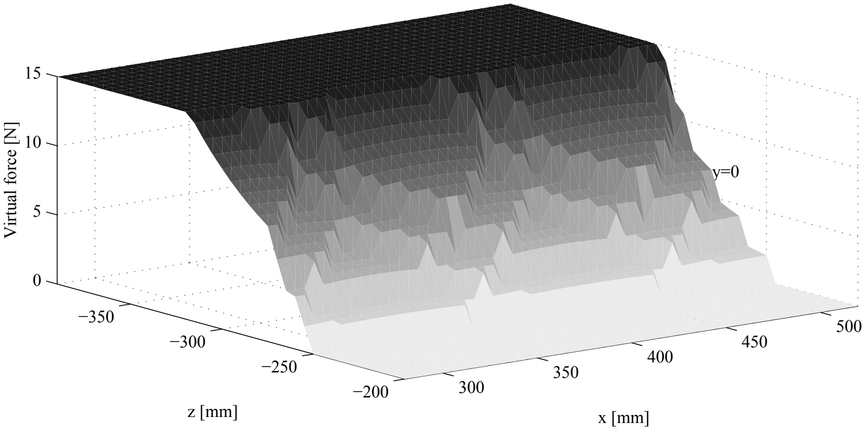


Fig. 3 Force field in dependence of obstacle distance.

generate the virtual force field. When the robot is guided closer to the obstacle the repulsive force increases. This can be seen observing the solid arrows. For better visualization the x-coordinate was kept constant by the operator. The combination

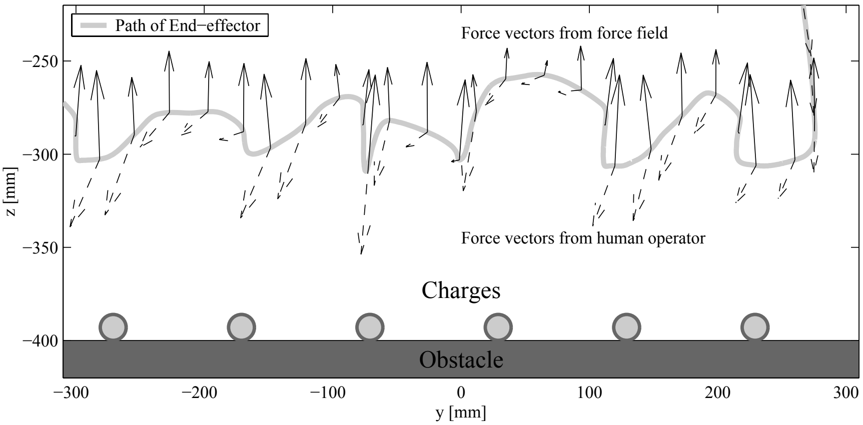


Fig. 4 End-effector path and force in the neighborhood of an obstacle with virtual charges.

of the human interaction force and the virtual force emitted by the charges, results in the corresponding end-effector path are also shown in Fig. 4.

3.3 Additional Aspects and Features

3.3.1 Limitation of Interaction Force Vector

Guiding the robot end-effector closely toward an obstacle a dangerous situation may occur if the interaction force is higher than the virtual repulsive force emitted by the obstacle. It is then possible that the end-effector crosses the charge layer and crashes into the object because behind the charge layer the direction of the virtual force vector suddenly changes. On the other side it will not be convenient to do without limitation of the force function when $r \rightarrow 0$ because dangerous situations for the operator may also occur.

A very simple way to prevent overrun of the charge layer and crash is the saturation of the interaction force of the human operator. It has to be guaranteed that the maximum value of $||\mathbf{F_H}||$ is always smaller than the maximum value of $||\mathbf{F_V}||$. This can be easily realized in the robot program. The charge layer is then an insurmountable barrier and protects the object.

3.3.2 Virtual Damping Field

Another aspect which has to be taken into consideration is the following one: The force field may be seen as a variable virtual non-linear spring which brings the robot back into a position far away from the obstacles. Regarding the dynamics of the hand-guided robot defined in (4) together with the virtual spring the behavior of the manipulator will result in a non-linear spring-mass-damper system. If one considers the additional time delay of the human operating the manipulator and feeling the obstacles it may easily occur that the whole system begins swinging. In the worst case it will become unstable.

One approach to avoid this situation is the implementation of an additional virtual damping field generated by the method of point charges. The simplest way is that the virtual damping $\mathbf{d}_V$ depends directly on the virtual force $\mathbf{F}_V$:

$$\mathbf{d}_V = \left[d_{Vx} \; d_{Vy} \; d_{Vz} \right]^T = \mathbf{d}_V \left(\mathbf{F}_V \right) \tag{9}$$

As the result $\mathbf{d}_V$ will decelerate the current robot motion together with the dampings of the linear impedance dynamics from (4) and stabilize the system.

3.3.3 Sources of Virtual Fields

Until now the virtual charges were placed manually during implementation of the robot program. This method is applicable when the layout of the robot workspace is simple. Another way might be to use the data from CAD of robot work cell. Then the locations of the charges could be generated automatically.

Yet another possibility is collision avoidance between multiple moveable objects using virtual force fields, e.g. between two robots [2] or a robot and a movable mechanism. In this case the positions of the charges located on the movable obstacle are changing in every computation cycle. If so, the primary application of intuitive collision avoidance during hand-operation becomes secondary. The effect of the virtual force field can rather be seen as a kind of the so called non-contact impedance control, [11]. Using this approach with commercial industrial robots, standard motion control algorithms have to be combined with impedance control.

Besides the application of virtual force emitting robots to avoid collisions in robot-robot collaboration the here presented approach will be extended for the purpose of human-robot collaboration. The position of a worker in the robot work cell can be detected by a 3D-camera, [4]. After that the human will be ”cropped” with virtual force charges. The resulting force field will prevent the manipulator from touching the worker.

4 Conlusion

In this paper an approach to generating artificial force fields within the robot workspace has been presented. The source of the field is a set of charges which can be seen as electric charges. This force field can be applied to intuitively control collision avoidance when hand-operating the robot manipulator. If the robot is equipped with a wrist force/torque sensor and the human performs hand-operation by pushing or pulling the end-effector he or she feels the virtual force field and brings the robot away from the restricted areas. Usually the charges have to be placed on the surfaces of obstacles. Thus, this approach is quite universal and can be easily implemented. The algorithm presented here was successfully verified on an industrial robot hand-operated by a human.

In the next future further analysis concerning the force functions seems to be necessary. The number of charges and the force functions assigned to them determine the characteristics of the force field.

Besides human-robot collaboration an interesting scope of application may be the robot-robot cooperation in the shared workspace. In this case virtual force fields emitted by the robots will help to avoid their collisions. For this purpose the commonly used algorithms to control the robot motion have to be extended to include the non-contact impedance control.

Furthermore, additional sensor information may be combined with the presented approach to the force field generation. A camera or laser scanner can monitor the work cell during human-robot collaboration. Human worker generates the virtual force field by charges placed on his body to keep the robot away from himself. The location of the charges is modified during every interpolation cycle by the camera information.

References

1. Arai, T., Ogata, H., Suzuki, T.: Collision avoidance among multiple robots using virtual impedance. In: Proc. of IEEE/RSI International Workshop on Intelligent Robots and Systems, pp. 479–485 (1989)
2. Chong, N.Y., Kotoku, T., Ohba, K., Tanie, K.: Virtual repulsive force field guided coordination for muli-telerobot collaboration. In: Proc. of IEEE International Conference on Robotics and Automation, pp. 1013–1018 (2001)
3. Choset, H., Lynch, K.M., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L.E., Thrun, S.: Principles of Robot Motion. MIT Press, Cambridge (2005)
4. Ebert, D.M., Heinrich, D.D.: Safe human-robot-cooperation: Image-based collision detection for industrial robots. In: Proc. of IEEE/RSI International Conference on Intelligent Robots and Systems, pp. 1826–1831 (2002)
5. Hogan, N.: Impedance control, an approach to manipulation: Part i, ii, iii. ASME Journal of Dynamic Systems, Measurement and Control 107, 1–24 (1985)
6. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. The International Journal of Robotics Research 5(1), 90–98 (1986)
7. Koeppe, R., Engelhardt, D., Hagenauer, A., Heiligensetzer, P., Kneifel, B., Knipfer, A., Stoddard, K.: Robot-robot and human-robot cooperation in commercial robotics applications. In: Dario, P., Chatila, R. (eds.) Robotics Research, ch. 4, vol. 11, pp. 202–216. Springer, Heidelberg (2005)
8. Schraft, R.D., Meyer, C., Parlitz, C., Helms, E.: Powermate - a safe and intuitive robot assistant for handling and assembly tasks. In: Proc. of IEEE International Conference on Robotics and Automation, pp. 4085–4090 (2005)
9. Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G.: Robotics - Modelling, Planning and Control. Springer, Heidelberg (2009)
10. Som, F.: Innovative robot control offers more operator ergonomics and personnel safety. In: Proc. of Joint Conference on Robotics - 37th International Symposium on Robotics and 4th German Conference on Robotics (2006)

11. Tsuji, T., Kaneko, M.: Noncontact impedance control for redundant manipulators. IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans 29(2), 184–193 (1999)
12. Winkler, A., Suchý, J.: Novel joint space force guidance algorithm with laboratory robot system. In: Proc. of 16th IFAC World Congress (2005)
13. Winkler, A., Suchý, J.: Force-guided motions of a 6-d.o.f industrial robot with a joint space approach. Advanced Robotics 20(9), 1067–1084 (2006)
14. Winkler, A., Suchý, J.: Sensorless force guided motions of an industrial robot based on motor currents. In: Proc. of Joint Conference on Robotics - 37th International Symposium on Robotics and 4th German Conference on Robotics (2006)

6D Pose Uncertainty in Robotic Perception

Wendelin Feiten, Pradeep Atwal, Robert Eidenberger, and Thilo Grundmann

Abstract. Robotic perception is fundamental to important application areas. In the Joint Research Project DESIRE, we develop a robotic perception system with the aim of perceiving and modeling an unprepared kitchen scenario with many objects. It relies on the fusion of information from weak features from heterogenous sensors in order to classify and localize objects. This requires the representation of wide spread probability distributions of the 6D pose.

In this paper we present a framework for probabilistic modeling of 6D poses that represents a large class of probability distributions and provides among others the operations of fusion of estimates and uncertain propagation of estimates.

The orientation part of a pose is described by a unit quaternion. The translation part is described either by a 3D vector (when we define the probability density function) or by a purely imaginary quaternion (which leads to a prepresentation of a transform by a dual quaternion). A basic probability density function over the poses is defined by a tangent point on the 3D sphere (representing unit quaternions), and a 6D Gaussian distribution over the product of the tangent space of the sphere and of the space of translations. The projection of this Gaussian induces a distribution over 6D poses.

One such base element is called a Projected Gaussian. The set of Mixtures of Projected Gaussians can approximate the probability density functions that arise in our application, is closed under the operations mentioned above and allows for an efficient implementation.

Wendelin Feiten · Thilo Grundmann
Department of Intelligent Autonomous Systems, Information and Communication,
Siemens AG, Otto-Hahn-Ring 6, 81739 Munich, Germany,
e-mail: {`wendelin.feiten`, `thilo.grundmann.ext`}`@siemens.com`

Pradeep Atwal
Institute for Applied Mathematics, Bonn University,
Regina-Pacis-Weg 3, 53115 Bonn, Germany,
e-mail: `pradeep.atwal@uni-bonn.de`

Robert Eidenberger
Department of Computational Perception, Johannes Kepler University Linz,
Altenbergerstraße 69, 4040 Linz, Austria,
e-mail: `robert.eidenberger.ext@siemens.com`

1 Introduction

As a basic tool in robotic perception, probability density functions of 6D poses need to be represented. In order to be able to represent and process weak information from imperfect sensors, widely spread densities need to be covered by the representation and the inference mechanisms.

The more critical part in the representation of a rigid transform is the rotation. The requirements concerning the parameterization of the rotation are contradictory, but our design goal is to satisfy them as well as possible:

- Unique: There should be only one representation for each orientation
- Minimal: The rotation should be represented with few parameters
- Composable: There should be an easy way to derive the parameters of the composed rotation from the parameters of two rotations in the composition
- Smooth: The rotation should be an at least continuous, or better still a differentiable function of the parameters.
- Distance and area preserving: Properties like areas or distances in the parameter space should be preserved under rigid transform. This is important when we deal with probability density functions over the rotations or transforms.

The formalism for the probability density function of the 6D poses should satisfy the following properties:

- Coordinate System Independent: A coordinate change should only change the arguments to the pdf, not the structure or the parameters of the pdf.
- Information Fusion: The formalism supports the fusion of two probability density informations (for example maximum likelihood estimation).
- Information Propagation: The formalism supports the propagation of uncertain information (i.e. a pose estimate) through an uncertain transform.
- The representation of the pdf uses not too many parameters, much fewer than for example a particle set.

Since each position and orientation w.r.t a given coordinate system is the result of a translation and a rotation. *Position* and *translation* can be and will be used synonymously in this paper, as well as *orientation* and *rotation*. Also, *pose* and *(rigid) transform* are used synomymously.

In Section 2 we will recapitulate various approaches to the parametrization of rigid transforms and corresponding probability density functions. None of them fulfills all requirements listed above, but they provide ingredients to our synthesis. In Section 3 we will present our approach to probability density functions over rigid transforms. In Section 4 we will recollect the presented system and indicate directions of future work.

2 Previous Work

The representation of rigid transforms, and especially of orientation, in 3D is a central issue in a variety disciplines of arts, science and engineering, and contributions from various disciplines are available.

The most popular representations of a 3D rotation are rotation matrix, Euler angles, Rodrigues vector and unit quaternions. For rotation matrices, renormalization is difficult, Euler angles are not invariant under transforms and have singularities, and Rodrigues vectors do not allow for an easy composition algorithm.

Stuelpnagel [9] points out that unit quaternions are a suitable representation of rotations in 3D with few parameters, but does not provide probability distributions.

Choe [7] represents the probability distribution of rotations via a projected Gaussian on a tangent space. However, he only deals with concentrated distributions, and he does not take translations into account.

Goddard and Abidi [4, 5] use dual quaternions for motion tracking. They also capture the correlation between rotation and translation. The probability distribution over the parameters of the state model is a uni-modal normal distribution. This is an appropriate model if the initial estimate is sufficiently certain, and if the information that is to be fused to the estimate is sufficiently well focused. Dual quaternions provide a closed form for the composition of rigid transforms, similar to the transform matrix in homogeneous coordinates (see also Kavan et al. [13]).

Antone [6] suggests to use the Bingham distribution in order to represent weak information. However, he does not give a practical algorithm for fusion of information or propagation of uncertain information. Also, Love [10] states that the renormalization of the Bingham distribution is computationally expensive. Furthermore, it is not (yet) clear to us how the Bingham distribution for rotations could be extended to rigid transforms.

Mardia et al. [12] use a mixture of bivariate von Mises distributions. They fit the mixture model to a data set using the EM algorithm. This allows for modelling widely spread distributions. However, they do not treat translations.

In general, the Jacobian is used to propagate the covariance matrix of a random variable through a non-linear function. Kraft et al. [11] use an unscented Kalman Filter - this technique could be applied also in our setting. However, it would have to be extended to the mixture distributions.

From the analysis of the previous work, we synthesize our approach as follows: We use unit quaternions to represent rotations in 3D, and dual quaternions to obtain a concise algebraic description of rigid transforms and their composition. The base element of a probability distribution over the rigid transforms is a Gaussian in the 6D tangent space, characterized by the tangent point to the unit quaternions and the mean and the covariance of the distribution. Such a base element is called a Projected Gaussian. We use mixtures of Projected Gaussians to reach the necessary expressive power of the framework.

3 Pose Uncertainty by Mixtures of Projected Gaussian Distributions

We assume that the quaternion as such is sufficiently well known to the reader. In order to clarify our notation, at first some basics are restated.

3.1 Quaternions

Let $\mathbb{H}$ be the quaternions, i.e $\mathbb{H} = \{q | q = a + ib + jc + kd\}$, where a is the real part of the quaternion, and the vector $v = (b, c, d)$ is the imaginary part. The imaginary units $\{i, j, k\}$ have the properties $i^2 = j^2 = k^2 = ijk = -1, ij = k, jk = i, ki = j$. The quaternions can be identified with $\mathbb{R}^4$ via the coefficients, $q = a + ib + jc + kd \sim (a, b, c, d)$. The norm of a quaternion is defined as $||q||^2 = a^2 + b^2 + c^2 + d^2$, the conjugate of a quaternion as $q^* = a - ib - jc - kd$ .With the above properties of quaternions we have $||q||^2 = q * q^*$.

Analogously to the way that unit complex numbers $z = \cos(\phi) + i\sin(\phi) = e^{i\phi}$ represent rotations in 2D via the formula $p_{\text{rot}} = zp$ for any point $p \in \mathbb{C}$, unit quaternions represent rotations in 3D.

A point (p_1, p_2, p_3) in 3D is represented as the purely imaginary quaternion $p = ip_1 + jp_2 + kp_3$; a rotation around the unit 3D axis v by the rotation angle θ is given by the quaternion

$q = \cos(\theta/2) + \sin(\theta/2)\,(iv_1 + jv_2 + kv_3)$.

The rotated point is obtained as $p_{\text{rot}} = q * p * q^*$. Clearly, q and $-q$ represent the same rotation, so the set U of unit quaternions is a double coverage of the special orthogonal group $\mathrm{SO}(3)$ of rotations in 3D.

The set U of unit quaternions is identified with the 3-dimensional unit sphere S_3 in $\mathbb{R}^4$, and probability density functions on U are defined by probability density functions on S_3.

3.2 Base Element

For a sufficiently expressive set of probability density functions on the rotations we choose a mixture of base elements.

Each base element is obtained by projecting a Gaussian distribution defined on a tangent space onto the sphere of unit quaternions. This technique is illustrated in Figure 1 for the example of a 1-dimensional unit sphere in $\mathbb{R}^2$. Note that the peaks are lower due to renormalization.

Definition 1: Let S_3 be the 3-dimensional unit sphere in $\mathbb{R}^4$ and r_0 be an arbitrary point on S_3. Further, let $T(r_0) \sim \mathbb{R}^3$ be the 3-dimensional tangent space to the sphere S_3 at the point r_0, with a local coordinate system that has the point q_0 as origin. Further, let $\mathcal{N}(\mu, \Sigma)$ be a Gaussian distribution on $T_R(r_0)$ and the corresponding probability density function be p_T. With the 2-valued central

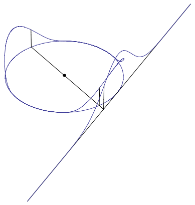


Fig. 1 A base element on the unit circle, obtained by projecting a Gaussian on a tangent line.

projection $\Pi_{r_0} : T_R(r_0) \longrightarrow S_3$ (Figure 1 illustrates how a probability density function is induced on the unit sphere $S_1 \subset \mathbb{R}^2$), a density function is given on S_3 by $p_S(r) := \frac{1}{C} p_T\left(\Pi_{r_0}{}^{-1}(r)\right)$ with $C = \int_S p_T\left(\Pi_{r_0}{}^{-1}(r)\right) dr$. The set of these pdfs p_S is called set of Rotational Projected Gaussians or RPG. The subset of pdfs for which $\mu = 0$ in the corresponding Gaussian on the tangent space T_R is denoted as RPG_0. Note that this definition is not valid for points $r^\perp \in S_3$ that are orthogonal to r_0. $p_S\left(r^\perp\right) := 0$ is the continuous completion.

In practice, the RPG is represented by its tangent point and the basis of the tangent space, and by the parameters of the corresponding Gaussian distribution: $p_S \sim \mathcal{N}\left(T_R(r_0), \mu, \Sigma\right)$. Note that the same distribution can be represented by RPGs with antipodal tangent points.

3.3 Pose Uncertainty

The pose uncertainty is modeled along the lines of the rotation uncertainty by including the translation.

Definition 2: Let $\mathrm{SE}(3)$ be the group of rigid transforms in $\mathbb{R}^3$, the rotation represented by a unit quaternion or equivalently a point on S_3 and the translation by a vector in $\mathbb{R}^3$, $\mathrm{SE}(3) \sim S_3 \times \mathbb{R}^3$. Let $x = (r_0, t)$ be a transform (or pose), with rotation r_0 and translation t. The tangent space to x is given by $T(r_0) := T_R(r_0) \times \mathbb{R}^3 \sim \mathbb{R}^6$, where $T_R(r_0) \sim \mathbb{R}^3$ is the tangent space to the rotation part. Let $\mathcal{N}(\mu, \Sigma)$ be a Gaussian distribution on $T_R(r_0)$. With the 2-valued mapping

$\Pi_{(r_0,t)} : T(r_0) \longrightarrow S_3 \times \mathbb{R}^3$,

$\Pi_{(r_0,t)}(y_1, y_2, y_3, y_4, y_5, y_6) = \left(\Pi_{(r_0)}(y_1, y_2, y_3), (y_4, y_5, y_6)\right)$

a density function is given on $S_3 \times \mathbb{R}^3$ by $p(r,t) := \frac{1}{C} p\left(\Pi_{(r_0,t)}{}^{-1}(r,t)\right)$ with

$C = \int_{S_3 \times \mathbb{R}^3} p\left(\Pi_{(r_0,t)}{}^{-1}(r,t)\right) drdt$. The set of these pdfs p is called set of Projected Gaussians or PG. The subset of pdfs for which $\mu_1 = \mu_2 = \mu_3 = 0$ in the corresponding Gaussian on the tangent space is referenced as PG_0.

Note that $S_3 \times \mathbb{R}^3$ is a double coverage of $\mathrm{SE}(3)$ in the same way that S_3 is a double coverage of $\mathrm{SO}(3)$. Note also that we do not shift the origin with respect to the position part of the pose. Again, in practice the PG is represented by the tangent space as $p \sim \mathcal{N}\left(T(r_0), \mu, \Sigma\right)$.

3.4 Fusion of Projected Gaussians

In analogy to the fusion of Gaussian pdfs pertaining to the same phenomenon, we now describe the fusion of two PGs. The approach is to find a common tangent space that can represent both of the original PGs reasonably well. A detailed analysis based on the approximation theory of the MPG framework exceeds the scope of this paper - a valid heuristic for PG_0 type distributions is to fuse PGs if the angle between the tangent points is less than 15°, or, equivalently, larger than 165°. Below we describe the fusion process for PG in general - in practice we use PG_0.

Let $p_1 \sim \mathscr{N}(T(r_{0,1}), \mu_1, \Sigma_1)$ and $p_2 \sim \mathscr{N}(T(r_{0,2}), \mu_2, \Sigma_2)$ be two pose pdfs with $\cos(r_{0,1} \cdot r_{0,2}) \geq 0.966$ (if $\cos(r_{0,1} \cdot r_{0,2}) \leq -0.966$, use $-r_{0,2}$ instead of $r_{0,2}$, the rest is unchanged).

1. Select $r_{0,3} = \frac{1}{\|r_{0,1}+r_{0,2}\|}(r_{0,1}+r_{0,2})$ as the first tangent point for a common tangent space $T(r_{0,3})$. The basis of the space can be selected arbitrarily (we use a random basis).
2. Restate p_1 in $T(r_{0,3})$: We define the transfer function $f_{1,3} : T(r_{0,1}) \longrightarrow T(r_{0,3})$ by $f_{1,3}(y) := \Pi_{(r_{0,3},t)}{}^{-1}\left(\Pi_{(r_{0,1},t)}(y)\right)$, and the Jacobian of this transfer function at the mean value μ_1 of the original distribution p_1: $J_{1,3} = \left.\frac{\partial f_{1,3}}{\partial y}\right|_{\mu_1}$. The statistical moments of the distribution p_1 represented in $T(r_{0,3})$ are then estimated as $\mu_{3,1} = f_{1,3}(\mu_1)$ and $\Sigma_{3,1} = J_{1,3} \cdot \Sigma_1 \cdot J_{1,3}{}^T$, so $p_{3,1} \sim \mathscr{N}(T(r_{0,3}), \mu_{3,1}, \Sigma_{3,1})$.
3. Restate p_2 in $T(r_{0,3})$ as $p_{3,2} \sim \mathscr{N}(T(r_{0,3}), \mu_{3,2}, \Sigma_{3,2})$. Note that while this is technically well defined even for large angle difference and wide spread distributions, it only makes sense for rather small angle differences and concentrated distributions. If wide distributions are needed, we use mixtures (Section 3.6).
4. Fuse $p_{3,1}$ and $p_{3,2}$: These pdfs are now stated in the same $\mathbb{R}^6$, so the fused pdf is $p_3 \sim \mathscr{N}(T(r_{0,3}), \mu_3, \Sigma_3)$, with the parameters $\Sigma_3 = \left(\Sigma_{3,1}{}^{-1} + \Sigma_{3,2}{}^{-1}\right)^{-1}$ and $\mu_3 = (\Sigma_{3,1} + \Sigma_{3,2})^{-1} \cdot (\Sigma_{3,2} \cdot \mu_{3,1} + \Sigma_{3,1} \cdot \mu_{3,2})$. The resulting probability density function on $S_3 \times \mathbb{R}^3$ needs to be normalized according to definition 2.
5. Generally, $\mu_3 \neq 0$. Since it is advantageous to refrain to base elements of type PG_0, p_3 is restated according to step 2, with the new tangent point $r_{0,4} = \Pi_{(r_{0,3},t)}(\mu_3)$. Finally, the resulting base element is renormalized.

3.5 Composition of Transforms

It is required to model uncertain transforms of uncertain poses, for example if a sensor is mounted on a mobile robot and the pose estimate is needed in world coordinates. In our framework, transforms and poses are represented as dual quaternions in order to calculate the probability distribution function of the composition, see Goddard [4] for more detail.

3.5.1 Dual Quaternions

A dual quaternion $\mathbf{q}_1 = q_{1,1} + eq_{1,2}$ is composed of two quaternions q_1 and q_2 and the dual number e, with $e \cdot e = 0$. Summation of dual quaternions is per component, $\mathbf{q}_1 + \mathbf{q}_2 = (q_{1,1} + eq_{1,2}) + (q_{2,1} + eq_{2,2}) = (q_{1,1} + q_{2,1}) + e(q_{1,2} + eq_{1,1})$.

The product of dual quaternions is

$$\mathbf{q}_1 * \mathbf{q}_2 = (q_{1,1} + eq_{1,2}) * (q_{2,1} + eq_{2,2}) = (q_{1,1} * q_{2,1}) + e(q_{1,2} * q_{2,1} + q_{1,1} * q_{2,2}).$$

The conjugate of a dual quaternion is $\mathbf{q}^* = q_1{}^* + eq_1{}^*$.

Let q_r be the rotation unit quaternion, and $q_t = [0, t_1, t_2, t_3]$ be the quaternion derived from the translation components, then the dual quaternion $\mathbf{q} = q_r + e0.5q_t * q_r$ represents the transform. A point (p_1, p_2, p_3) is embedded into the dual quaternions as $\mathbf{p} = [1,0,0,0] + e[0, p_1, p_2, p_3]$, and with this convention the rotation and translation is $\mathbf{q} * \mathbf{p} * \mathbf{q}^*$. The composition of two transforms, or of a transform and a pose, is represented by the product of the dual quaternions,

$$\hat{\mathbf{p}} = \mathbf{q}_2 * \mathbf{q}_1 * \mathbf{p} * \mathbf{q}_1{}^* * \mathbf{q}_2{}^* = \mathbf{q}_2 * \mathbf{q}_1 * \mathbf{p} * (\mathbf{q}_2 * \mathbf{q}_1)^*.$$

The composition function g will be used to derive the covariance:

$$\mathbf{q}_3 = g(\mathbf{q}_1, \mathbf{q}_2) = g_T(y_1, y_2),$$

where the y_i are the 6-dimensional vectors on the corresponding tangent spaces, and g and g_T are related via the central projections $\Pi_{(r_{0,1},t)}$ and $\Pi_{(r_{0,2},t)}$.

3.5.2 Calculation of Composition

This algebraic formulation justifies to set the tangent point of the composed base element to $r_{0,3} = r_{0,2} * r_{0,1}$, which for PG$_0$ is also the mean value. For base elements in PG$\setminus$PG$_0$, the mean values in the describing pdfs need to be projected to $S_3 \times \mathbb{R}^3$, then propagated and projected back to the tangent space.

The Jacobian of g_T is used to derive the covariance matrix of the base element describing the composition. With $J_C = \left.\frac{\partial g_T}{\partial(y_1, y_2)}\right|_{(0,0)}$ and $\Sigma_C = \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{pmatrix}$ the resulting covariance matrix of the composition is $\Sigma_3 = J_C \cdot \Sigma_C \cdot J_C{}^T$.

3.6 *Mixture of Projected Gaussians*

As stated above, a precondition for the fusion PG base elements is that their tangent points are sufficiently close to each other and that they are sufficiently well concentrated. For this reason, widely spread probability density functions should not be modeled in a single base element.

Instead, we use a mixture of PG or PG$_0$ base elements. Thus let $p_i \in$ PG or $p_i \in$ PG$_0$ be base elements, then the set of Mixtures of Projected Gaussians MPG or Mixtures of Projected Gaussians with zero mean MPG$_0$ is defined as

$\left\{p_m = \frac{1}{n}\sum_{i=1}^{n}\pi_i p_i \,|\, 0 \le \pi_i \le 1, \sum_{i=1}^{n}\pi_i = 1\right\}$. The techniques of fusion and composition carry over to mixtures in a similiar way they work for mixtures of Gaussians [1].

Let $p_{m,1}, p_{m,2} \in \mathrm{MPG}$, $p_{m,1} = \frac{1}{n}\sum_{i=1}^{n}\pi_{1,i}p_{1,i}$, $p_{m,2} = \frac{1}{l}\sum_{j=1}^{l}\pi_{2,j}p_{2,j}$. The base elements of the fused mixture are obtained from fusing the base elements of the original mixtures: $p_{m,3} = f\left(p_{m,1}, p_{m,2}\right) = C\cdot\sum_{i,j=1}^{n,l}\lambda_{i,j}\cdot\pi_{1,i}\cdot\pi_{2,j}\cdot f\left(p_{1,i}, p_{2,j}\right)$, with a normalizing constant $C = \left(\sum_{i,j=1}^{n,l}\lambda_{i,j}\cdot\pi_{1,i}\cdot\pi_{2,j}\right)^{-1}$. The weights $\pi_{1,i}$ and $\pi_{2,j}$ are those of the prior mixture.

The plausibility is composed of two factors, $\lambda_{i,j} = \alpha_{i,j}\cdot\delta_{i,j}$.

The factor $\alpha_{i,j} = e^{-a\arccos\left(\left(r_{0,1,i}\cdot r_{0,2,j}\right)^2\right)}$ says whether the mixture elements can share a tangent space and thus probably pertain to the same cases in the mixture. The angle distance is controlled by the factor a. Plausible results were obtained with $a = 5$, but this lacks a rigid mathematical justification.

The factor $\delta_{i,j} = \left(\mu_{3,1,i}-\mu_{3,2,j}\right)\cdot\left(\Sigma_{3,1,i}+\Sigma_{3,2,j}\right)^{-1}\cdot\left(\mu_{3,1,i}-\mu_{3,2,j}\right)^{T}$ is the Mahalanobis distance of the mean values and covariances transported to the common tangent space. It expresses that even if the mixture elements could share a tangent space, they could still not be compatible.

The composition carries over in a similar manner.

$$p_{m,3} = g\left(p_{m,1}, p_{m,2}\right) = C\cdot\sum_{i,j=1}^{n,l}\pi_{1,i}\cdot\pi_{2,j}\cdot g\left(p_{1,i}, p_{2,j}\right),$$

with $C = \left(\sum_{i,j=1}^{n,l}\pi_{1,i}\cdot\pi_{2,j}\right)^{-1}$. In this case, there is no question of whether two base elements could apply at the same time, since the two probability distributions are assumed to be independent, so the factor $\lambda_{i,j}$ is omitted.

Note that in both cases the individually fused or combined resulting base elements are assumed to be renormalized.

4 Conclusion and Outlook

In this paper we present the framework of Mixtures of Projected Gaussians that allows for modelling a large variety of possible probability distribution functions of 6D poses. In contrast to a particle filter approach, much fewer parameters are needed to describe the distribution. Like particle filter appraoches, it allows for classical probabilistic inference rules like the Bayes update.

The MPG representation of probability density functions is part of our overall architecture for robotic perception [3]. In this larger framework we also use particle filter representations [2]. We can transform the probability density models between the different representation forms. Currently, we use rejection sampling in order to sample from MPG distributions to obtain particle sets, and we use a variant of the EM algorithm in order to estimate MPG parameters from sample sets.

The operations of fusion, propagation or multiplication of MPG distributions generally result in a large number of mixture elements. However, many of them have practically zero weight, while others are approximately identical. The approach for identifying doublettes described in [1] will be carried over to the MPG. Similar components are merged, negligible ones are omitted and the weights are renormalized.

The algorithms for probabilistic inference (fusion, propagation, multiplication) are fully implemented in *Mathematica* and ported to C. The single time consuming step is the integration in renormalizing the components. Here, we hope to find a quadrature function that takes full advantage of the special structure of the integrand to speed up the processing.

The covariance matrices are currently estimated using the Jacobian of the nonlinear transforms. These estimates could be improved by using the unscented estimation technique (see Julier and Uhlmann [14])

In this paper we focus on the perception of static objects. The MPG framework can be extended to the dynamic case as well, following concepts by Goddard [4] and by Brox et al. [8].

Further work needs to be done on the analysis of the errors made in the approximations by mixtures of projected Gaussians. This includes also the selection of an appropriate statistical error measure, which in turn might very well depend on the application.

Acknowledgment. This work was partly funded as part of the research project DESIRE by the German Federal Ministry of Education and Research (BMBF) under grant no. 01IME01D.

References

1. Eidenberger, R., Grundmann, T., Feiten, W., Zoellner, R.D.: Fast Parametric Viewpoint Estimation for Active Object Detection. In: Proceeding of the IEEE International Conference on Multisensor of Fusion and Integration for Intelligent Systems (MFI 2008), Seoul, Korea (2008)
2. Grundmann, T., Eidenberger, R., Zoellner, R.D.: Local Dependency Analysis in Probabilistic Scene Estimation. In: Proceeding of the 5th International Symposium on Mechatronics and Its Applications (ISMA 2008), Amman, Jordanien (2008)
3. Grundmann, T., Eidenberger, R., Zoellner, R.D., Xue, Z., Ruehl, S., Zoellner, J.M., Dillmann, R., Kuehnle, J., Verl, A.: Integration of 6D Object Localization and Obstacle Detection for Collision Free Robotic Manipulation. In: IEEE International Symposium on System Integration, Nagoya, Japan (2008)
4. Goddard, J.S.: Pose and Motion Estimation from Vision using Dual Quaternion-based Extended Kalman Filtering, PhD Dissertation, University of Tennessee, Knoxville (1997)
5. Abidi, M.A., Goddard, J.S.: Pose and Motion Estimation from Vision using Dual Quaternion-based Extended Kalman Filtering. In: Proc. of SPIE Conf. on Three-Dimensional Image Capture and Applications, San Jose, CA, vol. 3313, pp. 189–200 (January 1998)
6. Antone, M.E.: Robust Camera Pose Recovery Using Stochastic Geometry, PhD Dissertation, Massachusetts Institute of Technology (2001)
7. Choe, S.B.: Statistical Analysis of Orientation Trajectories via Quaternions with Applications to Human Motion, PhD Dissertation, University of Michigan (2006)

8. Brox, T., Rosenhahn, B., Kersting, U., Cremers, D.: Nonparametric Density Estimation for Human Pose Tracking. In: Franke, K., Müller, K.-R., Nickolay, B., Schäfer, R. (eds.) DAGM 2006. LNCS, vol. 4174, pp. 546–555. Springer, Heidelberg (2006)
9. Stuelpnagel, J.: On the Parametrization of the Three-Dimensional Rotation Group. SIAM Review 6(4), 422–430 (1964)
10. Love, J.J.: Bingham statistics. In: Encyclopedia of Geomagnetism and Paleomagnetism, pp. 45–47. Springer, Dordrecht (2007)
11. Kraft, E.: A Quaternion-based Unscented Kalman Filter for Orientation Tracking. In: Proceedings of the Sixth International Conference of Information Fusion, vol. 1, pp. 47–54 (2003)
12. Mardia, K.V., Taylor, C.C., Subramaniam, G.K.: Protein Bioinformatics and Mixtures of Bivariate von Mises Distributions for Angular Data. Biometrics 63(2), 505–512 (2007)
13. Kavan, L., Collins, S., O'Sullivan, C., Zara, J.: Dual Quaternions for Rigid Transformation Blending, Technical Report (2006)
14. Julier, S.J., Uhlmann, J.K.: Unscented Filtering and Nonlinear Estimation. Proc. of the IEEE 92(3), 401–422 (2004)

3D Shape Detection for Mobile Robot Learning

Andreas Richtsfeld and Markus Vincze

Abstract. If a robot shall learn from visual data the task is greatly simplified if visual data is abstracted from pixel data into basic shapes or Gestalts. This paper introduces a method of processing images to abstract basic features into higher level Gestalts. Grouping is formulated as incremental problem to avoid grouping parameters and to obtain anytime processing characteristics. The proposed system allows shape detection of 3D such as cubes, cones and cylinders for robot affordance learning.

1 Introduction

Humans can solve even complex tasks such as manipulating or grasping objects by learning and using the relationship of object shape to the intended behaviour - also referred to as affordances. Robotics would profit considerably if a robot could learn these affordances [1] and relate them to visual percepts. However, at present visual perception delivers only image-based features. It is the objective of this work to present a method that extracts the basic 3D shape of objects for enabling more realistic robot learning. Fig. 1 gives an example of a robot that attempts to learn concepts such as moveability of objects.

There are several paradigms of making robots learn affordances. [2, 3] present an approach where a mobile robot observes the environment and autonomously learns a prediction model from its actions and observation data. An overhead camera is used to track colour blobs of the robot and the objects. Another possibility is to discover object affordances [4]. To track objects and estimate the object behaviour, they are also using colour blobs and a set of visual features to describe the 2D blob shape. As an example, a circle can roll but a rectangle cannot. Another possibility is learning high-level ontologies, as proposed in [5, 6], by clustering the point data from a laser range scanner with an occupancy grid map.

Andreas Richtsfeld · Markus Vincze
Automation and Control Institute, Technische Universität Wien,
Gußhausstraße 27-19 / E376, 1040 Wien, Austria,
e-mail: `{ari,vincze}@acin.tuwien.ac.at}`

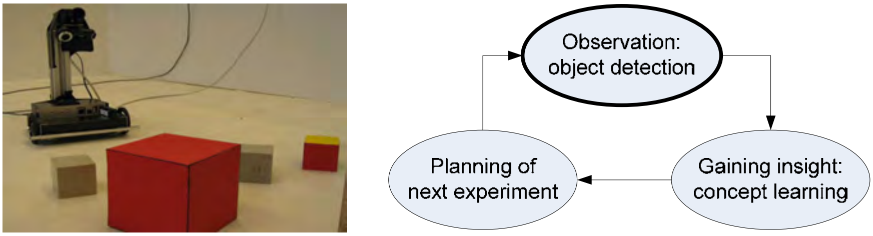


Fig. 1 Robot attempting to learn moveability of objects and the basic learning loop of planning robot motions, detecting objects (the focus of this paper) and gaining insights [2].

All these approaches for autonomous learning are dealing with simple vision data, which are determining relative positions of objects and use image-based point or blob features. However, to understand the usage of different objects the capability to abstract shapes from images into 3D shapes is necessary. This is particularly true for object grasping but also holds for navigation, where an abstraction of point data into geometric features reduces complexity, thus enables efficient learning and yields an effective description of affordances.

Hence we propose a system using methods of perceptual organisation to estimate basic object shapes in a hierarchical manner such that affordances can link to different abstractions as required. This aims at offering a generic tool for autonomous learning of robots in their environments.

2 Related Work

Perceptual recognition is a well known problem in computer vision and was research topic since the Eighties. Seminal works regarding the theory of perceptual recognition of simple objects are [7] and [8]. Biederman [7] proposed that there are non-accidental differences between simple objects, which can be derived through so called geons. [9] discusses the potential of geons in computer vision systems. Perceptual grouping is a bottom-up method to estimate Gestalts, where new hypotheses will be estimated from Gestalts of lower-level Gestalt principles. It can also be seen as geometric grouping, because most of the Gestalt principles implements geometric restrictions.

The main challenge of perceptual grouping is to limit the combinatorial explosion. Indexing and thresholding of less salient hypotheses are normally used to solve this problem [8, 10]. Indexing divides the search space for relations between elements into bins and each element will be allocated to a bin depending on the grouping relationship. Further all elements in a bin can be analysed whether they fulfil the given relation or not. A method for indexing in image space is proposed in [11], where search lines for a Gestalt feature are used to find intersections. Indexing into the image pixels limits the search space and avoids the problem of comparing all combinatorial possibilities to find connections. Depending on the type of search line, different types of groupings can be found.

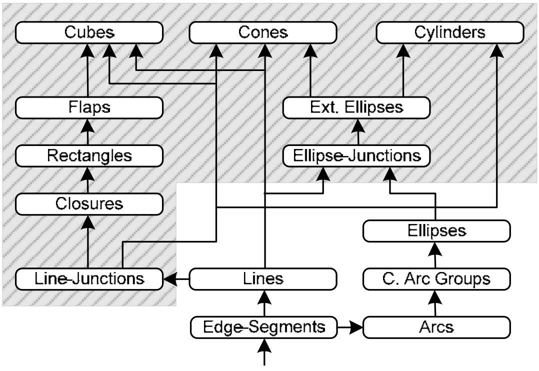
A dilemma of using search lines is the length definition. When the length of the search lines is too small, some important Gestalts could not be found and on the other side, if they are too long, enhancement of computation time could be provoked because many combinations may occur. This problem can be avoided with incrementally growing search lines [11], where in every processing circle another search line grows one pixel. When drawing into the indexing image, intersections with existing search lines are detected immediately in the respective bin. The biggest advantage of this method is that any preset parameters or thresholds are now moved into longer and longer processing. The results are no longer dependent to the line length, but to the processing time. The longer the process operates, the more junctions can be found and thus more other Gestalts can be detected. This gives a possibility to evaluate the best possible hypotheses for a limited processing time and on the other side this makes it feasible to obtain results at any time (a quality referred to as anytimeness). This grouping approach is suited for robotics, since it allows to sequential group higher level Gestalts such as cubes or cylinders in short processing time.

3 System Overview: Perceptual Grouping to Detect Object Shape

We propose an incremental grouping method to enable efficient abstraction of image pixel data into a hierarchy of basic geometric Gestalts shown in Fig. 2. Triggered by search lines in the image each Gestalt creates new hypotheses and delivers them to Gestalts of higher levels for robot learning.

When following the proposal of [11] it is possible to build a perceptual grouping system which processes most of the Gestalt principles incrementally. To extend from this work, we firstly applied the idea of incremental indexing to ellipses. Hence, we insert the Ellipse-Junction as Gestalt element. Boxes in the non-shaded area of Fig. 2 are referring to basic geometric features, whereas the boxes in the

Fig. 2 Gestalt tree: basic geometric features and higher-level Gestalts incrementally built up (shaded).



shaded area indicate Gestalts obtained by incremental processing. This shows that the essential Gestalts are the Line- and Ellipse-Junctions. They are are providing the system with incremental indexing capabilities.

In a first processing step the basic features are processed all at once and grouping uses neighbourhood constraints. Next, incremental processing of Line- and Ellipse-Junctions starts and the higher level Gestalts receive new input incrementally: each cycle extends one search line and all new hypotheses are processed at once to obtain new Gestalt hypotheses. If one of the Gestalts can hypothesise for a new Gestalt one level up the tree, the Gestalt from the next higher level begins to work, and so on. Hence the next junction will not be processed until all higher Gestalts are processed and no more hypotheses can be produced. This functionality of the processing tree guarantees that the best Gestalt hypotheses for the detected junctions are calculated in every processing circle.

The steps for building new Gestalts are the creation of new hypothesis, the ranking and masking, see Fig. 3. Whenever a Gestalt is informed about a new hypothesis, the processing starts and tries to build new hypotheses within this incoming Gestalt. The creation of new hypotheses exploits Gestalt principles from geometric constraints.

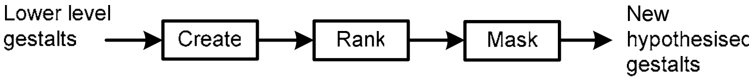


Fig. 3 Processing pipe for a Gestalt in the Gestalt tree.

After creation of new Gestalt hypotheses they will be ranked by quality criteria using the geometric constraints. Ranking is important to firstly calculate the best results and secondly for the masking of hypotheses. Masking is used to prune the poor results, e.g. when different hypotheses disagree about the interpretation of one element. Masking Gestalts is optional and is used to increase the performance and also to avoid combinatorial explosion in the following high-level principles.

4 Gestalt Principles

In this section we explain which Gestalt principles are used to build up the Gestalt tree of Fig. 2. The grouping of Edge-Segments into Lines, Arcs, Convex Arc Groups and Ellipses has been described in [11, 12]. Here we start with an explanation on how Line- and Ellipse-Junctions are formed to then introduce the approach to obtain the basic 3D shapes Cube, Cone and Cylinder.

Line-Junctions

The fundamental principles for incremental processing are Line and Ellipse- Junctions. Each processing circle starts with incrementing one search line of a line or an ellipse. Depending on which search line was chosen, the next hypothesis can be an Ellipse- or a Line-Junction. Line Junctions are connections between two lines. For each line six different search-lines are defined, as shown in Fig. 4, at each end one in tangential direction and two in normal direction. Different combinations of intersecting search lines are generating different junctions. We distinguish between Collinearities, L-Junctions and T-Junctions, where T-Junctions could also be interpreted as two L-Junction.

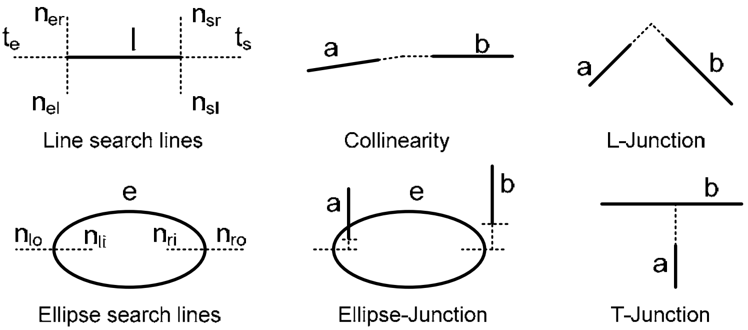


Fig. 4 Search lines for lines and ellipses and different junctions of lines and ellipses.

Ellipse-Junctions

Ellipse-Junctions are connections between a vertex of an Ellipse and a Line, which can also be found with search lines. Ellipse-Junctions have been defined to detect the higher-level Gestalts Cone and Cylinder. The search lines for Ellipses are growing on the main axis of the Ellipses, beginning at the vertices into both directions. For the growing of the search lines, we are using the same growing algorithm as for the search lines of Line-Junctions. Fig. 4 shows the search lines defined for Ellipses on both vertices.

Closures

Closures are closed convex contours, built from the Gestalts Lines and Line-Junctions, as proposed in [11]. Whenever the principle will be informed about a new Line-Junction, a new closed convex contour could be detected. The task is formalised as followed:

- Connect neighbouring lines to form a graph G = (V,E) with Lines as vertices (nodes) and Junctions between Lines as edges E of the graph.
- Perform a shortest path search on G, while making sure this path constitutes a roughly convex polygon.

To find these contours, Dijkstras algorithm for shortest path search is used, where only paths consisting of non-intersecting Lines, Collinearities and L-junctions of the same turning direction are allowed, thus ensuring simple convex polygon contours.

Rectangles

Rectangles can be directly derived from Closures by using geometric restrictions, which are given through the description of Rectangles in a perspective view. When considering these perspective projections as shown in [13] and neglecting one vanishing point (one-point projection), it is possible to recognize Rectangles, whenever a new Closure appears. We build a new Gestalt hypothesis, when:

- A Closure contains four L-Junctions, and
- Two opposing Lines between the four L-Junctions are approximately parallel.

Flaps

A Flap is a geometric figure consisting of two Rectangles that do not overlap and that have one line in common. Detecting Flaps is the intermediate step for detecting Cubes, because two sides of a Cube are always building a Flap in a perspective view of a camera. When seeing a Cube aligned to one side only a Rectangle or Flap is visible, making the Flap an obvious Gestalt element. We can formalise: Whenever a new Rectangle is hypothesised, a new Flap can be built, if

- It is possible to find another Rectangle, which shares one line with the new hypothesised Rectangle, and
- The two Rectangles do not overlap.

Cubes (Cuboids)

An image taken from a camera may show different perspective views of three-dimensional objects. In the case of a cube there may be one, two or three rectangles visible. The chance that one observed rectangle indicates a cube is small and increases for a flap, but both can occur accidentally. Only when we are able to find three adequate rectangles, we can conclude that the robot observes a cube. Cubes and cuboids can be built up from Flaps, Lines and Junctions in two different ways, shown in Fig. 5. We can formalise the detection of Cubes in the following way: Whenever a new Flap is hypothesised, build a Cube, if

- There are two Flaps (upper line in Fig. 5), which share one Rectangle with the new Flap (area 1 and 2) and the two Flaps are sharing one Rectangle (3), or
- There is one L-Junction and Lines that close the Flap from the two outer corners at the smaller inner angle of the Flap (lower line in Fig. 5).

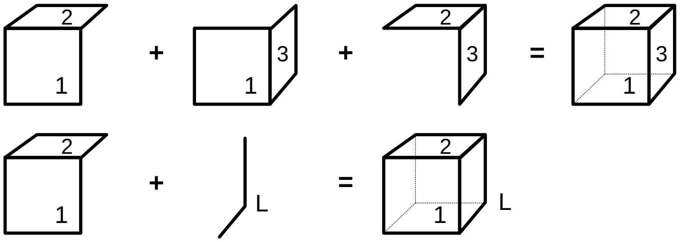


Fig. 5 Composing a cube from three Flaps or from a Flap with two Lines and an L-junction.

The second method for finding cubes is more general than the method with three Flaps and leads to more poor results, because connections between corners of one Flap can also occur accidently. On the other side it gives the possibility to detect cubes, where one rectangle could not be detected.

Extended Ellipses

Extended Ellipses are Ellipses with Lines attached trough Ellipse-Junctions. They are needed to build later the higher-level Gestalts Cone and Cylinder. We can describe the building of Extended Ellipses in the following way: Whenever a new Ellipse-Junction could be hypothesised:

- Build a new Extended Ellipse, if there is not one with the delivered Ellipse, or
- Assign the new Ellipse-Junction to the existing Extended Ellipse.

Cones

Cones are geometric figures consisting of ellipses (circles), lines and junctions between these components. We are using Extended Ellipses, Lines and Line-Junctions to find the object shape as given in Fig. 6. Building cones can be described in the following way:

- Whenever an Extended Ellipse (with at least one E-Junction at both vertices) could be hypothesised, build a Cone, if there is one L-Junction and Lines, which can close the Extended Ellipse.
- Whenever a new L-Junction can be hypothesised, build a new cone, if it is possible to close an Extended Ellipse with one Junction at both vertices.

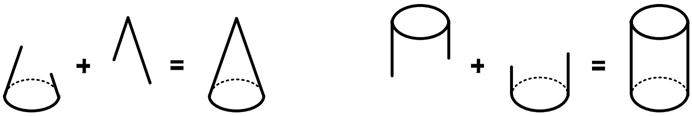


Fig. 6 Composing a cone and a cylinder from lower level Gestalts.

Cylinders

The object shape of a cylinder consists of an ellipse and lines, as shown in Fig. 6. Therefore Cylinders can be grouped from Extended Ellipses, Lines and Junctions. Building new Cylinders can be formalised to:

- Whenever an Extended Ellipse (with at least one E-Junction at both vertices) could be hypothesised, build a Cylinder, if there is another Extended Ellipse, whose lines are connected with the lines of the new Extended Ellipse.

5 Experiments and Results

The incremental grouping method has been evaluated with a mobile robot moving among simple geometric 3D objects. Fig. 7 shows an example image and the edge image containing several three-dimensional objects. The picture indicates the

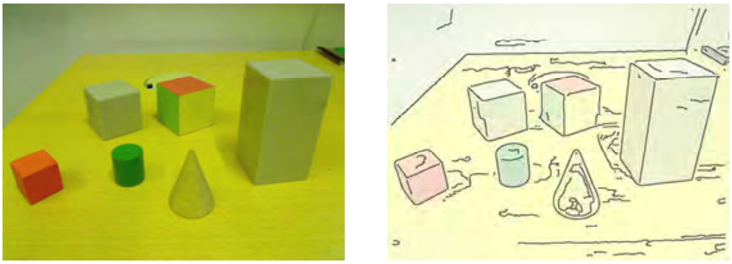


Fig. 7 Original image (left) and edge image with underlying half-transparent image (right).

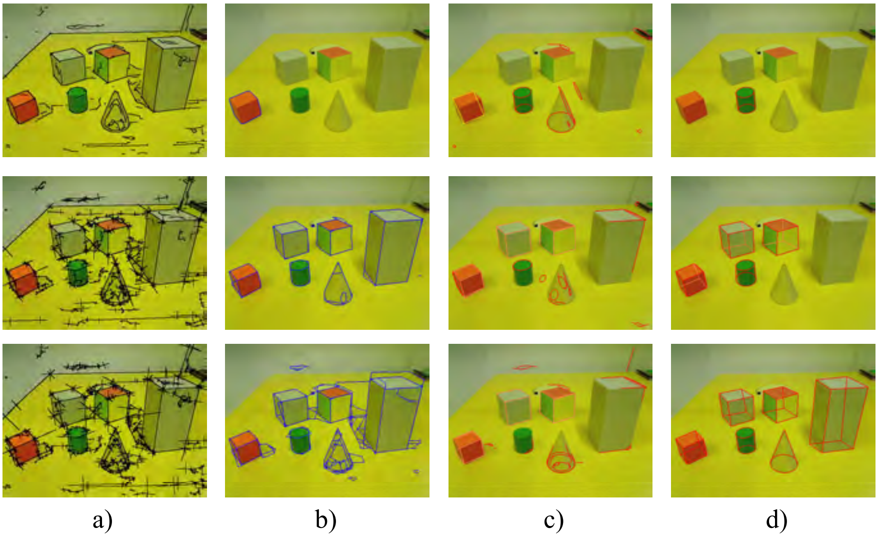


Fig. 8 Search lines (a), closures (b), the best ten lower level results (c) and simple object shapes (d) after 184ms (first row), 328ms (second row) and 468ms (third row).

typical problem of grouping, namely that shadows or image noise creates spurios features such as lines or arcs. A grouping into higher level Gestalts sometimes accidentially includes a wrong feature though, more often the Grouping principle constrain the search to actual higher level Gestalts.

With the incremental approach object detection depends on processing time. Fig. 8 shows in every row the results for different processing times. The first image (a) presents the resulting search lines, the second (b) the detected closures, the third (c) the best results of lower level Gestalts and the right column (d) the detected basic object shapes cube, cone and cylinder.

Fig. 9 shows eight images from a sequence of a robot to a playground-scene with cubes (see Fig. 1). The whole sequence consists of 148 images, where 404 cubes could be observed in all images. For the detection of some cubes the

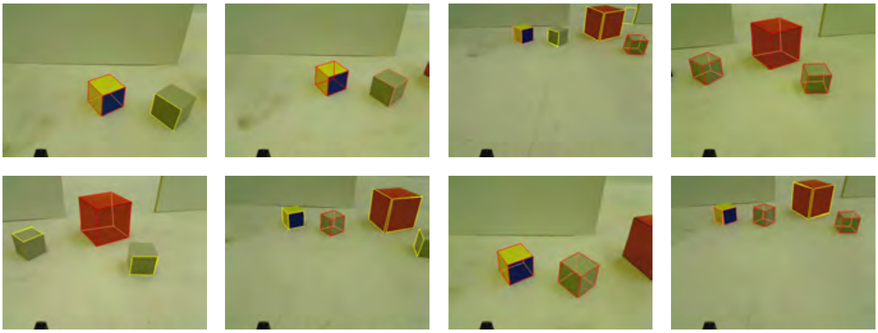


Fig. 9 Results for every 21 image from a 148 image sequence (from left to right) in a playground scene with 280ms runtime per image.

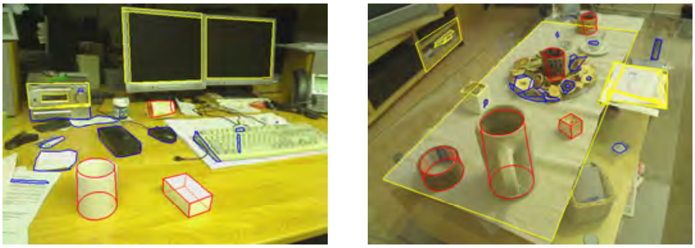


Fig. 10 Detected high-level Gestalts in an office and living room scene.

processing time was too short or the view to the object too bad to detect the cube. In these cases the highest level Gestalt detected is shown. In nearly every image is at least one rectangle or a flap of a cube visible. Hence it is fair to assume that tracking of rectangles would be sufficient to follow cubes over degenerate views. For this sequence we evaluated the cube detection rate depending on processing time. The detection rate is 46% at 220ms processing time per image, 71% at 280ms (examples shown in Fig. 9) and 82% at 350ms, calculated with an Intel Core Quad Q6600 with 2.40GHz. We can see that the detection rate grows for increased processing time as expected, but we note that increasing processing time to more than 500ms does not lead to further improvements: search lines are long enough to detect what is possible.

This clearly indicates the potential of the approach. Fig. 10 shows therefore results of real world images in an office and living room scene. While previous robot learning methods had to cope with pixel data [5] or 2D blobs [4] this could be exploited to learn from object to robot relationships in 3D in [3].

6 Conclusion and Further Work

We presented a hierarchical visual grouping system for the detection of basic geometric Gestalts such as cones, cylinders and cubes. With the incremental processing approach the problem of having parameters for each Gestalt principle is reduced to the single parameter processing time. The evaluation shows that basic shapes are detected with more than 80% detection rate. This makes it possible to follow the object motion over sequences. In [3] this has been used to learn affordances such as moveability and relate it to object size. And Fig. 10 indicates that this method is capable of extracting the outlines of objects in everyday scenes for future use in service and home robotics.

In future work it is interesting to investigate how the approach in [4] can be extended from learning affordance relations to 2D image blobs to different object shapes and their behaviour when pushed or grasped. This would also exploit the ability to calculate the relative 3D positions of the objects and also of size and orientation under the condition of knowing the mounting point of the camera to the ground plane when using a triangulation.

Presently added is shape tracking using lower-level Gestalts. With such a tracking algorithm there will be no need to observe the objects from a viewpoint where several or all sides are visible. Once a three-dimensional object is detected, it is possible to follow it, e.g., when only a rectangle or a flap of a cube is observable.

Acknowledgments. The work described in this article has been funded by the European Commission's Sixth Framework Programme under contract no. 029427 as part of the Specific Targeted Research Project XPERO ("Robotic Learning by Experimentation").

References

1. Gibson, J.: The Ecological Approch to Visual Perception. Hougton Mifflin, Boston (1979)
2. Zabkar, J., Bratko, I., Mohan, A.: Learning Qualitative Models by an Autonomous Robot. In: 22th International Workshop on Qualitative Reasoning, pp. 150–157 (2008)
3. Zabkar, J., Bratko, I., Jerse, G., Prankl, J., Schlemmer, M.: Learning Qualitative Models From Image Sequences. In: 22nd International Workshop on Qualitative Reasoning, pp. 146–149 (2008)
4. Montesano, M., Lopes, A., Bernardino, J., Santos-Victor, J.: Learning Object Affordances: From Sensory-Motor Coordination to Imitation. IEEE Transactions on Robotics [see also IEEE Transactions on Robotics and Automation] 24(1), 15–26 (2008)
5. Kuipers, B., Beeson, P., Modayil, J., Provost, J.: Boostrap learning of foundational representations. Connection Science 18.2; special issue on Developmental Robotics (2006)
6. Modayil, J., Kuipers, B.: Bootstrap Learing for Object Discovery. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 742–747 (2004)
7. Biederman, I.: Recognition-by-Components: A Theory of Human Image Understanding. Psychological Review 94(2), 115–147 (1987)
8. Lowe, D. G.: Three-dimensional object recognition from single two-dimensional images. Artificial Intelligence 31(3), 355–395 (1987)
9. Dickinson, S., Bergevin, R., Biederman, I., Eklund, J., et al.: Panel report: The potential of geons for generic 3-d object recognition. Image and Vision Computing 15(4), 277–292 (1997)

10. Sarkar, S., Boyer, K.L.: A computational structure for preattentive perceptual organization: Graphical enumeration and voting methods. IEEE Transactions on System, Man and Cybernetics 24(2), 246–266 (1994)
11. Zillich, M.: Making Sense of Images: Parameter-Free Perceptual Grouping. Ph.D. Dissertation, Technical University of Vienna (2007)
12. Zillich, M., Matas, J.: Ellipse detection using efficient grouping of arc segments. In: 27th Workshop of the Austrian Association for Pattern Recognition AGM/AAPR, pp. 143–148 (2003)
13. Carlbom, I., Paciorek, J.: Planar Geometric Projections and Viewing Transformations. Computing Surveys 10(4), 465–502 (1978)

3D Collision Detection for Industrial Robots and Unknown Obstacles Using Multiple Depth Images

Markus Fischer and Dominik Henrich

Abstract. In current industrial applications without sensor surveillance, the robot workcell needs to be rather static. If the environment of the robot changes in an unplanned manner, e. g. a human enters the workcell and crosses the trajectory, a collision could result. Current research aims at relaxing the separation of robot and human workspaces. We present the first approach that uses multiple 3D depth images for fast collision detection of multiple unknown objects. The depth sensors are placed around the workcell to observe a common surveilled 3D space. The acquired depth images are used to calculate a conservative approximation of all detected obstacles within the surveilled space. Using a robot model and a segment of its future trajectory, these configurations can be checked for collisions with all detected obstacles. If no collision is detected, the minimum distance to any obstacle may be used to limit the maximum velocity. The approach is applicable to a variety of other applications, such as surveillance of tool engines or museum displays.

1 Introduction

In current industrial applications, human and robot workspaces are completely separate and it is forbidden for them to share a common workspace. When a human enters the robot workcell, additional safety systems must ensure the safety of the human. There are many applications for coexistence of robot and human. Robots would require less space if no physical barriers are needed to isolate them and a technician could perform maintenance of other machines in the workcell. The first commercial product using image processing sensors is available [15]. This product checks for intrusion in virtual safety zones around robots and dangerous equipment.

As the main objective of such surveillance, collisions between the robot and any obstacle must be avoided, as these could lead to severe injuries if the robot

---

Markus Fischer · Dominik Henrich
Lehrstuhl für Angewandte Informatik III, Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth, Germany,
e-mail: `{markus.fischer,dominik.henrich}@uni-bayreuth.de`

collides with a human. The second objective is velocity control based on the proximity of the robot to any obstacle. By limiting its current maximum velocity, the robot can always be stopped before it encounters an obstacle.

The approach introduced here is the first that aims to use multiple depth sensors for online collision detection. Since depth sensors already provide volumetric data, no correspondence problem has to be solved, as is the case with raw data from stereo cameras. The sensors are placed around the workcell and observe a common space called the *surveilled space*. In each collision detection cycle, the sensors acquire new, synchronised depth images. An approximate and conservative representation of all objects within the surveilled space is computed using these images. With a geometric (CAD) model of the robot and the current robot configuration, it is possible to identify the robot as one of the objects, and the remaining objects are classified as obstacles for the robot. For a given segment of the future trajectory, the robot model may then be adjusted and checked for collisions with the obstacles. One way to avoid the potential collision is to decelerate and stop the robot [17]. It may then proceed with its trajectory if no collision is detected in succeeding collision detection cycles. Another way to avoid the potential collision is to calculate a new trajectory for the robot [4]. As long as no collision is detected, the proximity can be used to limit the maximum velocity [9, 10].

In summary, the contribution of this paper is a fast reconstruction of the entire robot environment, including multiple unknown objects, of multiple depth images for use in an online surveillance system.

2 State of the Art

This work is related to reconstruction in computer graphics, to human detection in computer vision, and to surveillance in safety systems.

In computer graphics, the visual hull concept [13] uses object silhouettes in various colour images. The silhouettes define cones containing the object and the object volume is defined by the intersection of these cones. In [12], a stereo algorithm is used to improve the volumetric approximation of the visual hull. The presented system calculates 4 frames per second *fps* for a scene that contains a single object.

A mesh-based CAD model of an object based on multiple depth images provided by laser scans from different perspectives is presented in [16]. Merging these images results in a precise CAD mesh model. Constructing the mesh and merging each perspective at a resolution of 110x128 *pixels* requires about 6 *min*.

These algorithms target reconstruction of single objects in empty space and are not suitable for fast fusion of multiple depth images containing multiple objects to obtain polyhedral representations.

In computer vision, several approaches for human detection are presented [7]. Most of those approaches use models for object recognition, but object models limit the capability to detect arbitrary unknown objects.

In safety systems, existing strategies provide 1 ½D, 2 ½D or 3 D surveillance, depending on the type and number of sensors used.

In [17] a single laser range finder acquires 1½D distance information within a plane just above the floor of the workcell. Any dynamic obstacles detected are assumed to be standing humans and are approximated by a vertical cylinder. The smallest distance between this cylinder and the robot limits the maximum robot velocity. However, other obstacles, humans in a stooping posture or humans behind the detected individual may not be correctly approximated.

In [18, 19] a tri-ocular camera system acquires colour images of the shared workspace from above. The human's hands and neck are detected based on the characteristic colour and texture features of the skin. The minimum distance between human body parts and the robot is used to limit the maximum robot velocity. This system was extended with a tri-ocular stereo vision system acquiring 2½D information from above. It remains unclear whether this method can cope with larger work pieces or obstacles other than exposed human body parts.

[15] calculates 2½D data about the surveilled space from a single stereo image and checks for intrusions into predefined safety zones. Entry into such a safety zone leads to deceleration or stoppage of the robot. During active surveillance, these zones are static and cannot be changed.

[21] presents an approach for 2½D surveillance of a robot workspace with a single time-of-flight camera. A robot model is calculated and suppressed in the depth image and a safety zone for each robot link is defined and checked for intersection with the depth image. A collision detection cycle is about 200 *ms*, with additional time necessary for acquiring the depth images.

All of the above systems use multiple colour cameras or a single depth sensor, but none of them involve full 3D surveillance by means of depth sensors.

This work extends now the SIMERO system [3, 6], which provides global surveillance of a robot workcell with multiple colour cameras and collision detection in 3D. By means of pixel classification [4], dynamic unknown objects are distinguished from the workspace background. Since no depth information can be obtained from colour images, only deviations from reference images can be used to detect dynamic unknown objects.

3 Problem Description

In this section, the theoretical background of the introduced surveillance approach is described.

The *visible volume* V of a sensor is defined as part of the space within the sensor's aperture angles. Here, multiple depth sensors observe a common volume of the three-dimensional workspace (Fig. 1a). The intersection of their Vs is called the *surveilled space* S. All collision detection results obtained are respective to S.

Each *image pixel* (x,y) describes a straight line $l_{x,y}$ from the *focal point* f of the sensor into V. This $l_{x,y}$ is provided by the sensor calibration. In every *time step* t, the depth sensor measures the distance to the nearest obstacle along $l_{x,y}$. The *point* $p_{x,y}(t)$ is defined as the first intersection of $l_{x,y}$ with this obstacle's surface. For each t, the depth image is a matrix of all distance values of the (x,y). Adjacent pixels in both image directions ($l_{x,y}$, $l_{x+1,y}$, $l_{x,y+1}$, $l_{x+1,y+1}$) define a *sub-volume* $V_{x,y}$ of V. The *mesh* $m_{x,y}(t)$ is a triangulated surface with $p_{x,y}(t)$ as vertices within $V_{x,y}$. Assuming that the angle between adjacent rays is small enough, none of the surface within $V_{x,y}$ will differ significantly from the real object's surface. $m_{x,y}(t)$ divides $V_{x,y}$ into two parts. All points in $V_{x,y}$ sharing a side with f are empty space $E_{x,y}(t)$, all other points in $V_{x,y}$ are occluded space $O_{x,y}(t)$. The union $m(t)$ of all $m_{x,y}(t)$ represents the measured surface within V and divides V into $E(t)$ and $O(t)$ (Fig. 1c). V is clipped to the volume of S (Fig. 1d). Adjacent portions of $O(t)$ within S are classified as detected objects $o_j(t)$ with $O(t) = o_0(t) +\ldots+ o_l(t)$ (Fig. 1f).

These definitions must be extended for a multi-sensor system. The index i of the current sensor is added to the parameters of a set as a superscript, e.g. V^i as V of the i^{th} sensor. The space surveilled by multiple depth sensors is determined by intersecting the V^i:

$$S = \bigcap_i V^i = ([\bigcup_i E^i(t)] \cap [\bigcap_i V^i]) \cup [\bigcap_i O^i(t)] \quad (1)$$

This is a lossless representation of the complete content of S. While the union of all $E^i(t)$ leads to a conservative approximation of empty space, the intersection of all $O^i(t)$ produces a conservative approximation of present objects (Fig. 1h).

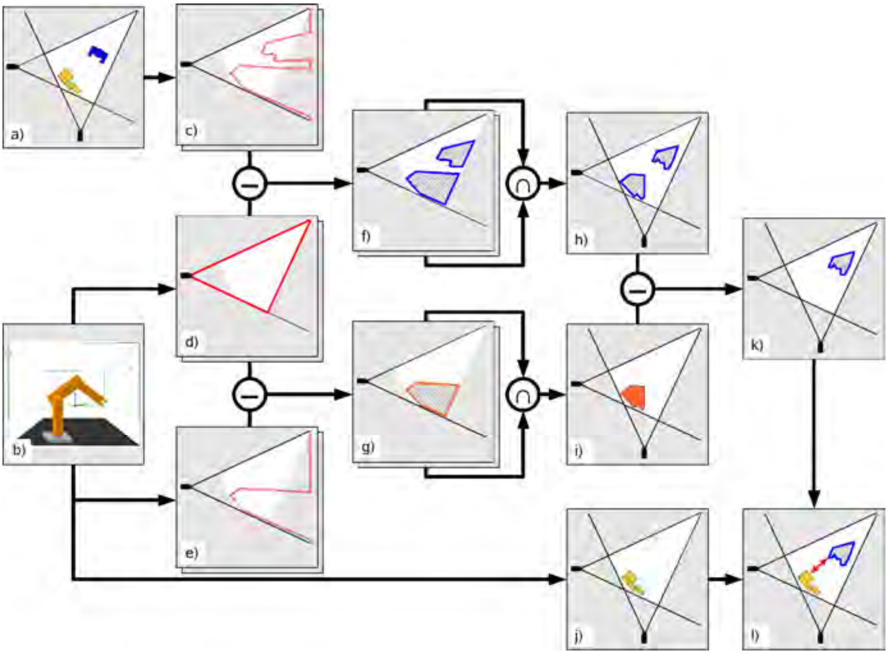


Fig. 1 Overview of the complete surveillance cycle viewed from above. Using the current depth images (c) and data about the robot (b), the minimum distance between the robot and any detected unknown object can be calculated (l).

Known objects should not be checked for collision with the *safety zones* $Z(t) = z_0(t) + \ldots + z_l(t)$, but are also detected in the depth images. A model database provides geometric data for known objects, e. g. a robot, and safety zones, e. g. an expanded robot model, at any time (Fig. 1b). Virtual depth images of an empty space containing only known objects (Fig. 1e) produce virtual detected objects $k_m(t)$ with $K(t) = k_0(t) + \ldots + k_l(t)$ (Fig. 1g). Their intersection (Fig. 1i) is a subset of $O(t)$ from real depth images and results in the *detected unknown obstacles* $O(t)\backslash K(t)$ within S (Fig. 1k). For each t, the $Z(t)$ are defined (Fig. 1j) and the intersection of $Z(t)$ with $O(t)\backslash K(t)$ results in the *detected collisions* (Fig. 1l):

$$C(t) = Z(t) \cap (O(t) / K(t)) \tag{2}$$

4 Fast Collision Detection Algorithm

Initially, the algorithm contains the following assumptions: (a) The algorithm is initialised with a user-determined number of depth sensors. (b) All of these sensors are calibrated. (c) The CAD model of the robot and its complete trajectory are known. (d) The number of configurations to be checked for collisions is user-determined. (e) S may include multiple unknown objects and additional objects may enter or leave S at any time. (f) The position, geometry and orientation of these objects may change over time.

The presented algorithm implements the collision detection defined in the previous section and consists of three computational steps:

A. Calculate the $o_j(t)$ within S by the fusion of all current depth images.
B. Calculate the $k_m(t)$ within S to suppress them in the collision detection.
C. Calculate the $z_n(t)$ and their minimum distance to any unknown obstacle.

The first two steps must be done once per surveillance cycle. The last step must be done once for each future robot configuration to be checked for collision.

4.1 Fusion of Depth Image Data

The first step is to calculate the objects in each depth image. The algorithm uses the bounding box of S within the workspace as the clipping volume for V^i. This is a conservative approximation of the portion of V^i within S. Adjacent portions of the $m^i(t)$ within this bounding box are segmented. The $o^i_j(t)$ are the result of a convex hull computation [1] for all segmented parts of $m^i(t)$. The convex hull is a conservative approximation of these detected objects. In computational geometry, efficient algorithms for convex hulls are presented. Since real sensor values include error pixels with incorrect distance values, one characteristic of the convex hull is very useful; if a single point is measured behind the real surface, it is "absorbed", since it lies within the convex hull. If a single point is measured above the real surface, the convex hull is extended by this point. This is a conservative approximation of the convex hull of the real surface.

To get the space occupied by all $o_j(t)$, the $o^i_j(t)$ of all depth images have to be intersected. Since this is computationally expensive and most intersections of $o^i_j(t)$

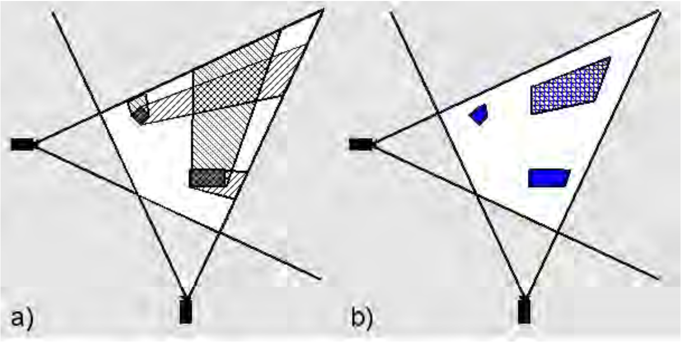


Fig. 2 Illustration of a pseudo-object calculation. *S* (white area) contains two real objects. Both depth sensors detect two objects $o^i_j(t)$ (a). The intersection of the $o^i_j(t)$ (b) results in real objects (blue) and a pseudo-object (blue hatched) that does not necessarily represent a real object.

from different real objects results in empty volumes, only the intersection of their bounding boxes is calculated initially and is then iterated for all cameras. The result of this computation is a list of bounding boxes for all possible objects. The detected objects within the non-empty bounding boxes are then intersected, resulting in the $o_j(t)$.

A non-empty intersection of $o^i_j(t)$ from different real objects can produce a virtual object with no correspondence in reality (Fig. 2). This virtual object is denoted a *pseudo-object*. Since only depth information is used to calculate the occupied space, this object is indistinguishable from a real object in *S*. No depth sensor has information about this portion of *S*. The pseudo-object may contain no or one or more real objects. For a conservative approximation in *S*, pseudo-objects [14] have to be treated as real objects.

4.2 Space Occupation of Known Objects

Since the number of sensors is limited, they cannot measure the entire surface of all known objects. Given the limited number of perspectives, the known objects must be approximated and some areas of *S* adjacent to known objects cannot be seen by any sensor and therefore produce obstacles during collision detection. Thus, virtual depth images containing only the $o^i_j(t)$ of known objects are generated and their intersections $k_m(t)$ are used instead of the real known object geometry. In the current implementation, this is done for the robot model and the current robot configuration. Additional known dynamic objects in *S* such as conveyor belts could be included in the same way, too.

4.3 Safety Zone Calculation and Collision Detection

The polyhedrons calculated in step A and B are used to calculate the minimum distance between the $z_n(t)$ and the detected unknown objects within *S*. A minimum distance of zero indicates a collision.

For the robot configuration to be checked for collisions, the volume occupied by the robot model is calculated. If objects other than the robot itself have to be checked for collisions with detected unknown objects, additional $z_n(t)$ can be defined. As with the robot model, these $z_n(t)$ may change their geometry in each new collision detection algorithm cycle, but in the current implementation, only the robot itself is used as a safety zone.

The detected unknown objects are the subset of detected objects excluding known objects. However, the difference operator results in non-convex polyhedrons. An intersection volume of $z_n(t)$ and $o_j(t)$ outside the $k_m(t)$ indicates a detected collision $C(t) \neq \emptyset$.

The collision detection step must iterate over all $z_n(t)$ and $o_j(t)$. The loop can be stopped if a collision is detected or continued with the next iteration if the distance [2] is greater than zero. If the distance between $z_n(t)$ and $o_j(t)$ is zero, there is an intersection and its volume is calculated by intersecting convex polyhedrons. For all $k_m(t)$, this intersection volume is checked for complete inclusion. Since the intersection volume and all $k_m(t)$ are convex, it is sufficient to check all vertices of the intersection volume for inclusion in the $k_m(t)$. There are three possible results: All vertices are included, none of them are included, or there are vertices both inside and outside of $k_m(t)$. If all vertices are included in $k_m(t)$, the volume is within the known objects and is assumed not to be an unknown obstacle and no collision is detected. Otherwise, the intersection volume has to be a detected collision, the minimum distance is set to zero and the loop is stopped. The minimum distance and the indices of $z_n(t)$ and $o_j(t)$ are then reported to the control program.

5 Experimental Results

The presented algorithm was tested in both a simulation system and an experimental setup.

5.1 Simulation Results

The simulation system is used with a virtual robot workcell. Its geometry is user-determined and was set during testing to the size of the robot workcell used for the prototype experimental setup. The simulated workcell includes a stationary robot model and additional defined objects such as humans or tables, which are all represented as polyhedrons. Up to six unknown objects were used during experimentation. A pinhole model is used to simulate the depth sensors and their number, position, orientation, resolution and aperture angles are user-determined. Up to 8 depth sensors were simulated in an experiment.

There is no structural difference between simulated and real depth images. The only differences are due to noise in measurements made with real sensors. With these simulated depth images, the presented surveillance algorithm can be applied.

The system was implemented in C++, compiled with gcc version 4.1.0 and optimised with the –O3 flag running on SUSE Linux 11 (i 586). A standard CPU (AMD Sempron 3000+, ~ 2GHz, 512 KB Cache) and 512 MB RAM were used.

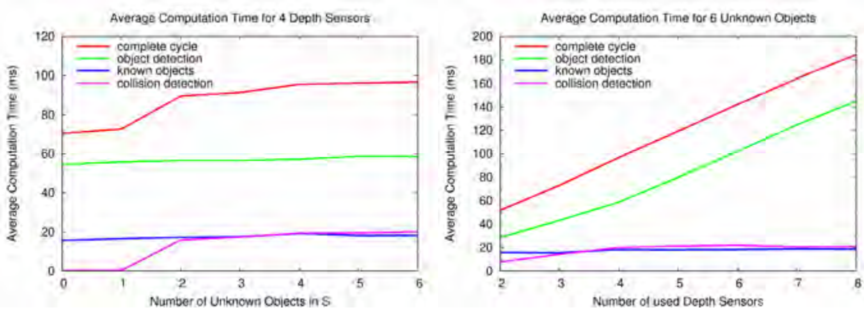


Fig. 3 Diagrams showing the average computation time for calculating a sequence of 1500 collision detection cycles. The algorithm was tested for an increasing number of unknown objects in S (left) and for an increasing number of depth sensors (right).

The average computation time for a sequence of 1500 frames containing the simulated workcell and an increasing number of objects within S and an increasing number of implemented depth sensors was calculated (Fig. 3).

As shown in Fig. 3 left, the number of objects within S has little influence on the surveillance cycle time. The object detection step calculation time directly depends on the number of objects detected by each depth sensor. However, most intersection-bounding boxes produced by $o^i_j(t)$ from different real objects in S are empty and are thus removed within the calculation. The intersection of bounding boxes itself is not expensive, leading to fast computation for an arbitrary number of objects. The collision detection computation time also directly depends on the number of detected objects. The performance for increasing numbers of objects is important, since robot workspaces generally contain multiple detected objects.

The number of depth sensors implemented is significant for use in real applications. As the number of perspectives available increases, the frequency of occlusions within S leading to pseudo-objects or small minimum distances between robot and detected objects will decrease. Since the fusion of depth images directly depends on the number of depth sensors, calculation time for object detection significantly increases proportionally to the number of depth sensors (Fig. 3) but the number of depth sensors does not influence the other computational steps.

5.2 Experimental Results

The prototype system is implemented as a master-slave architecture, with all data processing steps done on the master. Each depth sensor has an own slave, which only acquires the current depth images and transfers them to the master. This is done via a local area network.

The prototype setup is built in a workcell containing a Stäubli RX130 robot (Fig. 4, left). The CAD model of the robot is known and a direct connection between the robot control and the master is established. At any time, the current robot configuration can be requested from the robot controller and a velocity for the current trajectory segment can be sent to the controller.

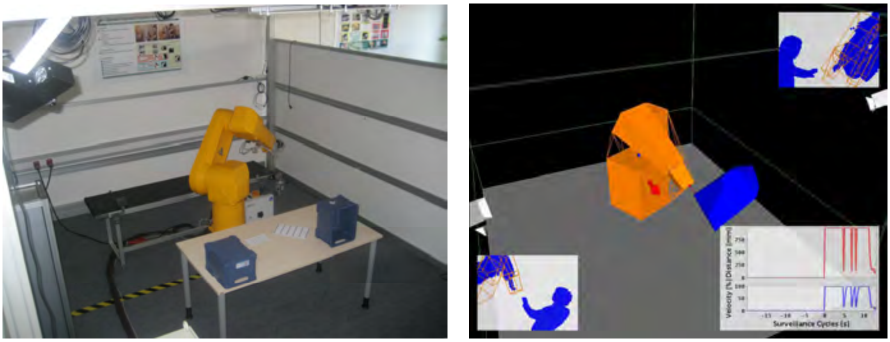


Fig. 4 Left: Prototype setup in a workcell containing an industrial robot and PMD cameras placed in the upper corners of the workcell. Right: Visualisation of the minimum distance calculation between robot (orange) and detected unknown objects (blue). The small images display the results of the object detection in the depth images. The fused objects are shown as 3 D representations. The minimum distance (red line) is used for velocity control (diagram).

Two PMD cameras [11] are used in the current configuration of the prototype setup. The model Vision 19k used has a resolution of 120x160 *pixels* and an aperture angle of 40°. Since PMD cameras are active sensors, each of them must use a different modulation frequency. Multiple PMD cameras using a single modulation frequency would cause interferences within the measured depth images. The shutter time is user-controlled and was set within an interval from 25 to 50 *ms*. Sensor noise increases with decreasing shutter time. Up to 15 *fps* can be provided by the camera to its slave using Firewire.

To obtain lateral calibration of the PMD cameras, a standard algorithm was used [20]. For depth calibration, constant offsets dependent on the modulation frequencies were calculated for the measured correspondences. A mean deviation of two pixels in lateral coordinates and 25 mm in depth values was the result.

Due to the variation in the measured values, the quality of the depth images needed to be improved with filters. In the prototype system, averaging filters were applied to the raw depth images and morphological filters (erosion, dilation, open, close) were used on the segmentation images. By adjusting filter parameters, the quality of the depth measurement and object detection was increased significantly. The best results were achieved for an 3x3 averaging filter and an 3x3 Open filter.

The detected objects were calculated for these depth images. The robot model (consisting of one cube per robot link) for the current robot configuration was generated and used in the known object calculation. In the 3D representation (Fig. 4, right), the model itself is displayed as an orange volume and the calculated known object as an orange mesh. The robot model generated was used as a safety zone for the collision detection step by calculating the minimum distance between the robot and the nearest detected unknown objects. The minimum distance measured was used to limit the maximum velocity of the robot (Fig. 4, right). A logarithmic mapping determined the velocity limitation for the current distance. The maximum acceleration since the last algorithm cycle is limited, too.

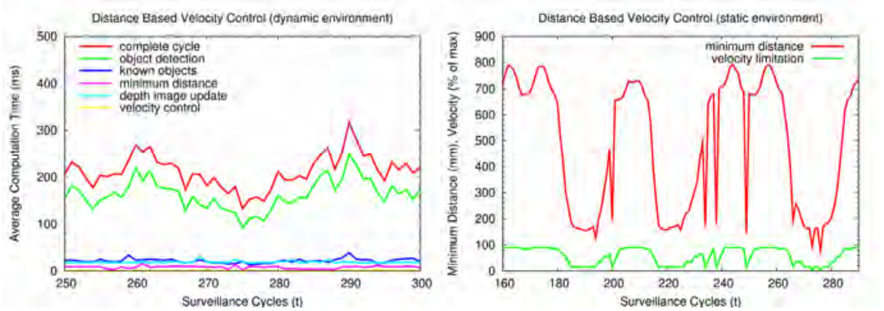


Fig. 5 Diagrams showing a sequence of logged surveillance cycle data from experiments in a static and dynamic robot environment with two depth cameras. The calculation times for each computational step and the complete cycle time are displayed (left). The minimum distance measured is used for simple velocity control (right).

The diagrams in Fig. 5 show logged sequences of surveillance cycle data from different experiments. The robot program loop was a circular movement between both ends of the conveyor belt. In the static environment, some boxes were placed on the table in front of the robot (Fig. 4, left). In the dynamic environment, a human entered *S*, left it and also placed boxes on the table or removed them.

Of all computational steps in the surveillance cycle, object detection consumes the majority of the algorithm cycle time. All other steps are almost constant. The measured distances in the right-hand diagram show that the repeated accuracy of measurements is within an interval of centimetres. It also shows that because of measurement errors, single frames lead to minimum distances much smaller than the real distances due to appearance of a small pseudo-object near the robot base.

6 Conclusions

This paper introduced the first approach for online collision detection of industrial robots with multiple depth cameras. The main contribution is fast, 3D reconstruction of robot environments containing multiple unknown objects. Previous surveillance and collision detection approaches use multiple colour cameras or only one depth sensor. Using an integrated robot model, a collision detection algorithm is applied to the space occupied by detected unknown objects within the surveilled area. Minimum distances between the robot and any detected unknown object may be used for velocity reduction or trajectory adjustment.

The main advantage over image-based surveillance systems like the current SIMERO system or SafetyEye is the high independence from external light sources. Dynamic lighting conditions or changes in object texture within the surveilled space cause fewer problems. The latter system also does not provide dynamic safety zones and require complete separation of humans and robots.

The algorithm introduced is usable with an arbitrary number of depth sensors. The current version should be used with a small number of sensors for applications with stringent time limitations, since the cycle time is directly dependent on the number of sensors used. The algorithm can be adapted to a variety of other surveillance and security applications, depending on a redefinition of the safety zones applied. For instance, in museum surveillance this can be a static safety zone around the displays, resulting in an intrusion or proximity alert.

Positioning of the depth sensors is important for the quality of results, because with few sensors, an object located between sensor and robot can lead to computed distances much smaller than actual distances. With optimised positioning, the robot and unknown objects can be separated in at least one depth image.

Further research will address automatic positioning of the depth sensors around the surveilled space. The quality of approximation is strongly related to an optimal number and position for variable geometry of robot workspaces.

References

1. Barber, C. B., Dobkin, D. P., Huhdanpaa, H. T.: The Quickhull algorithm for convex hulls. ACM Transactions on Mathematical Software 22(4), 469–483 (1996), `http://www.qhull.org`
2. Cameron, S.: Enhancing GJK: Computing Minimum and Penetration Distances between Convex Polyhedra. In: IEEE International Conference on Robotics and Automation (April 1997)
3. Ebert, D.: Bildbasierte Erzeugung kollisionsfreier Transferbewegungen für Industrieroboter. Schriftenreihe Informatik, Band 12 (2003) ISBN 3-936890-23-4
4. Gecks, T., Henrich, D.: Multi-Camera Collision Detection allowing for Object Occlusions. In: 37th International Symposium on Robotics (ISR 2006) / 4th German Conference on Robotics (ROBOTIK 2006) (2006)
5. Gecks, T., Henrich, D.: Path Planning and Execution in Fast-Changing Environments with Known and Unknown Objects. In: IEEE International Conference on Intelligent Robots and Systems (2007)
6. Henrich, D., Gecks, T.: Multi-camera collision detection between known and unknown objects. In: IEEE International Conference on Distributed Smart Cameras (2008)
7. Hu, W., Tan, T., Wang, L., Maybank, S.: A survey on visual surveillance of object motion and behaviors. IEEE Transactions on Systems, Man and Cybernetics 34(3), 334–352 (2004)
8. Ingensand, H., Kahlmann, T.: Systematic investigation of properties of PMD-sensors, 1st Range Imaging Research Day (2005)
9. Kuhn, S., Henrich, D.: Modelling intuitive Behaviour for Safe Human/Robot Coexistence and Cooperation. In: IEEE International Conference on Robotics and Automation (2006)
10. Kuhn, S., Henrich, D.: Fast Vision-Based Minimum Distance Determination Between Known and Unknown Objects. In: IEEE International Conference on Intelligent Robots and Systems (2007)
11. Kraft, H., et al.: 3D-Camera of High 3D-Frame Rate, Depth Resolution and Background Light Elimination Based on Improved PMD (Photonic Mixer Device)-Technologies, OPTO 2004, AMA Fachverband, `http://www.pmdtec.com/`
12. Li, M., Schirmacher, H., Magnor, M., Seidel, H.-P.: Combining Stereo and Visual Hull Information for On-Line Reconstruction and Rendering of Dynamic Scenes. In: IEEE Workshop on MMSP, pp. 9–12 (2002)
13. Matusik, W., Buehler, C., McMillan, L.: Polyhedral Visual Hulls for Real-Time Rendering. In: Proceedings of the 12th Eurographics Workshop on Rendering, pp. 116–126 (2001)

14. Meisel, A.: 3D-Bildverarbeitung für feste und bewegte Kameras., Vieweg Verlag, Reihe Fortschritte der Robotik Nr. 21 (1994)
15. Patent DE 10 2006 057 605 A1 Verfahren und Vorrichtung zum Überwachen eines dreidimensionalen Raumbereichs, Pilz GmbH & Co. KG (2006), `http://www.safetyeye.com/`
16. Reed, M.K., Allen, P.K.: 3-D Modeling from Range Imagery: An Incremental Method with a Planning Component. Image and Vision Computing 17(2), 99–111 (1999)
17. Som, F.: Sichere Steuerungstechnik für den OTS-Einsatz von Robotern. In: 4.Workshop für OTS-Systeme in der Robotik, IPA 2005 (2005)
18. Thiemermann, S.: team@work - Mensch/Roboter-Kooperation in der Montage. In: 2.Workshop für OTS-Systeme in der Robotik, IPA 2003 (2003)
19. Thiemermann, S.: Direkte Mensch-Roboter-Kooperation in der Kleinteilmontage mit einem SCARA-Roboter. IPA-IAO-Bericht (2005) ISBN 978-3-936947-50-2
20. Tsai, R.Y.: An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision. In: IEEE Conference on Computer Vision and Pattern Recognition (1986)
21. Winkler, B.: Safe Space Sharing Human-Robot Cooperation Using a 3D Time-of-Flight Camera. In: International Robots and Vision Show (2007)

Reducing Motion Artifacts in Mobile Vision Systems via Dynamic Filtering of Image Sequences

Christoph Walter, Felix Penzlin, and Norbert Elkmann

Abstract. Artifacts like motion blur are a common problem for vision systems on mobile robots, especially when operating under low light conditions. In this contribution we present a system that increases the average quality of camera images processed on resource-constrained mobile robots. We show a solution for estimating the magnitude of motion artifacts for every element of a continuous stream of images using data from an inertial measurement unit. Taking estimated image quality into account we describe an effective solution for congestion control between acquisition and processing modules. We build that upon a middleware that supports flexible flow control at a per-image level.

Keywords: robot vision, motion blur, congestion control, mobile robot.

1 Introduction

While the presence of motion artifacts in images from moving cameras can also be exploited in several ways, it is usually a troublesome effect. Objects may become unrecognizable because of blur; self-localization may yield poor results because of geometric distortion, and so on. At the same time, image processing tasks usually require significant resources and may easily exceed the capabilities of the computer hardware present on a mobile robot. In the following sections we describe our approach to lessen the effects of both problems. At first we discuss motion artifacts in more detail. After discussing related work we present our data processing scheme including approaches to image quality estimation and congestion control. We also present results achieved with our system.

Christoph Walter · Felix Penzlin · Norbert Elkmann
Fraunhofer Institute for Factory Operation and Automation, Sandtorstraße 22,
39106 Magdeburg, Germany,
e-mail: {`christoph.walter,felix.penzlin,`
`norbert.elkmann`}`@iff.fraunhofer.de`

2 Motion Artifacts

Cameras acquire images by exposing a light-sensitive element for a given period of time. Camera movement while the sensor is exposed may result in a number of image artifacts. The type and intensity of these artifacts depend on the exposure time as well as on several other parameters of the camera. In this section we discuss a number of common artifacts, in particular motion blur and geometric distortion. We use a pinhole camera model. Lens distortion is considered to have a negligible impact and is therefore not modeled here.

2.1 Motion Blur

Motion blur can be induced by moving either objects in the camera's field of vision or the camera itself. For simplicity we consider only a static scene and disregard any moving objects.

2.1.1 Translation of Camera

We distinguish between two kinds of camera movement. On the one hand there is translation in direction of the optical axis; on the other hand there is motion in the plane orthogonal to that axis. In the second case, the magnitude of blur b on the image sensor for an object is in inverse proportion to the distance to the camera plane d_{cam} (See Eqn. 1).

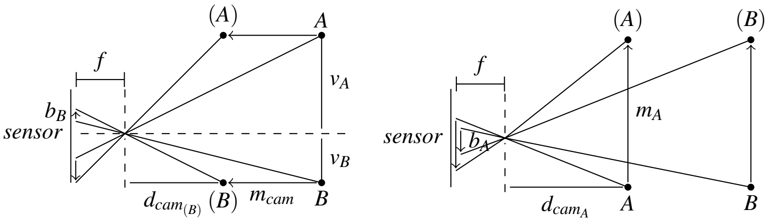


Fig. 1 Motion blur in case of vertical or horizontal translation (left) and translation parallel to the optical axis (right).

$$b = \frac{1}{d_{cam}} * m * f \tag{1}$$

For movements parallel to the optical axis the intensity of blur b for an object depends on its distances from line of view v and camera plane and the displacement d_{cam}. For a point at the optical axis, this kind of translation has no impact (See Fig. 1). If objects are relatively far away from the camera, translation becomes insignificant.

2.1.2 Camera Rotation

When rotating the camera, the magnitude of blur also depends on the position of a given object relative to the optical axis. Figure 2 shows that such a camera rotation results in a blur *b* roughly perpendicular to the axis of rotation. Its strength depends on the actual angle of rotation and on the angle between the rotation axis and the view direction. The distance to an object does not matter for rotational blur.

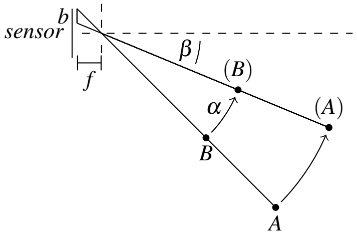


Fig. 2 Motion blur in case of rotation.

2.2 Distortion

Geometrical image distortion is another common artifact that can be found with moving cameras. It occurs when different portions of the image sensor are exposed sequentially to light. This mode is called “rolling shutter” and is implemented in various CMOS-based cameras.

Here, different phenomena can be observed. A sudden change in illumination may influence only portions of the image. If the camera is moved horizontally or rotated around the vertical axis, skew can be observed. Vertical lines appear to lean to the left for moving to the left or the right side for the opposite direction of movement. Vertical movements as well as rotations around the horizontal axis result in stretching respectively shrinking of objects vertically. Altering the direction of movement at a high speed (in case of vibrations) is called “wobble”. When rotating the camera around the optical axis, straight lines get bent around the image center.

3 Related Work

Researchers have dealt with techniques to prevent, detect and remove motion artifacts in the past. There are also several commercial solutions available that implement such techniques. Instead of removing motion artifacts it would be beneficial to avoid them all together. The first approach means compensating the motion of the camera. Here, special hardware with accurate actuators is required. One solution is to stabilize the whole camera on a special platform, as shown in [11]. Similar platforms are available for cine camera mounted on helicopters or cars to countervail vibrations actively. Other solutions are shiftable image sensors ([13], [1], [2]) that counteract camera shake. Therefor an inertial sensor captures the movements of the camera to compute the appropriate shift. Instead of a shiftable sensor other agile optical components can be used like a floating lens element, which is utilized in some digital SLR cameras, a variable fluid prism [10] or a movable mirror [7]. A simple solution for hand-held cameras tries to delay image exposure in case of camera shaking, which is determined using acceleration sensors.

A different class of solutions does not circumvent motion artifacts during acquisition, but tries to undo artifacts at a later point. For instance one approach ([4]) merges blurred and sharp but under-exposed images of the same scene to obtain an overall improved image. A number of algorithms for global shutter ([8], [6]) and for rolling shutter ([3], [9]) cameras have been developed. In general, a correction in software is time-consuming and consists of two steps. In the first step the artifacts are identified in a second step these artifacts are removed from the image. Another method for dealing with motion artifacts is to use redundant information sources. Using this method is highly application dependent. For example: The authors of ([12]) have used data from inertial sensors in case of fast and camera-data in case of slow or no motion when tracking motion of humans.

4 Data Processing Scheme

While researchers made various efforts to deal with motion artifacts, many of them are not well suited for mobile robot applications. Evaluating the content of individual images is time-consuming. Undoing blur is even more computationally expensive and yields mediocre results in real life. Adaptive triggering of image acquisition depending on current camera movement is a promising and computationally inexpensive approach. However, it is a disadvantage that a sudden increase in camera movement during exposure cannot be predicted.

In our system, we chose to continuously acquire images as well as motion data and apply a selection process at a later stage. One advantage is that the actual movement of the camera during image exposure is known. Another advantage is that potential 'bad' pictures are not prevented but may be used in case of adverse camera movement for a prolonged period of time. We achieve such behavior by introducing an acceptance test for individual images that incorporates not only estimated image quality but also a system load indicator.

In this section we discuss our solution for the estimation of the image quality by predicting image artifacts through evaluation of angular rates of rotation of the robot. We also describe our approaches to system architecture as well as congestion control when using time-consuming image processing algorithms as data consumers.

4.1 Data Flow and Flow Control

We based the implementation of the image filtering approach on our existing CORBA-based [5] software architecture. Its main focus is on data flow and distributed processing. In our system, an application is constructed by combining software modules that provide functionality to other modules via a generic interface.

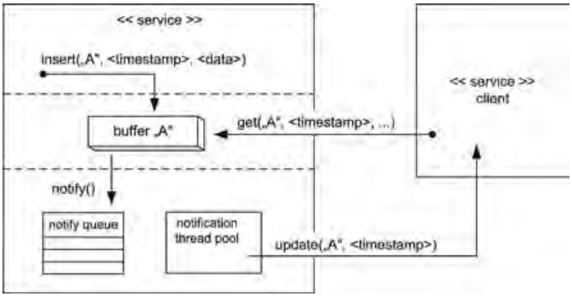


Fig. 3 A service provider offers data to clients via named buffers.

The modules are loosely coupled and may be arbitrarily distributed over a number of computers connected by a network. We call such a module a "service". Modules that only access functionality offered by other services are called "service-clients".

The interface of the services is data-centric. It allows services to offer data via named buffers to other services or clients. Each buffer represents a time series of specific data objects (See Fig. 3). A service can represent a data source, a processing module or a data consumer. Services that represent sensors or sensor systems may provide data that contains a single measurement, a time series of measurements, a vector, an image, or any other type of data that is generated by the sensor system. The data flow of the application is modeled by connecting the services via the publish/subscribe paradigm. In this communication model, the services play different roles. Please note that in our system a client does not subscribe to specific data elements but to notification on when such elements become available. We allow the subscriber to specify a notification queue size. This allows the data source to produce a certain number of data elements in advance, even though the subscriber is still busy processing a previous element. Flow control is achieved by preventing a service from generating new data elements in case the notification queue gets full.

The data objects that are being exchanged between services satisfy a given scheme. They consist of binary data and metadata in XML syntax. The most important element of the metadata is the timestamp, which indicates the recording time of the original data. The timestamp acts as a key for the data object within the given data buffer. The binary data can contain information in any format. A client can request individual data elements or arbitrary blocks of data elements from a buffer (See Fig. 4). This is achieved by sending structured queries with a simple syntax

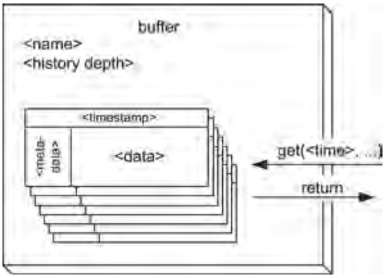


Fig. 4 Structure of a single buffer.

to the desired service. The client can then process these data items and/or eventually offer the results to further clients. The foundation of the used timestamps is the Newtonian time model. Timestamps are generated based on synchronized local clocks.

4.2 Image Quality Estimation

As discussed in 2, camera translation results only in marginal artifacts for objects at medium or large distances, which we consider dominant in many mobile robot scenarios.

For a camera mounted in the middle of the robot with a wheelbase of 0.5 m climbing an obstacle of 10 mm height the rotation results in motion artifacts for an object in a distance of 2 m seven times greater than by the translation component.

Hence we can simplify the tracking of camera movement by measuring rotation only. Angular rate gyroscopes were used for that matter. Because the strength of artifacts b depends on the angle between rotation axis and view direction, it varies across the image. In order to be able to decide, whether an image is too disturbed to be passed to image processing or not, we need a single quality estimate for the entire image. Several possibilities exist to calculate such an estimate, including the average or the maximum strength of motion artifacts of all regions of the image. We decided to go with another approach and focus on the region around the immediate image center only (See Eqn. 2).

$$b = \sum_{i=1}^{shutter} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \tag{2}$$

Reasons that support that decision are: (1) cameras are usually arranged so that objects of interest are located near the image center; and (2) it leads to further simplification of the tracking of camera movement. Rotation (roll) around the optical axis of the camera does not contribute to artifacts at the image center. Hence we only need to measure rotation (yaw and pitch) around two axes parallel to the image plane. This setup gives a good estimation of the projection of the actual rotation axis onto the image plane if the distance between the camera's pinhole and the gyroscope unit is small. The strength of motion artifacts can then be calculated in a trivial way by integrating rotation present during image exposure.

To express the image quality $q \in (0,1]$ we map the strength of motion artifacts b to the interval from 0 to 1, where 1 stands for an immobile camera. The quality reaches 0 for an infinitely high artifact strength (See Eqn. 3).

$$q_{image} = \frac{1}{e^b} \tag{3}$$

4.3 *Image Acceptance Test and Congestion Control*

We consider mobile platforms to have in general a limited computing capacity. At the same time we assume that image processing tasks consume significant resources. This is especially true if multiple image processing tasks are running simultaneously. Examples are visual SLAM, marker detection, terrain classification, object recognition, and so on. Here, the frame rate of a camera may easily exceed the image processing capacity. The system becomes congested and individual frames have to be dropped.

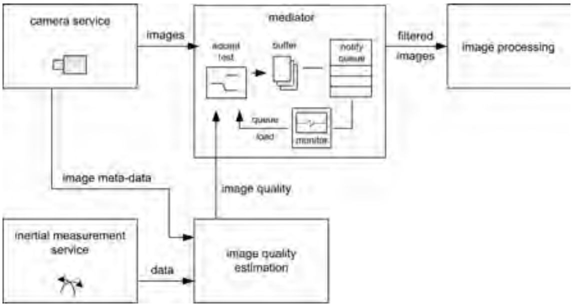


Fig. 5 Structure of a system containing a mediator service for congestion control.

Instead of dropping random frames, we can apply a simple, yet effective congestion control scheme. Having a normalized image quality value available for every frame enables us to compare it directly to the current system load, which is determined by the amount of pending images. Images are only accepted for further processing if their image quality is greater than the system load indicator. This implies that in case the system is idle, even images containing heavy motion artifacts are accepted for further processing. This ensures that it is not possible to reject every image for an extended period of time. However, a minimum required quality level can also be introduced if needed. The implementation shown in Figure 5 is self-explanatory. We chose to implement the acceptance test in form of a mediator that is arranged in between the data source (camera service) and a client that implements the image processing task. The system load can easily be computed by monitoring the current size of the notification queue as described earlier in this section. It is then normalized to the same interval as the quality.

5 System Evaluation

In this section we show results achieved with our approach to image quality estimation. We also present improvements of a scenario where markers are to be detected by a mobile robot while driving on a bumpy floor.

5.1 Evaluation of Motion Artifact Detection

In a first experiment we checked the correlation between motion blur in camera images and the calculated quality based on angular rate measurement. A mobile robot was equipped with a front facing camera and was driving towards a board placed in front of it. We did various measurements at different lighting conditions ranging from 300 lux to 600 lux, with a shutter time between 60 ms and 30 ms. The robot passed various bumps of a maximum height of 1 cm, which resulted in displacement as well as rotation of the robot. Two examples of images of different quality are shown in Figure 6. In image A (left) a strong motion blur in vertical direction can be observed, which resulted in a low quality estimate. Image B is significantly sharper, hence produces the better quality value. In Figure 7, the corresponding measurements from the gyroscopes are depicted. The intervals of image exposure are shown in gray.

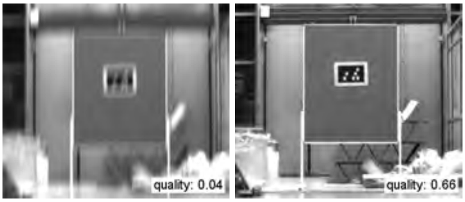


Fig. 6 Example images containing only slight (right) and strong motion blur (left).

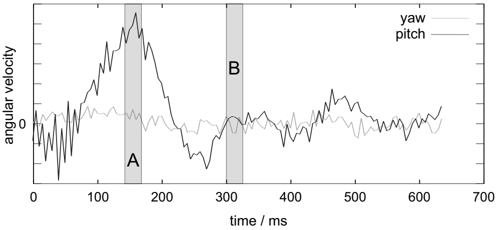


Fig. 7 Measured angular rates of rotation during image exposure (in arbitrary units).

For Figure 8 we compared the computed quality values 50 images to blur near the image center. The values for blur radius were generated by manual measurements.

5.2 Improving a Marker Detection Scenario

We applied our approach to dynamic image filtering to a scenario where optical markers were to be recognized by a moving mobile robot. The computing capacity onboard the robot is limited. Therefore not all images acquired by the onboard camera can be processed. The lights were set to about 300 lux, which resulted in an average integration time of 60 ms. A 2/3 inch monochrome CCD sensor and a

Fig. 8 Correlation between quality estimate and actual strength of motion blur.

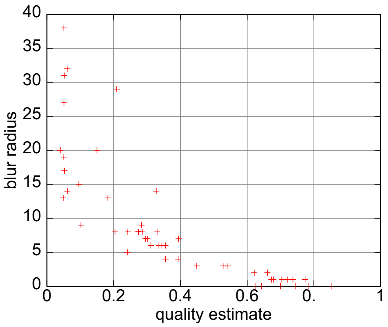
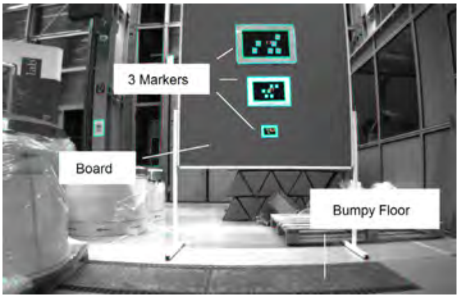


Fig. 9 Test setup of the marker detection scenario.



4.8 mm fixed focus C-Mount lens were used in the experiment. The iris was opened completely to allow as much light as possible to pass through. The robot was approaching a board from a distance of approximately 13 meters. Markers of different sizes were attached to the board (See Fig. 9). The goal when approaching the board was to recognize the markers as often as possible.

Images and motion data were recorded in order to compare results achieved with uncontrolled frame drops against results with dynamic filtering. In case of uncontrolled frame drops, the image quality was not taken into account. Here, the acceptance test was only considering queue utilization. The total number of images acquired during the approach was 319. The average processing time per frame required by the marker detection algorithm was approximately 1.6 times the inter-arrival time of new images. Table 1 shows the number of images in which a marker could be identified for one particular approach. The results were consistent when repeating the experiment. Please note that the largest marker was only visible in 246 out of 319 images. This is because it went out of view when the robot came close to the board. In general, markers could not be recognized in all 319 frames because at first they were too far away or they were obscured by motion blur. In the first run every image has been analyzed by the marker detection regardless of processing time. In the next run arriving images were dropped whenever the image-buffer was used to full capacity. In the third run images were discarded in the discussed manner.

It can be seen that the improvement in the total number of images with recognized markers increases with the decreasing size of the marker. This is because smaller markers are harder to recognize and are easily obscured by blur.

Table 1 Improvements of recognition results when applying dynamic filtering.

Marker	Images with marker recognizable	Marker recognized (uncontrolled)	Marker recognized (filtered)	Improvement (pct.)
large	246	154	161	4.5
medium	272	159	178	11.9
small	99	62	71	14.5

6 Conclusion

Here we presented an approach to improve the performance of image recognition tasks on mobile robots equipped with robust low-cost cameras. Concrete improvements have been achieved for an optical marker detection scenario. The basic idea presented was to improve the quality of processed images by estimating the amount of included motion artifacts for every image and rejecting bad ones. Our system is well suited for resource-constrained robots where the camera's frame rate exceeds the processing capabilities of the onboard computer. Based on our results, we are confident that the performance of a number of different image processing tasks can be improved through this approach.

Acknowledgements. This work has been funded in part by the German Federal Ministry of Education and Research under grant 01IM08002.

References

1. Cardani, B.: Optical image stabilization for digital cameras. Control Systems Magazine, IEEE 26(2), 21–22 (2006)
2. Chiu, C.W., Chao, P.P., Wu, D.Y.: Optimal design of magnetically actuated optical image stabilizer mechanism for cameras in mobile phones via genetic algorithm. IEEE Transactions on Magnetics 43(6), 2582–2584 (2007)
3. Cho, W.-H., Hong, K.-S.: A fast cis still image stabilization method without parallax and moving object problems. IEEE Transactions on Consumer Electronics 54(2), 197–205 (2008)
4. Choi, B.D., Jung, S.W., Ko, S.J.: Motion-blur-free camera system splitting exposure time. IEEE Transactions on Consumer Electronics 54(3), 981–986 (2008)
5. (eds.), O.M.G.: Common object request broker architecture (corba/iiop)
6. Fergus, R., Singh, B., Hertzmann, A., Roweis, S.T., Freeman, W.T.: Removing camera shake from a single photograph. ACM Trans. Graph 25, 787–794 (2006)
7. Gunthner, W., Wagner, P., Ulbrich, H.: An inertially stabilised vehicle camera system - hardware, algorithms, test drives. In: 32nd Annual Conference on IEEE Industrial Electronics, IECON 2006, pp. 3815–3820 (2006)
8. Ji, H., Liu, C.: Motion blur identification from image gradients. In: IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2008, pp. 1–8 (2008)
9. Nicklin, S.P., Fisher, R.D., Middleton, R.H.: Rolling shutter image compensation. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006: Robot Soccer World Cup X. LNCS, vol. 4434, pp. 402–409. Springer, Heidelberg (2007)

10. Sato, K., Ishizuka, S., Nikami, A., Sato, M.: Control techniques for optical image stabilizing system. IEEE Transactions on Consumer Electronics 39(3), 461–466 (1993)
11. Schiehlen, J., Dickmanns, E.: Design and control of a camera platform for machine vision. In: Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems 1994. Advanced Robotic Systems and the Real World, IROS 1994, vol. 3, pp. 2058–2063 (1994)
12. Tao, Y., Hu, H., Zhou, H.: Integration of vision and inertial sensors for 3d arm motion tracking in home-based rehabilitation. Int. J. Rob. Res. 26(6), 607–624 (2007)
13. Yeom, D., Park, N., Jung, S.: Digital controller of novel voice coil motor actuator for optical image stabilizer. In: International Conference on Control, Automation and Systems, 2007. ICCAS 2007, pp. 2201–2206 (2007)

A Software Architecture for Model-Based Programming of Robot Systems

Michael Geisinger, Simon Barner, Martin Wojtczyk, and Alois Knoll

Abstract. While robot systems become more and more elaborate, the need to simplify programming them grows as well. Regarding the high degree of internal heterogeneity in the sense that different microcontroller platforms, protocols and performance layers are used in a single robot application, it is not feasible to have specialists dedicated to each individual task. This motivates the need for tool support that allows an abstract view on a robot's sensors and actuators, means to program individual components as well as to define their interaction. In this work, we present how the model-based development and code generation tool EasyLab can be extended to support programming of all parts of a robot, including the main controller as well as peripheral devices like smart sensors. We show three typical use cases in the context of mobile platforms and highlight EasyLab's advantages in each domain.

1 Introduction

Complex robot systems often consist of highly heterogeneous components and communication protocols. A typical example is the TAMS Service Robot (TASER) presented in [17], whose parts are connected via Ethernet, CAN, RS-232 and others. Developing software for each component individually is a time-consuming process. A better approach is to specify the system components at a high level of abstraction. This specification is condensed in form of a *model*, which in turn can consist of various parts: the *device model* specifies the hardware components, their interfaces and dependencies. The *application model* formalizes the actual task to execute. Finally the *distribution model* specifies how software components are deployed in a distributed environment and how they interact with each other.

The model-based code generation tool EasyLab [3] uses this modular strategy to allow specification of application requirements and behavior at a high level of

---

Michael Geisinger · Simon Barner · Martin Wojtczyk · Alois Knoll
Institut für Informatik VI, Technische Universität München,
Boltzmannstraße 3, 85748 Garching bei München, Germany,
e-mail: `{geisinge,barner,wojtczyk,knoll}@in.tum.de`

abstraction. It currently supports device and application models. The integrated code generation engine can be used to transform the models into executable machine code optimized for the respective target platform.

EasyLab was originally designed for software development of *mechatronic systems* and includes components for mathematical computations, control systems (e.g., PID controllers) and communication over media typically used in industrial environments. However, due to its extensibility, EasyLab is also suitable for application in the field of robotics. This work will show how EasyLab can be extended to allow programming of all parts of a robot system. We show that different *modes of model execution* are a crucial point in this context and present uniform software interfaces for application deployment, debugging and monitoring of such systems. On the one hand, EasyLab can be used to generate code for the firmware of microcontroller systems. On the other hand, it also provides engine-driven execution of programs on the robot's controller system (based on an operating system or some kind of middleware).

In this paper, the term *programming* denotes the overall process of designing and implementing the control software of robot systems, whereas *deploying* means transferring the application to the *target* system. Since the latter one is our point of view, we also refer to it as the *local* system (in contrast to *remote* systems).

The rest of the paper is structured as follows: First, we summarize some related work in the domain of robot software architectures. In section 3, we show the main features of the model-based development tool EasyLab and point out the uniformity of some of our software architecture's interfaces in section 4. After three illustrative examples in section 5 we conclude with a summary and a section about future work.

2 Related Work

The usage of EasyLab in the domain of robotics requires to compare it to existing frameworks that tackle the challenging task of programming robot systems. We distinguish between different levels of abstraction:

- At the lowest level, a multitude of libraries have been created for robot systems to perform tasks like mathematical computations for kinematics, dynamics and machine vision [6, 10, 12]. EasyLab already provides this functionality partially through components for the mobile robot platform Robotino® introduced in section 5.1. In contrast to the approaches mentioned above, EasyLab also allows to specify required hardware resources of each component, which makes it possible to avoid inconsistent access to shared hardware already at the modeling level.
- In the field of robotics, middlewares are often used to hide complexity regarding inter-component communication. OpenRTM-aist [1] is a CORBA-based middleware for robot platforms that uses so-called robot technology components (RTCs) to model distribution of functionality. The Orca project [5] is a component-based framework for building robotic systems, which provides abstraction layers for communication and focuses on reuse of software. The Player

component of the Player/Stage project [9] provides a network interface to a variety of robot and sensor hardware based on TCP/IP. While EasyLab itself is not a middleware, it provides concepts to integrate them. On the one hand, this involves design and implementation of components, which is currently researched by integrating OpenRTM in such a way that the application logic of RTCs is modeled in EasyLab. On the other hand, support for distribution models is needed, which is subject to future work.
- Skill and behavioral based robot programming deals with the question how tasks and goals can be described intuitively and at a high level of abstraction. URBI [2] is a universal robotics software platform with built-in support for parallel execution and event-based programming. CLARAty [15] is a framework for generic and reusable robotic components and consists of a functional layer and a decision layer, whereas the latter allows high-level specification of tasks to execute. Until now, EasyLab was primarily applied to the field of automation and therefore only data-flow and state-based task specification and adequate execution concepts were required. However, EasyLab's component-based architecture and its language primitives for explicit parallelism form a possible approach for the integration of the high-level concepts mention above.

The Ptolemy project [8] has a broader scope and is not restricted to the field of robotics. It provides a wide range of models and modes of execution for the design of real-time systems, which is also one of the goals of EasyLab. In addition to that, EasyLab explicitly models the target hardware and provides means to generate hardware-dependent code for different platforms from the same model.

3 EasyLab

EasyLab is a model-based development tool for the software component of embedded systems. It is part of the EasyKit[1] project, which aims at providing a methodology for the efficient development of *mechatronic systems* (i.e., systems with a tight interaction of mechanics, electronics and software) using hardware/software co-design. While *EasyKit* provides a modular hardware construction kit for the assembly of hardware prototypes of mechatronic systems, *EasyLab* allows to define software models for interaction with the respective hardware and features both a template-based code generator (the current output language is C) and an engine-driven model execution mode. Models can be specified in EasyLab using visual programming languages (see section 3.1). Using predefined primitive components from an integrated library, a user does not have to write a single line of code in order to develop an application for a mechatronic system. EasyLab can also be used to monitor the running system and tune certain parameters over serial lines or bus communication protocols such as Modbus [14].

---

[1] See `http://easykit.informatik.tu-muenchen.de/`.

This work is funded by the German Ministry of Education and Research under grant 02PG1241.

3.1 Application Model

One of EasyLab's primary goals is to facilitate the way of programming target systems. For this purpose, it currently supports two visual modeling languages (for details see [3]):

1. **Structured flow chart:** The SFC language describes the states of a program, how state transitions are performed and which actions are taken when the program is in a certain state (typically execution of a data flow program as specified below). The SFC language was designed in the style of EN 61131-3 [11] (part "SFC").
2. **Synchronous data flow:** The SDF language describes a directed multigraph where each node is the instance of a certain actor type (also called *function block*). An actor type is defined by a set of typed input and output connectors as well as internal state variables. Furthermore, the actions **start()**, **step()** and **stop()** define an actor's effect on the input data and calculation of the output data. Edges in the SDF graph denote the data flow between actor instances. The SDF language was designed in the style of EN 61131-3 [11] (part "FBD") and also supports multi-rate data flow.

3.2 Device Model

In EasyLab, not only the application logic is modeled, but also hardware-related aspects of the system (e.g., the microcontroller being used). This is especially important for code generation, as low-level driver code must be generated differently for different types of hardware platforms. EasyLab's device descriptions are modular and can be nested to obtain composed devices (e.g., a robot system) from primitive devices (such as specific controllers, sensors and actuators). The library of primitive devices can also easily be extended.

Device behavior is defined by specifying handlers for the following actions: **open()** and **close()** are called when a connection to a device instance is to be established or dropped, respectively. **start()** and **stop()** are called just before a device will be used or when it should go "idle". **read()** and **write()** are called when data should be requested from or forwarded to the hardware (these operations are performed all at once for all data to ensure consistent sensor values and actuator control signals).

3.3 Modes of Execution

Once the application and device models are specified within EasyLab, they can be executed in different ways. The first approach is to generate code (currently: C code) from the models that is then executed *natively* on the target platform. If an operating system is available on the target, the model can also be transferred to the target platform and directly interpreted (*local execution* or *model interpretation*). A third

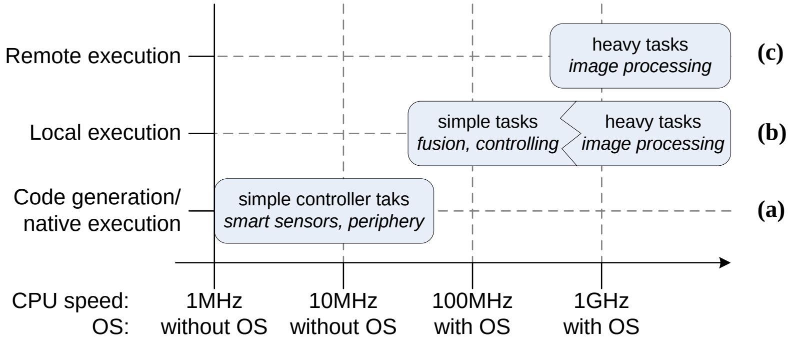

Remote execution
Local execution
Code generation/ native execution
heavy tasks
image processing
(c)
simple tasks
fusion, controlling
heavy tasks
image processing
(b)
simple controller taks
smart sensors, periphery
(a)
CPU speed:
OS:
1MHz
without OS
10MHz
without OS
100MHz
with OS
1GHz
with OS

Fig. 1 EasyLab modes of execution against requirements and typical usage scenarios in italics. **(a)** Code generation and native execution fits best the needs of simple controller applications as found in smart sensors. **(b)** Local execution can run on any target with operating system. **(c)** Using remote execution, the task is simulated on a host PC and control signals are exchanged with the hardware.

possibility is to execute the model directly within EasyLab and exchange control signals with the target (*remote execution* or *model simulation*), which requires an adequate proxy application running on the target.

Figure 1 shows EasyLab's modes of model execution in correlation with different application domains. Some execution modes imply the existence of an operating system (OS). The graph also shows rough boundaries for the clock speed of the respective processor.

3.3.1 Code Generation and Native Execution

The process of code generation and the provided interfaces are shown in figure 2. It is currently used for small applications without operating system (typically 8-bit microcontrollers). In the field of robotics, such systems are used to implement *smart sensors or actuators* that perform some kind of additional processing. EasyLab can be used to model the application logic of these systems and also the communication backend that allows other components to access them (see section 5.3 for an example). The template-based code generation approach provides the flexibility to generate code for different platforms such as middlewares or operating systems (see [3] for details).

EasyLab currently supports (but is not restricted to) the following compilers: `avr-gcc` for Atmel AVR[2], `mcc18` for PIC 18F[3] and `fcc907s` for Fujitsu F16[4]. Generating code for SFC programs is implemented using the continuation-passing style pattern. For SDF programs, static scheduling [4, 13, 16] is used to ensure

---

[2] http://www.nongnu.org/avr-libc/

[3] http://search.microchip.com/searchapp/searchhome.aspx?q=SW006011

[4] http://www.fujitsu.com/us/services/edevices/microelectronics/microcontrollers/datalib/softune/

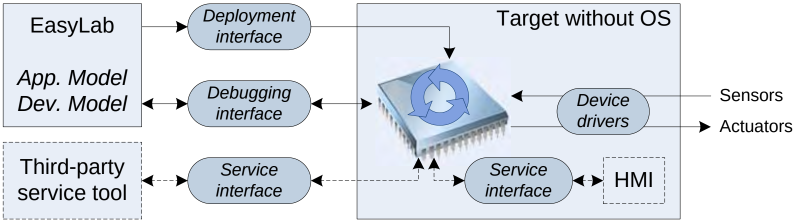

EasyLab
App. Model
Dev. Model
Third-party service tool
Deployment interface
Debugging interface
Service interface
Target without OS
Device drivers
Service interface
HMI
Sensors
Actuators

Fig. 2 Native execution use case: The compiled image is sent to the target via the deployment interface. The target executes the program (indicated by circular arrows) and interacts with the hardware using device drivers. A debugging interface enables inspection and step-wise execution during development. The optional service interface can be used to maintain and tune the system after deployment or to visualize and modify data using a human-machine interface (HMI).

efficient execution and predictable timing. Schedules for parallel periodic tasks (e.g. multiple SDF programs running at the same time) can be computed in advance using the concept of logical execution time [7]. However, programs cannot be modified after deployment without a new code generation run, which is different in the execution concepts described in the following sections.

3.3.2 Local Execution

By the term "local" we refer to execution of a program on the target hardware (i.e., on the robot). In contrast to code generation, local execution is performed by "interpreting" a program modeled in EasyLab using a console-mode version of EasyLab (i.e., a specially crafted version without graphical user interface). Figure 3 illustrates this use case.

An operating system is used to reduce the complexity of the interpreter and facilitates the implementation of tasks like host-to-target communication, model storage and dynamic loading of shared libraries. Interpretation of SFC programs

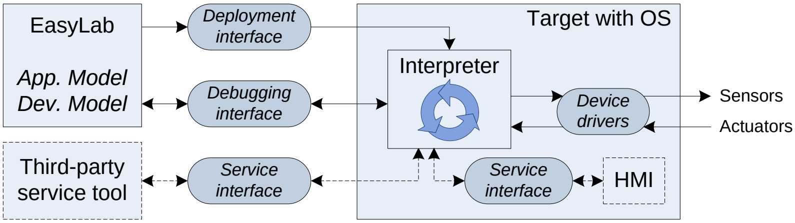

EasyLab
App. Model
Dev. Model
Third-party service tool
Deployment interface
Debugging interface
Service interface
Target with OS
Interpreter
Device drivers
Service interface
HMI
Sensors
Actuators

Fig. 3 Local execution use case: the complete application model is transferred to a console-mode version of EasyLab (interpreter) running on the target system. Debugging and service interface serve the same functionality and use the same communication protocols as in native execution mode.

is straightforward. For SDF programs, a dynamically linked library is specified for each actor type that provides the following C++ interface: The **start()** method is used to initialize the actor. The **step()** method is called each time the respective actor is executed according to the schedule. A **stop()** method is used to finalize the actor. The libraries for each actor are cross-compiled for the respective target's operating system and dynamically loaded on demand using a plug-in system. The same approach is also used for device drivers. This execution concept also allows to change the program while it is running by updating or retransmitting the model and the necessary plug-ins.

3.3.3 Remote Execution

Remote execution means that the application is not executed on the actual target system, but instead on a host PC running EasyLab (usually the machine where the application is being modeled). The target runs the console-mode version of EasyLab and only acts as a proxy that forwards sensor readings to and awaits control signals from the host. Figure 4 illustrates this configuration.

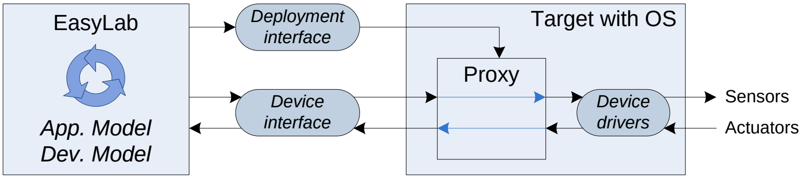


Fig. 4 Remote execution use case: The console-mode version of EasyLab running on the target is used to translate between requests from the host PC (on the left) and the actual device drivers. Deployment is realized by transferring device descriptions and drivers to the target. A service interface is not available on the target.

An advantage of this setup is that changes in the application model are immediately reflected in the behavior of the system. This makes it easy to modify or extend the program while it is already running. Only if the underlying device model is changed, the device proxy running on the target needs to be exchanged (redeployment). Another advantage is that the performance of the application is better if the processor of the host PC is more powerful than the one in the target system.

Note that remote execution could also be applied to target platforms without operating system. This would require to implement an adequate device interface and has not yet been researched.

4 Interfaces

As pointed out in [3], expandability is one of the primary design goals for EasyLab that is achieved by a clear separation of models, code-generation templates, execution and also GUI plug-ins. In addition to that, the concept of interfaces abstracts introspection and manipulation of execution instances. Since these interfaces are also used on embedded targets (native execution), the current implementation is based on C (a C++ version that integrates with the existing plug-in architecture is planned).

Access to variables is performed via unique identifiers. Possible implementations are the variable's RAM address (if known in advance) or arbitray identifiers that are mapped to the variable using look-up tables. The latter implementation is required in remote execution mode and for native execution in conjunction with dynamic memory allocation (i.e., when a variable's address can not be known in advance).

4.1 Debugging Interface

The debugging interface is only available during development. It is used to visualize and manipulate the current state of execution and all variable values directly in the graphical representation of the model and consists of the following functions:

getProgramId() retrieves a unique number that is used to match the model with the executed program instance. **getFlags()** retrieves variable flags (e.g., read/write access). **read()** retrieves the current value of a variable, while **write()** sets its value. There is support for transactions (atomic execution of multiple read and write operations). A transaction is initiated with **beginTransaction()** and executed on **commit()**. **getCurrentStack()** retrieves the execution stack and is used to determine and visualize currently active model items. **command()** sends commands to the target to run, break, step or reset it and to set or remove breakpoints.

4.2 Service Interface

In contrast to the debugging interface, this interface allows to monitor a running application and to change selected parameters of deployed programs (i.e., in "release mode"), which is a requirement for the use of programs generated by EasyLab in the field of automation. The *local service interface* can be used to connect to the running application from the target side. For native targets (code generation) this is implemented by linking another application into the final executable such as a human-machine interface (HMI). The *remote service interface* can be connected to over any communication medium supported by the target. An application scenario in context of the EasyKit project is to have a third-party service tool running on a mobile device to perform maintenance, diagnosis and benchmarking tasks.

5 Applications

In this section, we give three examples where model-based development using EasyLab is applied in the context of robot systems. They demonstrates how the different modes of execution pointed out in section 3.3 can be applied while using the same abstract types of device and application models at the topmost layer.

5.1 Mobile Robot Platform Robotino®

Robotino® is a mobile robot platform mainly used in education and research. It is not only equipped with a drive system allowing for omni-directional moves as well as distance, infrared and inductive sensors, but also features a camera with VGA resolution. Figure 5 (a) shows the default configuration.

The robot is equipped with a PC104 controller with a clock speed of 500 MHz (there are however newer versions with faster CPUs). The robot runs a real-time Linux operating system and can be controlled remotely over wireless LAN. The console-mode version of EasyLab runs directly on the robot (*local execution*, see section 3.3.2). However, the limited computational capacity of Robotino® prevents the implementation of extensive tasks like complex image processing, which is why the robot is mostly operated in *remote execution* mode (see section 3.3.3). For this purpose, the device model components in EasyLab establish the connection to the target. A library of customized SDF function blocks is available for reading the robot's sensor values and controlling its movements. To make this possible, a proxy for interactions with sensors and actuators is running on the robot. Typical applications are camera-based navigation (e.g., line following), induction-driven fine positioning and distance-based collision avoidance.

An advantage of this approach is its flexibility: users can easily explore the robot's functionality and intuitively understand how changes to the program are reflected in the robot's behavior. As the basic hardware components of the robot are fixed, the proxy application does not need to be adapted when modifying the program. The software Robotino®View is a special version of EasyLab that only supports remote execution with a fixed hardware model.

5.2 F5 Platform

The F5 platform is another mobile platform for which EasyLab device models have been defined. It is shown in figure 5 (b). In comparison to Robotino®, this platform is not only intended for education and research, but also for carrying out service

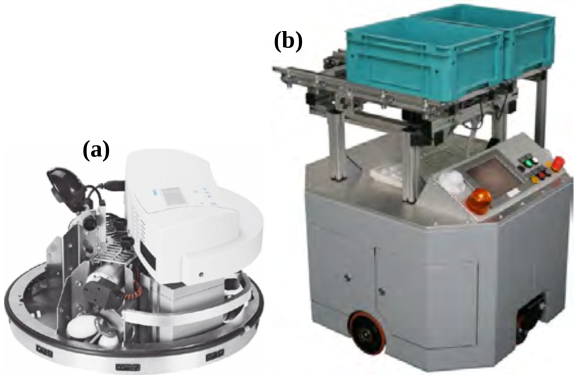


Fig. 5 Applications; **(a)** mobile robot platform Robotino®; **(b)** F5 platform in the current stage of expansion. Note that the images are not at the same scale. Images courtesy of Festo Didactic and REC GmbH.

tasks. It has larger proportions and is prepared to be extended by a robot arm to increase its versatility and ability to manipulate the environment. In previous projects, we demonstrated a robust and reproducible sample management process in a biotech pilot plant [18] carried out by an autonomous, mobile manipulator which is going to be succeeded by the F5. Precise movements and gentle device interactions are assured by the utilization of a camera for image processing and a force/torque sensor attached to the tool of the manipulator to prevent damages.

Since the platform is equipped with a rather powerful computer (standard PC running at 2 GHz and a touch screen), it is suitable to perform both *remote* and *local execution* of arbitrary EasyLab models such as image processing, localization, mapping and path planning. A typical scenario for *local execution* is the previously mentioned sample management process, which is carried out autonomously by the mobile platform. Alternatively, a remote user may connect to the platform and utilize the *remote execution* of EasyLab models to carry out surveillance tasks and lab walkthroughs.

5.3 Smart Sensors and Actuators

Modularization is a long-standing concept in system design that fosters reuse of well-tested components and thus both improves quality and helps to cut cost. Hence, also in the area of robotics decentralization and the need for local ”intelligence” for both sensors actuators is becoming more and more important. The term *smart sensors* refers to ordinary sensors that have been augmented with local preprocessing (like calibration, statistics, ...) and offer the additional benefit of a simple interface to clients. The term *smart actuator* is intended to transfer this principle to the domain of actuators (e.g., a drive systems with integrated control unit). For many of these components, already a microcontroller offers enough computational power. In the course of an integrated development process, code generation and *native execution* as described in section 3.3.1 can be used to implement the necessary signal processing and control software. Hence, the same level of abstraction can be used for all computation units of a robot.

As an example for a smart sensor application, we set up a PT100-based temperature sensor that was enhanced with a microcontroller and external EEPROM. The sensor provides calibrated and preprocessed data over Modbus. An example for a smart actuator is the “intelligent” pneumatic cylinder detailed in [3].

In both setups, we used the debugging interface to verify the application model during development. Compared to conventional smart sensor and actuator code development, the high level of abstraction fostered readability of the programs, facilitated debugging and thus reduced the potential for errors.

6 Summary

In this paper we argued that model-based development is a possible solution to the heterogeneous controller and communications architectures that are widespread in the field of robotics. While the raised level of abstraction hides implementation details and thus enables a more efficient and less error-prone development process, the approach followed in EasyLab still yields implementations that are adopted to the respective target platform. This is achieved by a threefold notion of model execution, namely native, local and remote execution, which were illustrated by three demonstrator implementations. The concept that is actually used depends on factors like processing power of the target system, presence of an operating system and whether tuning or altering the program while it is running should be supported.

7 Future Work

EasyLab's support for programming robot platforms is still to be extended in various directions. One of the most important features is *distributed modeling*, which will allow multiple components of a robot or networked system to be modeled within one EasyLab model. Data exchange can then be expressed at a very high level of abstraction by specifying producers and consumers of data. We consider to use the time-triggered approach because of its well integration with the SDF language. Going one step further, tasks such as acquiring and processing sensor data as well as controlling actuators can be distributed in the style of a *service-oriented architecture* (SOA), where each component offers services to the other components. An extension that goes in a different direction is support for *multicore architectures*. As both the SFC as well as the SDF language provide the possibility to explicitly model parallel execution of tasks, it is meaningful to distribute the load of these tasks in an intelligent manner among different computation cores, if available.

References

1. Ando, N., Suehiro, T., Kitagaki, K., Kotoku, T., Yoon, W.-K.: RT-middleware: Distributed component middleware for RT (robot technology). In: International Conference on Intelligent Robots and Systems 2005 (IROS 2005), pp. 3933–3938 (2005)
2. Baillie, J.-C.: URBI: towards a universal robotic low-level programming language. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), pp. 820–825 (2005) doi:10.1109/IROS.2005.1545467
3. Barner, S., Geisinger, M., Buckl, C., Knoll, A.: EasyLab: Model-based development of software for mechatronic systems. In: IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications, Beijing, China (2008)
4. Bhattacharyya, S.S., Buck, J.T., Ha, S., Lee, E.A.: Generating compact code from dataflow specifications of multirate signal processing algorithms. IEEE Transactions on Circuits and Systems – I: Fundamental Theory and Applications 42, 138–150 (1995)
5. Brooks, A., Kaupp, T., Makarenko, A., Oreback, A., Williams, S.: Towards component-based robotics. In: Proc. of 2005 IEEE/RSJ Int. Conf. on Intellegent Robots and Systems (IROS 2005), Alberta, Canada, pp. 163–168 (2005)
6. Bruyninckx, H.: Open robot control software: The OROCOS project. In: Proceedings of 2001 IEEE International Conference on Robotics and Automation (ICRA 2001), Seoul, Korea, vol. 3, pp. 2523–2528 (2001)

7. Buckl, C., Regensburger, M., Knoll, A., Schrott, G.: Generic fault-tolerance mechanisms using the concept of logical execution time. In: Proceedings of the 13th Pacific Rim International Symposium on Dependable Computing, pp. 3–10. IEEE, Los Alamitos (2007)
8. Eker, J., Janneck, J.W., Lee, E.A., Liu, J., Liu, X., Ludvig, J., Neuendorffer, S., Sachs, S., Xiong, Y.: Taming heterogeneity—the Ptolemy approach. In: Proceedings of the IEEE, vol. 91 (2003)
9. Gerkey, B.P., Vaughan, R.T., Howard, A.: Most valuable player: a robot device server for distributed control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Wailea, Hawaii, pp. 1226–1231 (2001)
10. Gourdeau, R.: A robotics object oriented package in C++. Ph.D. thesis, École Polytechnique de Montréal (2001)
11. International Electrotechnical Commission: Norm EN 61131 (2003)
12. Kapoor, C.: A reusable operational software architecture for advanced robotics. Ph.D. thesis, The University of Texas at Austin (1996)
13. Lee, E.A., Messerschmitt, D.G.: Static scheduling of synchronous data flow programs for digital signal processing. IEEE Trans. Comput. 36(1), 24–35 (1987)
14. Modbus IDA: Modbus application protocol specification v1.1a (2004), `http://www.modbus.org/`
15. Nesnas, I.A.D., Wright, A., Bajracharya, M., Simmons, R., Estlin, T.: CLARAty and challenges of developing interoperable robotic software. In: Proceedings of 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), vol. 3, pp. 2428–2435 (2003) doi:10.1109/IROS.2003.1249234
16. Oh, H., Dutt, N., Ha, S.: Memory optimal single appearance schedule with dynamic loop count for synchronous dataflow graphs. In: Asia and South Pacific Conference on Design Automation, p. 6 (2006)
17. Westhoff, D., Zhang, J.: A unified robotic software architecture for service robotics and networks of smart sensors. In: Autonome Mobile Systeme 2007, Kaiserslautern, Germany, pp. 126–132. Springer, Heidelberg (2007)
18. Wojtczyk, M., Marszalek, M., Heidemann, R., Joeris, K., Zhang, C., Burnett, M., Monica, T., Knoll, A.: Automation of the complete sample management in a biotech laboratory. In: Abramsky, S., Gelenbe, E., Sassone, V. (eds.) Proceedings of Visions of Computer Science, BCS International Academic Conference, pp. 87–97. The British Computer Society, London (2008)

Intuitive Robot Programming of Spatial Control Loops with Linear Movements

Katharina Soller and Dominik Henrich

Abstract. Programming by demonstration (PbD) is a promising approach to facilitate the use of robots. The bandwidth of PbD systems extends from simple play back to artificial intelligence methods. Our approach is based on the former. This paper aims to introduce and examine intuitive programming of spatial loops, i.e. program loops whose body changes according to a spatial variable. The loop is programmed by executing e.g. the first, the second, and the last run. The system then determines all components of the loop so they can be used to generate a program. The focus of this paper lies in automatic determination of the variables for the start, the step and the end of spatial loops. This method of robot programming has interesting applications in the field of industrial robotics, especially with respect to small- and medium-sized enterprises, as well as for service robotics.

1 Introduction

One of the largest tasks involved in expanding the application range of robots (e.g. for use in household applications or frequently changing applications in small enterprises) are the typically difficult and time-consuming methods of programming them. The usual way to program an industrial robot is to write a program (either offline or online) and teach the positions used in the program directly on the robot. One major drawback of this procedure is that the programmer must have some competence in the field of robot programming. In addition, this type of programming does not support the typical work flow used in small enterprises or households, because it is time-consuming and unwieldy.

For these reasons numerous efforts have been made to make robot programming more intuitive. One approach is called "Programming by Demonstration" (PbD). The idea behind this approach is to program a robot by executing the task instead of programming it textually. The task demonstration is either performed in the real environment or in a virtual environment [2]. The latter approach has some disadvantages: it is not easy to navigate in a virtual environment and such programming does not support the work flow. Data gloves and head-mounted displays are not typically tools cooks or carpenters want to deal with while working. Alone the time needed to put on such devices restricts the applicability.

Katharina Soller · Dominik Henrich
Lehrstuhl für Angewandte Informatik III, Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth, Germany,
e-mail: `{katharina.soller,dominik.henrich}@uni-bayreuth.de`

A further distinction may be made with regard to whether the demonstration is performed using the robot or not. In the former case, also called kinesthetic learning [5], the robot's internal sensors can be used for measuring trajectories, otherwise further sensors in the environment are required (e.g. cameras) or must be worn by the programmer while performing the demonstration (e.g. data gloves). The disadvantages of wearing sensors have already been mentioned above. Environmental disadvantages of sensors are possible occlusions and a restricted transportability. Our approach for PbD concentrates on the remaining alternative: measuring trajectories using the (internal) sensors of the robot while the user moves the robot via zero-force control. In general our approach is applicable to all of these approaches, as long as they make use of trajectories. Table 1 shows an overview of the different approaches and a small selection of references for each of them. A description of a system utilizing a robot model in a simulated environment for PbD could not be found in literature.

Table 1 Classification and evaluation of approaches to Programming by Demonstration (PbD).

	With Robot	*Without Robot*	
In real environment	[12] [10] [5] This paper	[7] [11]	– Need for highly structured environment and/or good object recognition techniques
In simulated environment	Just simulation, no PbD	[1] [2] [9]	– Need for integrating knowledge into simulated environment – Unwieldy VR-hardware
	– Need for high safety + Gives the user a better feeling for robot motions	– Need for adapting the human demonstration to the robot's kinematic + Independence of robot	

Work on PbD can also be classified with respect to the abstraction level [5]. High-level approaches regard complete actions and skills (see e.g. [7]), while low-level approaches concentrate on trajectories and basic robot actions. A high-level approach for loop structures can be found in [11].

Since the topic of our paper is the intuitive programming of spatial loops, composed of trajectories, it clearly involves a low-level approach. *Spatial loops* are repetitive movements of the robot that differ in one or more parameters (e.g. parameters for translation or rotation). For example, coating a surface consists of many similar movements (strokes of the brush) distributed equally over the surface. Such spatial loops may be programmed in a textual and ordinary manner by specifying three values for the loop parameter and a body that makes use of this loop parameter. Programming this task with the simple playback method, in which the robot directly repeats the demonstrated motion, requires one to perform each individual repetition. In addition, such an approach will not lead to a uniform loop due to demonstration insufficiency.

The system should be programmable via the following program flow structures, which are more than enough for writing general computer programs [3]: sequence, choice, loop, subroutine and break. The goal is to transfer the essential parts of general programming languages to intuitive robot programming, thus, inheriting the potential of computational completeness. In this paper, we concentrate on loops. With our approach it is possible to program a spatial loop in an intuitive way by demonstrating a portion of the desired task, for example the first, the second and the last cycle. The system then identifies the parameters of the loop, the values (start, step width and end) and the body, freeing the user from calculating or measuring these values. In addition such programming of loops makes sense even for applications which are required just once but contain many repetitions or for which the precision of the robot is needed to attain an exact distribution of the repetitive movements.

The rest of this paper is organized as follows: The next section gives an overview of the research related to PbD, especially low-level approaches. The third section briefly describes the proposed approach, where the component for loop programming should be embedded. The loops themselves and different ways to demonstrate them are covered in the fourth section. The fifth section formally defines the problem of deriving plausible spatial loops from a demonstration and presents a simple implementation. Based on this, we programmed translational spatial loops, and describe them in the sixth section.

2 Related Work

Our approach to PbD directly builds upon the playback (walk-through) method. This method is well known but its application was mainly restricted to early painting robots. Since these robots were heavy and dangerous, this kind of programming fell into misuse, but thanks to new technologies and standards these problems have been resolved [12].

Other problems associated with this method are demonstrations made insufficient by tremor and noise and high storage space requirements. [13] suggests smoothing of piecewise linear paths by removing unimportant points. Other approaches comprise B-Spline smoothing, Fourier transformation or Kalman-based smoothing (e.g. [6]). In [12] and [4] systems are described in which the user can modify the trajectory after demonstrating. Another approach to get rid of the insufficiency of a single demonstration is to interpolate several demonstrations. All these techniques just modify the trajectories and no interpretation of the user's intention can be drawn from them.

Some research addresses generalization of several demonstrations, allowing the robot to cope with uncertainties such as differing object positions. For example, the system described in [5] extracts the relevant features of trajectories (i.e. relative and absolute position constraints) from several varying demonstrations. The positions of the objects are obtained by a vision system. This way, the robot can apply the learned program to situations with new object positions. While this is a powerful method to generalize demonstrations, the number of needed demonstrations is dependent on the user's experience. In contrast to our work, this approach concentrates on the learning of task-specific constraints, while we concentrate on the kinesthetic definition of a loop structure.

To our knowledge, there is no approach that allows programming of looping movements by demonstrating just two or three movements. The idea is to give the user the possibility to implicitly define the variables and values needed for the execution of a loop structure.

3 Overview of System Concept

This section gives a brief overview of the system concept. The programmer uses the robot and a control panel to demonstrate the task he wants the robot to assume. For this purpose the demonstrator must be able to move the robot, e.g. via zero-force control, a joystick or a tracking device (for the latter, see e.g. [8]). Together with the signals from optional external sensors, this information forms the demonstration. The system then analyzes and interprets the demonstration and generates a corresponding robot program. The analysis and generation of the program are based on sets of control structures, movement primitives and variable types. The user then obtains a visualization of the spatial loop and can review the result.

4 Spatial Loops and Demonstration Types

The classical types of loops are condition-controlled loops (while loops), infinite loops and count-controlled loops (for loops)[2]. In the case of a *condition-controlled loop*, a Boolean variable or function is required that indicates e.g. if a button is pressed or a certain amount of time has passed. A loop is called *infinite loop* if there is no such condition or if this can never be met. In contrast to this kind of loops, *count-controlled loops* are stopped, when a variable – the counter – reaches a given end value. In all cases, the body of the loop either can be unchanged, i.e. the execution is a repetition of exactly the same behavior, or it can contain variations based on variables. In the case of condition-controlled and infinite loops, there is no general possibility to derive the break condition from the demonstration of two or three loop cycles, so for these loop types it is enough to determine the type and size of the step that is done between two consecutive cycles. In contrast, for count-controlled loops the number of cycles also has to be determined. For this reason we only regard count-controlled loops. The ideas are transferable to the other types.

In classical programming languages, a count-controlled loop consists of four components: three values for the loop parameter (initial value, boundary value and step size) and the body.

---

[2] Modern programming languages utilize further types of loops, e.g. for-each loops. We neglected these because they do not apply to the context of pure trajectories without further sensory input.

Table 2 Overview of loop types and their components.

	Condition-controlled loop	*Infinite loop*	*Count-controlled loop*			
			Demonstration Type 1	*Demonstration Type 2*	*Demonstration Type 3*	*Demonstration Type 4*
Condition	Boolean expression	–	– –		– –	
Start value	For varying the body	For varying the body	First trajectory	First trajectory	First trajectory	(auto)
Step			Second trajectory	Second trajectory	(auto)	First trajectory
Number of steps	Break by condition	–	Numerical input	(auto)	Numerical input	Numerical input
End value			(auto)	Third trajectory	Third trajectory	Second trajectory

In the case of spatial loops, the three values consist of spatial variables (e.g. coordinates). The start value and end value of such a spatial loop can be defined by movements. Such a movement is called a sub-trajectory. The size of the step can be regarded as transformation on these sub trajectories. The body includes some robot commands that use the spatial variable and create the motion. There are basically four[3] alternatives for specifying a count-controlled spatial loop (Table 2). Depending on some circumstances, which we will examine next, it is possible to determine the last of the four values (start value s, step size δ, number of steps k and end value e) using the other three. The idea is outlined in 2D in Figure 1: The left side shows the different Demonstration Types 1 to 4. They all define the same spatial loop shown on the right. It is easy to see that in this case three of the four values are enough to determine the fourth with the following formula: $k \cdot \delta = e - s$.

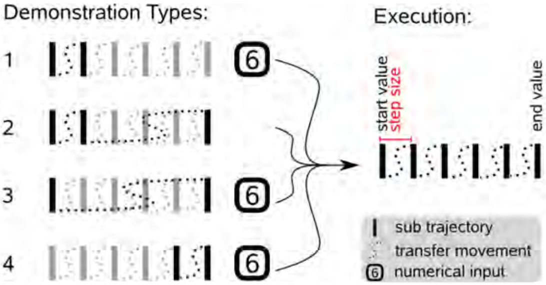


Fig. 1 Four alternatives for programming a translational loop consisting of six linear sub trajectories.

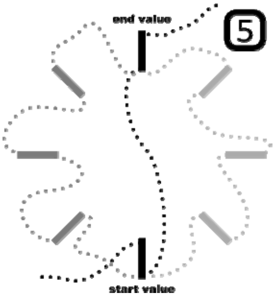


Fig. 2 Two possible spatial loops (dark grey and light grey) for Demonstration Type 3.

---

[3] Of course it would be possible to use all four components or even more sub-trajectories for the demonstration to increase robustness of the loop calculation, but we want to keep the demonstration as short as possible and focus on the minimum number of needed values.

We now look at the transformations that can be used for spatial loops. We do not consider the orientation of the tool center point here, i.e., we regard the orientation as fixed. In the context of this paper the transformation that describes how a trajectory for loop cycle $i + 1$ is derived from a trajectory for loop cycle i, can be translation and rotation, i.e. rigid body transformations. We do not regard reflection or scaling. Affine transformations that are more complex than the chosen transformations (e.g. shear) are also neglected because we regard them as less intuitive. We also restrict ourselves to linear interpolation for the generation of the loop, i.e. the steps between the cycles are all of the same size δ.

Considering the Demonstration Types 1, 3 and 4, we see that for translational transformations there is a unique mapping of the demonstrations to the spatial loops since a straight line is defined by two constraints. This does not hold for circular transformations, where three constraints are needed for a unique definition. Figure 2 shows an example with Demonstration Type 3. The input consists of two demonstrations (start value and end value) and the number of steps. Two spatial loops that both fit the demonstration are shown. Each rotation of a matching spatial loop around the axis defined by the start and end value leads to another matching spatial loop.

The situation is similar regarding the question whether a demonstration should be mapped to a translation or a rotation. Since a straight line can be considered as a circle with infinite radius, this distinction can not always be made with Demonstration Types 1, 3 and 4 without further knowledge.

On the one hand, since the use of always the same modality (e.g. position input) is more intuitive than multiple modalities (e.g. position and alphanumerical input), we suggest that Demonstration Type 2 is more intuitive than the other types. On the other hand, the other demonstration types have advantages as well. If the number of steps is essential for a certain application one of them might be the better choice.

5 From Demonstrations to Program Loops

In this section, the procedure of programming a spatial loop by Demonstration Type 2 and translational transformation is described. After instructing the system

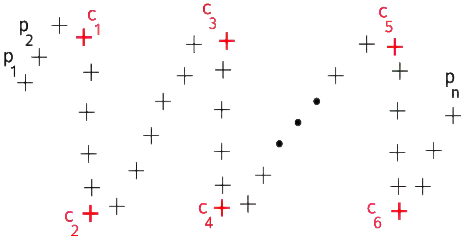


Fig. 3 Trajectory of Demonstration Type 2, divided into its sub-trajectories.

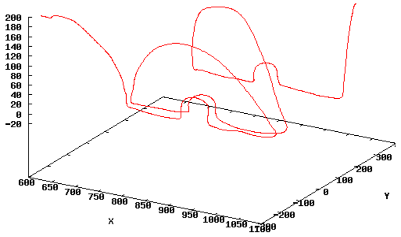


Fig. 4 The interpretation of the loop body depends on the set of motion primitives.

that a spatial loop is desired (e.g. via the control panel), the user demonstrates the first, the second and the last run of the loop and marks the beginning and the end of each cycle on the control panel, leading to a segmented trajectory that will then be analyzed.

At first the system has to decide whether the demonstration is a valid demonstration of Type 2. Since for Type 2 there have to be three demonstrated sub-trajectories, there must be six cuts, otherwise the system shows an error message.

5.1 Problem Description

A trajectory $r = (p_1, \dots, p_n)$ is a sequence of n coordinates $p_l \in \mathbb{R}^d$, $l \in \{1, \dots, n\}$, where d is the dimension of the coordinates. As mentioned above, we also have $m - 1$ cutting points c_j, $1 \leq j \leq m - 1$, on the trajectory, which divide the trajectory in sub-trajectories $r_1, \dots r_m$. Let $m(i)$ be the number of cuts needed for demonstration type i.[4] Then there are $m(i) + 1$ sub-trajectories $r_1, \dots, r_{m(i)+1}$ in a valid demonstration for Demonstration Type i. The trajectories we are interested in are those with an even index. All other trajectories are just transfer movements, which are neglected in this paper. Figure 3 shows an example with Demonstration Type 2.

Let R be the set of all trajectories in a given representation and T a set of transformations on these trajectories: $T = \ \ T = \{f \mid f\colon R \to R\}$. Let $\lambda\colon R \times R \to \mathbb{R}^{\geq 0}$ be a measure for the difference between two trajectories in R. Then the problem of finding a corresponding spatial loop to these trajectories can be stated as follows for Demonstration Type 2:[5]

$$\underset{f \in T; k \in \mathrm{N}^{\geq 2}}{\operatorname{argmin}} \left(\alpha \lambda(f(r_2), r_4) + (1-\alpha) \lambda(f^k(r_2), r_6) \right), \text{ with } \alpha \in [0,1].$$

With the first term of the formula, the loop r_2 is fitted to the second sub-trajectory r_4 and with the other to the last one (r_6). With the parameter α, it is possible to change the weighting between these terms.

There are many degrees of freedom in this problem formulation: The representation of the trajectories, the set of transformations and the measures. The concrete transformation of a demonstration to a spatial loop is dependent on the choice of those. Since the purpose of this paper is to introduce the idea of spatial loop programming, we restrict ourselves to comparatively simple representations (Section 5.2), transformations and measures (Section 5.3).

5.2 Representation of Sub Trajectories

There are various options for representing trajectories. The most obvious possibility is to let them as they are – an ordered list of points. They also can be represented for example as parameterized primitives (e.g. straight lines, circles ...), as Splines of different degrees or in Fourier space.

---

[4] $m(2) = 6$, $m(i) = 4$, $i \in \{1,3,4\}$;

[5] In the case of other demonstration types, this problem formulation differs slightly.

The body of a loop can consist of a single primitive motion or may be composed of different consecutive primitive motions. Of course, for a given sub-trajectory it depends on the set of primitive motions of the given system, whether it is atomic or composed.

The trajectory in Fig. 4 consists of three demonstrations for a loop (Demonstration Type 2). It is easy to see, that every sub-trajectory has a bulge in its center. Depending on the system's representation of sub-trajectories, this body can either be interpreted as one primitive (e.g. as a parameterized curve) or as composition (e.g. linear movements and a circle segment).

In this paper we start with the simplest loop bodies that consist of one single straight line represented by its two end points.

5.3 Transformations and Measures

The choice of the transformation is highly dependent on the representation of the sub-trajectories. If a sub-trajectory is defined via some points, the transformation can be any transformation that is applicable to points. Such a transformation might treat the different points of a sub-trajectory in a different manner or in the same way. The former allows the scaling of lines. In this paper we concentrate on translations and rotations.

For comparing two sub-trajectories, a measure is needed to describe the degree of similarity of the sub-trajectories. In the case of straight lines the most obvious measure is the sum of the Euclidean distances between the two pairs of corresponding end points. Like the transformations, the measure is also highly dependent on the representation of the sub-trajectories. Both have to be compatible with the representation.

5.4 Implementation

The concept was implemented on the industrial robot Stäubli RX130. The robot is equipped with a force/torque sensor (JR3 KMSi 90M31-63) on its wrist, allowing the user to demonstrate the trajectory. The cuts are set with a button on the teach pendant.

Given the trajectory and the cuts, the system has to decide what type of transformation fits best. Since we restrict ourselves to translation and rotation and do not consider combinations of them, we classify according to geometrical criteria. This section concentrates on transformations that work on the whole trajectory and not on the individual points.

Translation and rotation are rigid body transformations, so a common criterion for them is that the length of the strokes does no vary except for the user's imprecision.

For pure translations with Demonstration Types 1, 3 or 4 it must be checked, whether the two strokes have the same orientation.

For a Demonstration of Type 2 we regard the angle between the cutting points at the beginning and at the end of the sub-trajectories and the distance between them. Let γ_j be the point of a sub-trajectory that emerged from cut c_j and Θ_1, Θ_2,, $\Theta_3 \in [0,1]$ be pre-set thresholds. To be classified as translation, a demonstration has to comply with the following conditions:6

$$\| \gamma_5 - \gamma_1 \| - \| \gamma_6 - \gamma_2 \| \leq \Theta_1 ; \quad \| \gamma_3 - \gamma_1 \| - \| \gamma_4 - \gamma_2 \| \leq \Theta_1 ;$$

$$< \frac{\gamma_5 - \gamma_1}{\| \gamma_5 - \gamma_1 \|}, \frac{\gamma_3 - \gamma_1}{\| \gamma_3 - \gamma_1 \|} > \geq \Theta_2 ; \quad < \frac{\gamma_6 - \gamma_2}{\| \gamma_6 - \gamma_2 \|}, \frac{\gamma_4 - \gamma_2}{\| \gamma_4 - \gamma_2 \|} > \geq \Theta_3 ;$$

The threshold Θ_1 should be near zero while the others should be near one. The first two formulas assure that both points are translated equally. If they were not, the transformation can not be a translation. The other two formulas measure the angles $\sphericalangle(\gamma_3, \gamma_1, \gamma_5)$ respectively $\sphericalangle(\gamma_4, \gamma_2, \gamma_6)$. If

$$\text{round}\left(\frac{\| \gamma_5 - \gamma_1 \|}{\| \gamma_3 - \gamma_1 \|} \right) = \text{round}\left(\frac{\| \gamma_6 - \gamma_2 \|}{\| \gamma_4 - \gamma_2 \|} \right) = k,$$

there are $k + 1$ loop passes altogether. The new points of the spatial loop can now be calculated easily from the direction of $\gamma_5 - \gamma_1$ and k. A similar condition can be established for rotations but it is skipped here for lack of space.

As soon as a demonstration is assigned to a transformation, the points for all loop cycles can be calculated and used for the generation of the robot movement. The transfer movements consist of an approach to the first point of a line and a depart movement from the second point.

6 Experiments and Results

We demonstrated translational spatial loops with this system with Demonstration Types 1 to 3. We skipped Demonstration Type 4, since it is symmetrical to Demonstration Type 1. The robot was programmed to draw six straight lines with a resilient ball pen. The demonstrations were carried out via the zero-force control and the cutting points were set via the teach pendant. Figure 5 schematically shows the strokes the robot should be programmed to draw. At first, the robot drew a master on a fixed piece of paper. This provided the basis for measuring the deviation of the intuitively programmed loop from the textually programmed loop used for the master. We took the Euclidean distance in x- and y-direction (i.e. the points were projected onto a plane parallel to the table and perpendicular to the axis of the pen) as measure. The deviation in the z-direction was not included because of the compliance of the pen. The user then programmed the loop by repeating the strokes as accurately as possible by grabbing the pen (see Figure 6). For measuring the deviation we calculated the mean of all deviations of corresponding points of the ideal loop and the programmed loop. For each experiment and demonstration type, six experiments were conducted.

---

[6] The symbol ||•|| denotes the Euclidean norm and <•,•> is the inner product.

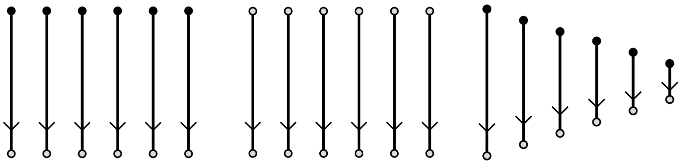


Fig. 5 Three experiments: pure point-based translation, pure translation working on the whole line and scaling of the lines by point-based translation.

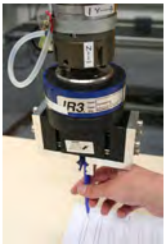


Fig. 6 The user grabs the pen and moves the robot.

In the first two experiments the strokes are simply translated. The first experiment was conducted with a point-based transformation, i.e. the translations for the start points and the end points are deduced separately. For the second experiment an implementation was chosen that calculated the translation vector from the mean of the two vectors derived with the implementation from the first experiment. The third experiment was conducted only for point-based transformation. With each cycle, the stroke becomes shorter, the points approach each other. This is a kind of scaling, but it is the result of the point-based transformation. This could not be achieved by a rigid body transformation that works on the whole line.

The diagrams in Figure 7 show the mean error of all points of each experiment. It is in evidence that Demonstration Type 1 is much more error-prone, since the error in the demonstration accumulates with the number of cycles while for Type 3 the error of the execution is limited by the error in the demonstrations. Demonstration Type 2 is dependent on the choice of parameter α. In our experiment, $\alpha = 0.5$, since we take the mean of the two vectors. For $\alpha = 0$, Demonstration Type 2 is like Demonstration Type 3 except that sub-trajectory 4 is used to derive the number of cycles instead of numerical input. For $\alpha = 1$ the same holds with Demonstration Type 1 and sub-trajectory 6. This explains why Demonstration Type 2 lies in between the other two demonstration types in most cases.

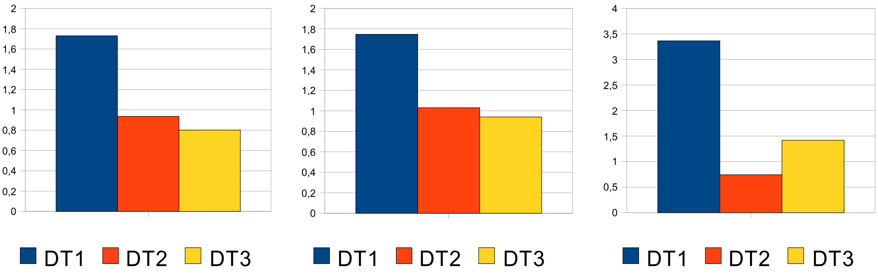


Fig. 7 The mean deviation in mm for the three experiments and for Demonstration Types 1, 2 and 3, measured in the x-y-plane.

7 Conclusions

We introduced the concept of intuitive spatial loop programming, where the parameters of a loop are derived automatically. We identified four demonstration types and compared them according to the uniqueness of the demonstration and the intuitiveness. Demonstration Type 2 turned out to be less ambiguous than the other types. After the comparison we analyzed the different components needed for spatial loop programming and presented possible candidates to be used. We implemented the concept and showed the applicability with an experiment.

Future work will focus on further kinds of transformations, particularly on combined transformations. Pushing a button to segment the sub-trajectories turned out to be annoying and error-prone, so an automatic segmentation will be considered. Since spatial loop programming was only tested on pure trajectories we intend to augment the concept with further sensor data such as forces.

References

1. Acker, J., Kahl, B., Henrich, D.: Environment Guided Handling of Deformable Linear Objects: From Task Demonstration to Task Execution. In: 37th International Symposium on Robotics (ISR, 4th German Conference on Robotics), Robotik, München, Germany (2006)
2. Aleotti, J., Caselli, S., Reggiani, M.: Toward Programming of Assembly Tasks by Demonstration in Virtual Environments. In: 12th IEEE Workshop Robot and Human Interactive Communication, San Francisco, CA, October 31, November 2 (2003)
3. Böhm, C., Jacopini, G.: Flow diagrams, turing machines and languages with only two formation rules. Communications of the ACM 9(5), 366–371 (1966)
4. Brageul, D., Vukanovic, S., MacDonald, B.A.: An Intuitive Interface for a Cognitive Programming By Demonstration System. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 3570–3575 (May 2008)
5. Calinon, S., Guenter, F., Billard, A.: On Learning, Representing, and Generalizing a Task in a Humanoid Robot. IEEE Trans. on Systems, Man and Cybernetics, Part B 37(2), 286–298 (2007)
6. Croitoru, A., Agouris, P., Stefanidis, A.: 3D Trajectory Matching By Pose Normalization. In: ACM international workshop on Geographic Information Systems, pp. 153–162 (November 2005)
7. Dillmann, R., Rogalla, O., Ehrenmann, M., Zöllner, R., Bordegoni, M.: Learning Robot Behaviour and Skills based on Human Demonstration and Advice: the Machine Learning Paradigm. In: 9th International Symposium of Robotics Research (ISSR 1999), Snowbird, UT, USA, October 1999, pp. 229–238 (1999)
8. Hein, B., Hensel, M., Wörn, H.: Intuitive and Model-based On-line Programming of Industrial Robots: A Modular On-line Programming Environment. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 3952–3957 (May 2008)
9. Kahl, B., Henrich, D.: Virtual Robot Programming for Deformable Linear Objects: System concept and Prototype Implementation. In: 12th International Symposium on Measurement and Control in Robotics (ISMCR 2002), Bourges/France (June 2002)

10. Mayer, H., Burschka, D., Knoll, A., Braun, E.U., Lange, R., Bauernschmitt, R.: Human-Machine Skill Transfer Extended by a Scaffolding Framework. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2866–2871 (May 2008
11. Pardowitz, M., Glaser, B., Dillmann, R.: Learning Repetitive Robot Programs From Demonstrations Using Version Space Algebra. In: The 13th IASTED International Conference on Robotics and Applications (2007), `http://wwwiaim.ira.uka.de/data/File/Publications/RA07-VSA.pdf`
12. Schraft, R.D., Meyer, C.: The Need for an Intuitive Teaching Method for Small and Medium Enterprises. In: ISR 2006 - ROBOTIK 2006: Proceedings of the Joint Conference on Robotics, May 15-17, Munich (2006)
13. Waringo, M., Henrich, D.: Efficient Smoothing of Piecewise Linear Paths with Minimal Deviation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3867–3872 (August 2006)

Model-Based Programming “by Demonstration”– Fast Setup of Robot Systems (ProDemo)

Jürgen Roßmann, Henning Ruf, and Christian Schlette

Abstract. This article describes a new integrated approach to robot programming which combines online and offline methods in an efficient, synergetic way. It aims at reducing the cost, the effort and the steepness of the learning curve to set up robotic systems, which are key issues to support the economic use of robots in small and medium enterprises. The innovative approach of the system lies in the use of a task-oriented description and modeling of the work cell as well as the intuitive commanding of the program flow. The input of a tracking system is used to define trajectories and to model obstacles in the work cell. All tasks are coordinated intuitively via a Graphical User Interface, which includes visual programming. A simulation system is then used for collision checking, visualization and optimization of the programs. This paper focuses on MMI’s developments of the GUI and the modeling capabilities of the system.

1 Introduction

ProDemo is a research project funded by the German Federation of Industrial Research Associations (AiF), which is a non-profit association for the promotion of applied research and development. The project is accompanied by a consulting committee consisting of representatives from small and medium enterprises (SME) that are interested in the use of automation technology. Our (MMI) main research partner in ProDemo is the Laboratory for Machine Tools and Production Engineering (WZL) of RWTH Aachen University. Besides other contributions, WZL developed a low budget, robot-based optical tracking tool for the use in ProDemo.

Online and offline are two different approaches to programming robot systems [10]. With the conventional *Teach-in* method, poses are collected by moving the robot to the intended positions and saving the joint values as future targets for the robot. Then, program logic has to be defined by coding in a

---

Jürgen Roßmann · Henning Ruf · Christian Schlette
Institute of Man-Machine-Interaction,
RWTH Aachen, Ahornstraße 55, 52074 Aachen, Germany,
e-mail: `{rossmann,ruf,schlette}@mmi.rwth-aachen.de`

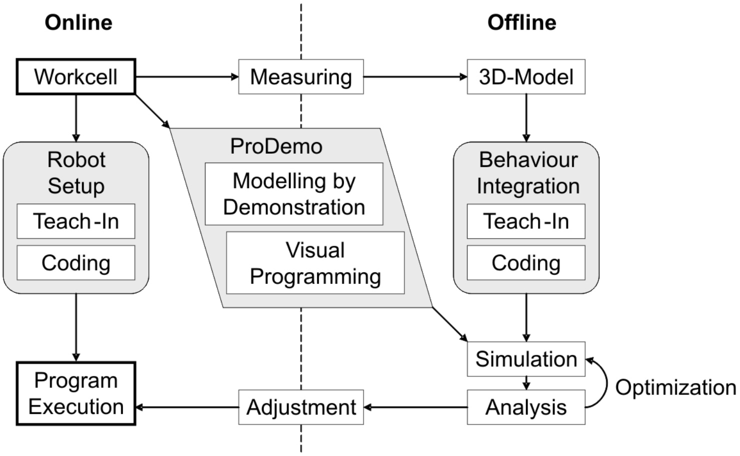


Fig. 1 Robot programming methods: online, offline and ProDemo.

manufacturer-specific programming language. With the offline method the same steps are carried out virtually in a simulated work cell. This comes with the advantages of collision detection, visualization, optimization and programming while the real robot system keeps being operational. The disadvantage of the offline method is the additional need to measure and model the work cell, which is time consuming and requires expert skills. The additional effort for modeling is tolerated only if the modeling of the work cell can be achieved easily and fast [9]. The offline approach to robot programming needs a simulation system as a basis, mainly for collision detection and visualization (Fig. 1).

The ProDemo project uses the CIROS simulation environment [3], which is being developed by RIF e.V., Germany. The Institute of Man-Machine-Interaction uses the CIROS environment as a basis for several projects, not only in robotics but in diverse fields such as *virtual reality* [8], geographic information systems (GIS) or work machine simulation. The method of *Projective Virtual Reality* [4] was developed for space robotics and allows the remote controlling of robots in space by using Virtual Reality.

The term “Programming by Demonstration” describes an intuitive interaction technique where a user carries out a certain task which is then repeated by the robot. One approach for teaching the task is moving the robot equipped with special force sensors manually [6], another is tracking the motion of a tool or a work piece with external sensors [5], which makes the collecting of spatial information much easier. In most works, Programming by Demonstration is only used to capture the motion of the user in order to repeat it by the robot in exactly the same way. Our approach is more similar to the conventional *Teach-in* insofar that the user only defines waypoints instead of complete trajectories. This way, it is easier to find motion segments that can be altered, reused or combined with program logic in a later stage of the programming process.

The spatial information gained by an external tracking system can be visualized in an Augmented Reality system, which allows the user to see the virtual and real objects at the same time [11] A method for defining program logic without the knowledge of special programming language lies in the use of graphical representations. Both research institutes participating in ProDemo gained experience with visual programming in former projects [2, 7]. Other authors developed a complete style guide for manufacturer-independent, icon-based programming [1].

These works focus on particular aspects of intuitive robot programming and flexible operation of automation technology. In contrast to that, our new approach integrates the advantages of the offline and online methods in a single system: the setup is done directly in the work cell and at the same time the optimization capacities of the simulation system are used without the time consuming step of modeling the work cell from the scratch in full detail (Fig. 1).

The measurement process required to collect spatial data is automated using a tracking system. The output of a pointing device is mapped to a cursor object in the simulation corresponding to it. This cursor is used for the definition of trajectories and for modeling purposes. Program logic is defined by using a visual program editor, while spatial information is included in the program by assigning collected poses to *Move* command blocks. The resulting visual program flow can be automatically translated into robot code and tested with the simulation system. After checking for collisions and other optimizations of the program, it is transferred to the robot control to be executed in reality. The process from measuring poses to the upload of the final program is shown in figure 2.

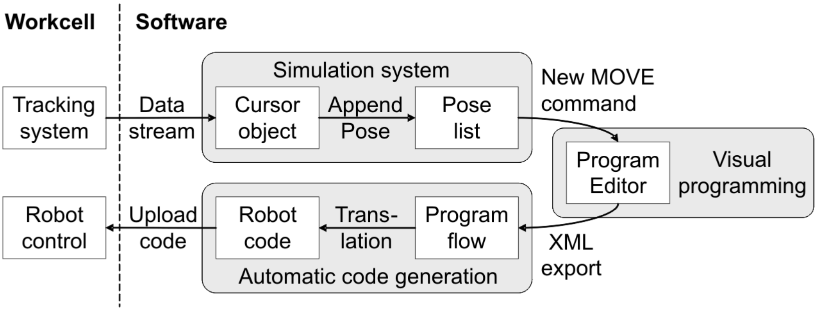

Workcell
Software
Simulation system
Tracking system
Data stream
Cursor object
Append Pose
Pose list
New MOVE command
Program Editor
Visual programming
Robot control
Upload code
Robot code
Trans-lation
Program flow
XML export
Automatic code generation

Fig. 2 Processing chain from tracking data to executable program code.

This paper is organized as follows: First, the GUI concept of the ProDemo system is presented in section 2, followed by a discussion of the modeling capabilities in section 3 and visual programming in section 4. Some early experimental results are presented in section 5 before the conclusion, section 6.

2 User Interface

The user interface is designed for intuitive usage and easy access. It consists of three major sections (Fig. 3). The *Tree Menu* is a new hierarchical menu structure that is used to present the available commands to the user in a given situation or step in the process. In terms of GUI programming, the *Tree Menu* is a mixture of the concepts *Menu*, *Toolbox* and *Wizard* and was developed for ProDemo in order to address the non-expert staff of SMEs. Every menu level can contain commands and submenus, while only the commands of the currently opened submenu are shown. This allows the presentation of many commands in a single menu while always having the option to return to each previous menu with a single click. For testing we use a tracking system featuring a pointing device with a thumb stick and additional buttons. These buttons can be used for navigation in the *Tree Menu* while operating in the work cell. For this purpose the text size of the menu is bigger than average so that it can be read even from longer distances. Next to the *Tree Menu* is an area for additional controls depending on the currently selected submenu, for example lists of available pose, lists of available CAD models for import or wizard-like instructions.

The most prominent section of the GUI is the 3D visualization of the simulated scene. The scene contains virtual representations of the robot and other machines in the work cell as well as graphical metaphors to show system states and relevant properties of the objects:

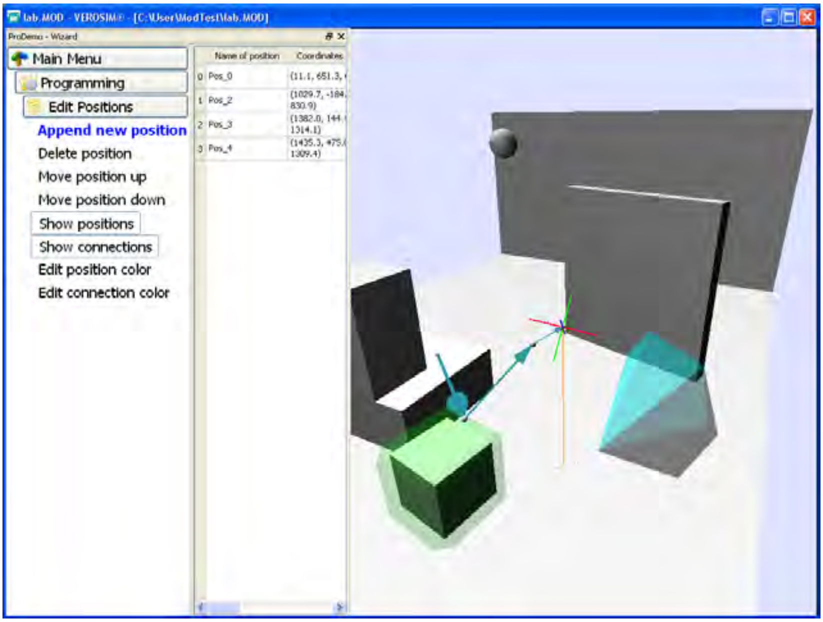
- Poses are shown as spheres
- Trajectories are shown as a series arrows connecting poses
- A cursor object shows the current position of tracked pointing device
- Transparent hulls show coarsely defined collision hulls for obstacles
- Inflated hulls show the amount of uncertainty that results from positioning with a pointing device

3 Modeling by Demonstration

The goal of modeling by demonstration is the fast acquisition of a work cell model without the use of specialized modeling software. Therefore, the model is only created as detailed as necessary for the intended interactions such as collision detection. A tracking system is used to acquire spatial data in order to avoid the elaborate measuring of the work cell. The system combines different approaches to modeling. First, it is possible to import detailed CAD models of machines and other components. A second way of modeling is assembling objects from geometric primitives such as boxes and cylinders. A third way of modeling is the coarse definition of obstacles with convex hulls generated from point clouds.

Import of CAD-Models

For simulation systems such as CIROS it is well-known practice to provide libraries of standard objects that are common in industrial applications. In the ProDemo project we extended the use of standard CAD libraries by the capability to place the inserted objects in the work cell by using a pointing device. To define the placement that corresponds to the object position in the real world, the user has to designate at least three key points. A number of candidates for key points are stored in the model that is to be inserted. After the user has made his choice from the library, the new object is pre-inserted in the middle of the work cell, using


lab.MOD - VEROSIM® - [C:\User\ModTest\lab.MOD]
ProDemo - Wizard
Main Menu
Programming
Edit Positions
Append new position
Delete position
Move position up
Move position down
Show positions
Show connections
Edit position color
Edit connection color

	Name of position	Coordinates
0	Pos_0	(11.1, 651.3,
1	Pos_2	(1029.7, -184, 830.9)
2	Pos_3	(1382.0, 144 1314.1)
3	Pos_4	(1435.3, 475. 1309.4)

Fig. 3 User Interface partition: Tree Menu, context-dependent area with pose list, 3D visualization using different metaphors (arrows for trajectory, cursor object, convex collision hull, box with collision hull displaying uncertainty).

transparency to visualize the temporary state of the insertion (Fig. 4). Using the *Tree Menu*, the user can now switch between the key points of the object and choose the ones that are easy to reach in the work cell. After three points have been registered, the positioning is executed, but not finished. Any additional point raises the precision of the placement and obvious errors can be corrected. After an initial placement, the user has the option to accept the result or to cancel the insertion. If accepted, the inserted object looses its transparency and gains an inflated hull to visualize the potential error that originated from the placement process.

Modelling with Geometric Primitives

Many common objects can be easily approximated by an assembly of geometric primitives such as boxes, cylinders and spheres. In ProDemo, the placement and the definition of geometric attributes of primitives are carried out at the same time by specifying characteristic points with the pointing device. After the user has chosen a geometry type from the *Tree Menu*, the context dependent area next to it shows an instruction which point to input next. These points do not necessarily correspond to corners of the object, but help to define its geometric parameters. For example, the first two points used for modeling a box define one edge. The third point defines the depth of the first face and does not have to be located on the edge, but can be anywhere on the straight line originating from it. A fourth point is necessary to define the height of the box in a similar way.

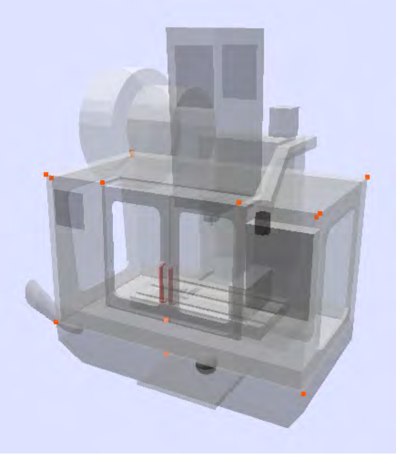


Fig. 4 Visualization of key points for the placement of imported CAD models.

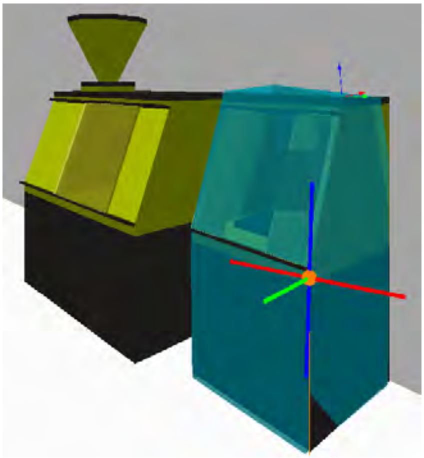


Fig. 5 Coarse approximation of a tool machine with a convex hull.

Modeling Convex Hulls

For many applications, the exact geometry of obstacles does not have to be known in full detail, for example if there is a complex machine in the work cell the robot does not interact with. In that case neither an exact CAD model nor an obstacle object assembled from geometric primitives is necessary. The easiest way to define a coarse obstacle is by using a convex hull. An advantage of this kind of geometry is that the order of point insertion does not matter. The user starts modeling by choosing the convex hull option from the *Tree Menu* and then extends the obstacle point by point until it resembles the intended shape close enough (Fig. 5). This method is especially easy because it is not required to plan the modeling in advance. With every new point the hull geometry is computed automatically from the set of input points. Naturally, it is impossible to model concave objects this way.

Integration with the Simulation System

One of the main advantages of using a simulation system is the ability to detect collisions between geometric objects while a robot program is being executed. The computation is not carried out between the objects themselves, but between collision hulls with a certain distance to the surface to the objects. These distances can be parameterized to consider safety intervals and the possible error of position measurements with a tracking system. A potential problem with modeling by demonstration is the possible loss of precision compared to exact modeling with a CAD system. Therefore the visualization of uncertainty is vitally important. The collision hulls discussed before are not only used for computations but for visualization as well. They are rendered transparently in order not to obscure the original object (Fig. 7 right). The simulation results can be visualized even more realistically by using a VR-system, which is supported by the CIROS environment [3] we used in our setup.

4 Visual Programming

The system uses a graphical editor that also enables non-expert programmers to define the intended program logic of the robot. Beyond defining movements, basic logic programming is possible on the basis of inputs and outputs. Conditional statements and break conditions of loops can be defined based on input values. Also input and output values can be queried or set as an interface to external components, which enables basic sensor integration. If the user opens the submenu for visual programming in the *Tree Menu*, a toolbox with all available command blocks appears in the area next to it. It is possible to assemble a program flow by dragging the icons from the toolbox to an appropriate position in the programming area (Fig. 7). The command set in use follows the Morpha Style Guide [1]. Each command belongs to one of the four color-coded groups *Program Control*, *Robot Motion*, *Input / Output* and *Application*.

As a prerequisite for execution, every program needs to be compiled by the simulation system. To decouple from any specific manufacturer-dependent programming language, the visual program editor uses a generic output in XML. In a second step, an external tool is used to translate this output into the specific programming language of a given robot. So far, only an elementary set of commands can be interpreted by this tool, which maps blocks of XML-tags to corresponding lines of code in the designated robot language.

Trajectories are defined as a series of spatial poses. The user can create new poses with the current position and orientation of the cursor object with a command from the Tree Menu. The order of move commands in the program editor defines a trajectory for robot motion. There are different blocks for each kind of movement (LIN, PTP) and their speed can be specified as well as additional options, for example continuous movement.

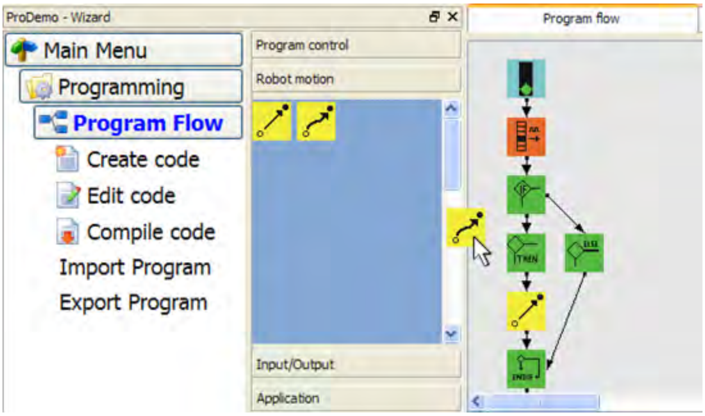


Fig. 6 Visual Programming using Drag and Drop to assemble the program flow.

5 Experimental Results

In cooperation with WZL, we set up a first test bed using an industrial robot with five degrees of freedom. A conventional camera is attached to the robot for tracking purposes. The work cell consists of a work piece carrier and milling machine (Fig. 7). The robot has to perform a simple pick-and-place task: taking a box-shaped work piece from the carrier, placing it into the wrench inside the milling machine, waiting for the execution of the process and after that taking it out and putting it back to the work piece carrier. Besides the opening and closing of the gripper, the task requires I/O operations when checking the state of the machine door and waiting for the machine to finish. In the setup, a pointing device is used for modeling. For the definition of poses a work piece equipped with a marker is tracked directly.

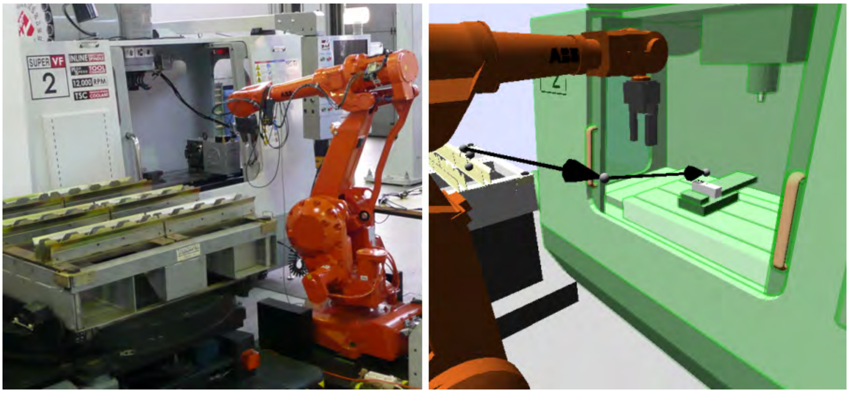


Fig. 7 Test scenario with a pick-and-place task in reality and in the simulation.

The following observations have been made with the test bed:

- Tracking system: Flexibility and good coverage are required. Ultrasonic systems are not applicable in industrial environments with noisy heavy machinery. The sensors need a good coverage of the work cell space because it can be necessary to track objects inside machines or on the ground.
- Placement of CAD models: The precise locating of a certain point that is highlighted in the 3D visualization can be difficult if it does not lie on a sharp edge of the real object. Also the tip shape of the pointing device influences the measurement. We are planning to measure the variation in precision of different users trying to locate the same point.
- Modeling with primitives: The modeling of boxes is easy and intuitive. Cylinders and Spheres are more difficult because they cannot be reasonably modeled by indicating their center points, if these are not reachable or identifiable. Instead, several points on the surface have to be indicated.

- Obstacle definition with convex hulls: The method is easy but unpractical when it is necessary to define points below knee level. For modeling obstacles standing on the ground a new feature could be implemented that extrudes a convex hull to the floor.
- Tracking of poses: For the tracking of a work piece it is necessary to define a gripping pose which specifies the position of the work piece coordinate system (the marker for optical tracking) relative to Tool Center Point (TCP) of the robot when gripping it. In the test scenario, that offset was determined with a user-guided calibration process.
- Suitability of robot hardware: The Tracking system developed by WZL requires an instantaneous influence on the robot control to move the camera during the tracking process. The interface of the robot we used in our initial setup proved not to be fast enough for the fluid motion aimed for. Therefore, our next test bed will be equipped with another robot, providing a more efficient access to robot control.
- Visual programming: The concept is easy enough to understand for non-professional users. We are planning to improve the synchronization between the program editor, the position list and the simulation process. As a result, the programming block that is currently being executed by the simulation will be highlighted to make understanding and debugging of the program easier.
- Overall precision: The precision we achieved with an optical marker-based tracking system is sufficient to define collision free trajectories. But in regions of the work cell where the robot manipulates objects in narrow spaces it can still be required to correct the poses.

6 Conclusion

This paper presents a new integrated approach to robot programming that combines methods from online and offline robot programming in a synergetic way. After 1½ years into the project, the consulting committee valued the newly developed methods as being promising for the application in SMEs. They can help non-expert technical staff to benefit from a simulation system's capabilities such as collision detection, visualization and optimization, without the time consuming steps of measuring the work cell in great detail and coding in a specific robot programming language. The major contribution of this work is the application of new ideas to " robust" graphical modeling and the seamless integration of advanced techniques for modeling and programming into a new, intuitively operable, task-oriented user interface. Compared to conventional approaches, the setup process for robot work cells using the developed system can be carried out safer, in less time and with less expert knowledge. Therefore, the new approach appears to be a promising enabling technology for small and medium enterprises to whom the easy and efficient setup of automation technology has not yet been available.

References

1. Bischoff, R., Kazi, A., Seyfarth, M.: The MORPHA style guide for icon-based programming. In: Proceedings of 11th IEEE International Workshop on Robot and Human Interactive Communication (2002)
2. Brecher, C., et al.: Intuitiv bedienbare Programmiersysteme zur effizienten Programmierung von Handhabungsaufgaben. In: Robotik 2004: Leistungsstand – Anwendungen – Visionen – Trends, VDI-Berichte, vol. 1841, pp. 303–310 (2004)
3. CIROS - Virtual Engineering (2008), `http://www.ciros-engineering.com` (Accessed, 20 March 2009)
4. Freund, E., Roßmann, J.: Projective Virtual Reality: Bridging the Gap between Virtual Reality and Robotics". IEEE Transaction on Robotics and Automation 15(3), 411–422 (1999)
5. Maeda, Y., Ishido, N., Kikuchi, H., Arai, T.: Teaching of Grasp/Graspless Manipulation for Industrial Robots by Human Demonstration. In: Proceedings of the 2002 IEEE/RSJ Int. Conference on Intelligent Robots and Systems EPFL, Lausanne, Switzerland, October 2002, pp. 1523–1528 (2002)
6. Meyer, C., Hollmann, R., Parlitz, C., Haegele, M.: Programmieren durch Vormachen für Assistenzsysteme - Schweiß- und Klebebahnen intuitiv programmieren. In: it - Information Technology, p. 49 (2007)
7. Roßmann, J., Schluse, M., Jung, T.: Multi-modal graphical programming of automation components in a 3D-simulation environment. ISR/Robotik, München (2006)
8. Roßmann, J.: New Man-Machine-Interaction concepts: Virtual Reality meets Robotics. In: Proceedings of the EUROBOT Conference. LNCS, pp. 89–96. Springer, Heidelberg (2008)
9. Taylor, S.: Distributed Simulation and Industry: Potentials and Pitfalls, In: Proceedings of the 2002 Winter Simulation Conference (2002)
10. Weck, M.: Werkzeugmaschinen. Automatisierung von Maschinen und Anlagen. Bd. 4. 5., erw. Aufl. Springer, Heidelberg (2001)
11. Zäh, M., Vogl, W., Munzert, U.: Beschleunigte Programmierung von Industrierobotern. Augmented-Reality-Einsatz zur intuitiven Mensch-Maschine-Interaktion. In: wt Werkstattstechnik online, Jahrgang, vol. 94(9), pp. 438–441 (2004)

New Concept for Wrist Design of the Humanoid Robot ARMAR

Albert Albers, Jens Ottnad, and Christian Sander

Abstract. The development of a humanoid robot within the scope of the collaborative research centre 588 has the objective of creating a machine that can cooperate with humans closely. In contrast to industrial robots – for which mechanical rigidity, precision or high velocities are primary requirements – the key aspects here are prevention of hazards to users, a motion space that corresponds to that of human beings, and a lightweight design. In order to meet these requirements, the robot must have humanlike appearance, motion space, and dexterity. This article gives insight into the design of a new wrist for the next generation of the humanoid robot ARMAR. The goals of the development project are both to improve the motion space and to achieve a humanlike appearance. The new mechanical design is described in detail completed by a study of a first prototype.

1 Introduction

The mechatronic design of a humanoid robot is fundamentally different from that of industrial robots. The key goal for humanoid robots is not accuracy, but the ability to cooperate with humans. In order to enable a robot to interact with humans, high standards are set for sensors and control of its movements. The robot's kinematic properties and range of movements must be adjusted to humans and their environment [1].

The collaborative research centre 588 "Humanoid Robots – learning and cooperating multi-modal robots" was established by the "Deutsche Forschungsgemeinschaft" (DFG) in Karlsruhe in May 2001. In this project, scientists from different academic fields develop concepts, methods, and concrete mechatronic components for a humanoid robot called ARMAR (see figure 1) that can share its working space with humans. The long-term target is the interactive work of robots and humans to jointly accomplish specified tasks. Communication between robots and humans should be possible in different ways, including speech, touch, and gestures, thus allowing humans to interact with the robots easily and intuitively.

Albert Albers · Jens Ottnad · Christian Sander
Institute of Product Development Karlsruhe, University of Karlsruhe,
Kaiserstraße 10, 76131 Karlsruhe, Germany,
e-mail: `{albers,ottnad,sander}@ipek.uka.de}`

Fig. 1 Upper body of the humanoid robot ARMAR III.

The focus of this paper is the design of a new wrist for the next generation of the humanoid robot ARMAR. The wrist serves as the connection between forearm and hand.

The wrist of the current version, ARMAR III, has two degrees of freedom [2] and its rotational axes intersect in one point. ARMAR III has the ability to move the wrist to the side (± 60°) as well as up and down (± 30°). This was realized by a universal joint in a compact construction. One of the main points of criticism on the current version of the wrist is the offset between the rotational axes and the flange, as shown in figure 2 (left). Due to the joint design, this offset distance is necessary in order to provide the desired range of motion. Also other wrists of humanoid robots show a similar design [3, 4, 5, 6]. That offset is even greater due to the placement of the 6-axis force and torque sensor. The resulting movement, a circular path performed by the carpus, does not appear as a humanlike motion.

The German Aerospace Centre DLR (Deutsches Zentrum für Luft- und Raumfahrt) has been working on seven degree of freedom robot arms for several years. The wrists of the lightweight arms of the third generation imitate human wrist movements by a pitch-pitch combination with intersecting axes (kardanic). An alternative pitch-roll configuration is also utilized, mainly for applications using tools [7]. Both versions have an offset comparable to the current wrist of ARMAR III.

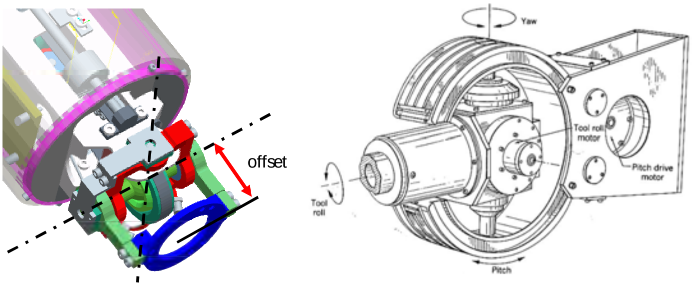


Fig. 2 Offset between the rotational axis and the hand flange at the wrist of the humanoid robot ARMAR III (left) and a concept for a “Powered Wrist Joint” [8] (right).

Henry J. Taylor and Philip N.P. Ibbotson designed a “Powered Wrist Joint” [8] (figure 2 right) in order to load and unload space shuttles. In a smaller version, the basic idea could be reused in humanoid robot’s wrist. The second degree of freedom (pitch) of the wrist is guided by a spherical joint. Such an assembly provides a slim design and relatively wide range of motion. The actuators for the second degree of freedom (yaw) are located directly at the joint; therefore, the drive units are quite simple. On the other hand, miniaturization seems to be very difficult due to the dimensions of common gears and motors.

2 New Concept

2.1 Requirements and Design Goals

Due to the fact that the robot is designed to interact with humans, the ability to move like a human is as important as a human-like appearance; therefore, specific demands [9] on kinematics, dynamics and the design space must be considered. A human wrist consists of many different elements and has a relatively wide range of motion. Figure 3 illustrates the different possible movements of the human wrist along with the corresponding reachable angular positions of the joints.

In order to implement a human-like wrist movement, two orthogonally arranged rotational degrees of freedom are necessary. Both axes are orthogonal to the forearm’s axis and intersect in one point. The two degrees of freedom need to be put in a kinematical series.

The requirements and design goals for a humanoid robot’s wrist can be deduced from the range of motion of the human wrist. The first degree of freedom should have a ±30° range of motion and the second ±90°.

The wrist will be attached to the forearm’s structure on one side and provides the connection to the hand. It should be possible to disconnect the mechanical joint between the hand and wrist in a simple way in order to enable a modular design. To measure the load on the hand, a 6-axis force and torque sensor must be fitted between wrist and hand. The electronic cables and pneumatic tubes supplying power to the hand actuators are the same as those used in the previous models of ARMAR [11, 12].

The design space for the robot’s wrist is based on human dimensions as much as possible; therefore, one aim is to keep a sphere of approximately 100 mm in diameter as a boundary. At the same time, the control strategy aims to operate all degrees of freedom as individually as possible.

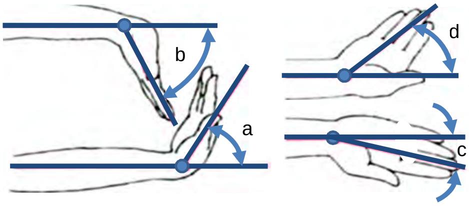


Fig. 3 Human wrist and range of motion: a = palmar flection 70°, b = dorsal flection 90°, c =radial abduction 20°, d = ulnar abduction 40° [10].

In keeping with the standardized drive concept of most modules of the robot, electronic motors are used as source for actuation. The drive units need to be dimensioned for a load of 3 kg. All gears are designed to be free from backlash and not self-locking. But friction, e.g. in case of a loss of power, leads to a slow and damped sinking of the arm instead of abrupt movement. That is of great importance for an interactive application of the robot in a human environment. On the other hand, stick-slip effects in the gears are avoided, which is a clear benefit for the control system.

Finally, the mechanical structures should be as light as possible in order to save energy during dynamic movements. A lower mass of the wrist can contribute significantly to a reduced energy consumption of the whole arm and has a strong influence on the gears and motors used for the drive units for the elbow and shoulder degrees of freedom.

2.2 Concepts

A reduction of the current wrist's length by only minor and simple modifications of certain elements is impossible. The main reason is that the joint design in combination with the drive unit for the second degree of freedom does not allow a mounting of the hand in the rotational axis. Formulated in an abstract way, the development goal is to shift material from the intersection point of the two axes to a different location in order to gain free space in the center position.

As a result of these basic considerations a new concept is depicted in figure 4. It consists of a curved track solution for the first degree of freedom (A). As this first rotation is limited to ± 30°, there is enough space left for the bearings of the second degree of freedom, which may be realized, e.g., by a simple shaft (B). This configuration allows a relatively wide range of motion and a high capability for a reduction of the wrist's length. The challenges for this concept include search for a technical solution for the curved track and designing it for the proper stiffness in the structures.

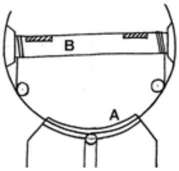


Fig. 4 The principle curved track solution.

2.3 Embodiment Design

By an appropriate design of the shaft (figure 4, named B) it is possible to gain still more space for the 6-axis force and torque sensor. Figure 5 illustrates a cross-section view of the modified shaft which allows efficient sensor integration.

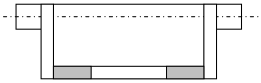


Fig. 5 Basic idea for the shaft of the second DOF of the wrist.

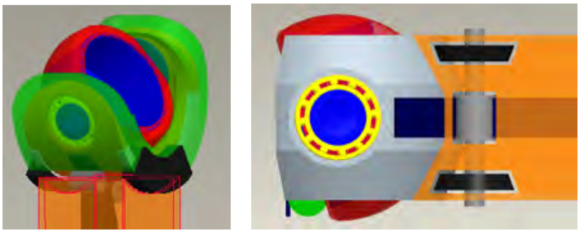


Fig. 6 Basic CAD implementation of the new concept by using a curved guide (left) and the alternative using a roll timing belt combination (right).

The depth of that shell corresponds with the radius of the curved track and enables a mounting of the hand exactly in the point where the rotational axes intersect. This is achieved by shifting the mechanical connection in the negative direction along the center axis of the forearm.

For the technical implementation of the curved track, a curved guide, called HCR manufactured by THK [13] could be selected. Used for medical applications, THK produces curved guides with a radius of approximately 100 mm. From a technical point of view a further reduction of the radius is possible in order to meet the requirements for a humanoid robot's wrist. For economic reasons, however, this was not a feasible option for the collaborative research centre.

Therefore the curved guide is replaced by rollers in combination with a timing belt. This allows the integration of two different functions in one machine element: the timing belt functions as a part of the drive unit while also providing sufficient preload to avoid a gap between the rollers and the track. Figure 6 shows the basic CAD models of each design. To investigate the new concept, different analyses and simulations were conducted and are described in the next section.

3 Simulation

3.1 Basic Geometric Considerations and FEM Simulation

Based on the new concept introduced and explained in section II, an analytic model can be set up. Therefore all geometric parameters based on the nature of the human body have to be adapted to the model. Using parameter optimization the best combination of values for a design proposal can be found in order to achieve reasonable preloads for the belt.

Two main load cases were used whereas the angle of the initiated force φ can vary. Figure 7 illustrates these two load cases. Here the calculated force F (36 N) is the substitute for all external loads and self-weight [2]. M is the appropriate torque resulting from the arm of lever and is about 3.14 Nm. To avoid a displacement of the cap the preload F_V has to be chosen great enough.

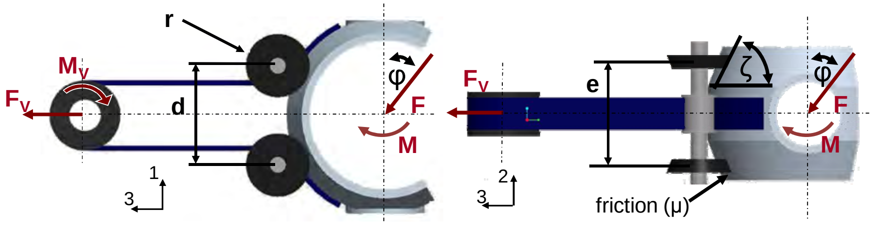


Fig. 7 Load case I (left) and load case II (right) in two different directions.

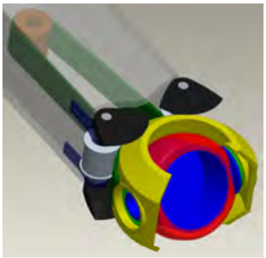
Load case I: The external load F is applied in a variable angle φ towards the vertical line. The maximum required preload for an offset of the bevel wheels (d) of 37.5 mm is about 1 kN. When d is increased to 42.5 mm, the required force is less than 0.63 kN. Thus, the required force decreases by about 37 % when the offset of the beveled wheels is increased by about 21 %. By doubling the distance from 35 mm to 70 mm, the required preload force is reduced by 90 %. The calculated critical angle of the load φ is 36°.

Load case II: Calculations have shown that the influence of the substituted shear force F is negligible for this load case. Therefore only the over-all torque M is used for the analysis. F_V is dependent upon the angle ς and the wheel distance e. The calculated maximum force for the timing belt is 0.28 kN.

The calculated forces are all in a reasonable range compared to the technical elements that can be used for the construction. Consequently the concept can be realized in a physical system with standard bearings and materials. Therefore on the basis of these analytic results a modified CAD model is implemented (see figure 8).

In a next step the CAD model is simulated numerically using finite element method (FEM) in order to gain further information of the system's behavior. Especially the elasticity of the different structures and the resulting interaction effects are of interest. The preload force and the orientation of the external force were varied systematically. The primary object is to get values for the displacement of the cap towards the global coordinate system. All deformable parts are simulated with isotropic material except the timing belt. Due to the fact that the timing belt is composed of a steel cord with polyurethane backing and teethes, an anisotropic material parameter is used in the model. The angle φ of the external load takes the value of 0° and 36° which is identified in the analytic calculation as the most critical. Figure 9 illustrates the result of the FEA.

Fig. 8 CAD model based on the results of the analytical considerations.



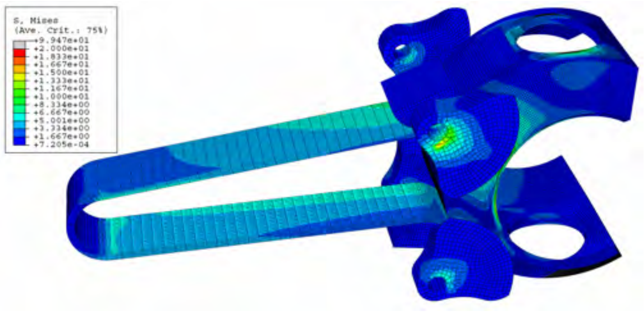


Fig. 9 Stress distribution (von Mises) for load case II.

The stresses, obtained by the FEA show a reasonable distribution. The displacement of the cap tested with high preload forces is minimal due to the FEA. For a preload of 0.6 kN the displacement for load case II is about $4.4 \cdot 10^{-3}$ mm and for load case I $1.39 \cdot 10^{-2}$ mm. Compared with a preload of 1 kN the displacement doesn't highly decrease. For load case II the displacement takes the value of about $4.07 \cdot 10^{-3}$ mm and for load case II $1.29 \cdot 10^{-2}$ mm. These values for the different preloads show that in the range between F_V=0.6 and 1.0 kN only a small increase of positioning accuracy due to less displacement can be reached. But the high additional costs in the construction of the wrist for preloads higher than 0.6 kN can't be justified. For this reason, and for practical implementation, it is not meaningful to use forces greater than 0.6 kN. For preloads lower than 0.25 kN the position deviation increases dramatically and the system becomes statically indeterminate. The displacement of load case I with φ=36° is for every point smallest compared with load case I (φ=0°) and load case II. Therefore, it appears that a preload between 0.25 kN and 0.6 kN would be most suitable.

3.2 Lightweight Design by Topology Optimization

In order to achieve a lightweight design, topology optimization [14] can be very useful for the determination of the basic layout of a new design. It involves the determination of features such as the number, location and shape of holes, and the connectivity of the domain. A new design is determined based upon the design space available, the loads, possible bearings, and materials of which the component is to be composed. A topology optimization scheme is basically an iterative process that integrates a finite element solver and an optimization module. Today it is also a common tool in the industrial design process [15].

For the topology optimization the complete system is disassembled and only the cap with the connecting shaft, still a rigid body element, is used for the simulation. This simplification is necessary to avoid enormous computing time caused by a very fine mesh for the cap and a huge number of load cases. All interactions between the cap and disassembled parts are replaced by corresponding forces and interactions, such as the preload of the timing belt of 450 N. For example five load case combinations are set for three different rotational positions of the second

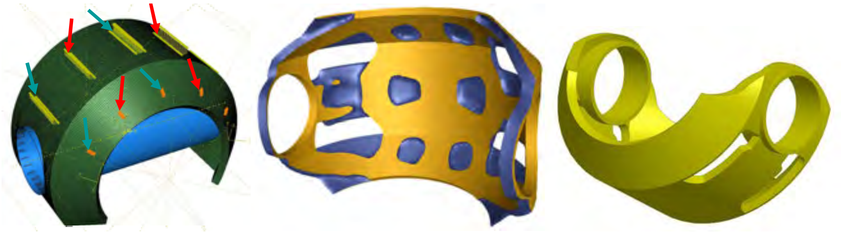


Fig. 10 Design space and FE model (left), design proposal as results of the topology optimization for a lightweight design (middle) and CAD implementation of the cap (right).

DOF (0°initial-red arrows in figure 10, ±30°). In total 15 different configurations of the cap are used for the topology optimization. The design proposal as a result of the topology optimization is shown in figure 10.

The mass is reduced by approximately 45 % of the original design space. For a final design the components (e.g. the bearing carriers) need to be modified and adapted to a certain manufacturing process. The final mass and structure of that part will of course depend also on the used materials but the optimization results can give a first design proposal. On the basis of the design proposals obtained from the topology optimization a CAD model is implemented (see figure 10, right).

3.3 Functional Prototype

Based on the positive results obtained by the different simulations, a functional prototype was developed. That was necessary mainly because different functions were integrated in the toothed belt, which is usually used in a different manner and not all material parameters were available so that estimated values were used.

As the purpose of the prototype is to prove basic functionality of the design, a few simplifications are made. For the bevel wheels complete rolls are used and the cap does not show the holes from the topology optimization for instance. Furthermore, the construction allows the possibility to implement an s-beam force sensor (Lorenz K-25). Figure 11 shows two pictures of the assembled functional prototype with a one kilogram weight attached at the hand's position.

Multiple static and dynamic tests show that this configuration is very accurate and has a high stiffness for small preloads of about 300 N. Hereby the wrist is hand-held at the forearm tube and statically loaded by huge forces between

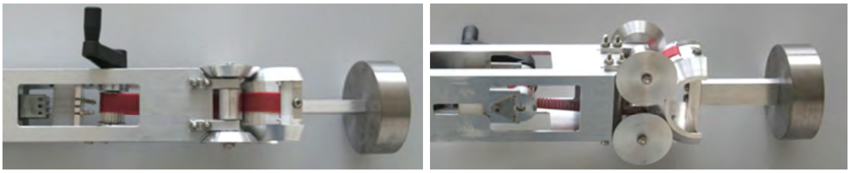


Fig. 11 Functional prototype.

20-80 N or moved dynamically in all different directions. Even for very fast "hand actuated" motions, which were approximately five times of the maximum velocity of the robot arm, the assembly remained free from backlash.

4 Conclusion

In this paper a new concept for humanoid robot's wrist is presented. Especially the different steps of the development process are described. Based on the basic ideas, different analyses and simulations are conducted. Due to the design proposal obtained by the topology optimization, a lightweight design is implemented in the CAD model. Finally, a functional prototype is presented which is a kind of proof of the concept.

The next step will be the integration of the drive units for the second degree of freedom. Here different solutions are possible, e.g. like Bowden Cables or a direct actuation in combination with harmonic drive gears. In order to achieve a further reduction of mass, composite materials may be used for some of the structural components. The new wrist will be developed in the next months and is to be manufactured and assembled during the next year.

Acknowledgment. The work presented in this article is funded by the "Deutsche Forschungsgemeinschaft" (DFG) within the Collaborative Research Centre 588 "Humanoid Robots - Learning and Cooperating Multimodal Robots".

References

1. Schäfer, C.: Entwurf eines anthropomorphen Roboterarms: Kinematik, Arbeitsraumanalyse, Softwaremodellierung, Dissertation Fakultät für Informatik, Universität Karlsruhe (2000)
2. Albers A., Brudniok S., Ottnad J., Sauter Ch., Sedchaicharn K.: Upper Body of a new Humanoid Robot – the Design of Armar III. In: Humanoids 06 - 2006 IEEE-RAS International Conference on Humanoid Robots, Genova, Italy, December 4-6 (2006)
3. `http://www.shadow.org.uk`: The Shadow Robot Company
4. Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., Hirata, M., Akachi, K., Isozumi, T.: Humanoid Robot HRP-2. In: Proc. IEEE Int. Conference on Robotics and Automation, pp. 1083–1090 (2004)
5. Kaneko, K., Harada, K., Kanehiro, F., Miyamori, G., Akachi, K.: Humanoid Robot HRP-3. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Acropolis Convention Center Nice, France, September 22-26 (2008)
6. Park, I.W., Kim, J.Y., Lee, J., Oh, J.H.: Mechanical Design of Humanoid Robot Platform KHR-3 (KAIST Humanoid Robot – 3: HUBO). In: Proc. IEEE-RAS Int. Conference on Humanoid Robots, pp. 321–326 (2005)
7. Albu-Schäffer, A., Haddadin, S., Ott, C., Stemmer, A., Wimböck, T., Hirzinger, G.: The DLR lightweight robot: design and control concepts for robots in human environments. Industrial Robot: An International Journal 34(5) (2007)
8. Rosheim, M.: Robot wrist actuators, 1. Auflage (1989) ISBN 0-471-61595-1

9. Asfour, T.: Sensomotorische Bewegungskoordination zur Handlungsausführung eines humanoiden Roboters, Dissertation Fakultät für Informatik, Universität Karlsruhe (2003)
10. Wirhed: Sportanatomie und Bewegungslehre, Schattauer Verlag, 3. Auflage
11. Beck, S., Lehmann, A., Lotz, Th., Martin, J., Keppler, R., Mikut, R.: Model-based adaptive control of a fluidic actuated robotic hand. In: Proc. GMA-Kongress 2003, VDI-Berichte, vol. 1756, S.65–S.72 (2003)
12. Schulz, S.: Eine neue Adaptiv-Hand-Prothese auf der Basis flexibler Fluidaktoren, Dissertation, Fakultät für Maschinenbau, Universität Karlsruhe, TH (2003)
13. THK, `http://www.thk.com`
14. Bendsoe, M., Sigmund, O.: Topology Optimization – Theory, Methods, Application. Springer, Heidelberg (2003)
15. Pedersen, C.B.W., Allinger, P.: Recent Developments in the Commercial Implementation of Topology Optimization. In: TopoptSYMP2005 - IUTAM-Symposium, Copenhagen, Denmark, pp. 123–132 (2005)

Biological Motivated Control Architecture and Mechatronics for a Human-Like Robot

Thomas Wahl, Sebastian Blank, Tobias Luksch, and Karsten Berns

Abstract. In this paper the lower levels of a biological inspired control architecture for dynamically moving legged robots are presented. This architecture features a hierarchical distributed reflex based high level controller as well as the possibility to adjust the compliance of a stiff actuated joint. Furthermore the respective mechatronical setup for that approach is presented, that includes the actuation and energy storage in parallel springs. The approach is verified on a prototype leg mounted on a vertical slider, that is capable of performing a cyclic squat jump. The reflex based control concept is tested in a physics simulation environment. The experimental validation shows that no series elastic elements are required to receive comparable results with respect to the resulting motion. The low level stiffness controller is implemented on a DSP-board and tested using an experimental setup with two motors.

1 Introduction

Bipedal locomotion has been a field of high interest in robotics during the last decades. This is among other things due to the fact that human environment favors this kind of locomotion over e.g. wheel based approaches. If mastered it offers highly versatile and energy efficient movements that will allow the robot to advance into areas that were not accessible to machines in the past.

Amongst the important aspects of a bipedal robot is a sound concept concerning mechantronics and control aspects. The biological representatives of two-legged locomotion show high energy efficiency and the ability to adapt to external disturbances compared to state-of-the-art technical implementations. Thus, biological

---

Thomas Wahl · Sebastian Blank · Tobias Luksch · Karsten Berns
Robotics Research Lab at the Department of Computer Sciences,
Kaiserslautern University of Technology, P.O. Box 3049, 67653 Kaiserslautern, Germany,
e-mail: `{t_wahl, blank, luksch, berns}@informatik.uni-kl.de`

mechanisms seem to be promising as a guideline for the design process without trying to copy them but rather make use of nature's concepts. The short-term goal is to deduce a system including mechanics, a reflex-based control architecture and a suited actuator control approach that can be used in a walking machine later on.

To evaluate these basic concepts, the work presented here advances a single leg prototype with biological motivated joints and control from an earlier project[1] [15]. The demonstrator is to perform cyclic squat jumps stabilized in a vertical slider. This highly dynamic motion is a meaningful benchmark as it poses considerable demands on the mechatronics and the control system.

2 State of the Art

Contemporary bipedal robots can be classified into two major groups: On the one hand the robots with fixed trajectory planning and no compliance, e.g. ASIMO, LOLA [12] and on the other the robots with elastic actuation respectively passive elastic elements inspired by the passive walkers by Tad McGeer et al. [16]. Those ideas were employed in the actuated passive walkers by e.g. Collins et al. [5] and Anderson et al. [2]. The second group tries to adapt more biological principles. These principles are optimized by evolution for energy efficiency. Elastic actuated robots can be divided once more in two basic approaches to acquire compliance. The first one makes use of specialized hardware with elastic elements while the other realizes the desired behavior using software with compliant control techniques.

A representative for the hardware approach is the Series elastic actuator (SEA) [6, 19]. This method combines an inelastic actuator like DC motor or hydraulic actuator with a fixed elastic element. A drawback of SEAs is non-trivial control of the desired stiffness. Other approaches that control the stiffness independently from the position have to make use of more than one actuator. The mechanically adjustable compliance and controllable equilibrium's position actuator (MACCEPA) [7] uses two separate servos. One servo controls the position and the second servo controls the stiffness of the joint. The drawback of these approaches is that the energy of the second actuator is lost in respect to movement. The fluidic muscles [10] use the antagonistic principle as can be found in biology. The antagonistic principle makes use of the nonlinearity of the elastic elements. Unfortunately this increases the control overhead at the same time.

Their are two major approaches that use a software solution: Virtual Model Control [18] and Impedance Control Methods [1] . Common for all these approaches is a stiff and retardant actuator. Obligatory for a software solution is a very fast sensor-controller-actuator-loop. This is reachable by distributing the control architecture on different CPUs.

---

[1] The initial leg prototype has been developed within the program 'Bionik - Innovationen aus der Natur" of the German Federal Ministry of Education and Research (BMBF).

3 Concept

Based on these biological motivated aspects a system should have the following components:

- actuator that can deliver high torques
- gearbox that allows free swing of the legs
- parallel elastic elements to store energy
- no explicit series elastic elements
- controller to adjust the stiffness of the joints
- reflex layer based on neurological concept
- easy to distribute functional units
- distributed control components for enhanced scalability

3.1 Mechanical Design

In consideration of the fact that the biped performs highly dynamic movements, the delivery of high torques is recommended. These high torques could be produced by either pneumatic or hydraulic actuators as well as DC motors. Due to the autonomy of the system the DC motors are the preferred solution. Another important detail is the controllability of such actuators. The control of pneumatic actuators is highly nonlinear and the antagonistic principle requires a second actuator.

In the swing phase of a walking gait the robot should use its mechanical dynamics to save energy. Hence the gear ratio has to be low. That allows the joint to be non-retardant in comparison to the commonly used harmonic drives.

For the energy storage during the squat phase parallel springs are attached to each joint [4] . These springs may not be too heavy such that all the benefit is compensated by their weight. There are three suitable kinds of springs. Two of these are the mechanical linear and rotary springs. They are very light and easy to handle. The third option is a pneumatic spring: they are slightly heavier and have a nonlinear force response.

3.2 Joint Control

In a jumping sequence an elastic configuration is required to avoid hard impacts at touchdown. Besides that in normal walking gaits the leg should use the mechanical dynamics by just relaxing the joint. On the other hand the joint has to be very stiff when the leg is in the support phase. These cases are nearly impossible to control with passive series elastic elements. Due to that the joint controller adjusts the stiffness of the joint. To perform these highly dynamic tasks the controller is built up hierarchical. The innermost loop has to be a very fast current-controller. The current is directly correlated to the output-torque of the motor. The current measurement is the most difficult for that controller, because it has to be very fast and synchronized to the PWM.

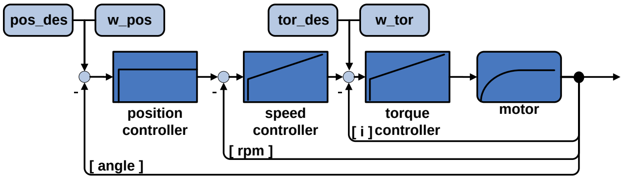


Fig. 1 Schematic view of the hierarchical closed loop controller.

Based on this current controller a speed and a position controller are implemented. A schematic layout and the interfaces of the controller are presented in Fig 1. The speed controller has no interface to the higher reflex layer. Biology shows that there is no need for an exact speed control. The desired position and torque impact of the controller output can be set using respective weight parameters named w_{pos} and w_{tor}. By decreasing w_{pos} the stiffness at a desired position is reduced. w_{tor} is proportional to the influence of a desired torque. This is e.g. the case during the pushoff phase: there is no need for an exact position since the maximum torque is required. In this phase w_{pos} is zero and w_{tor} is one with the maximum desired torque. To hold a fixed desired position w_{pos} is one and w_{tor} is zero.

3.3 *Reflex Control*

The joint control just mentioned presents the interface for the next layers of the control system. Complying with the biological motivation of this work, again concepts transferred from nature should be employed. The system described in the next paragraphs is based on a control approach for bipedal locomotion already published in more detail [14, 13].

Neurological control of cyclic motions seems to be a result of feedback and feedforward components. Bizzi, Ivanenko, and others have analyzed synergies of muscle activity during locomotion and suggest that the nervous system is of hierarchical layout [3, 11, 8, 9]. Based on the current phase of locomotion, coordinated patterns of activity are generated from a central unit and stimulate muscle groups to achieve the desired motion. Depending on this phase, reflex responses are modulated from spinal or supraspinal levels as well [21, 22].

Based on these and other results from biomechanics and neurology, the approach followed in this work suggests a hierarchical network of control modules to generate dynamic locomotion of legged robots. Control units are distributed and local to reduce the modelling effort and the complexity. Reflexes introduce a tight sensor/actor coupling for fast responses to stimuli and can be inhibited or modulated depending on the phase or mode of locomotion as it is the case in biological control. Motor patterns allow for temporal synergies of cooperating joints by generating synchronized torque impulses. No explicit joint trajectories are used so the natural system dynamics can be exploited and natural and energy-efficient motions emerge. Figure 2 illustrates the structure of the proposed approach. Skills represent control

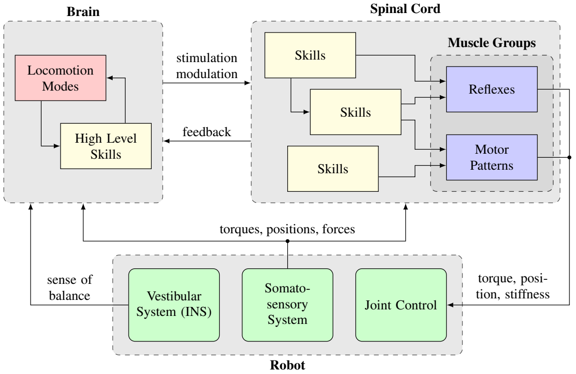

Brain
Spinal Cord
Muscle Groups
Locomotion Modes
High Level Skills
stimulation modulation
feedback
Skills
Skills
Skills
Reflexes
Motor Patterns
torques, positions, forces
sense of balance
Vestibular System (INS)
Somato-sensory System
Joint Control
torque, position, stiffness
Robot

Fig. 2 The proposed control approach is structured as a hierarchical network of skill, reflexes and motor patterns.

units hierarchically above reflexes and motor patterns. The actual control components selected and designed to achieve the aspired motion are based on reflexes, muscle synergies and EMG data found in biological research and adapted to the technical system. The control network is implemented using a behavior-based control framework that was successfully used before on various robots by the authors and others and allows to implement the characteristics just mentioned [20].

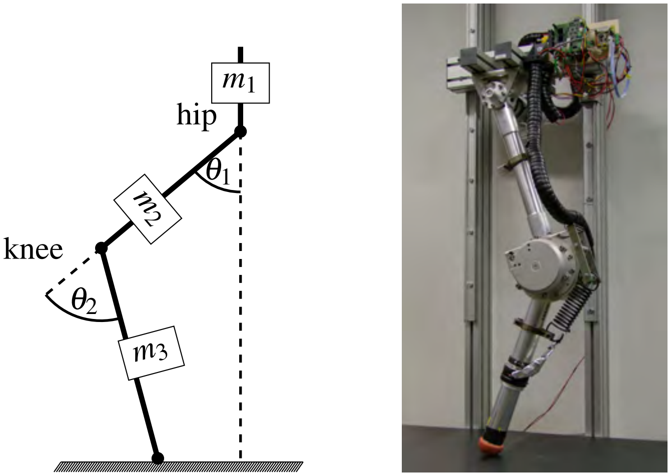
4 Prototype

4.1 System Description

The single leg prototype is comprised of two actuated joints (hip, knee) and two limbs (thigh,shank). The hip joint is fixed to a vertical slider that allows for free movement in z-direction while restricting lateral change of position as well as rotation. In order to emulate the very sparse biological sensor concept the prototype is only equipped with two positional encoders located at each actuated joint and a load cell mounted in the middle of the lower limb. The construction is intended to be highly modular in order to be able to replace components or expand the kinematic setup with e.g. a passive foot construction. The dimension of the leg is human like. The height of the leg is nearly one meter and the weight is around 16 kg.

In order to acquire valid simulation results a dynamic model of the leg is needed to represent the actual one as closely as possible while allowing to introduce a few simplifications to reduce the computational overhead. Thus, the weight points assumed for each part are located in a position that represents the actual load distribution in the best possible way. The model is presented in figure 3(a).

The actuated joints consist of a DC motor, a gearbox with low gear ratio and parallel elastic elements. Finding a suited actuator for this kind of application is not an easy task since the restrictions in respect of dynamic properties can only be met



(a) Dynamic simulation model (b) Photo of the prototype leg

Fig. 3 Schematic model used for simulation purposes (left) and a photograph of the prototype leg (right).

by very few actuators. A motor with a rotor disc seems to be ideal here since it offers very good dynamic properties due to its low inertia and high peak torque. The selected model offers a zero motion torque of approximately 13.72 Nm. Employing a gear ratio of 32 : 1 and neglecting the loss at the gearbox this would result in a maximum obtainable torque of 439 Nm. Unfortunately this can only be achieved for a very short time interval before the hardware would be destroyed due to a resulting current of more than 100 A. Thus, the sustainable peak torque is assumed to be 150 Nm for the simulation process.

4.2 Low Level Closed Loop Controller

The low level closed loop controller is implemented on a DSP-board. The DSP-board is connected via CAN-bus to the PC. Because of the encapsulation of the different reflexes there is no realtime capability required. The reflexes which need fast sensor informations, like touchdown reflex, can be directly implemented on the DSP.

The hip and the knee motor are connected to one DSP-board. For the speed and position measurement an optical-encoder is attached to the motor shaft. The encoders are connected via the CPLD to the DSP. The current measurement for each motor is realized by a shunt. Because of the non-continuous current due to the PWM the synchronisation of the measurement is a big problem. The solution is a timer that is synchronized in the DSP-hardware with the PWM and the AD-converter. The direction of the current is dedicated by the sign of the default PWM.

To reduce the noise of the measured current a simple lowpass-filter is implemented. The cycletime of the standard PI-controller is 1ms. That is fast enough to have an average risetime of about 8ms for a given desired current. This is concurrently the limiting setpoint setting of the speed controller and the desired torque. These two values are fusioned in a weighted sum (see equation 1). The fusion of the desired torque and the desired speed influences the stiffness of the joint. To avoid damage of the hardware the desired current is limited.

$$current_{des} = \frac{w_{pos}^2 * tor_{speed} + w_{tor}^2 * tor_{des}}{w_{pos} + w_{tor}} \tag{1}$$

The speed controller is also a classic PI-controller. The integral-portion is limited to avoid windup-effects when the weight of the position is very low. The third hierarchical controller is a position controller. Due to the integral portion of controlled system the position controller has no need for an integral portion. The acceleration of the speed controller is also limited. This is required to ensure the stability of the system (see [17]).

4.3 Reflex Control for Jumping

The jump is controlled by units on both the spinal cord and the muscle level. The spinal cord is the coordinating instance while the reflexes on the muscle level generate the actual commands for the joint controllers. The functionality on the spinal cord level is achieved through a behavior based module that acts as finite state machine (FSM) while the four reflexes are closely coupled with the hardware (see figure 4).

The Push off Reflex is intended to start in a squatted position that is reached using either the spinal cord level function to initialize the jump or the squat reflex during repetitive jumping. Once the reflex is stimulated, the leg is stretch out by applying torque to either actuated joints. Experiments have shown that the naive approve of applying the maximum momentum at both joints is contraindicated by the fact that this would cause undesired lateral force due to the closed kinematic chain of the leg. Thus the hip is relaxed almost entirely while the main share of the work is performed by the knee actuator. Torque is applied until leg remains slightly bend and is the entirely withdrawn to reduce the lateral movement in negative x-axis direction after the liftoff. Besides eliminating undesired movement this also helps to reduce the energy consumption of a jump motion. Once the foot point loses contact to the ground the activity of the reflex is withdrawn by the coordination function.

The Inflight Reflex is activated after lift-off. The intention behind the inflight reflex is on the one hand the necessity to ensure a proper landing posture in order to minimize the mechanical stress on the joints and segments and on the other hand to maintain favorable joint angles to maximize the amount of impact energy that can be restored in the subsequent push off attempt. The approach taken here is the

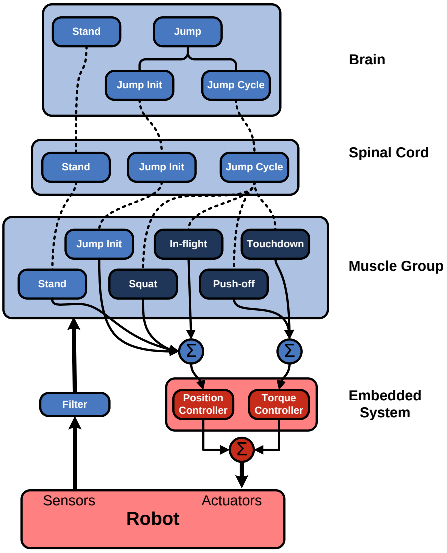

Stand
Jump
Jump Init
Jump Cycle
Brain
Stand
Jump Init
Jump Cycle
Spinal Cord
Jump Init
In-flight
Touchdown
Stand
Squat
Push-off
Muscle Group
Σ
Σ
Filter
Position Controller
Torque Controller
Embedded System
Σ
Sensors
Actuators
Robot

Fig. 4 Schematic view of the behavior based control structure.

combination of both by using two concurrent reflexes: the already mentioned inflight reflex and a touchdown reflex that will be introduced in the next section. The activity of this reflex (and therefore the stiffness of the joint angle controller) decreases the closer the sensed joint configurations approaches the desired one (slightly bend). Once the target configuration is reached within a certain threshold the activity is kept at a level of approx. 30 % to ensure the posture remains roughly the same. The reduction of the activity has proven to be very useful at the moment of touchdown since a desired amount of joint compliance can be realized this way. The stimulation is entirely withdrawn from the reflex as soon as a ground contact is detected and thus the landing can be assumed.

The Touchdown Reflex as mentioned above is pre-stimulated at a certain point in time while the leg is still in the air. Once ground contact is detected the reflex intends to gradually slow down the drop until the leg comes to a complete rest at a defined position. The former is achieved through the touchdown reflex while the latter is managed by the squat reflex to be described in the next section. The activity of this reflex is adjusted in respect to the angular velocity. The higher the angular velocity the more counter momentum (i.e. torque) is applied to the actuated joints.

The Squat Reflex is once stimulated at the beginning of the landing phase. The squat reflex is as already stated responsible for controlling the legs configuration into a defined resting position (hip angle $\approx 30°$, knee angle $\approx 60°$) by means of position control. By co-activating the touchdown reflex one can be sure to reach that position with only rather low velocity and thus tolerable mechanical stress for the hardware.

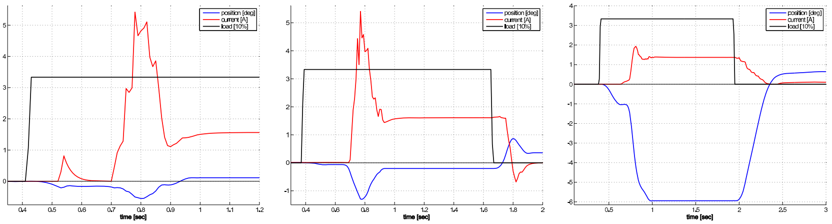
The Touchdown Preflex is intended to reduce the impact stress. The idea is to adjust the speed of the contact point to the ground. This results in adduction of the leg after the peak of the airborne phase is passed. The timing is very critical. If the preflex is activated too early it is no longer active during the impact since the inflight reflex is trying to maintain a safe landing position and thus useless. If it is activated too late it can not unfold its full effect.

5 Experiments and Results

5.1 Compliant Controller

In order to be able to test the implemented controller under the best possible conditions, the second actuator is mounted on the same shaft as the first one. This results in a direct and stiff connection of the two gearboxes. One of the motors is controlled using the hierarchical controller while the other is used to simulate a controlled load by directly applying the PWM ratio. To test the desired compliant properties, the controller was set to a "soft" position control ($w_{pos} < 1$, $w_{tor} = 0$) and the second motor was used to generate a sudden and heavy distortion. The results can be found in figure 5.

As expected the tolerance to deviation in position increases with decreasing w_{pos}. In figure 5(a) and 5(b) the controller is stiff enough to compensate the position deviation nearly completely. A very small deviation is left in the intermediate case. The compliant controller is not able to hold the desired position. With a higher w_{pos} setting the controller reacts more aggressive (in respect to magnitude and time) to an occurring distortion. With a lower value the behavior is more relaxed and the current overshoot is way smaller. The applied counter momentum in the static case is equal for all parameter setups. The time delay between the load jump and the reaction of the position is caused by the loose connection between the two motors.



(a) stiff (b) intermediate (c) compliant

Fig. 5 Qualitative performance of the real compliant position controller based on the stiffness weight. The dotted line represent the current measured in Ampere, the solid one stand for the current position denoted in degree while the dashed graph indicates the load applied by the second motor in voltage.

5.2 *Simulated Jump Cycles*

In order to be able to optimize the jumping process without putting the actual hardware in jeopardy a simulation environment consisting of an hardware abstraction layer and the physics simulation engine Newton was employed. The results of an undisturbed jump cycle can be found in figure 6.

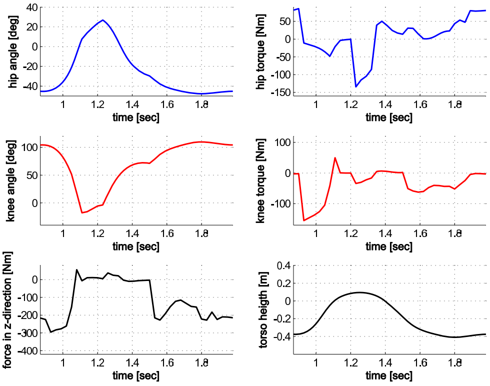


Fig. 6 Sensor data recorded during one jump sequence.

The dashed graphs (top) denote the hip torque and angle, while the dotted ones (middle) stand for the respective knee values. On the bottom the load cell data (left, solid) and the hip-joints z-position (right, dot-dashed) are marked. The cycle starts with the pushoff phase $t = 1.9$ *s* until the liftoff is reached at $t = 2.1$ *s*. The impact occurs at $t = 2.5$ *s* as visible in the load cell data. The recording continues until the initial squatted position is reached again after a cycle time of $t = 1.1$ *s*. The acquired jump height is approximately 15 cm. The lion's share of the work in the phase prior to liftoff is performed by the knee, because the hip actuator is only able to apply vertical forces to the leg. The ratio shifts after the leg is airborne. In this phase the hip motor has the role to bring the leg in the landing position. Due to the inertia of the tibia, the knee motor can be relaxed to reach the desired knee angle. The preflex helps to reduce the impact force of formally over 350 N to approx. 230 N. We have compared the hip and the knee angle with such of humans. Although it was not our goal to copy the trajectory of humans the behavior is approximately equal.

6 Conclusion and Outlook

In this paper the lower layers of a biologically motivated control architecture for biped robots were derived. The approach features compliant behavior and distributed control. The performance of the reflex-like control was evaluated using simulated cyclical jumps of a single leg with passive elastic elements. The biological motivated actuator controller was implemented and tested within an experimental setup of the prototype leg.

As can be seen in the experimental results, a behavior similar to a SEA could be obtained with a standard DC motor. This is possible by employing a low-friction gearbox and a fast hierarchical compliant controller. However the actively controlled stiffness can be altered by changing a single parameter. This might lead to an entirely stiff joint behavior on the one side or an unrestrained limb movement on the other. The approach has shown its potential during repetitive jumping in a simulated environment.

The next step will be to execute jumping motions with the prototype leg. Furthermore experiments concerning the effectiveness of a passive ankle joint will be pursued. After that the physical distribution of the reflexes will be taken into consideration.

References

1. Albu-Schaeffer, A., Hirzinger, G.: Cartesian impedance control techniques for torque controlled light-weight robots. In: Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA), Washington, DC, USA, vol. 1, pp. 657–663 (2002)
2. Anderson, S., Wisse, M., Atkeson, C., Hodgins, J., Zeglin, G.J., Moyer, B.: Powered bipeds based on passive dynamic principles. In: Proceedings of the 2005 5th IEEE-RAS International Conference on Humanoid Robotics, pp. 110–116 (2005)
3. Bizzi, E., Cheung, V., d'Avella, A., Saltiel, P., Tresch, M.: Combining modules for movement. Brain Research Reviews 57, 125–133 (2007)
4. Bobbert, M.F.: Dependence of human squat jump performance on the series elastic compliance of the triceps surae: A simulation study. The Journal of Experimental Biology (2001)
5. Collins, S., Ruina, A.: A bipedal walking robot with efficient and human-like gait. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA), Barcelona, Spain, pp. 1983–1988 (2005)
6. Curran, S., Orin, D.: Evolution of a jump in an articualted leg with series-elastic actuation. In: IEEE International Conference on Robotics and Automation (ICRA), Pasadena, CA, USA, pp. 352–358 (2008) ISSN 978-1-4244-1647-9
7. Ham, R.V., Damme, M.V., Vanderborght, B., Verrelst, B., Lefeber, D.: Maccepa, the mechanically adjustable compliance and controllable equilibrium position actuator. In: ACTUATOR 2006: The 10th International Conference on New Actuators, Bremen, Germany (2006)
8. Ivanenko, Y.P., Poppele, R.E., Lacquaniti, F.: Five basic muscle activation patterns account for muscle activity during human locomotion. Journal of Physiology 556 (2004)
9. Ivanenko, Y.P., Poppele, R.E., Lacquaniti, F.: Motor control programs and walking. The Neuroscientist 12(4), 339–348 (2006)
10. Kerscher, T., Zöllner, J., Dillmann, R., Stella, G., Caporaletti, G.: Model and control of compliant joints driven by fluidic muscles. In: ACODUASIS-workshop, Torino, Italy (2005)

11. Lemay, M.A., Galagan, J.E., Hogan, N., Bizzi, E.: Modulation and vectorial summation of the spinalized frog's hindlimb end-point force produced by intraspinal electrical stimulation of the cord. IEEE Transactions on Neural Systems and Rehabilitation Engineering 9 (2001)
12. Lohmeier, S., Buschmann, T., Ulbrich, H., Pfeiffer, F.: Humanoid robot lola - research platform for high-speedwalking. In: Motion and Vibration Control, pp. 221–230. Springer, Netherlands (2009)
13. Luksch, T., Berns, K.: Controlling dynamic motions of biped robots with reflexes and motor patterns. In: Fourth International Symposium on Adaptive Motion of Animals and Machines (AMAM), Cleveland, USA, pp. 115–116 (2008)
14. Luksch, T., Berns, K.: Initiating normal walking of a dynamic biped with a biologically motivated control. In: 11th International Conference on Climbing and Walking Robots (CLAWAR), Coimbra, Portugal (2008)
15. Luksch, T., Berns, K., Mombaur, K., Schultz, G.: Using optimization techniques for the design and control of fast bipeds. In: 10th International Conference on Climbing and Walking Robots (CLAWAR), Singapore (2007)
16. McGeer, T.: Dynamics and control of bipedal locomotion. Journal of Theoretical Biology 163, 277–314 (1993)
17. Pfaff, G., Meier, C.H.: Regelung elektrischer Antriebe II, 3rd edn. Oldenbourg Verlag (1992) ISBN 3-486-22376-3
18. Pratt, J., Chew, C., Torres, A., Dilworth, P., Pratt, G.: Virtual model control: An intuitive approach for bipedal locomotion. International Journal of Robotics Research 20(2), 129–143 (2001)
19. Pratt, J., Krupp, B., Morse, C.: Series elastic actuators for high fidelity force control. Industrial Robot Journal 29(3), 234–241 (2002)
20. Proetzsch, M., Luksch, T., Berns, K.: Robotic systems design using the behavior-based control architecture ib2c. Journal of Robotics and Autonomous Systems (submitted, 2008)
21. Rossignol, S., Dubuc, R., Gossard, J.P.: Dynamic sensorimotor interactions in locomotion. Physiological Reviews 86, 89–154 (2006)
22. Zehr, E., Stein, R.: What function do reflexes serve during human locomotion? Progess in Neurobiology 58, 185–205 (1999)

Using the Torso to Compensate for Non-Minimum Phase Behaviour in ZMP Bipedal Walking

Houman Dallali, Martin Brown, and Bram Vanderborght

Abstract. In Zero Moment Point (ZMP) bipedal walking, the conventional method is to use the cart-table model for generating the reference trajectory [1]. However, due to modeling and tracking errors and external disturbances, such as uneven terrain, the generated trajectory must be adapted by a stabilizer that uses sensory inputs from force and torque sensors placed in the robot's feet. The problem with the cart-table model is that it is non-minimum phase which causes a significant, undesirable undershoot in the ZMP in order to cancel the effect of disturbances. In this paper, a novel scheme is proposed for ZMP feedback stabilization that utilizes the upper body to balance the humanoid robot. This method increases the performance and robustness of walking by reducing the undershoot and maintaining a desired bandwidth. The effectiveness of the proposed scheme is demonstrated using simulation and open problems are discussed.

1 Introduction

A current and interesting robotics research area is walking bipedal robots. Studying bipedal robots introduces interdisciplinary challenges in applied mathematics, mechanical, computer and control engineering. During the past two decades, significant advances have been made in this area and current bipedal robots are capable of walking in well structured environments, on flat and sloped terrains [2],[3],[4],[5].

Houman Dallali · Martin Brown
The University of Manchester, Control Systems Centre, School of Electrical and Electronic Engineering, M60 1QD, United Kingdom,
e-mail: `houman.dallali@postgrad.manchester.ac.uk`, `martin.brown@manchester.ac.uk`

Bram Vanderborght
Italian Institute of Technology, Robotics, Brain and Cognitive Sciences Department,
Via Morego 30, 16163 Genova, Italy,
e-mail: `bram.vanderborght@iit.it`

In bipedal locomotion, cyclic walkers have been a focus of recent attention. This consists of energy shaping approaches which are based on nonlinear control theory [6],[7] and hybrid control approaches based on the notion of virtual constraints [8]. However, both approaches have only been implemented for 2D robots and the extensions to 3D is still under study. In addition, versatility, where a robot walks on discrete footholds and to turn or change its limb coordination, is an integral part of locomotion that can not be achieved using a cyclic walking approach. One practical scheme to control humanoid robots is based on the concept of Zero Moment Point (ZMP)[9]. ZMP is defined as the point on the ground plane about which the total moments due to ground contact forces become zero. For stability the ZMP must always reside in the convex hull of the ground contact points. In order to satisfy this criterion, [10] considered the non-minimum phase problem of balancing and proposed a nonlinear invariant control approach for planar robots. A conventional ZMP based trajectory generation method that can be implemented in 3D is the cart-table model that approximates the dynamics of a bipedal robot by a running cart on a pedestal table which is a linear and intuitive model [1]. It should be noted that the robots that fit in this scheme must be fully actuated and position controlled.

Due to the modeling and tracking errors, external disturbances such as the ground type and unknown external forces the actual ZMP trajectory will be different from the desired reference trajectory. The actual ZMP trajectory is obtained from measurements, such as force and torque sensors placed on the foot. The deviation of the actual ZMP trajectory from the desired reference trajectory has to be tracked by a controller (stabilizer) to generate the required change in the joint torques. However, these deviations can be large enough to destabilize the robot. HRP-1S can perform stable walking in simulation, but falls in practice [12]. The addition of a feedback stabilizer means that robot is also able to walk in experiments because it is able to cope with (limited) unknown disturbances. The tracking of the desired horizontal motion of Johnnie is suspended whenever the ZMP approaches an instability region [13]. However, the cart-table model is non-minimum phase and hence the stabilizers will have limited tracking performance of such errors, in terms of the bandwidth and the amount of undershoot

In this paper, an overview of the cart-table model for ZMP trajectory generation and its non minimum phase behaviour is given in section 2 and a two mass inverted pendulum is introduced to model the robot's upper body and introduce actuation in the hip. In section 3 these two models are combined to solve the non-minimum phase behaviour of the cart-table model, by including an upper body model. The dynamic performance is analyzed theoretically and in section 3.2, and an LQR optimal controller is formulated for the new multi-variable system to generate the reference walking trajectories. In section 3.3 a ZMP stabilizer is proposed to use the torso in balancing the robot. Finally, the effectiveness of the proposed scheme is evaluated by simulation and open problems are discussed.

2 Literature Review

In this section, an overview of the cart-table trajectory generation model is provided. This includes analyzing the model's unstable zero which causes the non-minimum phase behaviour and discussing the implications for control design. Next, a simple two mass inverted pendulum model of the robot's hip and torso is reviewed, and its benefit and limitations are mentioned [17]. This is to motivate the combination of the two models in section 3.2.

2.1 Linear Inverted Pendulum and Cart-Table Model

In [14] a 3D linear inverted pendulum model for bipedal walking trajectory generation was introduced. The 3D inverted pendulum is constrained to move along a plane which results in linear state space equations as illustrated in figure 1. In [1] it was shown that this directly corresponds to the dynamics of a running cart on a pedestal table and the cart-table dynamics are:

$$\begin{aligned} p_x &= x - \frac{z_c}{g}\ddot{x} \\ p_y &= y - \frac{z_c}{g}\ddot{y} \end{aligned} \tag{1}$$

where p_x,p_y are the zero moment points in X and Y directions, respectively, z_c is the height of the cart and g is gravity. In this paper, the problem is studied in the frontal plane (i.e. ZMP in X direction is considered) but under certain assumptions the motion in both frontal and lateral planes can be decoupled. Therefore, the models in this section and the proposed model in section 3 are also valid for 3D walking [14]. Equation 1 can be represented in state space form using a new variable u_x which is the time derivative of the horizontal acceleration (jerk) of center of mass, $\frac{d}{dt}\ddot{x} = u_x$:

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_x\,, \quad p_x = [1 \quad 0 \quad \frac{-z_c}{g}]\mathbf{x} \tag{2}$$

where $\mathbf{x} = (x, \dot{x}, \ddot{x})^T$. The cart-table model provides a convenient framework to generate a reference trajectory for the bipedal robot. However, in ZMP feedback

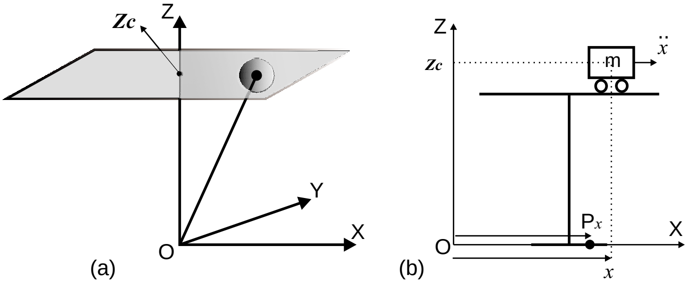


Fig. 1 In (a) constrained linear inverted pendulum is shown that assumes that ZMP is in the origin (0,0,0) (origin corresponds to the robot's ankle) and the ankle torque is zero. In (b) the cart-table model is shown for ZX (frontal) plane where the ZMP can move in the foot (table's contact with the ground) and the ankle's torque is not necessarily zero.

stabilization a problem arises from the non-minimum phase behaviour of equation 2. The transfer function of the equation 2 is:

$$G(s) = \frac{p_x(s)}{u_x(s)} = \frac{1 - (\frac{-z_c}{g})s^2}{s^3} = \left(\frac{-z_c}{g}\right) \frac{(\frac{-g}{z_c}) - s^2}{s^3} \tag{3}$$

This is unstable because it has three poles at the origin (triple integrator) but this can be overcome using pole placement. However, equation 3 also has two zeros. Using feedback will not affect the positions of the zeros and this will cause problems in the control design. The left half plane zero has a scaling effect on the response which can be solved by dc gain adjustment, but the right half plane zero will limit the achievable bandwidth of the system and produces an undesirable undershoot [15] as the inverse is unstable. For ZMP feedback stabilization, consider the case where the desired ZMP is in front of the actual ZMP of the robot. The hip must accelerate in the positive direction and according to equation 2 the ZMP will initially move backward and hence diverge as illustrated in figure 2. Therefore, it is important to investigate methods for minimizing/overcoming this behaviour in order to increase the robustness to disturbances such as uneven surfaces.

In general, perfect tracking control of such systems without future information of the tracking signal is not possible and it is necessary to approximate the non-minimum phase system with a minimum phase system to design controllers with bounded tracking error [16]. In addition, controller design based on right half plane pole-zero cancelation is fundamentally flawed, due to the loss of internal stability as such designs rely on unstable pre-filters. Therefore, performance limitations, due to the right half plane zeros, will be present in any design leading to inevitable compromises between the speed of response and the amount of undershoot in the system's step response. In figure 2 this is shown for the cart-table model which is stabilized using pole-placement.

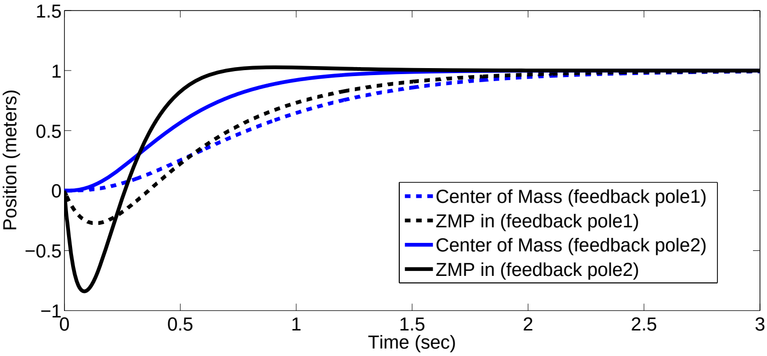


Fig. 2 Cart-table model step response with poles placed at pole1=[-2,-4,-6], pole2=[-4,-6,-10]. In the first case (dotted line) the ZMP rise time is 1.04s and ZMP undershoot is 27% (notice the slow response of center of mass) while in the second case (solid line) the ZMP rise time is 0.275s and ZMP undershoot is 84% (in this case the center of mass quickly approaches the steady state). This shows the trade off in bandwidth and undershoot criteria.

2.2 Two Mass Inverted Pendulum Model

The non-minimum phase problem of one-mass inverted pendulum model which represents the center of mass of a robot was first analyzed in [17] and a two mass inverted pendulum was proposed that results in a multi-variable minimum phase system, as illustrated in figure 3. Linearization about the origin produces the following state space equations:

$$
\begin{gathered}
\dot{\mathbf{x}} = \begin{bmatrix} 0 & I_{2\times 2} \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ I_{2\times 2} \end{bmatrix} \mathbf{u}\,, \quad p_x = [c_1 \; c_2 \; 0 \; 0]\mathbf{x} + [d_1 \; d_2]\,\mathbf{u} \\
c_1 = \frac{(m_1+m_2)l_1+m_2 l_2}{m_1+m_2}\,,\; c_2 = \frac{m_2 l_2}{m_1+m_2}\,,\; d_1 = -\frac{m_1 l_1^2+m_2(l_1+l_2)^2}{(m_1+m_2)g}\,,\; d_2 = -\frac{m_2(l_1+l_2)l_2}{(m_1+m_2)g}
\end{gathered}
\tag{4}
$$

where $\mathbf{x} = (\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2)^T$ and $\mathbf{u} = (u_1, u_2)^T = (\ddot{\theta}_1, \ddot{\theta}_2)^T$ and p_x are the input vector and scalar output of the two mass inverted pendulum, respectively. It can be shown that this has no right half plane zeros provided the second link's length is non-zero. The transfer function matrix of two mass inverted pendulum can be obtained as following:

$$
G(s) = C(sI-A)^{-1}B + D = [\frac{c_1+d_1 s^2}{s^2} \quad \frac{c_2+d_2 s^2}{s^2}]
\tag{5}
$$

It should be noted that the zeros in multi-variable systems are different from their SISO counterparts, and they are associated with the directions of input and output of the system. To obtain the zeros of equation 5, the transfer function, $G(s)$, is epxressed as:

$$
G(s) = \frac{\Pi(s)}{D_G(s)}\,, \quad \Pi(s) = [c_1 + d_1 s^2 \quad c_2 + d_2 s^2]\,, \quad D_G(s) = s^2
\tag{6}
$$

The roots of greatest common divisor of $\Pi(s)$ determines the zeros of the transfer function matrix $G(s)$, which in this case does not have a root.

Therefore, the transfer function does not have any zeros and hence the double inverted pendulum is minimum phase. However, as mentioned earlier, this scheme has limitations due to the linearization of the two masses around the vertical line. This can produce large errors as the robot's hip is required to move in a large operating range that cannot be approximated by linearizing about a single operating

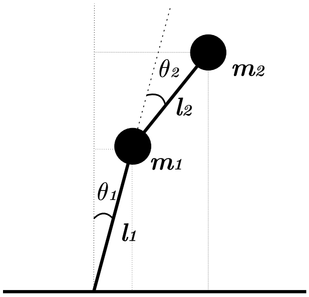


Fig. 3 The two mass inverted pendulum model relies on linearization of both upper and lower pendulum links. This is not the case for the robot as the motion of the hip is completely nonlinear and it can lead to large modeling errors for the trajectory generation.

point [11]. In addition, the inverted pendulum models assume ZMP in the ankle and ankle torque is zero, while in the cart-table model, the ZMP is located in the foot and ankle torque is not necessarily zero. In section 3, a new model is proposed that combines the advantages of cart-table model and two mass inverted pendulum.

3 Generalized Two Link Inverted Pendulum

In this section firstly a new model is proposed which combines the advantages of the cart-table and the two mass inverted pendulum models. The new model does not assume linearization of the lower mass that corresponds to the robot's hip motion. Secondly, a trajectory generation for the model based on preview control of ZMP is described in section 3.2 . Thirdly, a solution to ZMP stabilization is proposed in section 3.3.

3.1 Modeling

Consider the pendulum-cart-table model shown in figure 4. The equation for torques around the ZMP gives:

$$\tau_{zmp} = -Mg(x-p_x)+M\ddot{x}z_c-(mg+m\ddot{z}_2)(x_2-p_x)+m\ddot{x}_2(z_c+z_2)=0 \qquad (7)$$

where $z_2 = l\cos(\theta)$, $x_2 = x + l\sin(\theta)$, and l is the pendulum's length. . Assuming small deviation on the torso and linearizing around the vertical axis where $z_2 = l$ and $x_2 = x + l\theta$, this results in the following linear relationship:

$$p_x = x + \frac{ml}{(M+m)}\theta - \frac{Mz_c + m(z_c+l)}{(M+m)g}\ddot{x} - \frac{ml(z_c+l)}{(M+m)g}\ddot{\theta} = (1, c_1, 0, 0, c_2, c_3)\mathbf{x} \qquad (8)$$

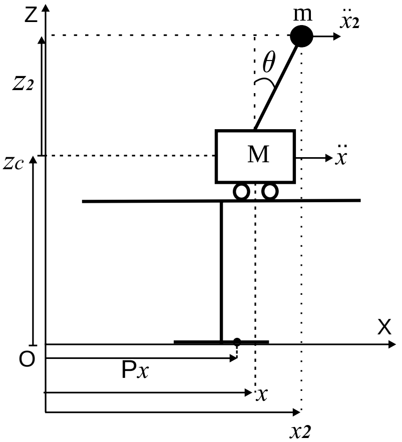


Fig. 4 The new pendulum-cart-table model in 2D. The cart represents the hip and the actuated pendulum represents the torso.

where $\mathbf{x} = (x, \theta, \dot{x}, \dot{\theta}, \ddot{x}, \ddot{\theta})^T$ is the state vector and the control input vector is $\mathbf{u} = (u_1, u_2)^T$ where $u_1 = \frac{d}{dt}\ddot{x}$ and $u_2 = \frac{d}{dt}\ddot{\theta}$. Therefore, the system for control of ZMP can be define as:

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & I_{4\times 4} \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ I_{2\times 2} \end{bmatrix} \mathbf{u}\,, \quad y = p_x = \begin{bmatrix} 1 & c_1 & 0 & 0 & c_2 & c_3 \end{bmatrix} \mathbf{x} \tag{9}$$

The transfer function matrix for this system is given by:

$$G(s) = C(sI - A)^{-1}B = \frac{1}{s^3} \begin{bmatrix} 1 + c_1 s^2 & c_2 + c_3 s^2 \end{bmatrix} \tag{10}$$

and in a similar fashion to section 2.2 the system does not have any zeros. Therefore, one can design a multi-variable controller to achieve the desired bandwidth required for the purpose of feedback stabilization.

3.2 Trajectory Generation Using Preview Control of ZMP

Consider the dynamics of pendulum-cart-table model given by equation 9. The discrete time dynamics can be expressed by:

$$\mathbf{x}(k+1) \;=\; A\mathbf{x}(k) \;+\; B\,\mathbf{u}(k), \qquad \mathbf{y}(k) \;=\; C\mathbf{x}(k) \tag{11}$$

where $\mathbf{x}(k) = (x(k), \theta(k), \dot{x}(k), \dot{\theta}(k), \ddot{x}(k), \ddot{\theta}(k))^T$, $\mathbf{u}(k) = (u_1(k) \;\; u_2(k))^T$ and A, B are the discrete time versions of equation 9 with sample time T. To add robustness to the control, integral action is introduced by using incremental control, $\Delta\mathbf{u}(k) = \mathbf{u}(k) - \mathbf{u}(k-1)$, and state, $\Delta\mathbf{x}(k) = \mathbf{x}(k) - \mathbf{x}(k-1)$, [19]. The state vector is augmented as $\tilde{\mathbf{x}}(k) = (p_x(k), \Delta\mathbf{x}(k))^T$ and the dynamics now become:

$$\tilde{A} = \begin{bmatrix} I & CA \\ 0 & A \end{bmatrix} \quad \tilde{B} = \begin{bmatrix} CB \\ B \end{bmatrix} \quad \tilde{C} = \begin{bmatrix} 1 & \mathbf{0}_{1\times 6} \end{bmatrix} \tag{12}$$

An optimal control problem is formulated by minimizing:

$$J = \sum_{i=k}^{\infty} [e^T(i) Q_e e(i) + \Delta\mathbf{x}^T(i) Q_x \Delta\mathbf{x}(i) + \Delta\mathbf{u}^T(i) R \Delta\mathbf{u}(i)] \tag{13}$$

where $e(i) = p_x(i) - p_{ref}(i)$, and the optimal control is given by [19]:

$$u^o(k) \;=\; -G_I \sum_{i=0}^{k} e(k) \;-\; G_x x(k) - \sum_{i=1}^{NL} G_d(i) p_{ref}(k+i) \tag{14}$$

where p_{ref} is the reference ZMP trajectory in x direction. The parameter N_L determines the horizon of the future desired ZMP. The optimal gain is determined by solving the discrete time algebraic Riccati equation:

$$\tilde{P} = \tilde{A}^T \tilde{P} \tilde{A} - \tilde{A}^T \tilde{P} \tilde{B} (R + \tilde{B}^T \tilde{P} \tilde{B})^{-1} \tilde{B}^T \tilde{P} \tilde{A} + \tilde{Q} \tag{15}$$

where, $\tilde{Q} = diag\{Q_e, Q_x\}$. Hence the optimal gain is defined by:

$$\tilde{K} = [G_I \quad G_x] \; = \; (R + \tilde{B}^T \tilde{P} \tilde{B})^{-1} \tilde{B}^T \tilde{P} \tilde{A} \tag{16}$$

and the optimal preview gain is calculated using the recursive formula:

$$\begin{aligned} G_d(l) = \; & (R + \tilde{B}^T \tilde{P} \tilde{B})^{-1} \tilde{B}^T \tilde{X}(l-1), \\ \tilde{X}(l) = & \tilde{A_c}^T \tilde{X}(l-1), \quad l = 2, \ldots, N_L, \end{aligned} \tag{17}$$

where $\tilde{A_c} = \tilde{A} - \tilde{B}\tilde{K}$, $G_d(1) = G_I$ and $\tilde{X}(1) = -\tilde{A_c}^T \tilde{P}[1 \quad \mathbf{0}_{1\times 6}]^T$. A numerical example is given in the section 4 where the generator is producing the desired motion of center of mass. In the next section, the stabilization issue associated with ZMP based walking is considered.

3.3 Stabilizer Design

In section 3.2, a walking reference trajectory generator based on the new pendulum-cart-table model was introduced. However, due to imperfect ground conditions, modeling errors and unknown disturbances, a real time feedback stabilizer must be used to adapt the generated trajectories based on the sensor information. This section proposes a ZMP feedback stabilization scheme to increase the robustness in bipedal walking.

Consider equation 9 with outputs modified as:

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & I_{4\times 4} \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ I_{2\times 2} \end{bmatrix} \mathbf{u} \, , \quad \mathbf{y} = \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & c_2 & 0 \\ 0 & c_1 & 0 & 0 & 0 & c_3 \end{bmatrix} \mathbf{x} \tag{18}$$

where p_1 and p_2 are the ZMP outputs corresponding to hip and torso, respectively. Reference tracking using feedback can then be implemented by considering the system defined in equation 18 in compact format as:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \;\; , \;\; \mathbf{y} = C\mathbf{x} \tag{19}$$

Define the integral of error, $\mathbf{z}$, between reference vector $\mathbf{r}$ and the system's output vector $\mathbf{y}$, ie. $\dot{\mathbf{z}} = \mathbf{r} - \mathbf{y}$, then the new state space system is:

$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{z}} \end{bmatrix} = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \mathbf{u} + \begin{bmatrix} 0 \\ I \end{bmatrix} \mathbf{r} \;\; , \;\; \mathbf{y} = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix} \tag{20}$$

Since the system in equation 20 is controllable, it is possible to use feedback $\mathbf{u} = -[K_1 \quad K_2]\mathbf{x}$ to stabilize the system and track the reference vector $\mathbf{r}$. As a result of feedback the closed loop system is:

$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{z}} \end{bmatrix} = \begin{bmatrix} A - BK_1 & -BK_2 \\ -C & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} \mathbf{r} \;\; , \;\; \mathbf{y} = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix} \tag{21}$$

A numerical example for the ZMP stabilization is given in section 4 and the results are compared with the old cart-table model.

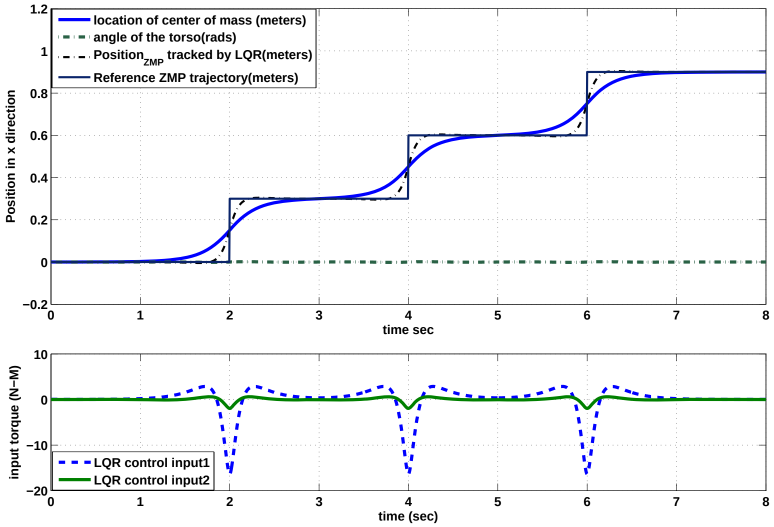


Fig. 5 An illustration of the reference ZMP trajectory generation using the new pendulum-cart-table model. The location of torso is kept in the vertical direction, however it can be moved depending on the ground slope. The control input signals are shown in the lower figure.

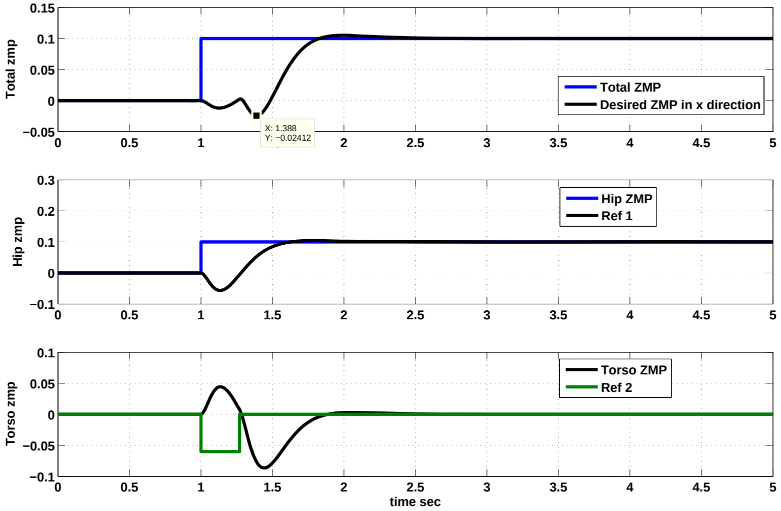


Fig. 6 An illustration of the use of torso in minimizing the amount of undershoot, for the purpose of stabilization and balancing.

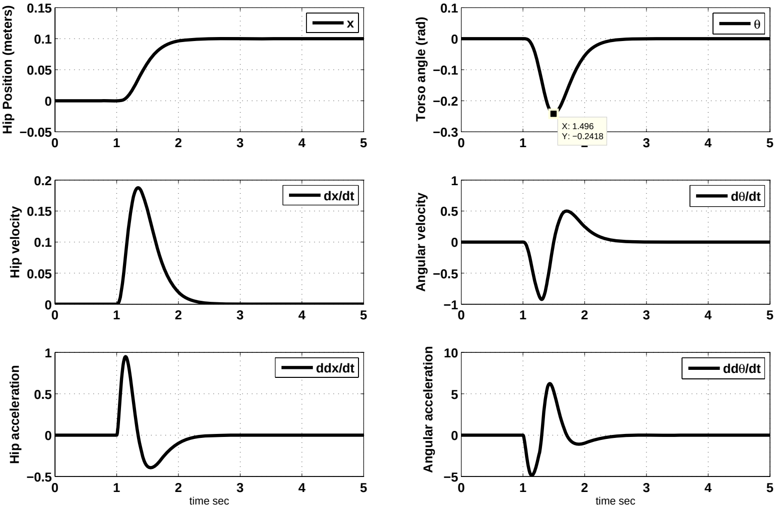


Fig. 7 An illustration of the state evolution in the pendulum-cart-table model. Note that the upper body is moving by maximum of 0.24rad.

4 Comparison and Simulation Results

In order to implement ZMP based walking, a stable trajectory of center of mass that satisfies the desired ZMP must be generated and during trajectory following a real time stabilization mechanism must be used to adapt the trajectories and react to unknown disturbances. Therefore, the results of this paper are two fold. Firstly, a numerical example is provided to illustrate the reference trajectory generation introduced in section 3.2. Simulation results of ZMP tracking and the location of center of mass (robot's hip) associated with the pendulum-cart-table model is given in figure 5, and the controller and model parameters are given as follows: $g = 9.81$, $z_c = 0.6m$, $m = 8Kg$, $M = 12.5Kg$, $l = 0.3m$, $T = 0.005s$; preview time $Tprev = 2s$, $Q_e = I_{2\times2}$, $Q_x = I_{6\times6}$, $R = 10^{-3} \times I_{2\times2}$ and the simulation time is $8sec$. Secondly, a numerical example of the real time feedback scheme that was introduced in section 3.3 is illustrated in figure 6. Note that the step height is 0.1 m and the system poles are placed at $[-6-6-8-8-10-10-12-12]$. The undershoot corresponding to the hip is reduced from 5.6cm to 2.4cm by using the upper body movement, which is a 57% reduction in the total ZMP of the new pendulum-cart-table model. At the same time the settling time in cart ZMP step response is 0.53s and in the total ZMP response the settling time is 0.74s. That is the new model can achieve a reasonable tracking speed, and avoid the undesired undershoot. In addition, due to practical limitations, the range of upper body movement must be as small as possible. This is achieved and demonstrated in figure 7.

5 Conclusions and Future Work

In this paper, the non-minimum phase problem associated with using the cart-table model for ZMP reference trajectory generation has been described and a solution has been proposed based on actuating the torso. The new pendulum-cart-table model combines the advantages of existing models and makes use of the torso to compensate the unknown ZMP deviations. The effectiveness of this model has been shown by numerical examples both for pattern generation and for ZMP feedback stabilization. In ZMP stabilization the torso is used for reducing or ideally canceling the undesired effects of undershoots and bandwidth limitations. However, this method can be used in combination with existing stabilization schemes that use ankle actuation. All the results mentioned in this paper are valid for the three dimensional case where the sagittal and lateral motions can be decoupled. The future, work should combine the ankle and torso actuation for the purpose of stabilization, as it can be observed in human walking.

Acknowledgements. This research has been done on the CICADA project which is financed by the EPSRC grant EP/E050441/1 and the University of Manchester. The authors would like to thank IIT and especially Prof D. Caldwell and Dr N. Tsagarakis for their support.

References

1. Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., Hirukawa, H.: Biped walking pattern generation by using preview control of zero-moment point. In: ICRA, vol. 2, pp. 1620–1626 (2003)
2. Kanehira, N., Kawasaki, T., Ohta, S., Ismumi, T., Kawada, T., Kanehiro, F., Kajita, S., Kaneko, K.: Design and experiments of advanced leg module (HRP-2L) for humanoid robot (HRP-2) development. In: IROS, vol. 3, pp. 2455–2460 (2002)
3. Sakagami, Y., Watanabe, R., Aoyama, C., Matsunaga, S., Higaki, N., Fujimura, K.: The intelligent ASIMO: system overview and integration. In: IEEE/RSJ IROS, vol. 3, pp. 2478–2483 (2002)
4. Vanderborght, B., Van Ham, R., Verrelst, B., Van Damme, M., Lefeber, D.: Overview of the Lucy-project: Dynamic stabilisation of a biped powered by pneumatic artificial muscles. Advanced Robotics 22(6-7) (10), 1027–1051 (25) (2008)
5. Pfeiffer, F., Lff er, K., Gienger, M.: The Concept of Jogging JOHNNIE. In: ICRA, vol. 3, pp. 3129–3135 (2002)
6. Holm, J.K., Spong, M.W.: Kinetic energy shaping for gait regulation of underactuated bipeds. In: Proceedings of the 17th IEEE International Conference on Control Applications, pp. 1232–1238 (2008)
7. Ames, A.D., Gregg, R.D., Spong, M.W.: A Geometric Approach to Three-Dimensional Hipped Bipedal Robotic Walking. In: 46th IEEE Conference on Decision and Control (2007)
8. Chevallereau, C., Abba, G., Aoustin, Y., Plestan, F., Westervelt, E., Canudas-de-Wit, C., Grizzle, J.W.: RABBIT: A testbed for advanced control theory. IEEE Control Systems Magazine 23(5), 57–79 (2003)

9. Vukobratovic, M., Juricic, D.: Contribution to the synthesis of biped gait. IEEE Trans. Biomed. Eng. BME-16(1), 1–6 (1969)
10. Sobotka, M., Wolff, J., Buss, M.: Invariance Controlled Balance of Legged Robots. In: European Control Conference (2007)
11. Kajita, S., Tani, K.: Study of Dynamic Biped Locomotion on Rugged Terrain- theory and basic experiment. In: ICAR, Robots in Unstructured Environments, vol. 1, pp. 741–746 (1991)
12. Hirukawa, H., Kanehiro, F., Kajita, S., Fujiwara, K., Yokoi, K., Kaneko, K., Harada, K.: Experimental evaluation of the dynamic simulation of biped walking of humanoid robots. In: ICRA, vol. 2, pp. 1640–1645 (2003)
13. Pfeiffer, F., Lff er, K., Gienger, M.: Humanoid robots. In: CLAWAR, pp. 505–516 (2003)
14. Kajita, S., Kanehiro, F., Kaneko, K., Yokoi, K., Hirukawa, H.: The 3D linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. IEEE/RSJ IROS 1, 239–246 (2001)
15. Nise, N.S.: Control systems engineering, 4th edn. John Wiley and Sons, USA (2004)
16. Slotine, J.J.E., Li, W.: Applied nonlinear control. Printice-Hall international editions, London (1991)
17. Napoleon, N.S., Sampei, M.: Balance control analysis of humanoid robot based on ZMP feedback control. IEEE/RSJ IROS 3, 2437–2442 (2002)
18. Okumura, Y., Tawara, T., Endo, K., Furuta, T., Shimizu, M.: Realtime ZMP compensation for biped walking robot using adaptive inertia force control. In: IROS, vol. 1, pp. 335–339 (2003)
19. Katayama, T., Ohki, T., Inoue, T., Kato, T.: Design of an optimal controller for a discrete-time system subject to previewable demand. Int. J. of Control 41(3), 677–699 (1985)

Object-Specific Grasp Maps for Use in Planning Manipulation Actions

Franziska Zacharias, Christoph Borst, and Gerd Hirzinger

Abstract. Humans use learned knowledge to solve reaching tasks and to manipulate objects and tools. We believe that representations of manipulation characteristics of an object and of the reaching capabilities of a robotic arm can speed up low-level planners, like grasp planners. They also enable sophisticated scene analysis and reasoning for high-level planners, like task planners.

We present object-specific grasp maps to encapsulate an object's manipulation characteristics. A grasp planner is shown to use the grasps maps and a representation of the reachable workspace. The exploitation of the provided knowledge focuses the planning on regions of the object that are promising to yield high quality grasps. Speed ups of factor 2-12 are reported.

1 Introduction

In the course of life a human learns to use his arms but also to grasp and use tools and objects. Thus the human can rely on knowledge about the world and about himself to decide which regions are reachable for him and how to approach objects. It could be imagined that this knowledge is saved in the form of a model that can be referenced in action planning. Our goal is to use models of a robotic arm's capabilities and an object's manipulation properties to guide the task planning process at a high level of abstraction. The models can furthermore be used to parameterize path and grasp planners and enable them to solve a problem more quickly (Fig. 1). We introduced a representation of a robotic arm's reachable workspace [13] that characterizes regions w.r.t. their reachability from various directions. Using this representation it can easily be determined whether an object in the scene is reachable and from what

---

Franziska Zacharias · Christoph Borst · Gerd Hirzinger
German Aerospace Center (DLR), Institute of Robotics and Mechatronics,
Münchner Straße 20, 82234 Weßling, Germany,
e-mail: {`franziska.zacharias,christoph.borst,`
`gerd.hirzinger`}`@dlr.de`

directions. While an object may be reachable for the robotic arm, the question whether stable grasps can be realized from the corresponding approach direction is still open. An object like a coffee pot for instance is not graspable equally well in all regions of its geometry model. Grasps which are not suited to the hand kinematics (Fig. 2(a)) should be discarded early. Due to the object's orientation a prefered grasp (Fig. 2(b)) may not alway be reachable and other grasps (Fig. 2(c)) have to be used. Or all grasps may be unreachable (Fig.2(d)). A model of an object's manipulation properties should reflect these issues. A high-level planner could use this model to determine if good approach directions are accessible. In case of cluttered scenes, it could be analyzed whether a rearrangement of the scene or a repositioning of the mobile manipulator is required. In path planning research, workspace knowledge [14] and knowledge about the capabilities of a robotic arm [5] is used to speed up planning. However to the authors' knowledge neither current high-level planners like a manipulation planner nor low-level planner like a grasp planner use knowledge about an object's manipulation characteristics or representations of the reachable workspace. Instead state of the art grasp planners [9], [10], [7] integrate additional algorithmic frameworks to ensure particular characteristics. For instance a robotic arm's inverse kinematics is directly integrated into the grasp planner to ensure that grasps are reachable by the robot arm. In the long run this leads to huge monolithic planning systems. The representation of good grasps has also received attention. In manipulation planning often fixed manipulation points on the objects [12], predefined grasps [1] or simple rules for grasp generation are used [2] instead of versatile grasp planners. These methods discard a lot of manipulation possibilities especially for multifingered hands. Pelossof et al. [11] use an SVM approach to learn from a set of good grasps how to generate grasps for similar objects. Gienger

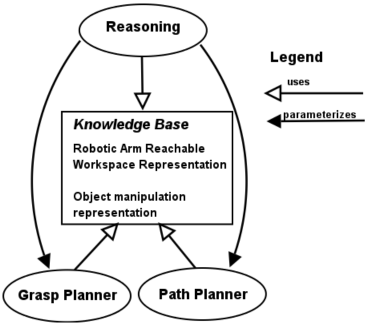


Fig. 1 Knowledge about the world and about the robot is used to guide and parameterize planning processes.

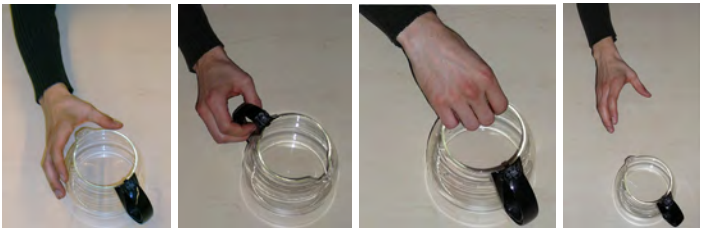

(a) (b) (c) (d)

Fig. 2 Various ways to reach for and grasp a coffee pot.

et al. [6] represent a continuous set of power grasps by a Rapidly-exploring Random Tree. The object-specific grasp information is used in motion optimization to speed up motion planning. However, these representations where not designed to be efficiently exploited by planners like grasp planners, or task planners.

In this paper we take a first step towards this goal and introduce a simple representation of an object's manipulation capabilities. We show how a grasp planner can use this representation and a previously published representation of a robot arm's reachable workspace to avoid wasting planning efforts on regions of the object to be grasped that do not participate in reachable high quality grasps.

2 Grasp Maps for Use in Grasp Planning

In this section we demonstrate the benefits of including object information into grasp planning. We summarize the grasp planner introduced by Borst et al. [4]. We describe the derivation of simple grasp maps and adapt the grasp planner presented by Borst et al. [4] to use this object knowledge. We assume that a geometric triangulated model of an object is available. No preprocessing steps are necessary.

2.1 The Grasp Planning Algorithm

The main aspect for a planned grasp is to hold an object firmly and safely, also in the presence of disturbances acting on the object, e.g. during transport motions. Grasps that have this property are called force-closure grasps. Unbound forces in the contact points are assumed. In this paper a grasp is composed of k point contacts with friction, with one contact per finger (precision grasp). We focus on multi-fingered force-closure precision grasps. Borst et al. [4] presented an intuitive measure for evaluating the quality of a grasp. They furthermore proposed an efficient generate and test algorithm to plan good but not optimal grasps for robotic hands independent of a robotic arm in very short time.

The grasp quality measure is based on the two concepts of the grasp wrench space GWS and the object wrench space OWS. A wrench is composed of the forces and torques acting in a contact point. The GWS of a given grasp describes disturbances that a grasp can counterbalance. The OWS of a given object describes which disturbances may occur. The largest factor by which the OWS can be scaled to just fit in the GWS was identified as an intuitive physically motivated measures of a grasp's quality. Based on this measure a generate and test grasp planner was introduced. It includes the following steps:

1. Contact points are generated by randomly sampling a face of the triangulated object model (uniform distribution).
2. Heuristic fast prefiltering steps roughly check force closure and gripper constraints. The tests are conservative since no solutions are discarded.
3. Grasp quality and robustness is evaluated
4. Gripper configuration is determined
5. Collision checks of the hand with the object to grasp are performed

All combined the grasp planner can produce good grasps for arbitrary 3D objects online. Depending on the object complexity to generate a grasp the planner needs between 20 ms-180 ms on an Intel Pentium D 3 GHz with 2 GB main memory.

2.2 Grasp Maps

A grasp planner based on random sampling strategies produces high and low quality grasps. We believe that some regions of an object's geometry contribute to high quality grasps more than others. When thinking of objects like the coffee pot (Fig. 2), a martini glass or a cup this immediately becomes obvious. In the grasp map, we intend to represent how specific regions of an object participate in manipulation. The geometric object model will be attributed with region specific aspects. Did a specific region participate in force closure grasps? What quality did these grasps have? The grasp planner presented in the last section places contact points on the object without any knowledge of the kinematics of the robotic hand. Therefore a lot of grasps are sampled that are kinematically infeasible or cause collisions between the object and the hand. The object-specific grasp maps will provide this knowledge to the grasp planner at the earliest stage. Using these grasp maps a grasp planner can automatically bias the generation process to concentrate the search for grasps in regions promising to yield higher quality force-closure grasps faster.

We illustrate the grasp map generation using an object constructed of primitive shapes (Fig. 3). A cube, a cylinder and a cone are concatenated. Using the grasp planner described in the preceding section we generate a number of n grasps per geometric object model. For each face we register its participation in valid force closure grasps. Furthermore we iteratively compute for each face the minimum, the maximum and the mean of the grasp quality measure for the grasps it participated in. Using these attributes important manipulation characteristics can be visualized. Fig. 3 shows two views of the grasp map for our example object. In Fig. 3 (left) the color encodes the absolute frequency of a face participating in a force closure grasp. Regions that were often part of grasps are light gray. Those that were seldom used are dark gray. In Fig. 3 (right) the color encodes the mean grasp quality value from low quality (dark gray) to high quality (light gray). This object shows a clear regional preference w.r.t. where high quality grasp can be found i.e. they are most often found on the cube part. Combining the information in both pictures it can be seen that high quality grasps are also found on the cylinder but seldom.

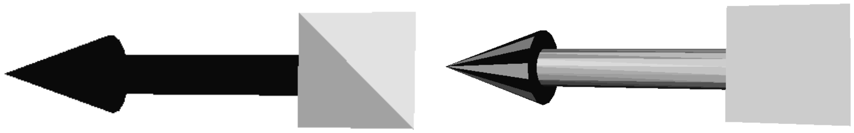


Fig. 3 (Left) Absolute frequency of a face participating in a grasp. (Right) Mean quality per face.

2.3 Feeding the Grasp Planner with Object Information

As a first step the object specific manipulation knowledge represented by the grasp maps will be used in a modified version of the grasp planner presented in section 2.1. We will show that by integrating the grasp map into the grasp planning algorithm higher quality grasp can be generated significantly faster, especially for complex objects. Borst et all. showed that for 4 or 5 finger grasps on a glass only 8% to 15 % of the sampled grasps were force closure grasps. Research in motion planning has already discovered workspace biasing methods and resulting non-uniform sampling distributions to lead to significant speedup in planning [3], [14]. Therefore we will modify the contact generation step of the grasp planner presented in section 2.1. Here contact points were placed on randomly sampled faces using a uniform distribution. Using the object's grasp map, we bias this sampling process to prefer regions of an object that promise to yield good force-closure grasps. We use the biasing method presented in [14]. Specifically we bias the distribution to favor faces that have a high max grasp quality measure and were often part of valid grasps. These two characteristics need to be combined. Otherwise given a high max grasp quality measure a face that e.g. participated only in one grasp would be treated the same as a face that was often part of valid grasps.

2.4 Discussion of Results

We evaluate the impact of using object-specific grasp maps in the grasp generation process using the objects in Fig. 4. For each object we generated the grasp map offline based on a set of 20000 grasps. The grasps were generated with the original grasp planner described in section 2.1. It should be noted that the coffee mug and the martini glass have interior regions which are also considered by the grasp planner. To measure the improvement we computed the time to generate one grasp and the mean grasp quality measure (Tab. 1). Since the grasp planner incorporates a random-sampling process the values were averaged using 1000 grasps. As expected using grasp maps for objects of rotational symmetry achieves practically no improvement. The reason for this is that valid force closure grasps are uniformly distributed across the object surface. However, a clear improvement w.r.t. time and quality was achieved for the martini glass and especially for the coffee mug. Here our assumption that not every region of an object participates in valid grasps is intuitive and valid. The object-specific grasp map prevents the grasp planner from exploring unpromising regions. For the coffee mug these regions are

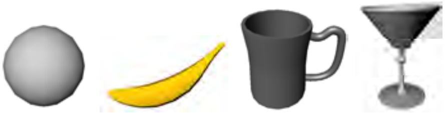


Fig. 4 Objects for testing the enhanced grasp planner.

Table 1 Time (in ms) and grasp quality measurements for all objects.

	original planner		grasp map planner	
	time	mean quality	time	mean quality
sphere	22	0.59	26	0.575
banana	17	0.09	24	0.098
martini glass	75	0.085	35	0.1
coffee mug	119	0.127	21	0.164

the inner regions and the handle. For the martini glass the regions on the stem are found to be rather insignificant for force closure grasps.

3 Reachable Grasps

Up to now it could be argued that a set of valid force closure grasps could equally well be generated for each object and saved in a database instead of biasing the grasp planner using grasp maps and computing grasps online. However as soon as the hand is attached to a robotic arm the advantages of using the grasp map strategy become evident. Reachable grasps cannot be saved for each object since they depend on the object's position w.r.t. the robot. Assuming a database of grasps for an object is available two strategies could be pursued. In the first strategy each grasp in the database would need to be checked for reachability using the robotic arm's inverse kinematics. They would furthermore have to be checked for collisions with the environment. Especially for cluttered scenes and complex objects that are difficult to grasp this can be an expensive process. A second strategy could be similar to that presented by Berenson et al. [1]. Here a database of grasps for each object is assumed to exist. The grasps are then ranked by combining several heuristic values that take the robot kinematics and the environment into account. However each grasp has to be tried out and evaluated till a valid reachable and collision-free configuration of arm and hand is found. In this section we show that reachable grasps can be generated without introducing heuristics. We generate grasps online by using the grasp map approach. By using a representation of a robot arm's reachable workspace we decouple the grasp planner from the inverse kinematics of the arm.

3.1 Ensuring Modularity by Using the Capability Map

Humans use internal models to solve reaching tasks [8], to infer which regions are reachable from what directions. Possessing a similar abstraction of a robot arm's capabilities in its workspace is important for task planning. We previously introduced a model that can be used for this purpose [13] and will summarize its main points.

3.1.1 The Capability Map of the Robot

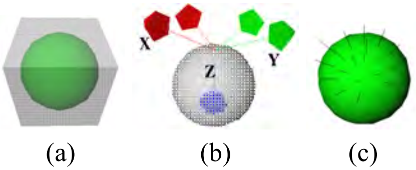


Fig. 6 Sphere inscribed into the cube (a), exemplary frames for a point on the sphere (b), valid inverse kinematics solutions on a sphere (c).

The theoretically possible robot arm workspace is enveloped by a cube and subdivided into equally-sized smaller cubes. Into each cube a sphere is inscribed and on this sphere *n* points are uniformly distributed (Fig. 6 (a), (c)). Frames are generated for each point and serve as the target tool center point (TCP) for the inverse kinematics of the robot arm (Fig. 6 (b)). The point on the sphere determines the z-axis of the TCP frame. The orientation of the x-axis and y-axis is subsampled. The result of the inverse kinematics is registered in the data structure that is visualized by the sphere. Reachable points on the sphere are visualized as black lines. The spheres visualize the reachability for a region and are therefore called reachability spheres. Fig. 6 (c) shows a sphere where only some points are reachable. The reachability sphere map is the aggregation of all spheres. It can be used to visualize and inspect the reachability across the workspace and to approximate the shape of the robot arm workspace. It was shown that the reachability sphere map of the DLR robot arm is regularly structured [13]. The lines on the reachability spheres form cone-like structures in outer workspace regions. These cones then open out and become cylinders in the center of the workspace. Towards the inner workspace regions the cylinders change into cones again. Due to this observation, cones and cylinders were fitted to the reachability spheres to capture the data. The quality of the fit was measured by

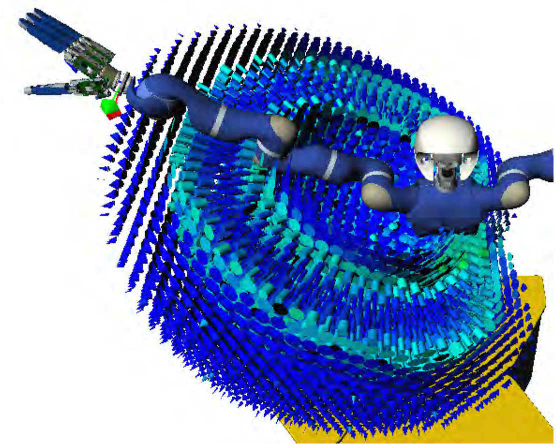


Fig. 5 The capability map for the right arm of the humanoid robot Justin.

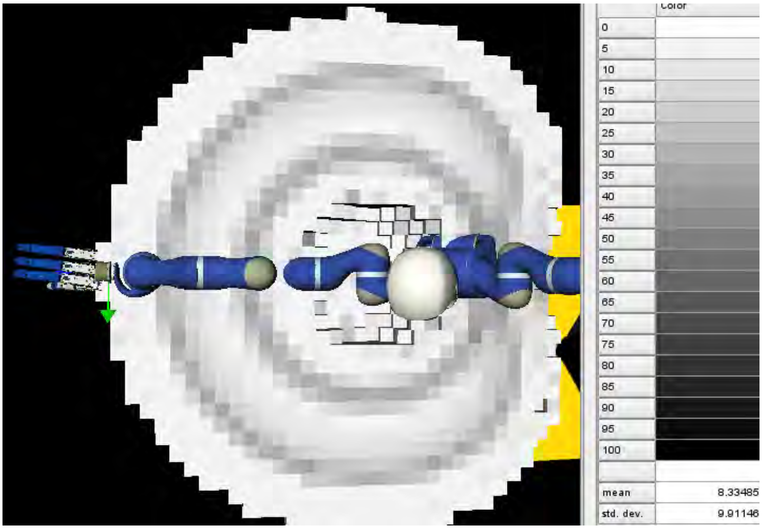


Fig. 7 Gradient of the SFE in grayscale for the capability map for the right arm of the humanoid robot Justin.

the relative error of an approximation, called the shape fit error (SFE). The capability map of the robot arm is a representation of the reachability sphere map. It is derived by replacing every sphere by the best fitting shape. Fig. 5 shows the capability map for the right arm of the DLR robot Justin. The location where the hand was attached to the robot arm serves as TCP for the map computation. It is marked by a coordinate system. The SFE is represented in grayscale in Fig. 7. Here cubes replace the cones and cylinders to allow the isolated inspection of the gradient of the SFE. Light gray indicates low error (0) and dark gray indicates high error (100). The mean SFE of 8.33 shows that the capability map is a valid representation of the reachability sphere map.

3.1.2 Filtering Grasps Using the Capability Map

Using the capability map, it can easily be determined whether an object of the scene is reachable and from which directions. Fig. 8 (a) shows a bottle placed near the outer border of the workspace. In this region cones are used to represent the directions from which the areas can be reached. Fig. 8 (b) shows the same bottle placed at the center of the robot arm workspace. Here cylinders are predominantly used to capture the reachability data. The possibilities to approach and manipulate the object are more numerous as is directly evident from the capability map. We provide the grasp planner with the capability map as a model of the robot arm's reachable workspace. The grasp planner then uses this model to predict the reachability of a grasp. A grasp is composed of 4 finger contact points on the object, 4 surface normals at these contact points and a frame T_H defining the position and orientation

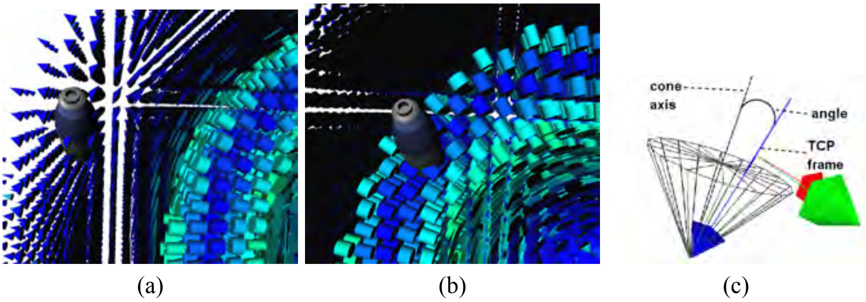


Fig. 8 (a,b) A bottle in different areas of the workspace, (c) test for reachability.

of the hand in a reference coordinate system. This frame T_H is tested against the capability map to determine the reachability of the grasp. The frame is mapped to a region of the capability map. To test its reachability the inner product between the z-axis of the TCP frame and the axis of the respective shape is computed. This is then compared to the opening angle of the shape (Fig. 8 (c)). If the test fails the grasp is discarded as unreachable. The filtering step using the capability map is added to the *Fast Prefiltering* step of the grasp planner. To evaluate the achieved improvement we measure the time to generate one reachable grasp with the original system and our improved grasp planner that uses grasp maps and the capability map of the robot arm. The original grasp planner does not determine a grasp's reachability by a robotic arm. Therefore we first create a grasp using the original grasp planner and check it for reachability using the arm's inverse kinematics. If it is not reachable a new grasp is generated. The execution times averaged over 1000 grasps are listed in table 2. A clear improvement is visible. The objects were placed in the center of the reachable workspace. The speed up ranged from factor 2 to 6 depending on object geometry. For different positions and orientations of the objects in the workspace speed ups up to factor 12 were observed. The effect was more pronounced in the bordering workspace regions since here fewer grasps are actually reachable. Orientation dependency can be observed for objects like the coffee cup. The orientation of the handle in the workspace influence the speed up to range from factor 3 to 12 given the object position is kept.

Table 2 Time (in ms) and grasp quality measurements for all objects.

	original		improved	
	time	mean quality	time	mean quality
sphere	120	0.58	95	0.576
banana	91	0.067	94	0.074
martini glass	306	0.075	158	0.084
coffee mug	395	0.130	68	0.157

4 Outlook and Future Work

In this paper we introduced the idea of object-specific grasp maps. We presented the capability map and the grasp maps as resources for a grasp planner. The randomized grasp planer was able to exploit the provided knowledge to bias the exploration towards the most promising regions. In the best cases a good grasp was found 12 times faster than before. In most cases also the mean grasp quality was significantly improved w.r.t. to the original planner. Thus the grasps produced with the grasp map based system have a higher quality than grasps produced with the original planner. In the long run we expect the capability map and the object-specific grasp maps to be a valuable resource for a number of planning subsystems (Fig. 1). Future versions of the grasp maps will contain not only information w.r.t. good finger positions but also identify good approach directions. In a cluttered environment a reasoning component could use the representations to determine whether an object is reachable and graspable at all. Using this information, a scene could be judged to be too difficult if a lot of approach directions are obstructed for the object manipulation. A rearrangement planner [12] could be triggered to simplify the scene setup and make it possible to achieve the task goal.

The quality of the grasps produced using the grasp maps is dependent on the features chosen, e.g. max quality measure, and their corresponding weights. To obtain an even better performance these weighting factors could be learned using the method described in [14]. If a robot often handles a specific object or if it does not know an object before hand and is able to build a geometric model online it is also imaginable to learn the grasp maps iteratively online and thereby improve the grasping process. This is a clear advantage compared to relying on a database of grasps.

References

1. Berenson, D., Diankov, R., Nishiwaki, K., Kagami, S., Kuffner, J.: Grasp planning in complex scenes. In: IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids 2007) (2007)
2. Berenson, D., Kuffner, J., Choset, H.: An optimization approach to planning for mobile manipulation. In: Proc. IEEE Int. Conf. on Robotics and Automation (ICRA) (2008)
3. Boor, V., Oversmars, M., van der Stappen, A.: The gaussian sampling strategy for probabilistic roadmap planners. In: Proc. IEEE Int. Conf. on Robotics and Automation (ICRA), pp. 1018–1023 (1999)
4. Borst, C., Fischer, M., Hirzinger, G.: Efficient and precise grasp planning for real world objects. Springer Tracts in Advanced Robotics 18, 91–111 (2005)
5. Diankov, R., Ratliff, N., Ferguson, D., Srinivasa, S., Kuffner, J.: Bispace planning: Concurrent multi-space exploration. In: Proc. Int. Conf. on Robotics: Science and Systems (2008)
6. Gienger, M., Toussaint, M., Goerick, C.: Task maps in humanoid robot manipulation. In: Proc. IEEE Int. Conf. on Intelligent Robots and Systems (IROS), pp. 2758–2764 (2008)
7. Harada, K., Kaneko, K., Kanehiro, F.: Fast grasp planning for hand/arm systems based on convex model. In: Proc. IEEE Int. Conf. on Robotics and Automation (ICRA) (2008)

8. Kawato, M.: Internal models for motor control and trajectory planning. Curr. Opin. Neurobiol. 9(6), 718–727 (1999)
9. Lopez-Damian, E., Sidobre, D., Alami, R.: A grasp planner based on inertial properties. In: Proc. IEEE Int. Conf. on Robotics and Automation (ICRA), pp. 754–759 (2005)
10. Miller, A.T., Knoop, S., Christensen, H.I., Allen, P.K.: Automatic grasp planning using shape primitives. In: Proc. IEEE Int. Conf. on Robotics and Automation (ICRA) (2003)
11. Pelossof, R., Miller, A., Allen, P., Jebara, T.: An svm learning approach to robotic grasping. In: Proc. IEEE Int. Conf. on Robotics and Automation (ICRA) (2004)
12. Stilman, M., Schamburek, J.U., Kuffner, J., Asfour, T.: Manipulation planning among movable obstacles. In: Proc. IEEE Int. Conf. on Robotics and Automation (ICRA) (2007)
13. Zacharias, F., Borst, C., Hirzinger, G.: Capturing robot workspace structure: Representing robot capabilities. In: Proc. IEEE Int. Conf. on Intelligent Robots and Systems (IROS), pp. 3229–3236 (2007)
14. Zucker, M., Kuffner, J., Bagnell, J.A.: Adaptive workspace biasing for sampling-based plaanners. In: Proc. IEEE Int. Conf. on Robotics and Automation (ICRA), pp. 3757–3762 (2008)

Vision Controlled Grasping by Means of an Intelligent Robot Hand

Giulio Milighetti, Moritz Ritter, and Helge-Björn Kuntze

Abstract. In this paper a new visual servoing concept for dynamic grasping for humanoid robots will be presented. It relies on a stereo camera in the robot head for wide range observing in combination with a miniaturized close range camera integrated in a five finger hand. By optimal fusion of both camera information using a fuzzy decision making algorithm a robust visually controlled grasping of objects is achieved even in the case of disturbed signals or dynamic obstacles.

1 Introduction

Within the last years a new challenging field of robotics has arisen in which assisting robots cooperate and interact with humans, both imitating their behavior patterns and performing with them and/or in their environment diverse complex tasks. Due to the ambitious tasks and the complexity of a time varying environment, these “humanoid” robots have to be provided with much more intelligence in comparison with the current industrial robots available on the market. That means first of all, that the robot has to be equipped with human-like senses which enable it to cope with these extreme conditions. Thus, the robot has to dispose of a large variety of sensors with the capability of extracting, combining and analyzing their information in order to react autonomously in the most appropriate way. Moreover, in order to cope with a manifold of different possible situations, the robot has to be able to manage specific skills in a human-like manner.

In order to properly develop these issues the Deutsche Forschungsgemeinschaft (DFG) has established the long-term Collaborative Research Centre SFB 588 ”Humanoid Robots” in which the various aspects of the multifaceted interaction

---

Milighetti Giulio · Moritz Ritter · Helge-Björn Kuntze
Fraunhofer-Institute for Information and Data Processing (IITB),
Fraunhoferstraße 1, 76131 Karlsruhe, Germany,
e-mail: `{milighetti,ritter,kuntze}@iitb.fraunhofer.de`

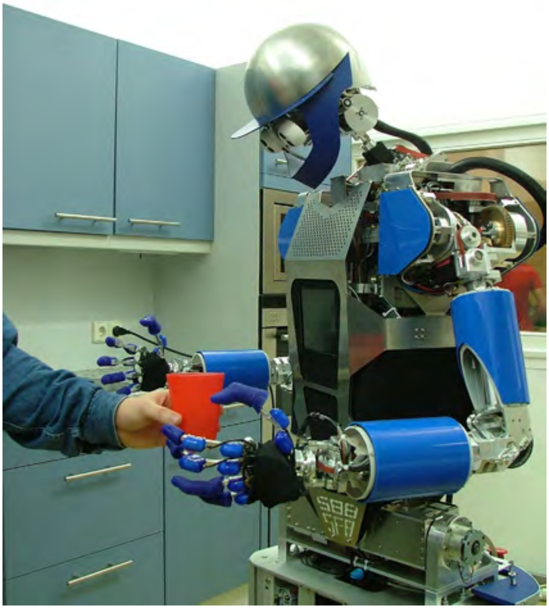


Fig. 1 ARMAR III grasping a cup [4].

between robots and humans are investigated. Within this framework, the Fraunhofer Institute IITB is responsible for the development of a modular and flexible supervisory control concept which allows the robot to cope with different skills. Typical implemented basic skills are for example the fitting of parts into holes (e.g. key into key hole), the balancing instable parts on a tray or the lifting and carrying of smooth objects by adherence [1].

One of the most important basic skills which is needed for a robot in almost every daily situation is the capability of grasping randomly placed objects by means of vision (see Figure 1). Contrary to industrial robots with very stiff and accurate arm mechanics, the required accuracy of a humanoid light weight arm can only be achieved by sensor based feedback control concepts. Moreover, the position of the target object in an unstructured environment could change during the task execution and dynamic obstacles could hinder the robot motion. For these reasons, the desired trajectory has to be planned dynamically and a constant supervision of the whole process has to be assured by appropriate vision based control concepts ("Visual Servoing" [2, 3]).

Assuming simplified geometric conditions and limited robot dynamics, visual servoing problems like grasping objects from a moving conveyor belt have been solved at Fraunhofer IITB even in the early eighties [5]. Thanks to tremendous improvements in processor performance, the capability of vision based control concepts has grown considerably within the last years. Thus, more and more ambitious robotic applications within complex environments under critical conditions can now be solved. Especially in the automotive industry camera driven robotic systems are used more and more frequently, for example for assembly and manipulation [6] or to guide welding robots [7]. Robot positioning [8] and collision free motion in a dynamic environment [9] are applications still distant from the industrial field but essential in service robotics.

In this contribution a new visual servoing concept for a grasping task in a complex unstructured environment will be presented. It relies on two cameras: one wide range stereo camera integrated in the robot head as well as one close range camera embedded in a five finger robotic hand. By optimal fusion of both camera information using a fuzzy decision making algorithm a robust visually controlled grasping of objects is achieved even in the case of disturbed signals or dynamic obstacles. The paper is organized as follows: after a brief description of the vision-based grasping concept in section 2, the experimental set-up is presented in section 3. The theoretical background for a fuzzy-based combination of the two cameras is then summarized in section 4. Finally first experimental results are presented in section 5.

2 Vision-Based Grasping Concept

As already pointed out, a humanoid robot has to act in a time-varying and unstructured environment where it has to cope with both internal (kinematic) uncertainties and uncertainties about its surroundings. Hence, there is the need for a constant visual feedback during the task execution in order to ensure an optimal positioning of the robot hand with a collision free trajectory.

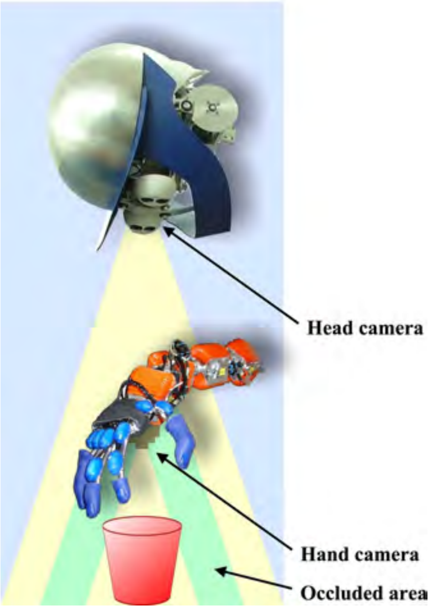


Fig. 2 Coordination of head and hand camera.

The simultaneous use both of a close range camera in the hand (hand camera) and of a wide range camera in the head (head camera) makes the grasping process more robust and accurate. By means of the head camera the robot can obtain a first rough estimation of the goal position and can align its hand camera with the object. During the approach, while the accuracy of the head camera measurement remains almost constant, the object localization by means of the hand camera becomes more and more accurate. By intelligently fusing the two sensors, the robot can obtain in every condition an optimal estimation of the object position. Moreover, the concurrent use of two cameras is advantageous in the case of occlusions due for example to the robot arm or to some obstacle (see Figure 2). The allocation of the head camera for the execution of parallel tasks (e. g. giving the attention to a human during a dialogue) is also permitted without loosing accuracy in the grasping. In order to use the most appropriate camera for assuring at any instant a reliable measurement, the robot has to evaluate both sensors depending on their properties (e. g. their position relative to the object) and on the actual environment conditions (e. g. illumination of the room) using some optimal criterion. The adopted decision algorithm is presented in section 4.

3 Experimental Set-Up

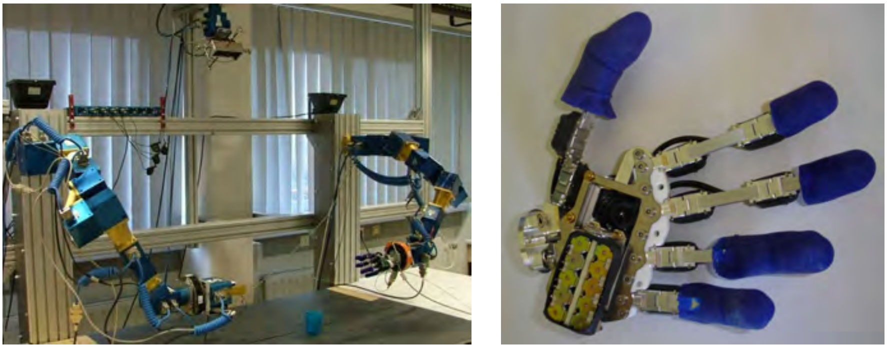
For developing the presented vision-based basic skill, the experimental set-up established at Fraunhofer IITB has been used. This section gives a short description of the robotic platform and more detailed specifications of the five-finger hand. An overview about the used control architecture is also given.

3.1 Multi-sensor Robot Platform

The experimental test platform available at Fraunhofer IITB consists of two robot arms each with 7 degrees of freedom (DoF), a 2DoF pan-tilt sensor head and a 5 finger fluid hand which will be described in detail in the next paragraph.

For coping with a variety of basic skills the robot is equipped with several redundant and complementary sensors (see Figure 3(a)). The head is equipped with a stereo camera able to track predefined objects and with an acoustic sensor (microphone array) able to determine the position of sound sources. Moreover, a miniaturized camera for accurately localizing objects at close range is integrated in the palm of the hand. For the tactile inspection two force-torque sensors are mounted on the wrists and the fingers of a gripper are equipped with tactile arrays. Moreover, a slip sensor can be installed in one of the grippers in order to detect the relative motions between end-effector and surfaces in contact with it.

Both cameras as well as the acoustic and slip sensors are connected to a dedicated computer where a first processing of the data takes place. The results are then sent via UDP/IP communication to the main computer where the robot control is implemented.



(a) Two arms system (b) Smart hand with camera

Fig. 3 Multi-sensor robot platform at IITB.

3.2 Smart Hand

The five-finger hand mounted in one robot arm at IITB is a collaborative development with the Forschungszentrum Karlsruhe. The requirements of a hand for humanoid robots can be summarized by light weight, high dexterity, compactness, robustness and by the ability to provide the needed grasping forces. The developed hand provides all these human capabilities by a design which mimics a natural hand and its fundamental kinematics. Thus, it has 8 active DoF (one DoF in the small finger and in the ring finger and two in the other fingers) with the possibility of opposition movement of the thumb (see Figure 3(b)). Every DoF is driven by a flexible fluidic actuator integrated in the finger joint, which is suitable for a weight saving construction, granting at the same time the needed forces [10]. In order to investigate new solutions for vision controlled grasping processes, a small camera has been mounted in the ball (see Figure 3(b)). The chosen camera is a Point Grey color camera with a resolution of 640x480 pixels, able to grab images with a frame rate of 30 fps.

3.3 Control Architecture

The basic discrete-continuous structure of the supervisory concept developed in order to drive the robot throughout its visual-controlled task is shown in Figure 4 and presented in detail in [11]. The state of the robot and of the environment in which it acts are supervised with the help of proprioceptive (e.g. encoder) and exteroceptive sensors (e.g. camera, microphone, force-torque). A hierarchical two-level control is then responsible for the interpretation of such an information.

In the upper level, a discrete control processes the measurements coming from the sensors and uses them in order to generate diagnosis signals that contain quantitative information about the continuous state of the system (e.g. position of objects, sounds, forces). As a second step, this information serves for the identification of the discrete state or event providing the qualitative information about the present situation (e.g.

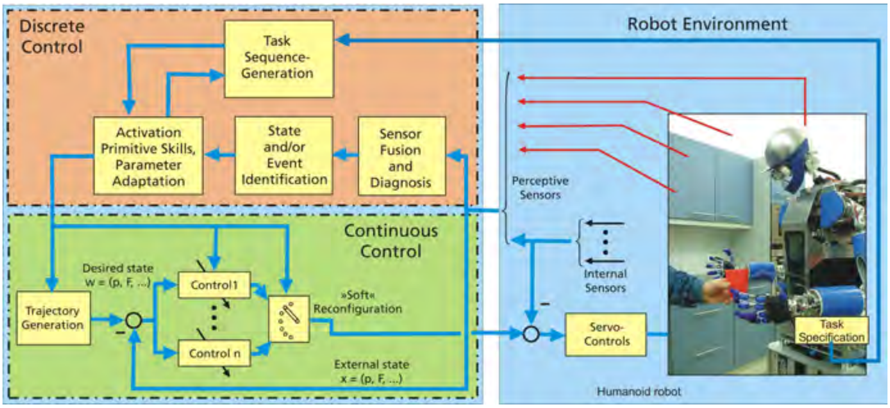

Discrete Control
Task Sequence-Generation
Activation Primitive Skills, Parameter Adaptation
State and/or Event Identification
Sensor Fusion and Diagnosis
Continuous Control
Desired state w = (p, F, ...)
Control1
Control n
Trajectory Generation
»Soft« Reconfiguration
External state x = (p, F, ...)
Robot Environment
Perceptive Sensors
Internal Sensors
Servo-Controls
Task Specification
Humanoid robot

Fig. 4 Discrete-continuous control concept.

occurrence of a certain event). By interpreting the acquired knowledge about both the continuous and discrete state of the system, a decision unit supervises the actual task sequence, introducing a new appropriate action or adapting the present one for overcoming unexpected situations and reaching the initial desired goal.

In order to generate a flexible task sequence and to have the possibility of a fast on-line task adaptation, a discrete task structure has been developed which assures a transparent and efficient instrument for an on-line decision making algorithm. In the proposed architecture every task is handled as a chain of elementary actions called Primitive Skills (PS), each with its own sub-goal. Once the decision unit has determined the PS sequence that has to be performed and the most appropriate controller to execute the currently active PS, in the lower hierarchy level the continuous control ensures an optimal system response and the attainment of the desired values.

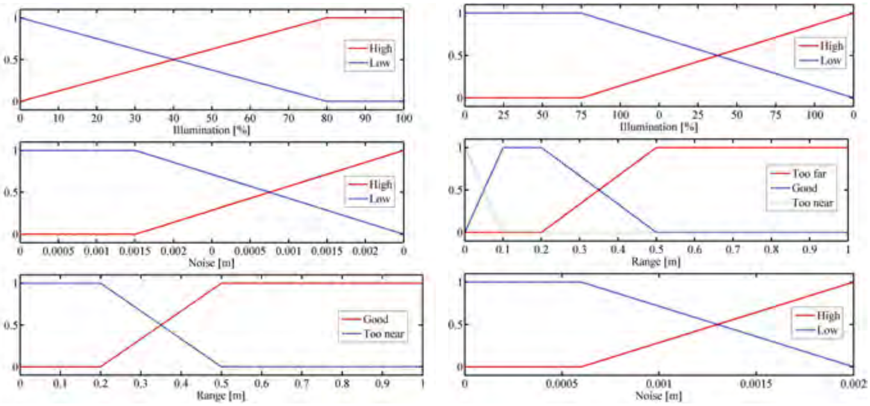
4 Decision Making Algorithm

An online supervision concept for monitoring the conditions of the two cameras has been realised by means of a fuzzy-based decision making algorithm. Using the theoretical background given in [11] the two optical sensors have been associated with the two basic actions ”approach by hand camera” and ”approach by head camera” which are implemented as two primitive skills. The decision about how to optimize the use of the two available cameras, is based on the heuristic utility function given by

$$v = E \cdot a \tag{1}$$

which has to be maximed by choosing the most appropriate PS.

In (1) the factor *a* represents the affinity of a PS, that is an a priori suitability for achieving the given goal. It is used to give a time independent preference between different PS. Its value is related for example to the nominal (or average) accuracy in reaching the goal and to the nominal (or average) execution time. The value 1 corresponds to the optimal PS for the given goal. In this case an affinity of 1 has



(a) Head camera (b) Hand camera

Fig. 5 Membership functions for the quality evaluation of the two cameras.

been given to the approach by hand camera (more accurate and faster) and a value 0.8 to the approach by head camera.

The factor E measures the efficiency of the PS, that is its current suitability by analysing how good the resources needed for its execution are presently functioning with respect to the actual state of the system. The efficiency of a resource k can be expressed by

$$\begin{aligned} &E_k(t) = av_k(t) \cdot \vartheta_k(t), \\ &av_k \in \{0,1\}, \vartheta_k \in [0,1], \qquad \forall k = 1 \dots n_{res} \end{aligned} \tag{2}$$

where av_k is the availability of the resource (1 available, 0 not available) and ϑ_k its quality (1 good functioning, 0 not functioning). For the two PS considered the evaluation of the efficiency has been carried out on the basis of three main factors:

- availability of the communication between sensor and robot (1 = available, 0 = not available)
- availability of an actual measurement (1 = received, 0 = no new measurement in the last n time steps)
- quality of the last measurement (1 = good and reliable, 0 = bad and/or unreliable)

The quality of a measurement is evaluated by taking into account three more factors that mostly influence a camera:

- luminosity of the environment
- noise of the measurement
- working range of the sensor

The membership functions associated with each of these three factors are shown in Figure 5. Once the values of luminosity, noise and range have been fuzzified, they are evaluated with very intuitive rules like for example

If *(Noise is High)* or *(Range is Too far)* or *(Illumination is Low)* then *(Quality is Low)*
If *(Noise is Low)* and *(Range is Good)* and *(Illumination is High)* then *(Quality is High)*

After the defuzzification process a value of the quality between 0 and 1 is obtained and the corresponding efficiency is weighted with the affinity in order to obtain the utility v. The so evaluated values are used for combining the measurements m_head and m_hand relative to the two sensors by the following fusion strategy:

$$m = k_1 \cdot m_{hand} + k_2 \cdot m_{head} \tag{3}$$

with

$$k_1 = \frac{v_{hand}}{v_{hand} + v_{head}}, \; k_2 = \frac{v_{head}}{v_{hand} + v_{head}} \tag{4}$$

5 Experimental Results

In order to validate the proposed concept, some experiments have been conducted on the robotic platform at Fraunhofer IITB. As test scenario the grasping of a blue cup has been investigated. The head camera used is a Mega-DCS Megapixel Digital Stereo Head from the company Videre Design. Based on its hardware a pattern matching algorithm for the localization of the goal object has been implemented. The algorithm delivers a new position with a frame rate of 15fps. The hand camera works with a frame rate of 30fps thanks to a marker-based tracking.

Figure 6 shows the efficiencies of the two sensors during the approach towards the goal object. The time scale has been stretched in order to better appreciate the variations of the presented values. Both efficiencies are influenced by the noise in the measurements and by some false or completely missed measurement (i.e. at 20 seconds for the hand camera or at 21 seconds for the head camera). Except for these variations, the head camera efficiency remains constant during the analyzed time interval because the object is not moving and therefore the working range of the camera is not changing. Only at the end of the task, while the hand is grasping the object, is the head camera no longer able to localize it and its efficiency sinks to zero (at ca. 32 seconds). On the contrary, the closer the hand camera comes to the object, the better its working range becomes and its efficiency grows accordingly till the optimal value of 1 has been reached. At a certain instant (at ca. 23 seconds), the hand camera is too close to the object and its efficiency begins to sink once again. In the last grasping phase, shortly before the object has been reached, the localization is no longer reliable as shown also by the extreme fluctuations in the efficiency.

On the basis of the calculated efficiencies, the robot is able to emphasize by (3) the control strategy based on the currently optimal sensor depending on the actual situation. In the presented example the approach is started stressing in the control loop the action of the stereo camera. In the central phase of the task the hand camera provides measurements with a higher quality and therefore its weight gets bigger. In order to avoid a blind grasping phase at the end of the approach (where the hand camera is no more able to localise correctly the object) the robot has to switch back again to the head camera.

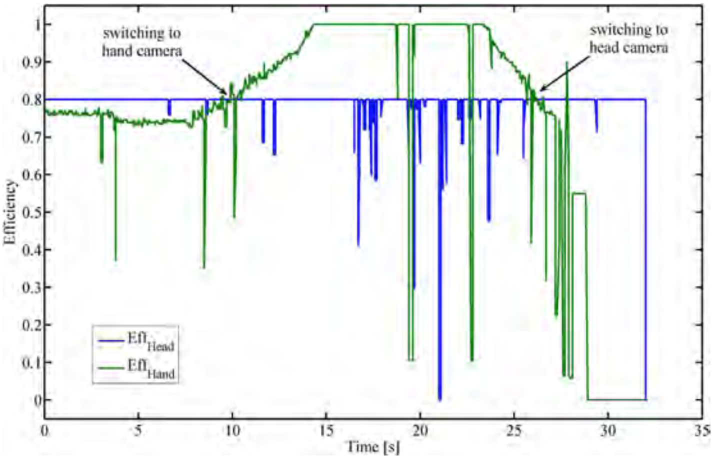
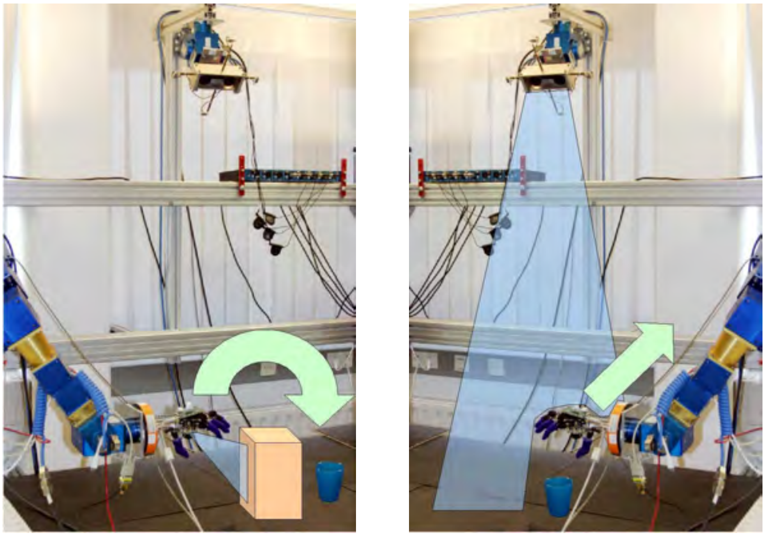


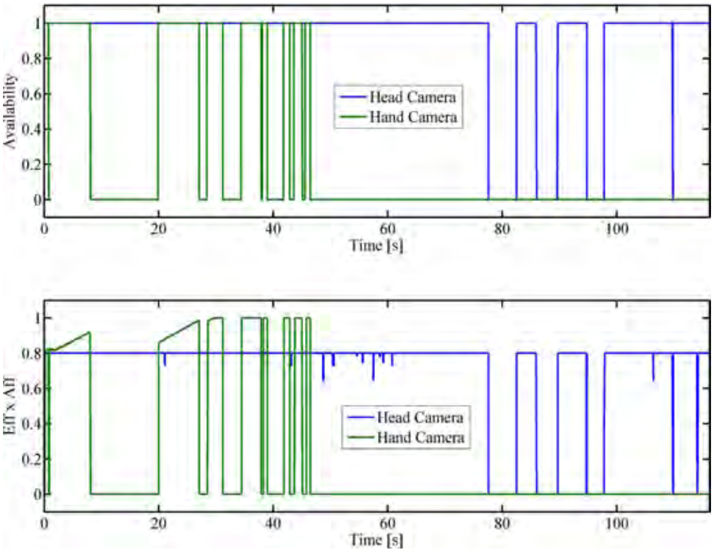
Fig. 6 Sensor efficiencies during the grasping task.



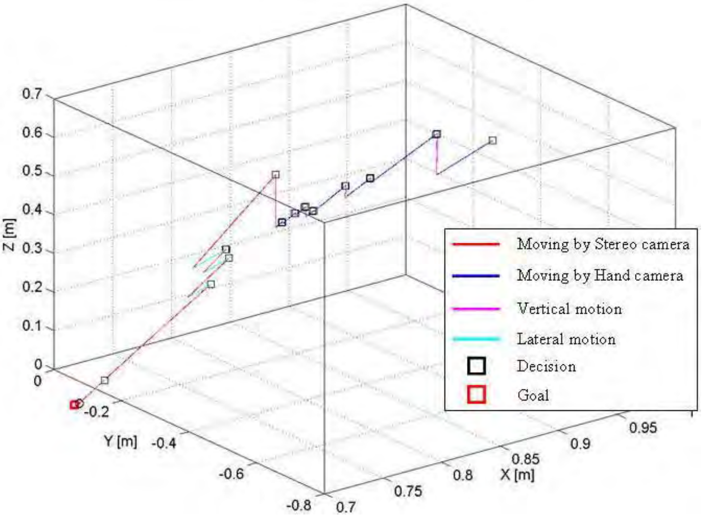
(a) Occlusion of hand camera (b) Occlusion of head camera

Fig. 7 Unexpected situations during the task execution.

Also in case of unexpected events like for example the presence of an obstacle (Figure 7(a)) or the occlusion of the view due to the motion of the arm (Figure 7(b)), the values of the efficiencies of the two sensors and of their associated PS can be used in order to overcome such situations by activating some correction in the plan. In the two presented situations, a vertical and a lateral motion can be respectively executed for overcoming the obstructions. In Figure 8(a) and Figure 8(b) the efficiencies of the two PS and the cartesian trajectory of the robot TCP during a sce-



(a) Estimated efficiencies



(b) Cartesian trajectory

Fig. 8 Approach with several occlusions of the cameras.

nario with several occlusions are respectively shown. Firstly, four occlusions of the hand camera have been simulated. The correspondence between the lowering of the availability (in this time interval no measurements are received) and the value of the efficiency is clearly observable. The robot reacts with vertical motions until a new measurement is available. Switching off the hand camera (at ca. 45 seconds) leads the robot to choose an execution by means of the stereo camera, although its optimal range was not yet reached. Finally, the stereo camera was shortly occluded three times and the robot moves laterally until it has again free field of view. Some isolated peaks in the two efficiencies are visible as in the previous experiment.

6 Conclusion

In this paper a vision-based control concept for a smart five-finger hand of a humanoid robot has been presented. It relies on two cameras: a miniaturized camera embedded in the robot hand for close range localisation and a stereo camera integrated in the robot head for wide range observation. For the dynamic fusion of close and wide range measurements a fuzzy-based decision-making algorithm has been developed. The experimental results of different fusion strategies have been presented and discussed.

Acknowledgements. The authors gratefully acknowledge that the research project presented in this paper has been supported by the Deutsche Forschungsgemeinschaft (DFG).

References

1. Milighetti, G., Kuntze, H.-B.: Multi-sensor controlled skills for humanoid robots. In: Proceedings of the 8th International IFAC Symposium on Robot Control (SYROCO) (September 2006)
2. Hill, J., Park, W.T.: Real time control of a robot with a mobile camera. In: Proceedings of the 9th International Symposium on Industrial Robots, Washington, DC (March 1979)
3. Martinet, P., Chaumette, F., Hashimoto, K., Malis, E.: Tutorial on advanced visual servoing. In: International Conference on Robotic Systems, Tutorial (September 2004), `http://www.robot.uji.es/EURON/visualservoing/tutorial/`
4. Asfour, T., Regenstein, K., Azad, P., Schröeder, J., Bierbaum, A., Vahrenkamp, N., Dillmann, R.: Armar-iii: An integrated humanoid platform for sensory-motor control. In: Proceedings of the IEEE International Conference on Humanoid Robots, Genoa, Italy (December 2006)
5. Kuntze, H.-B.: Closed-loop algorithms for industrial robots - a state of the art. Process Automation 1, 35–46 (1984)
6. Lippiello, V., Siciliano, B., Villani, L.: Position-based visual servoing in industrial multirobot cells using a hybrid camera configuration. IEEE Transactions on Robotics and Automation 23(1), 73–86 (2007)
7. Agapakis, J.E., Katz, J.M., Friedman, J.M., Epstein, G.N.: Vision-aided robotic welding: An approach and a flexible implementation. International Journal of Robotic Research 9(5), 17–34 (1990)
8. Namba, K., Maru, N.: Positioning control of the arm of the humanoid robot by linear visual servoing. In: Proceedings of the IEEE International Conference on Robotics and Automation, Taipei, Taiwan (September 2003)

9. Morikawa, S., Senoo, T., Namiki, A., Ishikawa, M.: Realtime collision avoidance using a robot manipulator with light-weight small high-speed vision systems. In: Proceedings of the IEEE International Conference on Robotics and Automation, Rome, Italy (April 2007)
10. Schulz, S., Pylatiuk, C., Kargov, A., Oberle, R., Bretthauer, G.: Progress in the development of anthropomorphic fluidic hands for a humanoid robot. In: Proceedings of the 4th International Conference on Humanoid Robots vol. 2 (November 2004)
11. Milighetti, G., Kuntze, H.-B.: On the discrete-continuous control of basic skills for humanoid robots. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (October 2006)

Learning an Object-Grasp Relation for Silhouette-Based Grasp Planning

Nicolas Gorges and Heinz Wörn

Abstract. This work addresses the problem of experienced-based grasping of unknown objects. A relation between objects and grasps is learned based on example grasps for a set of given objects. This machine-learning approach is applied to the problem of visual determination of grasp points based on a given silhouette of an object. The approximated function allows computing a grasp quality for any object-grasp combination. For the dimension reduction of the object silhouettes, a combination of image normalization and principal component analysis is used. A Support Vector Regression is used to learn the object-grasp relation. For the evaluation, the objects are grasped with a two-finger gripper and an imprint of a planar object on a tactile sensor matrix is used as an imaging method.

1 Introduction

In order to succeed in his daily activities, a household robot has to manipulate objects which were made for the human. Such a natural task for a human being is a more difficult procedure for the robot. The robot has to find a proper grasp position for the robot hand with respect to the object's shape and position. This works well if the object is known and the robot can choose a grasp from a grasp database, as introduced in [1].

A common approach for handling unknown object is the visual determination of grasp points using geometric properties of an object. In [2] the shape of an object is represented by the morphological skeleton and the contour of the object. The generation of grasp candidates and the calculation of their qualities are based on heuristics provided by the user. A machine learning approach is introduced in [3] where a description of an object and a grasp is mapped on a quality measure. Here,

---

Nicolas Gorges · Heinz Wörn
Institute for Process Control and Robotics, University of Karlsruhe,
Engler-Bunte-Ring 8, 76131 Karlsruhe, Germany,
e-mail: `{gorges,woern}@ira.uka.de`

the shape of the object is represented by a superquadric and the training data is automatically generated and evaluated in a simulation environment.

We want to extend the machine learning approach to the visual determination of grasp points and show that this approach is applicable to a real grasping scenario. Instead of using heuristics, we let the robot learn how to determine proper grasp points. For gaining a set of example grasps, the human shows the robot how to grasp an object. Then, the robot performs an abstraction of this knowledge and finally applies it to grasp known and unknown objects.

The abstraction of given object-grasp examples is made by the approximation of a function f which maps the object parameters **O** and the grasp parameters **G** on a quality measure q: $f(\mathbf{O},\mathbf{G}) \rightarrow q \in [0..1]$. Then, given a description of an unknown object, the task is to find a corresponding grasp **G** which maximizes the function f.

In this work, an imprint of a planar object on a tactile matrix[4] is used to obtain a silhouette of the object to be grasped. A two-finger gripper is used for grasping, as described in [5].

2 Object Description

In general, an object silhouette can be obtained by

1. a single view of a camera using foreground-background segmentation
2. an imprint of a planar or a cylindrical object on a tactile sensor matrix
3. a palpation sequence of an object's surface with a gripper - constrained to a plane

How the object silhouette was obtained is only important for the transformation from object coordinates to robot coordinates.

The proposed procedure for gaining an object description goes as follows:

1. At first, the silhouettes of several test objects are acquired. The images are divided into a training and a test set.
2. Each object silhouette is normalized according to its position and orientation.
3. The normalized training data is used for a principal component analysis. A dimension reduction is applied to each normalized image using only a subset of the obtained principal components.

2.1 Object Normalization

The normalization of the object concerning the object's position and orientation is based on a moment analysis. The process of object normalization is shown in Fig. 1. As the data of the tactile sensor matrix corresponds to a two-dimensional planar image, we analyze these images using moments up to the 2^{nd} order [6]. The two-dimensional $(p+q)^{th}$ order moment $m_{p,q}$ of an image is defined as the following double sum over all image pixels (x,y) and their values $f(x,y)$:

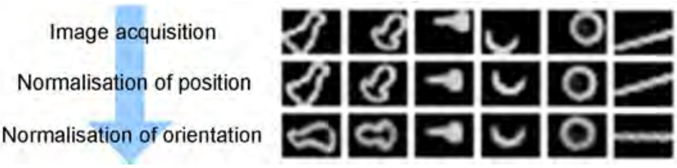


Fig. 1 Process of object normalization.

$$m_{p,q} = \sum_x \sum_y x^p y^p f(x,y) \qquad p,q \geq 0 \ .$$

The moment $m_{0,0}$ constitutes the total area of the object imprinted on the sensor. The centroid $\underline{x}_c = (x_c, y_c)^T$ of this area can be computed to

$$x_c = \frac{m_{1,0}}{m_{0,0}}, y_c = \frac{m_{0,1}}{m_{0,0}} \ .$$

Using the centroid, the object center is shifted to the image center and hence the object is normalized regarding its position. Furthermore, the centroid is needed to calculate the higher order moments, the so-called *central moments* $\mu_{p,q}$:

$$\mu_{p,q} = \sum_x \sum_y (x - x_c)^p (y - y_c)^q f(x,y) \qquad p,q \geq 0 \ .$$

The 2^{nd} order central moments can be used to compute the object's orientation given by the angle θ between the principal axes and the sensor coordinate system. The object's orientation can be readily computed by

$$\theta = \frac{1}{2} \arctan \frac{2\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}} \ .$$

To normalize the object with respect to its orientation, the object is rotated by θ around the image center.

2.2 Dimension Reduction

A principal component analysis (PCA) is used for dimension reduction. Therefore, any image matrix I of the size $w \times h$ can be written as a vector $\mathbf{V}$ of the size $s = w \cdot h$:

$$I = \begin{pmatrix} I_{11} & \cdots & I_{w1} \\ \vdots & \ddots & \vdots \\ I_{1h} & \cdots & I_{wh} \end{pmatrix} \Rightarrow \mathbf{V} = \left(I_{11} \ldots I_{w1} \ldots I_{12} \ldots I_{w2} \ldots I_{wh} \right)^T$$

Given a set of image vectors $\mathbf{V_i}$ of n normalized images, the mean vector μ and the covariance matrix K of size $s \times s$ are computed to

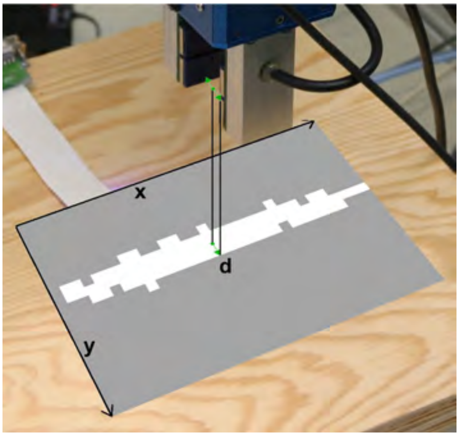


Fig. 2 Relation between the grasp description and the actual grasp with the gripper.

$$\mu = \sum_{i=1}^{n} \frac{1}{n} V_i, \quad K = \frac{1}{n} \sum_{i=1}^{n} (V_i - \mu)(V_i - \mu)^T.$$

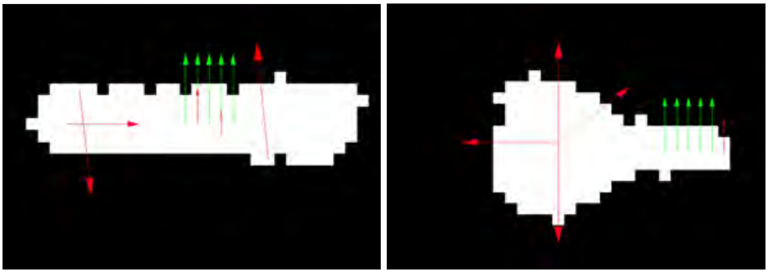
The principal component analysis of K results in s eigenvalues Λ_i and the accordant eigenvectors γ_i, which span the orthogonal eigenspace $\Gamma = (\gamma_1, \cdots, \gamma_s)$. This eigenspace describes the highest variance between the images. The $s \times s$-matrix, with each eigenvector as a column, is reduced to a $s \times d$-matrix E by taking only the first d eigenvectors with highest eigenvalues. Given any image in vector presentation $\mathbf{V}$ and the reduced eigenvector matrix E the image vector is reduced to a vector $\mathbf{k}$ of size d: $\mathbf{k} = E^T \mathbf{V}$. The vector $\mathbf{k}$ is used as the final object presentation.

3 Grasp Description

3.1 Evaluation Platform

The robot we will use as our first test bed is a 7 degrees of freedom (DOF) humanoid manipulator and is composed of PowerCubes by amtec robotics in a modular fashion. For sensing, it is equipped with tactile sensors as an artificial skin on the arm (see [7] for details) and a 6DOF force/torque sensor at the wrist. On the force-torque sensor we mounted a two-finger gripper. The latter is actuated electrically and equipped with tactile sensors. The robot control software runs under Linux and builds on the freely available Modular Controller Architecture (MCA) as a software framework [8].

The sensor modules we use for the gripper (Weiss Robotics, type DSA 9335) provide a spatial resolution of $3.8mm$ with 28 sensor cells. They are organized as 4×7 matrices. Using 4 modules (2 on each gripper finger), our gripper thus is equipped with 112 sensor cells. The tactile sensor pad we use as an imaging method for the object's silhouette (Weiss Robotics, type DSAMOD-6 [9]) provides an image



(a) Imprint of a text marker (b) Imprint of a key

Fig. 3 Representation of good and bad grasp samples. Green arrows describe positive, red arrows describe negative examples. The starting point of the arrow describes the grasp position of the gripper (**x** und **y**). The length of the arrow describes the initial opening position of the gripper **d**. The angle α corresponds to the orientation of the gripper.

resolution of 12 × 16. A more detailed description is available in [4]. Fig. 2 shows the setup of the evaluation platform with the gripper and the tactile sensor pad.

3.2 Grasp Parameters

As a two-finger gripper is a relatively simple version of an artificial hand, only a few parameters are needed to describe a grasp. The 4 required parameters are:

- The **position of the gripper**, given as the center point (x,y) of the gripper in image coordinates.
- The **orientation of the gripper** regarding the x-axis of the image coordinate system, given by the angle α.
- The **pregrasp position** of the gripper's fingers used for approaching the object, given by d. For grasping, the distance is decreased until the tactile sensor system of the grippers reports a tight grip.

A similar representation could be used for any other hand performing a precision grasp with two or three fingers. The final grasp representation is given by $\mathbf{G} = (x,y,\alpha,d)^T$.

3.3 Generation of Training Data

In this work, the user is requested to mark two points on any object silhouette. The first point represents the first two grasp parameters, the position (x,y) of the gripper. The angle of the line given by the two points represents the orientation α of the gripper. The distance between the two points provides the pregrasp position of the gripper's fingers. For visualization, a good grasp is represented by a green arrow and a bad grasp by a red arrow, as shown in Fig. 3. The intensity of the color refers to the quality of the grasp.

The grasp of an object should fulfill the following criteria:

1. The object must be grasped close to the object's center of gravity — for planar and cylindrical object close to the centroid of the silhouette.
2. The fingers should not collide with object while approaching.
3. The object should not be moved while grasping the object.

A positive grasp example does fulfill all criteria above. A negative example does not fulfil the first or the second criteria.

3.4 Registration of the Sensor Pad

To finally grasp an object based on the imprint on the tactile sensor matrix, the transformation between the image plane of the tactile sensor and the robot's coordinate system is required. The sought transformation T allows to transform each grasp point $\mathbf{P_i} = (x,y)^T$ to the coordinate system of the robot $\mathbf{R_i} = (X,Y,Z)^T$ with $T \cdot \mathbf{P_i} = \mathbf{R_i}$. The transformation matrix T is obtained by solving the following system of equations:

$$TP = R \Leftrightarrow T = RP^{+} = RP^{T}(PP^{T})^{-1}$$

with P as a set of tactile sensor points and R as a set of corresponding tool center points. At least 3 point correspondences are required. In the general case, the system of equations is overdetermined. This can be solved by computing the pseudo-inverse P^{+}, if P has maximum rank. To calculate the needed transformation, a collection of point correspondences is acquired. Therefore, a stick is pinched between the jaws of the gripper. By the means of a zero-force control, the gripper is aligned perpendicularly to the sensor pad and the robot arm is moved towards the sensor pad until the top of the stick touches the sensor pad. With the presence of a contact, the coordinates of the contact point on the sensor pad as well as the position of the tool center point are stored. This procedure is repeated several times until the required amount of point correspondences is collected. For the given setup, the mean square error of the transformation is $1.6mm$.

4 Object-Grasp Relation

The proposed procedure for learning an object-grasp relation goes as follows:

1. For each training image, adequate grasps are provided manually. Positive and negative examples must be given. Currently, a binary decision is made — a positive sample is labeled by 1 and a negative sample by -1.
2. Now, a training set of object-grasp relations $(\mathbf{O}, \mathbf{G}, Q)$ is given with $\mathbf{O}$ as the reduced and normalized representation of an object imprint, $\mathbf{G}$ as the grasp parameters and $Q \in \{-1, 1\}$ as the quality of this relation. By the means of this

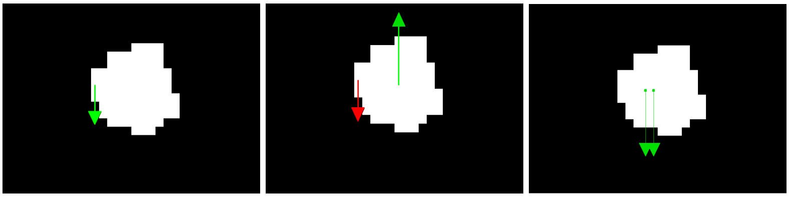
training set, a function is approximated which maps an object-grasp pair onto a quality measure. In this work, a Support Vector Regression is used to approximate the target function.

4.1 Support Vector Regression

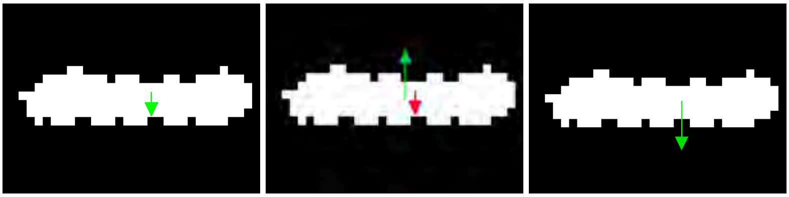
Support Vector Machines (short: SVM)[10] are a method of machine learning. One of its advances is the separation of non-linearly separable data by a linear function using the kernel-trick. The assumption is that data, being projected to higher-dimensional hyperspace, can be linearly separated. The samples closest to the hyperplane are called support-vectors. The Support Vector Regression (short: SVR) works analog to the SVM. Instead of finding a hyperplane, which separated the training data, a hyperplane is sought, which interpolates the positive training samples best. The benefits of the SVM are preserved.

4.2 Finding Grasps

Given any object-grasp combination, the previously computed SVR can be used to provide a corresponding quality. To find the best grasp for an unknown object, the grasp search has to be discretized. Therefore, grasp candidates have to be generated and finally be tested. Due to the good speed performance of the SVR, thousands of grasp-objects combinations can be easily tested. The dataset is built by generating



(a) Bad result (b) New dataset (c) Improved result



(d) Bad result (e) New Dataset (f) Improved result

Fig. 4 Procedure for assisted generation of training data: The SVR gives an suboptimal result (a), the user adds a new dataset (b) and the SVR gives an improved result (c). This is repeated for every sample ((a)-(f)) until an adequate dataset has been generated.

several angles α and pregrasp positions d for each object pixel. The step size of these parameters and therefore the resolution of the discretization can be easily adjusted.

4.3 Assisted Generation of Training Data

Obviously, the result of the SVR depends on the given set of grasp examples. To help the user with the generation of the training data, an additional step is introduced. At first, the user generates a first set of training data and a first SVR is trained. Then, for each image of the training dataset, the best grasp is sought based on the trained SVR and visualized. The user has to evaluate the given object-grasp pair. If the result is not satisfying, the assumed best grasp is taken as a dataset for a bad grasp and the user is prompted to add a new dataset. By repeating this procedure with the all training images, the user easily adds new training grasp samples.

5 Evaluation

The performance of the SVR depends on several factors. The first factor is the input dimension. The 4 grasp parameters are already the minimum size. Therefore, a variation of the input dimension is only possible by changing the number of object parameters given by the number of principal components used for the dimension reduction of the images. Another aspect is the number of used training samples and the resulting success rate.

5.1 Realization

For the evaluation, 41 training and 50 test examples were given. The samples were taken from 17 different objects — 10 objects were used for training while the remaining 7 object were used for the testing. The setup as introduced in Chapter 3.4 has been used.

The input resolution of the images is 12×16. During the normalization procedure the images are scaled by factor 2 and finally reduced to a dimension of 8 and 12 respectively. The first two experiments start with 164 training samples which were generated by the user without any assistance. Then, the user is assisted while generating further samples until a final set of 464 grasp samples in the first and 362 in the second experiment are generated. For the third experiment, the user starts with a small set of 10 training samples and finally gains 178 samples.

For the implementation of the SVR, a matlab version[11] of SVMlight was used. SVMlight[12] is implemented in C and provides fast algorithms for solving optimization problems of the SVR. Here, the RBF-Kernel (Radial Basis Function) was used. This has been used for several non-linear problems[3]. For the generation of grasp candidates (cp. Ch. 4.2) only 2 angles, 0° and 90°, were used due to the object

Table 1 Results of the evaluation: exemplary, three experiments are shown. Each experiment consist of several testruns. With each testrun of the same experiments the amount of training samples is increased. The experiments differ by the amount of eigenvalues used for the object presentation and the number of training samples.

Testrun number	Number of Eigenvalues	First amount of training samples	Final amount of training samples	Good grasps	Acceptable grasps
1	8	164	164	50 %	60 %
2	8	164	326	76 %	86 %
3	8	164	396	82 %	88 %
4	8	164	464	76 %	86 %
1	14	164	164	22 %	22 %
2	14	164	236	64 %	70 %
3	14	164	298	78 %	78 %
4	14	164	362	72 %	74 %
1	8	10	94	66 %	84 %
2	8	10	132	67 %	69 %
3	8	10	178	66 %	74 %

normalization and the symmetry of the gripper. An orientation of 0° or 180° represent the same grasps. The pregrasp position were set to 5 lengths between 2 and $10cm$.

5.2 Results

Based on the previously stated grasp criteria (cp. Ch. 3.3), the found grasp can be assigned to three categories:

1. A **good** grasp fulfills all requirements.
2. **Moved** while grasping fulfills all requirements except that the object has been moved while closing the gripper.
3. A **bad** grasp fulfills none of the requirements. It might also happen that no grasp is found for a given object.

The results of the evaluation are presented in Table 1. It shows the success rate of grasping the test objects with respect to the dimension of the object description and the amount of training samples. The good grasps build a subset of the acceptable grasps. It shows that the training data of each experiment has been increased with each new testrun. The proposed strategy provides acceptable results. Almost 90% of the object could be grasped by the means of the SVR trained in the first experiment. The additional step of increasing the training set by evaluating the SVR has been proven as a reliable method. All experiments haven shown that with every new testrun and an increasing amount of training data the success rate increases.

6 Conclusions

This work combines visual determination of grasp points with a machine learning approach. A Support Vector Regression was used to learn the relation between an object and a grasp. The learned function provides for any combination of object and grasp parameters a qualitative measure. Together with a descritization of the grasp search space, an adequate grasp can be sought for a given object. The approach was applied to a two-finger gripper. The imprint of planar objects on a tactile sensor matrix was used as an imaging method. The performance of the approach could be improved by assisting the user while providing the grasp examples. The user can add new examples which directly result from the errors of the SVR. Consequently, the amount of objects to be successfully grasped could be increased.

Although the evaluation scenario addresses a rather simple case of grasping, the results show it is possible to grasp known and unknown object based on experience and a learning approach. Future work should include a comparison of this learning approach to a grasping approach using heuristics. Furthermore, future work has to prove that this learning approach is applicable to more complex grasping scenarios.

References

1. Morales, A., Asfour, T., Azad, P., Knoop, S., Dillmann, R.: Integrated Grasp Planning and Visual Object Localization For a Humanoid Robot with Five-Fingered Hands. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS), Beijing, China (2006)
2. Hauck, A., Ruttinger, J., Sorg, M., Farber, G.: Visual determination of 3D grasping points on unknown objects with a binocular camera system. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 1999, vol. 1, pp. 272–278 (1999)
3. Pelossof, R., Miller, A., Allen, P., Jebara, T.: An SVM learning approach to robotic grasping. In: Miller, A. (ed.) Proceeding of IEEE International Conference on Robotics and Automation ICRA 2004, vol. 4, pp. 3512–3518 (2004)
4. Weiss, K., Woern, H.: The working principle of resistive tactile sensor cells. In: Proceedings of the IEEE International Conference on Mechatronics and Automation, Canada (2005)
5. Schmid, A., Gorges, N., Göger, D., Wörn, H.: Opening a Door with a Humanoid Robot Using Multi-Sensory Tactile Feedback. In: IEEE International Conference on Robotics and Automation, Pasadena, CA (2008)
6. Hu, M.-K.: Visual Pattern Recognition by Moment Invariants. IEEE Transactions on Information Theory 8(2), 179–187 (1962)
7. Kerpa, O., Osswald, D., Yigit, S., Burghart, C., Woern, H.: Arm-hand-control by tactile sensing for human robot co-operation. In: Proceedings of Humanoids 2003, Karlsruhe, Germany (2003)
8. Scholl, K.-U., Albiez, J., Gassmann, B.: MCA - An Expandable Modular Controller Architecture. In: 3rd Real-Time Linux Workshop (2001)

9. Weiß, K.: Weiss Robotics, `http://www.weiss-robotics.de` (accessed 31/05/2007)
10. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
11. Briggs, T.: MATLAB/MEX Interface to SVMlight, `http://webspace.ship.edu/thbrig/mexsvm/`
12. Joachims, T.: Making large-scale support vector machine learning practical, pp. 169–184. MIT Press, Cambridge (1999)

Efficient Parallel Random Sample Matching for Pose Estimation, Localization, and Related Problems

Daniel Kubus, René Iser, Simon Winkelbach, and Friedrich M. Wahl

Abstract. This paper introduces a parallel variant of the Random Sample Matching (RANSAM) approach leading to a very time and memory efficient enhancement of the traditional Random Sample Consensus (RANSAC). This improvement exploits the theory of the birthday attack whose mathematical background is known from cryptography. The RANSAM technique can be applied to a wide field of applications, for example, localization in mobile robotics and object pose estimation in automated assembly. Standard computers feature multi-core processors nowadays – hence a considerable speedup can be obtained by distributing selected subtasks of RANSAM among the available processors. Primarily, this paper addresses the parallelization of the RANSAM approach and the selection of the quality function that evaluates the validity of a matching – since each application scenario and sensor setup requires an individually appropriate quality measure. Apart from general performance results, experimental results regarding a special application – pose estimation with tactile sensors – are presented.

1 Introduction

In various robotics applications, matching several sets of points, also known as point clouds, is a mandatory capability. The matching problem is often defined as follows: Given two sets of points A and B, find a translation t and a rotation R resulting in a maximum overlap of these two sets of points. Typical applications can be found in mobile robotics [1, 2, 3, 4, 5, 6], computer vision [7], and medical robotics [8]. Moreover, matching algorithms can be employed to recognize objects or to estimate their poses based on tactile (and vision) information [9, 10, 11]. These features are especially relevant in flexible assembly and service robotics applications.

Daniel Kubus · René Iser · Simon Winkelbach · Friedrich M. Wahl
Institut für Robotik und Prozessinformatik, Technische Universität Carolo-Wilhelmina zu Braunschweig, Mühlenpfordtstraße 23, 38106 Braunschweig, Germany,
e-mail: `{d.kubus,r.iser,s.winkelbach,f.wahl}@tu-bs.de`

One of the most popular matching algorithms is the Iterative Closest Point (ICP) algorithm [12]. This approach minimizes the squared distance between points of two point clouds by iteratively computing a homogeneous transformation (R,t) between the sets of points until convergence. It has been proven that ICP converges to at least a local minimum. The computation of the transformation matrix requires identifying point-to-point correspondences. The generation of these point pairs requires significant computing resources. Langis *et al.* [13] proposed a parallel version of the ICP algorithm which they called pICP. They assume a system with C processors. The data set is subdivided by a parent process and distributed among C child processes which calculate the local correspondences and partial sums. The results are sent back to the parent process which generates the global cross-covariance matrix that is essential for the transformation. Langis *et al.* observed a compelling speedup compared to the original ICP technique. Nüchter [1] utilized pICP in his approach to the *Simultaneous Localization and Mapping* (SLAM) problem in 6D. The implementation of pICP was optimized by using the OpenMP framework. A SICK laser scanner is applied to generate 3D scans, which are matched incrementally by pICP. A parallelized version of the GraphSLAM method (LUM) [14] – called pLUM – is then employed for global relaxation. A maximum speedup of 1.92 and an average speedup of 1.25 could be observed on a dual core machine.

Johnson *et al.* [16] introduced spin images for efficient object recognition. In their work a spin image is assigned to each surface point of the object that is to be matched. It comprises a 2D accumulator array which conveys the geometrical relationship of adjacent surface points. This approach is capable of detecting occluded objects in cluttered scenes but a remarkable preprocessing effort is necessary for generating the spin images of the model library.

Another very popular approach to matching problems is the Random Sample Consensus (RANSAC) method proposed by Fischler *et al.* [15]. It randomly generates a set of hypotheses by matching a minimal portion of both point sets and rating each hypothesis by counting the number of points which could be matched. Tanaka and Kondo [6] localized a mobile robot by incrementally building a local map which is matched to the global map by the RANSAC method. Lowe *et al.* [4, 5] also apply RANSAC for localization by matching features extracted from a camera image to a set of features stored in a data base.

In this paper we describe a parallelized variant of the Random Sample Matching (RANSAM) approach introduced by Winkelbach *et al.* [7] which is an efficient enhancement of the RANSAC technique[15]. This approach exploits the theory of the so-called birthday attack known from cryptography [17] and determines an upper bound for the probability of a collision in a hash table. This technique has originally been developed for solving the 3D-puzzle problem [7], e.g., matching a set of fragments of a broken object. The approach has successfully been applied by Westphal [8] in medical robotics for estimating the relative pose of two bone fragments of a broken femur. In [2] a mobile robot is localized by matching the current observation to a map of the environment. In [3] a solution to the SLAM problem is proposed where a set of local maps is connected to a topological map. The RANSAM technique is part of the loop-closing detection system. Apart from matching problems, RANSAM and pRANSAM may also be applied to recognition problems since each matching result includes a quality estimate. Thus, sensor data may be matched to several objects and the obtained quality estimate may be used to decide which object has been sensed. In addition to introducing a parallel variant of RANSAM, we address the problem of finding a quality function which is suitable for the target application. For instance, localization of mobile robots requires a different measure of matching quality than pose estimation of objects with tactile sensors while the rest of the pRANSAM algorithm remains unchanged. To demonstrate the performance of pRANSAM and the influence of the contact criterion on matching performance, a concrete application scenario – pose estimation with tactile sensors – is presented.

This paper is structured as follows: In Section 2 we briefly review the original version of the RANSAM method and introduce the parallelized variant of the algorithm – called pRANSAM. In Section 3 we focus on selecting/designing an appropriate quality function, which strongly influences the performance of the approach. In Section 4 general performance results and especially pose estimation results with tactile sensors are presented. Limitations of the current pose estimation approach as well as future work are addressed in Section 5. Section 6 concludes the paper.

2 Brief Review of the RANSAM and the pRANSAM Algorithms

In the following we will give a brief summary of the original RANSAM algorithm and introduce the parallelized variant called pRANSAM.

2.1 The RANSAM Algorithm

Let $A = \{x_1,...,x_m\}$ and $B = \{y_1,...,y_n\}$ be two arbitrary sets of points given w.r.t. a world frame $\mathscr{W}$. The matching operation works as follows. First, three points $x_i,x_j,x_k \in A, i, j, k \in [1,m]$ are sampled randomly. These points form a triangle and define a 3D relation $rel(x_i,x_j,x_k)$ determined by the side lengths of the triangle. This relation allows the points to be stored in a three dimensional hash table denoted by R_A:

$$R_A[rel(x_i,x_j,x_k)] = (x_i,x_j,x_k) \tag{1}$$

Subsequently, the same process is repeated for set B: three points $y_p,y_q,y_r \in B, p,q,r \in [1,n]$ are drawn randomly and stored in a second hash table R_B: $R_B[rel(y_p,y_q,y_r)] = (y_p,y_q,y_r)$. This process is alternatingly repeated until a collision occurs, i.e.,

$$rel(x_i,x_j,x_k) = rel(y_p,y_q,y_r) \tag{2}$$

In this case a pose hypothesis to compute the relative transformation between A and B is obtained. The run-time complexity to compute the first hypothesis is $O(n)$ where n is the size of the hash table (assuming the computation of rel to be unique given a concrete point triple, only $1.2 \cdot \sqrt{n}$ point triples need to be processed until a collision occurs [17]). This process converges to $O(1)$ because the hash table is continuously filled with new relations. A pose hypothesis H can be obtained by computing the centroid of each triangle defining two transformation matrices whose position vectors correspond to the centroids. The transformation from $\mathscr{W}$ to the centroid of the triangle of A is denoted by ${}^{\mathscr{W}}T_A$. Similarly, the transformation to the triangle frame of B is denoted by ${}^{\mathscr{W}}T_B$. Now each point $x_l \in A$ is transformed using the following equation:

$$x_l{}' = {}^{\mathscr{W}}T_B \cdot {}^{\mathscr{W}}T_A{}^{-1} \cdot x_l \tag{3}$$

After this matching operation the quality of the hypothesis is checked. In principle, this can be done by checking for each point of A whether the distance to its closest point in B is below a threshold ε. Hence, an appropriate measure of quality would be

$$\Omega = \frac{\sum_{i=1}^{|A|} contact_B(x_i)}{|A|}. \tag{4}$$

The contact evaluation function $contact_B$ is discussed in detail in Section 3. If the value of Ω either exceeds a predefined threshold Ω_{th} or a maximum number of iterations is reached, the computation of hypotheses is aborted. The computation of Ω can be speeded up considerably by an extrapolation [7]. Instead of checking each point for contact, a confidence interval within which the actual value lies with a probability of 0.95 is defined by:

$$\Omega \approx \frac{\sum_{i=1}^{k} contact_B(x_i)}{k} \pm \frac{1.96}{2 \cdot \sqrt{k}} \tag{5}$$

The points considered in this extrapolation are drawn randomly. If the upper bound of the interval is lower than the quality of the current best match Ω_{best}, the check is cancelled. Note that correct hypotheses might be discarded due to the randomized structure of the extrapolation technique; but this behavior does not influence the average quality of the matching result since a large number of hypotheses is checked.

2.2 Parallel RANSAM – pRANSAM

To accelerate matching or object recognition processes, either application-specific optimizations of the RANSAM algorithm may be implemented, which can significantly speed up the algorithm as discussed below, or the RANSAM algorithm may be distributed to C processing nodes. In contrast to the original RANSAM algorithm, pRANSAM exploits the advantages of systems with multiple processing nodes. The implemented approach to concurrent RANSAM features $T = C$ threads, each thread performing point sampling, collision detection, and hypotheses checking.

Fig. 1 shows its structure. Each thread Θ_i fills one of the two relation tables, checks for collisions, and evaluates hypotheses ${}^{A}H(\Theta_i)_B = {}^{\mathscr{W}}T(\Theta_i)_B \cdot {}^{\mathscr{W}}T(\Theta_i)_A{}^{-1}$ representing transformations between the two sets (resp. point clouds).

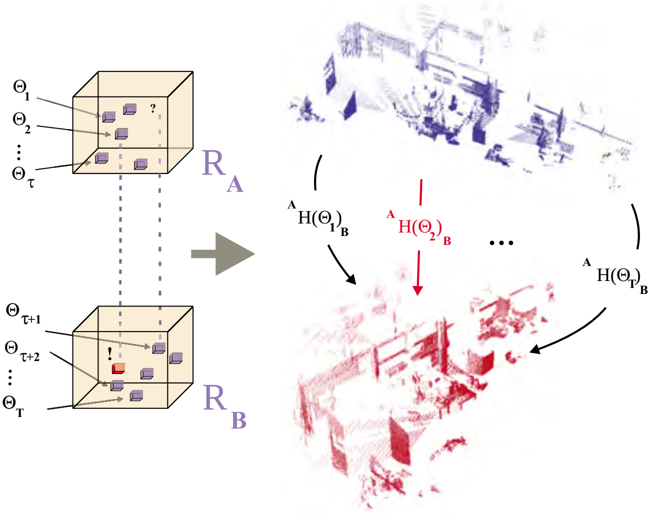


Fig. 1 Principle of pRANSAM: The left part of the figure shows the two relation tables R_A and R_B and two groups of threads entering point triples; the right part shows two 3D fragments of our robotics lab that are to be matched. In contrast to thread $\Theta_{\tau+1}$, thread Θ_2 has detected a collision after writing a point triple to R_A. The corresponding hypothesis ${}^{A}H(\Theta_2)_B$ is then evaluated by Θ_2.

After each completed hypothesis evaluation, a thread checks whether the quality $\Omega_{i,curr}$ of its current hypothesis is higher than that of the best hypothesis and overwrites Ω_{best} as well as ${}^{A}H_{B,best}$, if necessary.

For pRANSAM important characteristics concerning the optimal distribution of threads to processing nodes and the dependency of the average matching time on the distribution of the points in the sets can be derived. The central findings are:

- In case of T threads, the maximum success probability and hence the minimum average matching time is obtained by assigning $\frac{T}{2}$ threads to fill relation table R_A and $\frac{T}{2}$ threads to fill relation table R_B; regardless of the sizes of the sets or other characteristics.
- The *theoretical* average number of matchings per time Σ_{match} increases linearly with the number of cores C in the system. As a matter of fact, experimental results even show superlinear speedups.
- Both, average matching time and average matching quality strongly depend on the geometrical properties of the point sets, e.g., matching a small segment of a map to a complete map need neither take less time nor yield a more precise result after a certain computation time than matching two maps of similar sizes.

Details on the aforementioned characteristics are contained in [19].

3 Selecting an Appropriate Hypothesis Test Function

Depending on the target application, different quality measures should be applied to evaluate the quality of a pose hypothesis. The matching quality function is define as a function

$$\Omega = f({}^{A}H_{B}, A, B) \rightarrow I\!R \in [0,1] \tag{6}$$

In the original RANSAM approach, the quality of a hypothesis is assessed by the ratio of points of set A that are in contact with a point of set B to the total number of points in set A, cf. Eq. (7). To determine whether two points are in contact, their Euclidian distance is regarded. If the Euclidian distance is below a predefined threshold ε_h, the points are said to be in contact. This measure, however, may be suboptimal in various applications. For instance, it may be beneficial to consider

- a negative contribution if a point is not in contact, i.e., a penalty,
- a fuzzy threshold for the evaluation of the contact state,
- the confidence in a measured point to weight its influence on matching quality,
- a combination of these heuristics.

In several application areas real-time capability and low matching times are key requirements while accuracy is of secondary importance. Since the execution time per matching depends on the input data, the state of the random number generator, scheduling issues, etc., a time limit for a matching operation has to be specified to employ pRANSAM in soft real-time applications.

As pRANSAM provides a quality measure, each matching may be evaluated w.r.t. its plausibility. Occasional low-quality matchings may be compensated by employing probabilistic filters or context information. Generally, the average number of iterations and thus the average matching time may be reduced by modifying the contact criterion. Instead of a hard decision (cf. Eq. (7)) whether points are in contact, a simple fuzzy-like approach (cf. Eq. (8)) for contact evaluation may be employed. Let $\Delta = |y_j - x'_i|$ denote the distance between a transformed point (cf. Eq. (3)) and its closest point in set B.

$$contact_B^{hard}(x_i) = \begin{cases} 1 & \Delta < \varepsilon_h \\ 0 & \Delta \geq \varepsilon_h \end{cases} \tag{7}$$

$$contact_B^{fuzzy}(x_i) = \begin{cases} 1 & \Delta < \varepsilon_h \\ \frac{\varepsilon_s - \Delta}{\varepsilon_s - \varepsilon_h} & \varepsilon_h < \Delta < \varepsilon_s \\ 0 & \Delta \geq \varepsilon_s \end{cases} \tag{8}$$

As can be seen in the next section, the average matching time can thus be reduced while at the same time maintaining accuracy.

4 Experimental Results

This section presents general performance results of our matching approach focusing on the dependency between matching accuracy and matching time. Moreover, a concrete application scenario is regarded: Estimation of object poses using tactile sensors in the context of automated assembly. The first subsection describes the experimental setup. In Subsection 4.2 general performance results are presented while Subsection 4.3 focuses on pose estimation results obtained with tactile sensors.

4.1 Experimental Setup

A system running Windows XP SP2 equipped with an Intel Core 2 Quad Q9300 processor ($256 KBytes$ L1-Cache, $6144 KBytes$ L2-Cache) working at $2500 MHz$, an ASUS P5Q-E motherboard, and 2 GB of RAM with a clock rate of $1066 MHz$ has been employed. The code has been optimized for speed using standard compiler options of the Visual Studio 2003 tool chain.

The tactile sensing system employed in the experiments of Subsection 4.3 consists of a Schunk PG70 electric parallel jaw gripper and two Weiss Robotics tactile sensors DSA 9205 with a DSACON32-H sensor controller. The gripper offers position, velocity, and current control at a rate of approx. $1 kHz$. The tactile sensor matrix consists of 6 by 14 tactile elements and has a spatial resolution of $3.4 mm$ in both directions. The measuring range of the sensors is $250 kPa$. Fig. 2 shows the experimental setup.

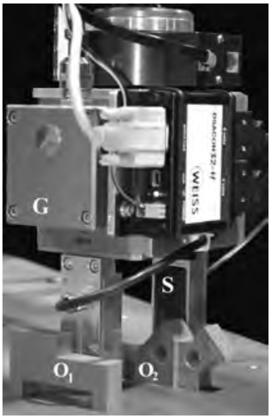


Fig. 2 Experimental setup with two sample objects O_1 and O_2 as well as the gripper G and the jaws with the tactile sensors S.

4.2 General Matching Performance

To demonstrate the general performance of the pRANSAM approach, CAD models of O_1 and O_2 as well as a map of our lab M_{lab} have been matched against themselves. The map is based on scans of a Sick LMS 200 laser range finder with a typical accuracy of $\pm 15 mm$. If the matching is perfect, ${}^{A}T_{B,best}$ should be the identity matrix. Therefore, the translational σ_t and rotational σ_α standard deviation between the

Table 1 Matching performance depending on contact thresholds.

Object	ε_h	ε_s	Avg. matching time	Percentage	σ_α	σ_t
O_1	$1.0mm$	–	$484.33ms$	100%	1.78°	$1.09mm$
O_1	$1.5mm$	–	$340.81ms$	70.4%	3.21°	$1.48mm$
O_1	$1.0mm$	$2.0mm$	$416.71ms$	86.0%	2.12°	$1.19mm$
O_1	$1.0mm$	$3.0mm$	$375.19ms$	77.5%	2.58°	$1.31mm$
O_1	$1.0mm$	$5.0mm$	$343.79ms$	71.0%	2.86°	$1.58mm$
O_2	$1.0mm$	–	$289.23ms$	100%	1.31°	$1.11mm$
O_2	$1.5mm$	–	$164.50ms$	56.9%	1.31°	$1.21mm$
O_2	$2.0mm$	–	$77.66ms$	26.9%	3.37°	$1.74mm$
O_2	$1.0mm$	$2.0mm$	$243.59ms$	84.2%	1.14°	$1.14mm$
O_2	$1.0mm$	$3.0mm$	$193.58ms$	66.9%	1.37°	$1.18mm$
O_2	$1.0mm$	$5.0mm$	$135.15ms$	46.7%	1.59°	$1.36mm$
M_{lab}	$1.0mm$	–	$42.58ms$	100%	$< 0.00001°$	$0.45dm$
M_{lab}	$1.5mm$	–	$30.52ms$	71.6%	$< 0.00001°$	$0.69dm$
M_{lab}	$4.0mm$	–	$21.86ms$	51.3%	8.05°	$1.85dm$
M_{lab}	$1.0mm$	$2.0mm$	$36.75ms$	86.3%	$< 0.00001°$	$0.54dm$
M_{lab}	$1.0mm$	$3.0mm$	$35.37ms$	83.1%	$< 0.00001°$	$0.58dm$
M_{lab}	$1.0mm$	$5.0mm$	$33.06ms$	77.6%	$< 0.00001°$	$0.61dm$

resulting ${}^{A}T_{B,best}$ and the identity matrix as well as the average matching time to reach a given quality threshold ($\Omega_{O_1} = 0.8$, $\Omega_{O_2} = 0.8$, and $\Omega_{M_{lab}} = 0.9$) serve as performance measures. Table 2 shows the matching performance for the above mentioned data sets. If no ε_s is given, the hard contact criterion (Eq. (7)) is employed; Eq. (8) is used otherwise . The 'percentage' column shows the matching time with the respective parameter set related to the maximum matching time for this object. The models of the objects consist of $N_{O_1} \approx 80.000$ and $N_{O_2} \approx 23.000$ points respectively. The lab map depicts an area of approx. $250m^2$ and contains $N_{M_{lab}} \approx 18.000$ points. To evaluate the matching performance, $\Sigma_{tr} = 500$ matching trials have been executed with each parameter set.

As can be seen, the matching time can be reduced significantly by increasing the contact thresholds ε_h and ε_s respectively while at the same time only marginally reducing the matching accuracy. However, if the contact threshold is selected too high, matching may fail completely. In the case of the lab map, selecting $\varepsilon_h = 4.0mm$ leads to occasional 180° rotations of the map, and hence to the high σ_α. Regarding O_2, selecting $\varepsilon_h = 2.0mm$ leads to an abrupt increase of σ_α. Generally, the selection of an appropriate threshold strongly depends on the characteristics of the employed data sets and hence the objects and the sensors. If occasional matching failures can be tolerated, e.g., if probabilistic filters are employed, high thresholds may nevertheless be selected.

4.3 Application Scenario – Pose Estimation with Tactile Sensors

This subsection first addresses problematic characteristics of the employed tactile sensors and then presents pose estimation results obtained with pRANSAM and the tactile sensors.

4.3.1 Peculiarities of the Tactile Sensors

The employed Weiss Robotics tactile sensors DSA 9205 show some undesired characteristics which render their application difficult for pose estimation. Especially the following characteristics deteriorate the quality of the matching results:

- Hysteresis: Even if texels are not loaded to their maximum value, the sensors show a significant hysteresis.
- Mechanical low-pass effect of the sensor cover: The sensor elements are covered with an elastic rubber layer and a second polymer layer of several millimeters. Due to these layers, a considerable low-pass characteristic of the spatial impulse response can be observed.
- Deviation between sensors: Although both sensors experience exactly the same force when an object is gripped, the integral of the texel values differs remarkably between the sensors.

To further improve the input data for pRANSAM and hence the matching performance, these issues will be addressed in future work.

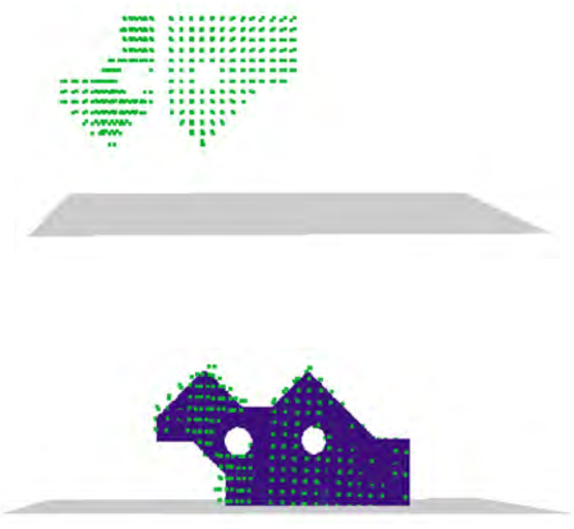


Fig. 3 Matching results for tactile sensor data. The top plot shows the point cloud obtained by sampling O_2 with the tactile sensors. The bottom plot depicts a typical matching result.

Table 2 Pose estimation results using tactile sensor data.

Object	ε_h	ε_s	Avg. matching time	Percentage	σ_α	σ_t
O_2	$1.0mm$	–	$265.93ms$	100%	0.77°	$4.06mm$
O_2	$2.0mm$	–	$29.24ms$	11.0%	2.01°	$4.86mm$
O_2	$3.0mm$	–	$5.35ms$	2.0%	35.16°	$27.01mm$
O_2	$1.0mm$	$2.0mm$	$236.16ms$	88.8%	0.76°	$4.03mm$
O_2	$1.0mm$	$5.0mm$	$89.86ms$	33.8%	0.94°	$4.04mm$
O_2	$1.0mm$	$10.0mm$	$31.55ms$	11.9%	1.11°	$5.24mm$

4.3.2 Pose Estimation Results

The tactile sensor data has been obtained by sampling the object surface at different locations. To reach a desired force per texel, a PID controller is used to control the position of the gripper jaws based on the deviation between the desired average force per element and the actual average force applied to the sensor elements considering constraints on the maximum load per element.

To evaluate the performance of the approach, ground truth data is required. The object has been placed on the table and the relative transformation to the gripper reference frame has been determined. Subsequently, the object surface has been sampled with the tactile sensors and its pose has been estimated using the pRANSAM approach. Fig. 3 shows the point cloud based on the tactile sensor data in its original orientation as well as a typical matching result. Table 2 shows performance measures of the achieved matching results. Again, the 'percentage' column shows the matching time with the respective parameter set related to the maximum matching time. The translational and rotational standard deviations have been calculated based on $\Sigma_{tr} = 500$ matching trials. Similar to Subsection 4.2, the fuzzy contact criterion reduces average matching time while at the same time maintaining σ_α and σ_t. As can be observed, a threshold of $\varepsilon_h = 3.0mm$ leads to a considerable increase of the standard deviations σ_α and σ_t. When the issues addressed in Subsection 4.3.1 have been solved, a matching performance with sub-texel accuracy is expected; the current translational standard deviation w.r.t. to the ground truth pose is approx. 120% of the texel width.

5 Limitations and Future Work

Currently, the application of the pose estimation approach is limited to (nearly) plane-parallel rigid objects since arbitrarily formed surfaces lead to pressure profile that cannot be interpreted correctly due to the ambiguity between partially loaded texels and curvature influences. Moreover, the pose of nearly symmetric parts with differences that cannot be resolved by the tactile sensors may not be estimated correctly due to the ambiguity arising by symmetry. Apart from trying to resolve the ambiguity problem whether a partial load on a texel results from the curvature of the object or a partial coverage of the texel, a differentiation between surface points and points on the edge of the object will probably improve matching performance.

Currently, subsequent matchings are merged using the relative transformation of the robot between gripping poses. Small pose variations when gripping the object, however, are currently neglected which may generally deteriorate matching accuracy. Probabilistic filters will be employed to mitigate this influence. Moreover, the object surface is sensed sequentially instead of first sensing distinctive features to reduce the number of hypotheses and thus the number of necessary grasps and the matching time.

6 Conclusion and Outlook

The presented parallel approach to random sample matching features a roughly linear to super-linear speedup compared to the conventional approach (depending on the number of processing nodes and the operating system). The computational costs of pRANSAM can be decreased by employing a fuzzy contact evaluation function instead of a hard contact criterion while at the same time maintaining matching accuracy. Regarding our application scenario, acceptable pose estimation times between $32ms$–$265ms$ depending on the envisaged matching accuracy have been obtained. Therefore, pRANSAM is an interesting approach for applications requiring a high number of matching operations per time. For example, pRANSAM facilitates real-time pose estimation or object recognition using laser range finders, tactile sensors, etc. in assembly or service robotics applications or localization tasks in mobile robotics.

Acknowledgments. We would like to thank *QNX Software Systems* for providing free software licenses, Harald Meyer-Kirk for designing and manufacturing all mechanical accessories, and Christian König for coding a tactile gripper driver for the RTOS QNX Neutrino.

References

1. Nüchter, A.: Parallelization of Scan Matching for Robotic 3D Mapping. In: Proc. of the 3rd European Conf. on Mobile Robots (ECMR 2007), pp. 198–203 (2007)
2. Iser, R., Spehr, J., Wahl, F.: Mobile Robot Localization Using the Fast Random Sample Matching Approach. In: Proc. of Robotik 2008, VDI-Berichte Nr. 2012, pp. 163–166 (2008)
3. Iser, R., Wahl, F.: Building Local Metrical and Global Topological Maps Using Efficient Scan Matching Approaches. In: Proc. of the, IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (2008)
4. Se, S., Lowe, G.D., Little, J.J.: Vision-Based Global Localisation and Mapping for Mobile Robots. IEEE Trans. on Robotics 21(3), 364–375 (2005)
5. Se, S., Lowe, G.D.: Global Localization using Distinctive Visual Features. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 226–231 (2002)

6. Tanaka, K., Kondo, E.: Towards Constant-time Robot Localization in Large Dynamic Environments. In: Proc. of IEEE Int. Conf. on Networking, Sensing and Control (ICNSC 2006), pp. 113–118 (2006)
7. Winkelbach, S., Molkenstruck, S., Wahl, F.M.: Low-Cost Laser Range Scanner and Fast Surface Registration Approach. In: Franke, K., Müller, K.-R., Nickolay, B., Schäfer, R. (eds.) DAGM 2006. LNCS, vol. 4174, pp. 718–728. Springer, Heidelberg (2006)
8. Westphal, R., Winkelbach, S., Gösling, T., Oszwald, M., Hüfner, T., Krettek, C., Wahl, F.: Robot Assisted Long Bone Fracture Reduction. Journal of Robotics Research - Special Issue: Medical Robotics. SAGE Publications (to be published in mid, 2009)
9. Haidacher, S., Hirzinger, G.: Estimating Finger Contact Location and Object Pose from Contact Measurements in 3-D Grasping. In: Proc. of the, IEEE Int. Conf. on Robotics & Automation, pp. 1805–1810 (2003)
10. Payeur, P., Pasca, C., Cretu, A.-M., Petriu, E.M.: Intelligent Haptic Sensor System for Robotic Manipulation. IEEE Trans. on Instrumentation and Measurement 54(4), 1583–1592 (2005)
11. Petchartee, S., Monkman, G.: Contact Identification using Tactile Arrays. In: Proc. of IEEE Int. Symposium on Signal Processing and Information Technology, pp. 1105–1110 (2007)
12. Besl, P.J., McKay, N.D.: A method for registration of 3d shapes. IEEE Trans. PAMI 14(2), 239–256 (1992)
13. Langis, C., Greenspan, M., Godin, G.: The parallel iterative closest point algorithm. In: Proc. of the 3rd IEEE Int. Conf. on Recent Advances in 3D Digital Imaging and Modeling (3DIM 2001), pp. 195–204 (2001)
14. Lu, F., Milios, E.: Globally Consistent Range Scan Alignment for Environment Mapping. Autonomous Robots 4(2), 333–349 (1997)
15. Fischler, O., Bolles, R.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Communications of the Association for Computing Machinery, 381–395 (1981)
16. Johnson, A.E., Hebert, M.: Using spin images for efficient object recognition in cluttered 3d scenes. IEEE Trans. Pattern Recognition Anal. Mach. Intell. 21(3), 443–449 (1999)
17. Weisstein, E.W.: Birthday Attack. From MathWorld - A Wolfram Web Resource (August 2008), `http://mathworld.wolfram.com/BirthdayAttack.html`
18. König, C.: Implementation of a Tactile Sensor System for Estimating Object Parameters, Master Thesis, Institute of Robotics and Process Control (2008)
19. Iser, R., Kubus, D., Wahl, F.M.: An Efficient Parallel Approach to Random Sample Matching (pRANSAM). In: Proceedings of the International Conference on Robotics and Automation (May 2009)

Automated Robot Assisted Fracture Reduction

Ralf Westphal, Simon Winkelbach, Thomas Gösling, Markus Oszwald,
Tobias Hüfner, Christian Krettek, and Friedrich M. Wahl

Abstract. The preferred treatment of femoral shaft fractures nowadays is the minimally invasive technique of intramedullary nailing. However, besides its advantages, this technique also has a number of drawbacks like the frequent occurrence of malaligned fracture reductions and high X-ray exposures, especially to the operating team. The aim of our research is to overcome these shortcomings by utilizing 3D imaging and surgical navigation techniques in combination with a robot assisted reduction procedure. Even though telemanipulated reductions as developed in our earlier work can achieve high accuracies for simple fracture types, we show how reductions of more complex fractures can benefit from the developed automated methods by ensuring high reduction accuracies and attaining gentle reduction trajectories. We outline a system, which automatically computes the desired target poses of the fragments from 3D imaging data and focus on the algorithms, which plan reduction paths complying with soft-tissue constraints. In addition we propose how to automatically execute these plans by means of hybrid (position and force/torque) controlled skill primitives. In a feasibility study we finally show the applicability of this approach.

1 Introduction

With approximately 37 per 100,000 persons per year, fractures of the human thigh bone, the femur, are a very frequently encountered injury. The minimally invasive surgical operation of intramedullary nailing has prevailed as a standard technique for a definite stabilizing treatment in those fractures [1, 2]. Due to its minimally invasive nature and the rigid implant, this technique has a number of benefits like a very low incidence of surgery induced infections and high union rates [1-3]. Besides its advantages, this technique also has a number of drawbacks, which are well-known in clinical practice and revealed in the open literature. Malalignments of the fracture segments, with significant rotational deviations of more than 10° or

---

Ralf Westphal · Simon Winkelbach · Friedrich M. Wahl
Institut für Robotik und Prozessinformatik,
Technische Universität Carolo-Wilhelmina zu Braunschweig,
Mühlenpfordtstraße 23, 38106 Braunschweig, Germany,
e-mail: {ralf.westphal,s.winkelbach,f.wahl}@tu-bs.de}

Thomas Gösling · Markus Oszwald · Tobias Hüfner · Christian Krettek
Unfallchirurgische Klinik, Medizinische Hochschule Hannover,
Carl-Neuberg-Straße 1, 30625 Hannover, Germany,
e-mail: {goesling.thomas,oszwald.markus,huefner.tobias,
krettek.christian}@mh-hannover.de

shortenings of the leg by more than 2cm, is the most significant one which has a high impact on functional biomechanics [2, 3]. The high radiation exposure to the patient and especially to the operating staff, with average X-ray image intensifier usage between 158 and 316 seconds [1, 4], is a second well-known problem.

By utilizing and combining modern and emerging techniques like 3D imaging for a detailed and precise fracture visualization, surgical navigation for an interactive tracking of fragment motions, and robotics for a powerful yet well-controlled fracture manipulation, the aim of our research is to develop and evaluate methods to increase the quality and reliability of such operations and at the same time achieve improvements in ergonomics, e.g. by reducing the amount of the surgeons' X-ray exposure and relieving the surgeons from the exhaustive task of manually moving the fragments, sometimes requiring more than 400 Newton [5].

1.1 Motivation for an Automated Fracture Reduction

In our previous work, we developed and extensively evaluated different types of telemanipulated fracture reduction procedures by means of a robot attached to the distal (knee-side) fragment. Starting with 2D telemanipulated reductions using CCD cameras emulating X-ray imaging [6], continuing with a 2D system based on real X-ray images evaluated in cadaver studies [7], we finally developed our current 3D telemanipulator system [8, 9]. Utilizing a joystick with haptic feedback as input device, a special methodology of human-robot interaction was developed, which allows for an intuitive and efficient telemanipulated robot control to solve the complex fracture reduction problem. This has been achieved by reducing the six degrees of freedom (DoF) reduction to an iterative process of several but simpler sequential three DoF interactions. The haptic feedback of forces acting in the operation field, which are measured by a force/torque sensor mounted at the robot's hand, turned out to be an invaluable information for the surgeons to perceive the strain on soft-tissue and therefore support a gentle reduction procedure. Evaluation test series on exposed – or dry – bones as well as on human specimens demonstrated the applicability of this approach. We could show that telemanipulated reductions based on 3D imaging data can achieve high accuracies while offering an intuitive robot interface to surgeons at the same time [8, 9].

However, for complex fracture types, without a direct connection between the two major/outer fragments, the achievable accuracies degrade. As smaller fragments of complex fractures (cp. Fig. 2. right) can not be tracked by the surgical navigation system, the surgeon has no direct intraoperative visual feedback of the surgical situation on the whole. Apparently, the accuracy degradation is therefore due to the fact that the transformation between the two major fragments, and especially the size of the fracture gap, has to be guessed by the surgeon. In order to overcome these limitations of telemanipulated methods, we propose in this contribution a (semi-)automated reduction procedure. By automatically, virtually solving the 3D puzzle for all fragments, i.e. computing the correct poses of *all* bone parts, the target transformation between the two major fragments can be determined. A subsequent path planning step leads to an automated reduction procedure performed by a robot, taking geometrical information as well as force/torque-measurements into account. As a result, this leads to an accurate and at the same time gentle reduction procedure.

1.2 Related Work

Although robot assisted fracture reduction has been described conceptually early in 1995 by Bouazza-Marouf et al. [10], until today no commercially available solutions exist for supporting any fracture reduction procedure by means of a robotic system. Only a few research groups world-wide are working in the field of robot assisted reduction of femur fractures. However, to our knowledge no algorithms about automated planning or automated robotized executions of reduction trajectories have been published so far.

Warisawa et al. from Tokyo, Japan, are developing a fracture reduction robot, which follows the principle of orthopedic traction tables (i.e., an operating table, which allows the application of a constant and adjustable pull). Mounted to the patient's foot, their robot performs – or more precisely: supports – the reduction procedure of the femur via the knee joint [11]. Seide et al. have been conducting extensive research in developing a fracture reduction robotic system based on a parallel kinematic (hexapod) [12]. Xie et al. are also working on a research project utilizing a parallel kinematic robot mounted to the patient’s foot as substitute for an orthopedic traction table and furthermore plan to take soft tissue interaction into account during the planning step of reduction paths [13].

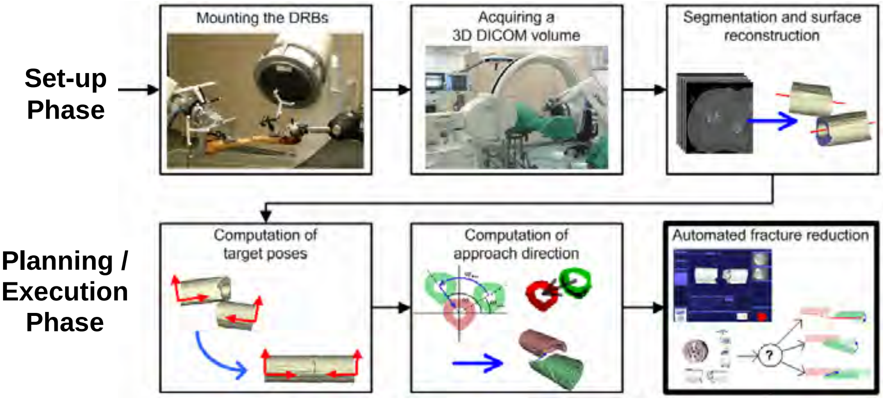

Set-up Phase
Mounting the DRBs
Acquiring a 3D DICOM volume
Segmentation and surface reconstruction
Planning / Execution Phase
Computation of target poses
Computation of approach direction
Automated fracture reduction
?

Fig. 1 Automated fracture reduction based on intraoperative 3D imaging, surgical navigation, and automated computation of target poses as well as automated execution of reduction paths.

2 Methods

The process of automated fracture reduction (cp. Fig. 1 bottom row) consists of three consecutive steps, which will be described in the following subsections. After acquiring a 3D voxel image of the fracture region, the fragments are segmented, surface models are reconstructed in 3D and the desired target poses of the fragments are computed. Next, a path to the target poses in compliance with soft tissue constraints is computed, which finally is decomposed into simple robot motion commands.

2.1 Computation of the Target Poses

The desired target poses are computed from surface models of the fragments in 3D. Therefore, the 3D voxel image is segmented and surface models are reconstructed using a marching cube algorithm. The basic principle behind computing target poses is trying to get as much matching fracture surface of two fracture segments in contact as possible. Concepts and implementations for solving such 3D puzzles are described in detail in [14-16]. These algorithms use a fast random sample matching algorithm to generate a large number of randomly distributed pose hypotheses, which are efficiently evaluated against a surface matching quality criterion. A subsequent ICP step might be used to further optimize these solutions. The results of these computations are not the absolute poses of the fracture segments, but the anatomically correct relative transformation between the two major fracture segments (between the proximal (hip-side) and distal (knee-side) fragments), denoted as ${}^{PROX}T_{DIST}^{\text{target}}$ (cp. Fig. 2).

For complex fractures with more than two fragments, the complete 3D puzzle has to be solved applying these algorithms pair-wise for matching and overall evaluation of all fragments. Even though only the outer fragments are finally repositioned and therefore moved by the robot, the complete puzzle including all small fragments in-between has to be solved virtually during the planning step in order to yield the correct target transformation between the main fragments.

A final evaluation of achievable computation accuracies in large scale test series using real human bones is still subject of future work. However, first tests using a virtual manual reposition as reference revealed satisfying pose accuracies of about 1mm and 2° in mean [16].

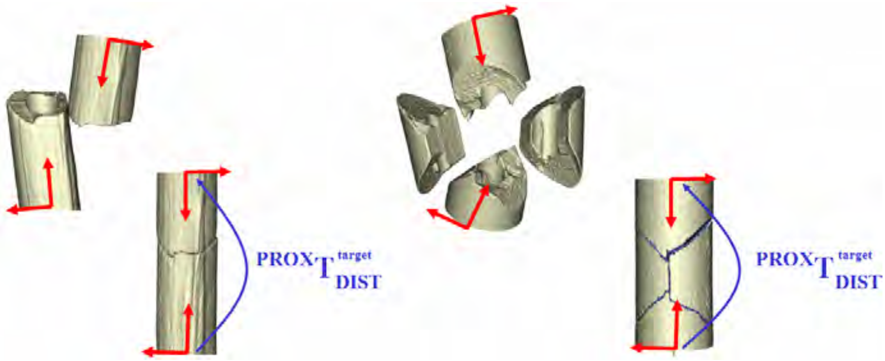


Fig. 2 From the initial poses of the fracture segments (top row), the required relative target transformation ${}^{PROX}T_{DIST}^{\text{target}}$ is computed (bottom row). For simple fractures (left) the target transformation can be obtained by matching the fracture surfaces of the two fracture parts. For complex fractures (right), the whole 3D puzzle including *all* fracture segments has to be solved.

2.2 Computation of the Reduction Paths

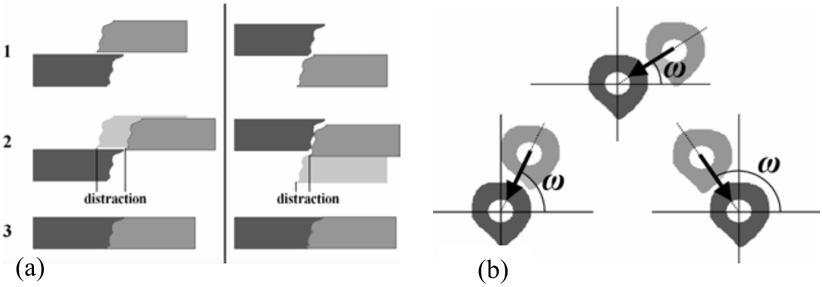
In order to adjust the target transformation, a path from the current to the target pose of the distal fragment has to be planned in compliance with anatomical constraints. In order to minimize the risk for additional soft tissue trauma, this path has to (a) minimize the required forces and (b) to avoid unnecessary movements.

A typical initial fracture situation is a lateral overlapping of the fragments. As shown in Fig. 3a, the amount of distraction required to move the fragments into the right position, depends on the direction from which the fracture region is approached. In order to minimize forces induced by the surrounding soft tissue, the amount of distraction should be minimized. In order to reduce the amount of required distraction, it might be favorable to move the distal fragment around the proximal one, as illustrated in Fig. 3b. On the other hand, the path of the distal fragment should be kept as short as possible. A suitable cost function for computing an appropriate approach direction to the fracture region, weighting distraction against additional motions, turned out to be the following:

$$\boldsymbol{C}(\omega) = \frac{\boldsymbol{a} \cdot \boldsymbol{R}(\omega) + \boldsymbol{b} \cdot \boldsymbol{D}(\omega)}{\boldsymbol{a} + \boldsymbol{b}} \rightarrow \mathbf{min} \tag{1}$$

where ω specifies the approach direction as an angle, by which the distal fragment has to be moved around the proximal one (cp. Fig. 3b). $\boldsymbol{R}(\omega)$ is a cost function for the amount of rotation, which can be expressed by a Gaussian, $\boldsymbol{D}(\omega)$ is a cost function for the required amount of maximum distraction for a given approach direction ω. $\boldsymbol{a}$ and $\boldsymbol{b}$ are weighting factors for both components.

The computation of the maximum distraction $\boldsymbol{D}(\omega)$, required to move into the fracture region from a given approach direction ω assumes that the three rotational DoFs of the two main fracture segments are aligned correctly. Practical experiences showed that this precondition can be fulfilled within a first step of the



(a) (b)

Fig. 3 (a) A simplified 2D example to illustrate the necessity of computing appropriate approach directions. In the left part an unfavorable approach direction is shown, which would result in a huge distraction. The approach direction shown in the right part is conspicuously better, minimizing the required distraction. (b) Three different approach directions, expressed as angle ω, for moving the distal fracture segment (outer silhouette) into the fracture region of the proximal fracture segment (inner silhouette).

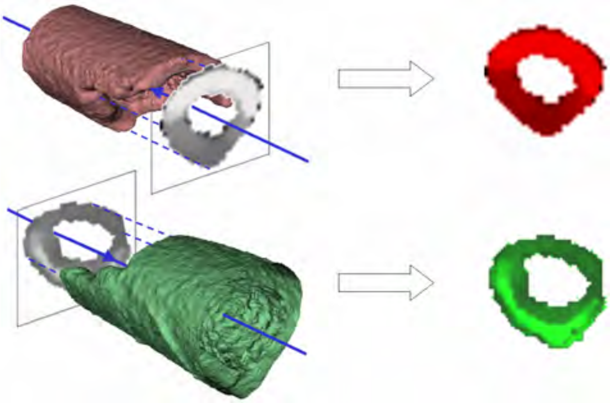


Fig. 4 Z-buffer computation for the proximal (top) and distal fragments (bottom).

reduction process. Given this assumption, the amount of required distraction can be computed utilizing a Z-buffer like approach. Two Z-buffers are generated along the axes of the two fragments as shown in Fig. 4.

Next, the Z-buffers of the two fragments are moved over each other from every possible approach direction ω and the maximum distraction expressed by the maximum added value of corresponding/overlapping Z-buffer elements is stored for that particular approach direction. This process is visualized in Fig. 5. A detailed description of this approach can be found in [8].

For complex fractures with more than two fragments, the motion planning and execution in general is considerably easier, as usually no contact between the two major fracture segments occurs during fracture reduction. Therefore the first step of the reduction path planning is to determine whether there is an overlap of the two main fragments in the initial and goal poses. If not, no further planning is required and the target motion can be performed on a straight path. However, a

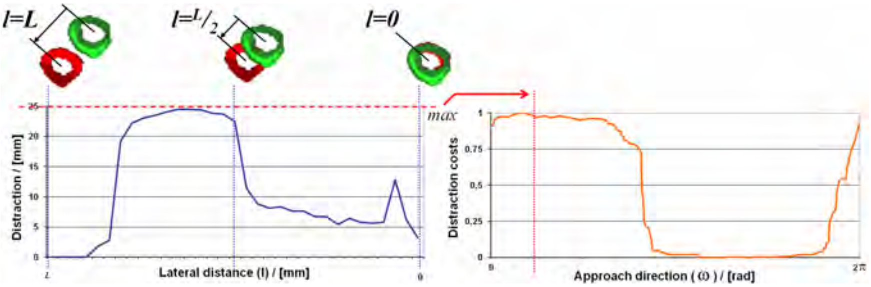


Fig. 5 For a given approach direction ω the Z-Buffer of the distal segment (upper) is moved pixel by pixel over the proximal (lower) one towards its center. Three samples are shown here along that path. For every step the amount of distraction is computed resulting in the left diagram. The maximum value of this computation is stored in the right diagram for all approach directions ω, resulting in the cost function relating approach directions to distractions.

precise biomechanical modeling of the fracture region with all of the smaller fragments and the surrounding soft-tissue might be a reasonable extension to the concepts presented in this paper and therefore are subject of further research.

2.3 Execution of the Planned Reduction Paths

Using the same set-up as for the telemanipulated reductions described above, the robot moves the knee-side fragment to which it is rigidly attached in order to obtain the computed target transformation between the two main fragments. In the first place, the path to the target pose is executed in such a way that collisions between the two major fragments are avoided as described in the previous subsection. Due to limited segmentation accuracies and unpredictable soft tissue influences during the reduction process, in some cases a fragment motion might not be executed as planned. In such a case an alternative motion command, based on the available sensor and on the 3D model data, has to be generated and executed. Furthermore, an intuitive and efficient human machine interface (HMI) for communicating the planned path to the supervising surgeon, who must always be able to modify this path if needed, has to be developed. In order to meet both constraints, the reduction path is decomposed to a sequence of small robot motion commands, called skill primitives (atomic robot motion commands providing the specification of sensor-guided and -guarded robot motions in multiple DoFs) [17]. Utilizing such skill primitives, the fracture reduction algorithm benefits from the following two advantages:

- **Intuitive supervision:** The reduction process can be supervised intuitively by a human operator, i.e. a surgeon.
- **Discrete planning space:** The state space, in which the planning algorithm is working, is discrete and limited, allowing to efficiently weight the prospects of execution success for the available motion commands.

These primitives are pushed on an execution stack in reverse order of execution (cp. Fig. 6), so that the first primitive to be executed (the correction of rotational malalignments) is executed first, followed by a motion of the distal fracture segment to the correct approach pose for the computed approach direction. On the bottom of the stack (and so executed last) the target motion with an optional additional distractive motion is placed.

Now the commands on this stack are executed one after another in the following way (cp. Fig. 6 and 7): The top most primitive is proposed to the surgeon who can accept this command or alternatively select a different skill primitive for execution. When the skill primitive was successfully executed, it is removed from the stack and the new top most one is scheduled for execution. However, the execution of a skill primitive might fail for the mentioned reasons. Whenever a skill primitive fails, it will not be removed from the stack. Instead an evasive skill primitive automatically will be proposed and pushed on the stack. This new skill primitive will be responsible for removing the cause of the previous skill primitive execution failure. The evasive skill primitive is computed in the following way:

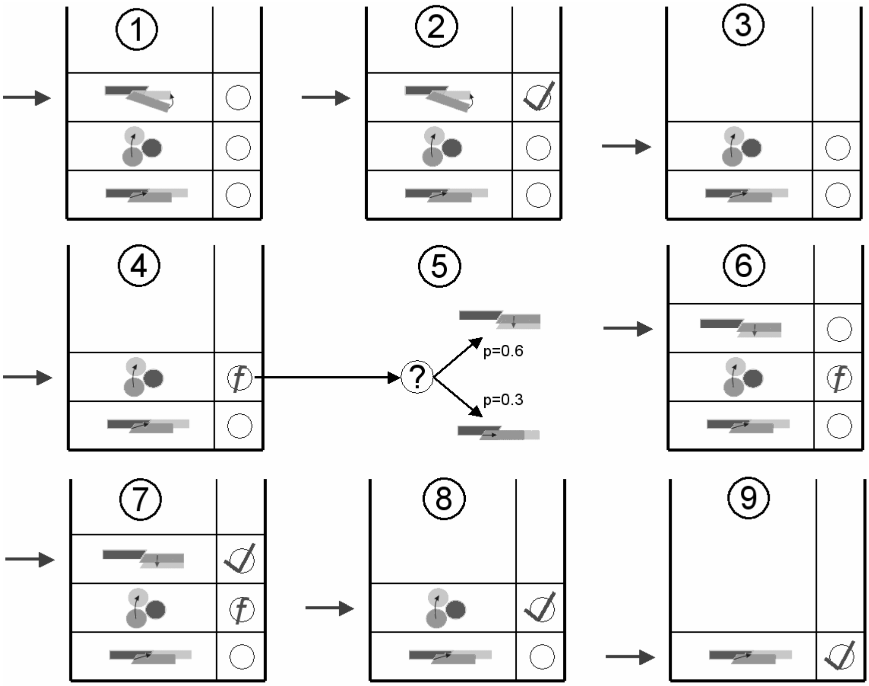


Fig. 6 The automated execution of the skill primitive execution stack. (1) The top-most element of the execution stack is executed. When the execution finishes successfully (2), the skill primitive is removed from the stack and the new top-most one is scheduled for execution (3). When a skill primitive execution fails (4), possible alternative motion commands are weighted (5) and the one with the highest vote is pushed on the stack and scheduled for execution (6). If the execution of this new evasive motion command succeeds (7), the previously failed command is executed again (8) until finally the stack is empty (9).

Let SP be the set of all available skill primitives and $s \in SP$ be one skill primitive in this set. When selecting an evasive motion skill primitive s_{evasive}, one has to search for the skill primitive with the maximum prospect of success rating according to the following equation:

$$s_{\text{evasive}} := \underset{s \in SP}{\arg\max}\left(P(s, s', f) \cdot 0.5^{m}\right) \tag{2}$$

Where s' is the skill primitive, whose execution just failed, m is the number of how many times the skill primitive s has already been marked as failed on the stack, and $P(s, s', f)$ is an empirically derived a-priori probability of the skill primitive s for a given previous skill primitive s' and a failure reason $f \in F$ of s'. In our current implementation, f can have the occurrences $F := \{\textit{interference, force\_limit}\}$.

2.3.1 Interference Detection

The skill primitive execution is continuously monitored and guided by data from a force/torque sensor mounted at the robot's hand, and an optical navigation system, tracking the motions of the two major fragments. If acting forces or torques exceed predefined thresholds or if executed robot motions don't result in the expected changes in the relative transformation between the two fragments, the skill primitive execution is stopped.

Given ${}^{PROX}T_{DIST}$ as relative transformation between the two major fragments and a commanded robot motion $\vec{c}$, we denote the resulting desired change in the relative transformation as $\vec{d}_{des}$. The actual change of this transformation as measured by the navigation system is denoted $\vec{d}_{act}$. The amount of interference i_j as complement of the effectiveness of a commanded robot motion, with j being one of the n controlled DoFs, is described as follows:

$$i_j = 1 - \frac{d_{act_j}}{d_{des_j}} \mid \forall j \in [1, n] \tag{3}$$

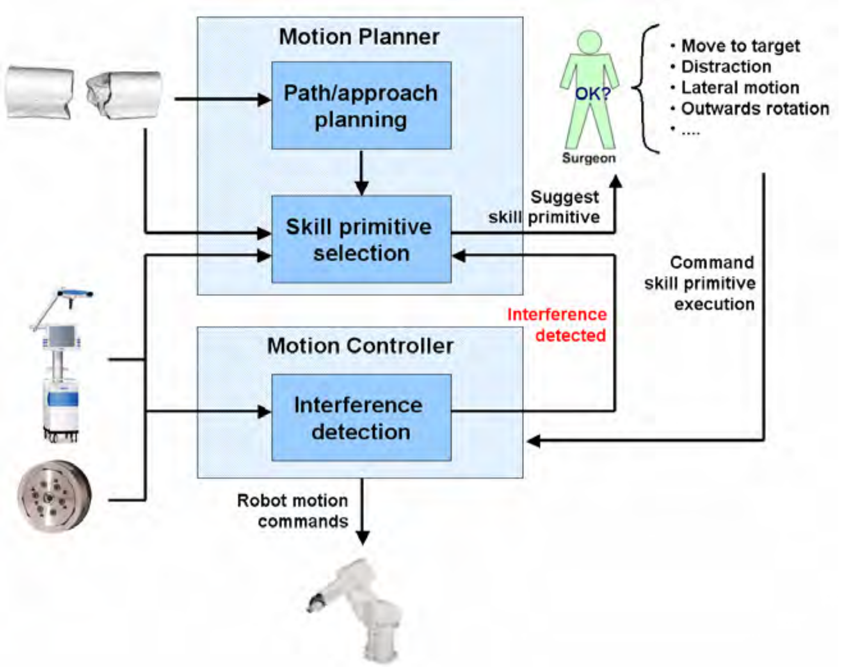

Motion Planner
Path/approach planning
Skill primitive selection
Motion Controller
Interference detection
OK?
Surgeon
• Move to target
• Distraction
• Lateral motion
• Outwards rotation
• ....
Suggest skill primitive
Command skill primitive execution
Interference detected
Robot motion commands

Fig. 7 Architecture of the automated fracture reduction planning and control system. The module "Motion Planner" computes the approach direction and appropriate skill primitives, which are suggested to the surgeon. The "Motion Controller" module generates the robot motion commands and continuously monitors the execution, stopping the execution when system parameters are about to be exceeded or situations are detected, which are likely to avert a successful execution.

Because of complex soft tissue interactions and noisy measurements from the navigation system, usage of equation 3 directly results in unstable/unreliable interference predictions. Therefore a model selection algorithm based on two independent Kalman filters is proposed here. Each Kalman filter describes the linear motion model of the relative transformation change resulting from a given robot motion command. The first Kalman filter $\boldsymbol{K}_1$ represents successful motions, which are almost unaffected by soft-tissue interactions; the parameters for the motion model have been obtained from cadaver experiments, while parameters for the error matrices are obtained from measurements from the navigation system. The second filter $\boldsymbol{K}_2$, representing failed motion commands, is parameterized very similar to the first one except for the motion model, where the influence matrix of the control parameters is set to zero, therefore describing motions which are almost completely prevented due to soft-tissue influences. During reduction, both Kalman filters are executed in parallel. Interferences are detected, when the error values of $\boldsymbol{K}_2$ are smaller than those of $\boldsymbol{K}_1$ in several (for example ten) consecutive control cycles. More details on the parameterization of the filters can be found in [8].

3 Experiments and Results

In order to evaluate the automated reduction procedure, we performed reductions on formalin embalmed human specimens with intact soft tissues and measured the achieved accuracies and execution times. To measure achievable accuracies, we first recorded a reference transformation of the intact bone geometry with our optical navigation/tracking system by means of dynamic reference bases (DRBs) attached to both ends of the unbroken bone. Subsequently the bone was broken in the mid-shaft and the reduction was performed. A more detailed description of this set-up can be found in [8, 9]. So far, the automated reduction was tested on one fractured bone in one human specimen. This fracture was reduced five times, starting from different initial poses applying the algorithm presented in this paper. In order to focus this evaluation on the performance of the robotized target motion and path planning, the unbroken reference transformation and not an automatically computed target pose was used to guide the reduction procedure. All five fracture reductions during the experiments were finished successfully without any intervention by a human supervisor. The target pose was reached with a very high mean accuracy of about 1.19mm and 0.16°. The reduction times were also very satisfactory with an average of 4:13 minutes. Based on the computed approach direction, all fracture reductions were achieved along an almost ideal path without any or with only few evasive motions.

4 Conclusion and Outlook

It was shown that an automated fracture reduction is possible when using the techniques presented in this paper. The performance of the automated reduction is very satisfactory with respect to the achievable reduction accuracy and the procedure execution time. Compared to clinically achievable results, the performance of our automated reduction procedure is potentially a considerable improvement. From the first tests presented here, which have the character of a feasibility study, one can conclude that automated reduction achieves conspicuously higher reduction accuracies when compared to telemanipulated fracture reductions. And most remarkably, the variance in accuracy and operation time is much smaller when compared to the telemanipulated approach. Therefore, the automated fracture reduction has the potential to become a reproducible surgical procedure leading to a reliable and high reduction quality.

The next steps in further developing and evaluating our automated reduction procedure is to perform extensive test series on fresh – i.e. un-embalmed – human cadavers, which, in contrast to embalmed specimens, have the advantage of having a soft tissue situation very close to that of living and perfused tissue. This allows us to finally evaluate robot assisted, and especially automated reduction procedures in a set-up very close to real surgeries.

Acknowledgments. This project was funded by the German Research Foundation (DFG: Deutsche Forschungsgemeinschaft) under the grants WA 848/10-1/2 and KR 2161/1-1/2 "Roboterunterstütze Femurmarknagelung". We would like to thank BrainLAB (Munich, Germany) for supporting us with an interface to their navigation system. We thank the Department of Anatomy of the Hannover Medical School for making specimens available for our experiments.

References

1. Kempf, I., Grosse, A., Beck, G.: Closed locked intramedullary nailing. Its application to comminuted fractures of the femur. J. Bone Joint Surg. Am. 67, 709–720 (1985)
2. Winquist, R., Hansen Jr., S., Clawson, D.: Closed intramedullary nailing of femoral fractures. A report of five hundred and twenty cases. J. Bone Joint Surg. Am. 66, 529–539 (1984)
3. Wolinsky, P., McCarty, E., Shyr, Y., Johnson, K.: Reamed intramedullary nailing of the femur: 551 cases. J. Trauma 46, 392–399 (1999)
4. Sugarman, I., Adam, I., Bunker, T.: Radiation dosage during AO locking femoral nailing. Injury 19, 336–338 (1988)
5. Gösling, T., Westphal, R., Faulstich, J., Sommer, K., Wahl, F.M., Krettek, C., Hüfner, T.: Forces and torques during fracture reduction: Intraoperative measurements in the femur. J. Orthop. Res. 24(3), 333–338 (2006)
6. Gösling, T., Westphal, R., Hüfner, T., Faulstich, J., Kfuri, M., Wahl, F.M., Krettek, C.: Robot-assisted fracture reduction: A preliminary study in the femur shaft. Med. Biol. Eng. Comput. 43, 115–120 (2005)
7. Westphal, R., Winkelbach, S., Gösling, T., Hüfner, T., Faulstich, J., Martin, P., Krettek, C., Wahl, F.M.: A Surgical Telemanipulator for Femur Shaft Fracture Reduction. Int. J. of Medical Robotics and Computer Assisted Surgery 2(3), 238–250 (2006)
8. Westphal, R.: Fortschritte in der Robotik: Sensor-based Surgical Robotics: Contributions to Robot Assisted Fracture Reduction. Shaker-Verlag, Dissertation (2007)

9. Westphal, R., Winkelbach, S., Gösling, T., Oszwald, M., Hüfner, T., Krettek, C., Wahl, F.M.: Robot Assisted Long Bone Fracture Reduction. Int. J. of Robotics Research – Special Issue: Medical Robotics (to be published, 2009)
10. Bouazza-Marouf, K., Browbank, I., Hewit, J.: Robotic-assisted internal fixation of femoral fractures. In: Proc. Inst. Mech. Eng., pp. 51–58 (1995)
11. Maeda, Y., Sugano, N., Saito, M., Yonenobu, K., Nakajima, Y., Warisawa, S., Mitsuishi, M.: Case reports of robot-assisted intertrochanteric fracture reduction. In: Proceedings of the 21the International Congress and Exhibition, Berlin, Germany, June 27-30 (2007); International Journal of Computer Assisted Radiology and Surgery, June 2007, vol. 2(suppl. 1), pp. 258–259 (2007)
12. Seide, K., Faschingbauer, M., Wenzl, M., Weinrich, N., Juergens, C.: A hexapod robot external fixator for computer assisted fracture reduction and deformity correction. Int. J. Medical Robotics and Computer Assisted Surgery 1(1), 64–69 (2004)
13. Graham, A.E., Xie, S.Q., Aw, K.C., Xu, W.L., Mukherjee, S.: Design of a Parallel Long Bone Fracture Reduction Robot with Planning Treatment Tool. In: Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1255–1260 (2006)
14. Winkelbach, S., Westphal, R., Gösling, T.: Pose Estimation of Cylindrical Fragments for Semi-automatic Bone Fracture Reduction. In: Michaelis, B., Krell, G. (eds.) DAGM 2003. LNCS, vol. 2781, pp. 566–573. Springer, Heidelberg (2003)
15. Winkelbach, S., Rilk, M., Schönfelder, C., Wahl, F.M.: Fast Random Sample Matching of 3d Fragments. In: Rasmussen, C., Bülthoff, H., Schölkopf, B., Giese, M. (eds.) DAGM 2004. LNCS, vol. 3175, pp. 129–136. Springer, Heidelberg (2004)
16. Winkelbach, S.: Das 3d-Puzzle-Problem - Effiziente Methoden zum paarweisen Zusammensetzen von dreidimensionalen Fragmenten. Fortschritte in der Robotik, Band 10,, Shaker-Verlag, Dissertation (2006) ISBN 3-8322-5567-2
17. Finkemeier, B., Kröger, T., Wahl, F.M.: Executing assembly tasks specified by manipulation primitive nets. Advanced Robotics 19(5), 591–611 (2005)

Path Planning for Robot-Guided Endoscopes in Deformable Environments

Markus Rilk, Friedrich M. Wahl, Klaus W.G. Eichhorn, Ingo Wagner,
and Friedrich Bootz

Abstract. In this paper we present novel concepts that allow to plan time optimal paths under velocity constraints for an endoscopic robot acting in deformable environments. These environments are described by means of a simplified biomechanical model stored as a 3d volume. It permits the estimation of forces and torques caused by soft body deformations. Moreover, the parameters of the model are dynamically updated when structures are removed during an endoscopic surgery. All objectives of the path planning task are mapped to velocity constraints. In order to find an optimal path within the 5d configuration space of the endoscope, we propose a pivot point constrained global path planning in 3d employing fast marching and relax the pivot constraint afterwards for a 5d local optimization.

1 Introduction

In recent years, *Functional Endoscopic Sinus Surgery* (FESS) has been established as one of the most important standard techniques in the field of ear, nose and throat medicine. However, the major disadvantage of this technique is that the surgeon has to hold the endoscope himself. This results in unsteady endoscopic images during long surgeries caused by tiredness of the surgeon or in frequent instrument changes, e.g. between a knife to cut away tissue and a sucker to remove this tissue and blood. Since this is also true for other endoscopic surgeries like laparoscopy, robotic

---

Markus Rilk · Friedrich M. Wahl
Institut für Robotik und Prozessinformatik, Technische Universität Carolo-Wilhelmina zu Braunschweig, Mühlenpfordtstraße 23, 38106 Braunschweig, Germany,
e-mail: `{m.rilk,f.wahl}@tu-bs.de`

Klaus W.G. Eichhorn · Ingo Wagner · Friedrich Bootz
Klinik und Poliklinik für Hals-Nasen-Ohrenheilkunde/Chirurgie, Universitätsklinikum Bonn, Sigmund-Freud-Straße 25, 53105 Bonn, Germany,
e-mail: `{klaus.eichhorn,ingo.wagner,`
`friedrich.bootz}@ukb.uni-bonn.de`

systems are in development able to guide the endoscope during an endoscopic surgery (some of these developments are listed in [1, 14, 17]). Our ultimate aim in *Robot Assisted FESS* (RAFESS) is that the robot guides the endoscope during a surgery as autonomous as possible. One of the main challenges is the close proximity of critical regions like brain, eyes and carotid artery to the workspace of the robot, which, of course, is not allowed to collide with these structures. Therefore, the motions of the robot should be restricted as much as possible without loosing the flexibility needed to satisfy the demands of the surgeon. Part of these requirements are that (1) the tip of the surgical instrument is always in the center of the endoscopic view, (2) the surgeon has enough free space to operate his instruments, (3) the motions of the robot harmonize with the motions of the surgeon, (4) the robot can automatically clean the camera lens, and (5) the surgeon can direct the robot to specified locations (e.g. on the basis of CT data).

Especially the last two points require a sophisticated path planning algorithm. In contrast to most industrial applications it has to deal with soft structures. And due to the limited space in the (para-)nasal cavities, the robot has to deform some of these soft structures in order to lead the endoscope to the desired region of interest. The amount of these deformations should be restricted in such a way that the risk of an injury is minimal for the patient. Furthermore, as some of the hard and soft tissue structures are removed during a surgery, dynamic data structures are required. Despite the advances in soft body simulation [12, 18] and handling by flexible robots [3, 13], to the best of our knowledge, up to now no general path planning algorithm is available working under these conditions at interactive response times.

In this paper we present an important step towards path planning in deformable environments. We show how deformations can be modeled by a simplified biomechanical model and how they can be integrated into the planning process. Furthermore, we derive constraints that can be intuitively defined by the user.

The paper is organized as follows: In chapter 2 we show how to model the environment for motions of the endoscope within (para-)nasal cavities. How the robot is allowed to move in RAFESS is described in chapter 3. A path planning algorithm is presented in chapter 4. Some first experimental results are given in chapter 5. Chapter 6 summarizes the contributions of this paper and indicates further research.

2 Model of the Mixed Hard / Soft Tissue Workspace

In RAFESS the robot moves in a complex and dynamic environment that includes the inner structures of the patient's head, the complete body of the patient, the acting surgeon, a moving medical assistant, different hardware devices and so on. Especially w.r.t. human-machine interaction it is very important to model the environment as secure as possible without limiting the robot motions too much. In our work we distinguish between the interior and exterior environments. The interior environment desribes the inner structures of the patient's head whereas the exterior environment includes all other structures mentioned before. In this paper we concentrate on modelling the interior environment.

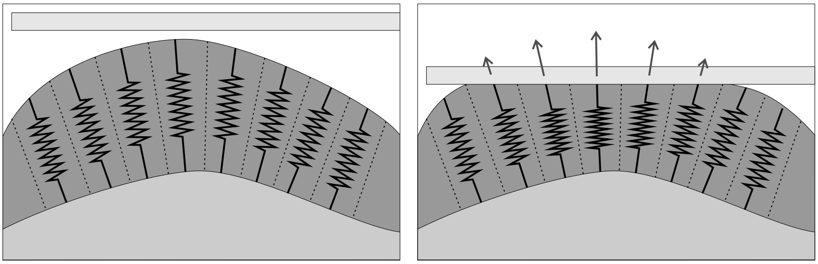
The (para-)nasal structures are very complex and the properties of their materials differ strongly from most materials involved in industrial robotics. In medical robotics we have to deal not only with hard materials like bones, but also with soft tissues that have to be deformed by the robot because of the limited space available to navigate an endoscope. Hard structures and other critical regions (e.g. brain, eyes) can be treated as prohibited zones for the endoscope held by the robot. Contacts with soft structures can not be avoided during navigation. But the amount of deformation should be limited so that the risk of injuries is minimal for the patient. Furthermore, some of the hard and soft structures are removed during a surgery. The model of the environment has to be flexible enough to allow for these dynamic changes.

Prior to FESS a computer tomography (CT) scan is taken from the patient's head. These CT data can be used by a preprocessing tool to generate an initial 3d model of the interior environment. Currently, we use a semi-automatic segmentation as shown in [19]. The initial 3d model is subsequently converted into a (planned) biomechanical FEM model. This allows for offline simulations of soft body deformations caused by endoscope motions. As a result, we get simulation data (e.g. describing maximum allowed deformations) that can be used to initialize the parameters of a simplified biomechanical model. In contrast to the FEM model, the simplified model has to be suitable for online path planning.

In the following, we propose a simplified biomechanical model that does not need any numerical simulation steps. Instead, on the basis of our model it is possible to directly compute the distances to obstacles and the forces acting on the endoscope for any configuration In Fig. 1 the idea is roughly illustrated. We assume that soft tissues can be divided into several independent rods of isotropic linear elastic material and that deformations along the axis of the rod obey Hook's law given in the following form:

$$F = EA\varepsilon \tag{1}$$

F and A denote the resulting force and the cross-sectional area of the rod respectively. The material parameter E is the modulus of elasticity and is assumed to be



(a) (b)

Fig. 1 Soft structures are divided into rods of isotropic linear elastic material. (a) Initial state of the simplified biomechanical model. (b) Restoring forces are estimated with the help of springs.

known. The dimensionless strain ε is the ratio between the length l of the deformation and the initial length l_0 of the rod.

Our simplified biomechanical model is represented as a 3d volume (typical size is 256^3) with the following properties defined for each voxel:

1. $\tau \in \{HARD, SOFT, FREE\}$: Type of the voxel.
2. $\rho \in \{false, true\}$: Is the voxel marked as removable?
3. k_{SOFT}: Index of the nearest SOFT voxel.
4. k_{HARD}: Index of the nearest HARD voxel.
5. $\varepsilon \in [0,1]$: Describes the amount of deformation (strain).
6. E: Modulus of elasticity as a measure of the stiffness $[N/mm^2]$.

A voxel is classified as either (1) *HARD* structure (also including prohibited zones), (2) *SOFT* structure, or (3) *FREE* space (air). Moreover, non-critical voxels of type *HARD* and *SOFT* can also be marked as removable. Instead of storing the Euclidean distance d_{SOFT} to the nearest *SOFT* voxel and d_{HARD} to the nearest *HARD* voxel directly, we use the indices k_{SOFT} and k_{HARD} for an efficient distance calculation as suggested by several authors (see [8] for an overview). In order to estimate the strain in an efficient and robust way, we use an isotropic diffusion process within the adjacent *SOFT* voxels:

$$\varepsilon_{i,t+1} = \frac{1}{7}\left(\varepsilon_{i,t} + \sum_{j \in N(i)} \varepsilon_{j,t}\right) \tag{2}$$

If a voxel j in the 6-neighbourhood is of type *HARD*, then $\varepsilon_j = 0$; if it is of type *FREE*, then $\varepsilon_j = 1$. Initially, $\varepsilon_{i,0}$ is set to 0.5. The modulus E does not have to be defined for each voxel individually, but by doing it this way we gain additional freedom and we can directly look-up the value.

Next, we will explain how to estimate a restoring force vector **F** caused by a small sphere probe located at position $\mathbf{c} = (x,y,z)^T$ with radius r. Let $\mathbf{n}$ be the voxel nearest to the center $\mathbf{c}$. Thus, the force vector can be estimated by

$$\mathbf{F} = -\varepsilon(\mathbf{n})E(\mathbf{n})r^2\pi \frac{\nabla\varepsilon(\mathbf{n})}{||\nabla\varepsilon(\mathbf{n})||} \tag{3}$$

For the magnitude of **F** Eqn.1 is used. The direction of **F** is calculated by the normalized spatial derivative of the strain. In order to estimate the resulting force of a more complex object, we decompose the complex object into several overlapping spheres, calculate the forces for each sphere individually and sum them up afterwards (also taking into account the overlapping areas). For some objects this might be not the optimal method, but considering the simple shape of an endoscope and how the endoscope interacts with the soft structures, we are confident that this is a good approximation. Additionally, we estimate the torque vector τ from the calculated forces.

It was important to us to keep the computational complexity and memory requirements for a single voxel low, because we are planning to implement the proposed 3d model on a GPU. Given our current CPU implementation, we anticipate that one iteration step over all voxels will take a few milliseconds on a fast GPU for a 256^3 grid. In each iteration step we update the distances (indices) and the strain based on the information in the 6-neighbourhood.

3 Configuration Space of the Endoscope

In this chapter we define an appropriate configuration space for RAFESS and derive kinematic and velocity constraints. Some of these constraints are based on the assumption that while a path planning request is executed no structures are removed. In the following, we are only interested in the part of the robot relevant for path planning within the (para-)nasal cavities. This is the front of the endoscope that can be precisely modeled as a cylindrical shaft.

3.1 Configuration Parameters

The pose of the endoscope can be described by a 6d vector (three rotations and three translations). Since rotations around the symmetry axis of the endoscope do not change anything w.r.t. collision detection, we can reduce the 6d configuration space to 5d (three translations and two rotations). In endoscopic surgeries the endoscope and the surgical instruments are inserted through small openings. Let us preliminary assume these openings to be fixed and so small that the endoscope can only be moved in such a way that the symmetry axis of the endoscope always passes through the center of the opening. In this case the 5d configuration space is further reduced to 3d (one translation and two rotations). The center of the opening is usually called *pivot point*, entry point or invariant point; we will use the first term in the following. In RAFESS we have to consider a *pivot region* not only a pivot point. This motivates the following definition of the configuration of the endoscope also shown in Fig. 2:

$$\mathbf{q} = (l,\ a,\ b,\ \alpha,\ \beta) \tag{4}$$

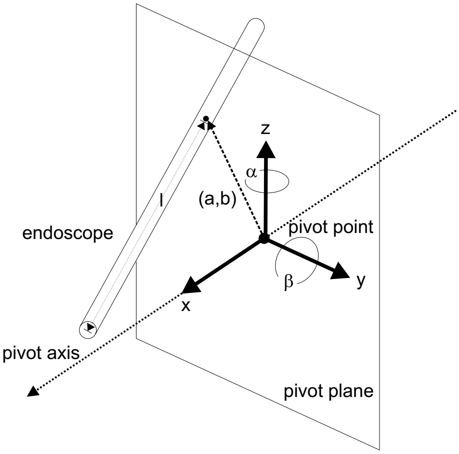
This definition has the advantage that the user can constrain the parameters in an intuitive way. For example setting $(a,\ b)$ to $(0,\ 0)$ results in the 3d configuration space mentioned above.

3.2 Kinematic Constraints

We have tracked the endoscope poses in real FESS surgeries. Based on this, we could derive the following kinematic constraints:

$$l < 80mm,\ |a| < 8mm,\ |b| < 4mm,\ |\alpha| < 45^\circ,\ |\beta| < 30^\circ, \tag{5}$$

Fig. 2 The parameters of **q** are defined as follows: *l* is the distance between the tip and the intersection point of the endoscope axis and the pivot plane, $(a,\ b)$ is a displacement vector within the pivot plane, α is the rotation angle around the *z*-axis (yaw), β is the rotation angle around the *y*-axis (pitch).



Environmental constraints are given by checking the configurations for possible collisions. For this purpose we need a geometrical model of the endoscope. As mentioned before, the front of an endoscope is exactly described by a cylindrical shaft. Given some volume data with distance values to the obstacles, it is straightforward to sample the cylindrical shaft along the symmetry axis of the endoscope with spheres. Depending on the distance δ between the sphere origins, we loose some accuracy here (see also Fig. 3), but we have the great advantage that a single distance query runs in $O(m)$, where m is the number of spheres. For a configuration $\mathbf{q} = (l, a, b, \alpha, \beta)$ each sphere $\mathbf{s}_i$ is described by the equation

$$\mathbf{s}_i = \begin{Bmatrix} x \\ y \\ z \\ r \end{Bmatrix} = \begin{Bmatrix} cos(\alpha)\, cos(\beta)\, \lambda \\ sin(\alpha)\, cos(\beta)\, \lambda + a \\ -sin(\beta)\, \lambda + b \\ \sqrt{(0.5\,\delta)^2 + (0.5\,D)^2} \end{Bmatrix} \tag{6}$$

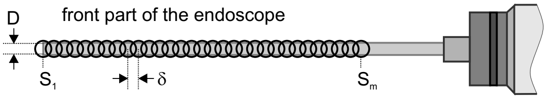
with

$$\lambda = l - (i-1)\,\delta \tag{7}$$

The first three parameters $(x,y,z)^T$ of a sphere define its origin in workspace. The last parameter r is its radius and is constant for our application. Based on this we can define the distance function $d(\mathbf{s})$ in the following way:

$$d(\mathbf{s}) = d(x,y,z,r) = max(d(x,y,z) - r,\, 0) \tag{8}$$

Fig. 3 Front part of the endoscope sampled with spheres.



Before we define the conditions for a configuration to be free, we have to consider some important points in FESS. We do not want to come into contact with any hard structures. Moreover, the tip of the endoscope (sphere $\mathbf{s}_1$) is not allowed to get in touch with soft structures, too, because the lens might get covered with blood resulting in a loss of image quality. Thus:

$$\mathbf{q}\ is\ free \Leftrightarrow d_{SOFT}(\mathbf{s}_1) > 0 \wedge d_{HARD}(\mathbf{s}_i) > 0\ \ ,\forall i \in [1,m] \tag{9}$$

3.3 Velocity Constraints

So far we could plan a shortest path between given start and goal configurations using classical path planning approaches (see e.g. [9]). But this is not desired in RAFESS, because of sensor inaccuracies, dynamic constraints of the robot and limited allowed soft body deformations. A straightforward way would be to include all objectives in one common cost function and then find a path with minimal costs. The disadvantage of a general cost function is that sometimes parameters are hard to balance, because an intuitive formulation is missing. Therefore, we map the objectives discussed above to velocity constraints for a given configuration and then plan a time optimal path under these constraints. Two constraints are defined in the following: First, a *strictly kinodynamic constraint* [2], accounting for unavoidable collisions due to limited dynamics of the robot, and second, a force/torque constraint, which limits the maximum allowed deformations.

Since the maximum possible de- and acceleration of the robot is limited by $\ddot{\mathbf{q}}_{max}$, it needs some time and space before coming to a halt. Using Newton's laws of motion we estimate the covered distance in Euclidean space for every sphere $\mathbf{s}_i$:

$$d(\mathbf{q},\mathbf{s}_i) = \sqrt{\left(\frac{\dot{q}_l^2}{2\ddot{q}_l}\right)^2 + \left(\frac{\dot{q}_a^2}{2\ddot{q}_a}\right)^2 + \left(\frac{\dot{q}_b^2}{2\ddot{q}_b}\right)^2 + \left(\lambda_i\frac{\dot{q}_\alpha^2}{2\ddot{q}_\alpha}\right)^2 + \left(\lambda_i\frac{\dot{q}_\beta^2}{2\ddot{q}_\beta}\right)^2} \tag{10}$$

This is a conservative approximation of the dependencies exactly described by the Jacobian matrix. λ_i is the same value as defined in Eqn. 7 and is used to calculate the arc length. Assuming $\dot{\mathbf{q}} = \phi\,\dot{\mathbf{q}}_{max}$ and $\ddot{\mathbf{q}} = \ddot{\mathbf{q}}_{max}$, we calculate the minimum velocity scale factor $\phi \in [0,1]$ so that the estimated distances are smaller than the distances to the obstacles for all spheres:

$$\phi = \sqrt{min\left(1,\ \frac{min(d_{HARD}(\mathbf{s}_1), d_{SOFT}(\mathbf{s}_1))}{d(\mathbf{q},\mathbf{s}_1)},\ \frac{d_{HARD}(\mathbf{s}_2)}{d(\mathbf{q},\mathbf{s}_2)},\ \ldots,\ \frac{d_{HARD}(\mathbf{s}_m)}{d(\mathbf{q},\mathbf{s}_m)}\right)} \tag{11}$$

In order to limit the forces/torques and thus the amount of deformation, we define the following velocity scale factor:

$$\omega = min\left(\frac{F_{max} - min(||\mathbf{F}||, F_{max})}{F_{max}},\ \frac{\tau_{max} - min(||\tau||, \tau_{max})}{\tau_{max}}\right) \tag{12}$$

If forces and torques are zero, $\omega = 1$ indicates that no reduction of $\ddot{\mathbf{q}}_{max}$ is needed. On the other hand, a value of $\omega = 0$ excludes configurations due to forces and torques above the given limits F_{max} and τ_{max}. These limits can be set globally or locally, just as needed.

Both scale factors are combined within the resulting velocity bounds vector:

$$\mathbf{v} = (v_l, v_a, v_b, v_\alpha, v_\beta)^T = min(\phi, \omega)\,\dot{\mathbf{q}}_{max} \tag{13}$$

4 Velocity Constrained Path Planning

Given velocity constraints, we want to find a path $\gamma(s)$ that minimizes the cumulative travel cost from a start configuration $\mathbf{x_0}$ to a goal configuration $\mathbf{x_1}$. The minimum cumulative cost at $\mathbf{x}$ is defined as

$$T(\mathbf{x}) = \min_{\gamma} \int_{\mathbf{x_0}}^{\mathbf{x}} F(\gamma(s))\,ds \tag{14}$$

In our case $T(\mathbf{x})$ is the minimum time needed to reach $\mathbf{x}$. We use the definition of [5] to formulate the cost function F with a metric tensor M in the following way:

$$F(\gamma(s), \gamma'(s)) = \sqrt{\gamma'(s)^T M(\gamma(s))\,\gamma'(s)} \tag{15}$$

Given the velocity bounds vector $\mathbf{v}$, we can concretize M as the inverse of a diagonal matrix with diagonal entries $(v_r^2, v_a^2, v_b^2, v_\alpha^2, v_\beta^2)$, which gives us in contrast to [4] an anisotropic cost function F.

Global path planning in 5d is a computational expensive task. In these cases probabilistic methods have been successfully applied. The RRT framework (e.g. [7]) allows for fast single path queries and also enables an elegant integration of kinodynamic constraints at the expense of more dimensions [11]. But in RAFESS we have to deal with many narrow passages that are a problem for RRTs. Furthermore, we are not looking for the first path found, but for a (near) optimal path. In this case, the PRM framework (e.g. [10]) has the advantage that we can preplan a roadmap before the surgery starts. In addition, the difficulties with narrow passages can be solved with a preferred sampling along the medial axis as proposed in [20]. Though, we would need many samples in order to get an optimal path and these samples need to be updated during the surgery. Therefore, we investigated another promising approach.

In RAFESS the translation within the pivot plane is quite limited and the kinematic constraint arising from collisions of the endoscope tip (sphere $\mathbf{s}_1$) with obstacles can always be fulfilled for $(a,b) = (0,0)$ and sufficient freedom for α and β. Thus, it makes sense to constrain the 5d path planning to a 3d problem with the endoscope axis fixed at the pivot point. In 3d efficient deterministic approaches are available. One of them is the fast marching method of [15]. According to [15] the level set $T(\mathbf{x}) = t$ is the set of all points in R^3 that can be reached with minimal travel time t, and the minimal time paths are orthogonal to the level curves; hence we have

$$\nabla T(\mathbf{x})^T M(\gamma(s))^{-1} \nabla T(\mathbf{x}) = 1 \tag{16}$$

If M is the identity matrix multiplied with a scalar, this eikonal equation reduces to the isotropic version

$$\|\nabla T(\mathbf{x})\| = 1/v \tag{17}$$

with the local speed value v. The standard fast marching method can be used to solve this equation for all $T(\mathbf{x})$ in R^3. If F is anisotropic (Eqn. 16), we have to switch to methods proposed in [4] or [16]. A special case occurs, if M is a diagonal matrix (this is also true for our case). Then the standard fast marching is applicable with a slight modification [16]. After the fast marching, we get the optimal path from $\mathbf{x_1}$ to $\mathbf{x_0}$ by following the gradient of $T(\mathbf{x})$ with respect to the inverse metric M^{-1}.

In our current implementation the search space is divided into a regular grid; this is not strictly necessary for fast marching methods. We do not precompute the constraints for all configurations, but instead we calculate them on demand, as our environment is dynamic and configurations are only processed once. This is a property of fast marching, since it works like a wave front (narrow band) that propagates always outwards with positive speed. In RAFESS the free space is very limited, often resulting in a small narrow band. Another very nice property of fast marching is that the user can easily setup more constraints to limit the search space. For example, we could use a heuristic that ignores configurations far away from the straight line segment connecting $\mathbf{x_0}$ and $\mathbf{x_1}$.

Our global 3d path planner delivers a (near) optimal path for the case of (a,b) frozen at $(0,0)$. Now we want to relax this limitation and optimize the path in 5d

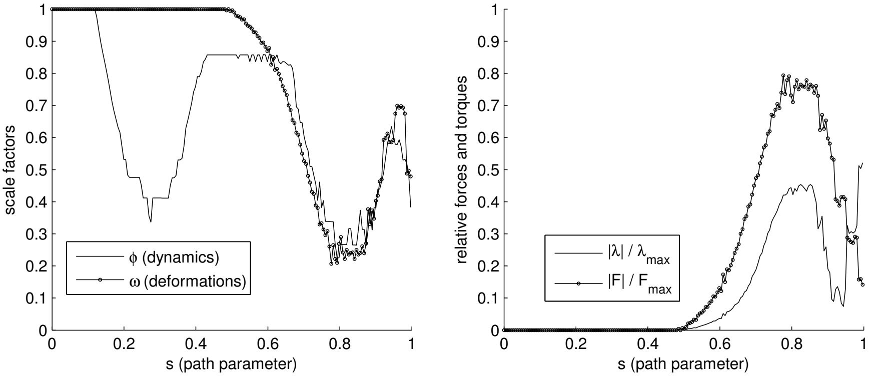


Fig. 4 Shows the computed scale factors and relative forces/torques along the upper path visible in Fig 5. *(Left)* For the range $s \in [0, 0.12]$ the endoscope is allowed to move at maximum velocity. Between $s = 0.12$ and $s = 0.6$ the velocity is reduced due to dynamic constraints. From $s = 0.6$ on, the velocity is primarily bounded by the estimated forces. *(Right)* Between $s = 0.86$ and $s = 1$ forces are dropping, because deformations nearby the pivot point are relaxing and opposing forces are summing up to zero. At $s = 0.9$ the torques are rapidly increasing due to forces nearby the tip of the endoscope applied to a long lever.

locally. We represent the path as a sequence of configurations; this sequence could also be converted into a B-spline representation. A straightforward way is to use a gradient descent scheme for optimizing $T(\mathbf{x_1})$. This has to be done in a careful way to keep the distance between two neighbouring configurations below a certain threshold and assuring fast convergence. Another option would be, to directly use $T(\mathbf{x})$ as a navigation function within a motion controller, which would allow for an elegant way to react to sensor events (e.g. force torque sensor values too high).

5 Experiments

Up to now, we have done most of our experiments in a special 2d scenario, in which we have set b and β to constant zero. The concepts remain the same, but the

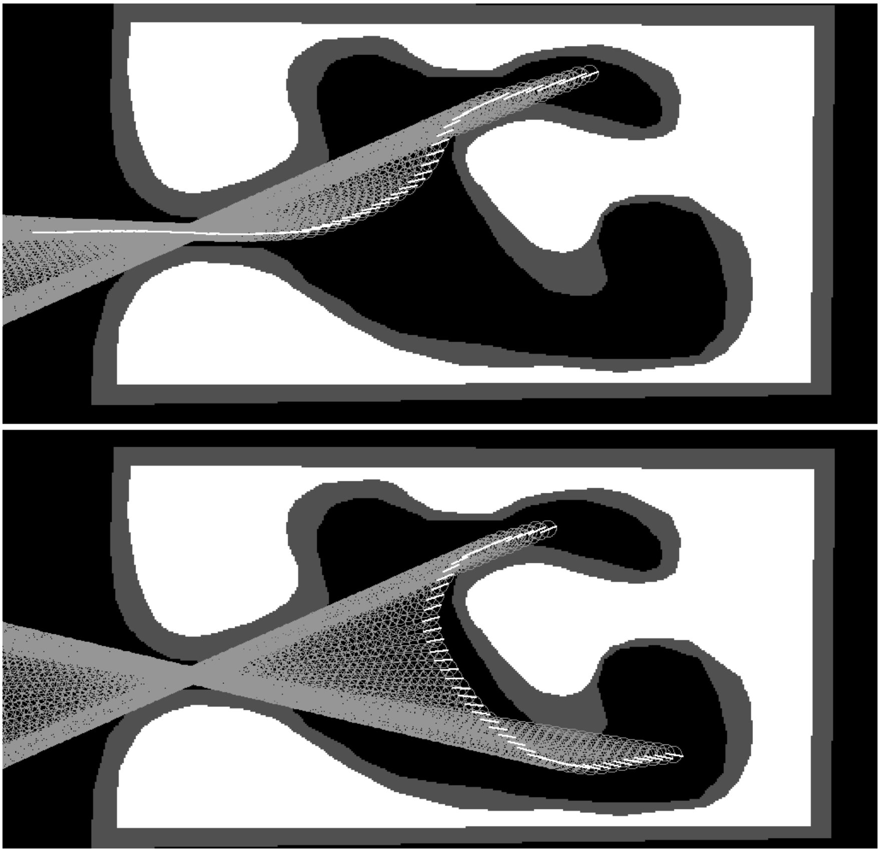


Fig. 5 Two planned paths of our 2d experiments. *HARD* structures are white and *SOFT* structures are dark gray. Only every tenth endoscope pose is drawn. *(Top)* The endoscope moves from outside into the upper right cavity. *(Bottom)* The endoscope moves from this position into the lower cavity.

testing is a lot easier. Our toolchain allows us to create arbitrary 2d environments like the one in Fig. 5. In this example the environment was of size 512x256 pixels and the configuration space had a resolution of 512x32x256 (l, a, α). The run time for a path planning request was at about one second for the worst case (single threaded, E6600 CPU, unoptimized code). In all tested environments so far, the path planning worked as expected: The paths were smooth with a good balance between limited deformations and straight driving to the goal. In Fig. 4 the computed scale factors ϕ (Eqn. 11) and ω (Eqn. 12) together with the estimated forces/torques are given for one of the planned paths. Although the resulting forces and torques are good indicators for deformations, not all critical situations might be detected in our current implementation, because the forces/torques, which are individually estimated for each sphere by Eqn. 3, can cancel each other. Therefore, we are investigating other constraints that directly use these individual forces/torques. A detailed performance analysis especially for the 3d case and a comparison between estimated forces and measured ones will be part of future publications.

6 Conclusion

Motion planning in deformable and dynamic environments is a highly demanding task. Our proposed novel concepts are an important step in that direction. The presented simplified biomechanical model is easy to implement and is well suited for GPU processing. How it compares to FEM and other models is still an open question; but we are confident that at least for small deformations the accuracy will be enough for our demands in RAFESS. It might even be possible to approximate non-linear behaviors, since the modulus of elasticity can be altered for each voxel. Furthermore, the derived velocity constraints are intuitive and integrate nicely into an eikonal solver framework. On a CPU the fast marching algorithm is very elegant, but it is a bit too slow for our demands in 3d. Therefore, we are also looking at good heuristics and other methods like the eikonal solver proposed in [6], which allows an efficient implementation on a GPU.

Acknowledgements. This project is funded by the German Research Foundation (DFG: Deutsche Forschungsgemeinschaft) under the grants WA 848/13-2 and BO 1909/3-2 ”Roboterunterstuetzte, erwartungskonforme Endoskopfuehrung in der endonasalen Chirurgie”.

References

1. Dario, P., Hannaford, B., Menciassi, A.: Smart surgical tools and augmenting devices. IEEE Transactions on Robotics and Automation 19(5), 782–792 (2003)
2. Donald, B., Xavier, P., Canny, J., Reif, J.: Kinodynamic motion planning. Journal of the ACM 40(5), 1048–1066 (1993)
3. Gayle, R., Lin, M.C., Manocha, D.: Constraint-based motion planning of deformable robots. In: Proc. IEEE International Conference on Robotics and Automation ICRA

2005, pp. 1046–1053 (2005)

4. Hassouna, M.S., Abdel-Hakim, A.E., Farag, A.A.: Pde-based robust robotic navigation. Image and Vision Computing 27(1-2), 10–18 (2009)
5. Jbabdi, S., Bellec, P., Toro, R., Daunizeau, J., Pélégrini-Issac, M., Benali, H.: Accurate anisotropic fast marching for diffusion-based geodesic tractography. International Journal of Biomedical Imaging 2008(1), 320, 195 (2008)
6. Jeong, W.K., Fletcher, P.T., Tao, R., Whitaker, R.T.: Interactive visualization of volumetric white matter connectivity in dt-mri using a parallel-hardware hamilton-jacobi solver. IEEE Transactions on Visualization and Computer Graphics 13, 1480–1487 (2007)
7. Kuffner, J.J., LaValle, S.: Rrt-connect: An efficient approach to single-query path planning. In: Proc. IEEE International Conference on Robotics and Automation ICRA 2000, vol. 2, pp. 995–1001 (2000)
8. Jones, M.W., Baerentzen, J.A., Sramek, M.: 3d distance fields: a survey of techniques and applications. IEEE Transactions on Visualization and Computer Graphics 12(4), 581–599 (2006)
9. Kavraki, L.E., LaValle, S.M.: Springer Handbook of Robotics, ch. 5, pp. 109–131. Springer, Heidelberg (2008)
10. Kavraki, L.E., Svestka, P., Latombe, J.C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation 12(4), 566–580 (1996)
11. LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. In: Proc. IEEE International Conference on Robotics and Automation, vol. 1, pp. 473–479 (1999)
12. Miller, K., Joldes, G., Lance, D., Wittek, A.: Total lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation. Communications in Numerical Methods in Engineering 23, 121–134 (2007)
13. Moll, M., Kavraki, L.E.: Path planning for deformable linear objects. IEEE Transactions on Robotics 22(4), 625–636 (2006)
14. Pott, P.: Meroda, the medical robotics database (2008), `http://www.ma.uni-heidelberg.de/apps/ortho/meroda/`
15. Sethian, J.A.: Fast marching methods. SIAM Review 41, 199–235 (1999)
16. Sethian, J.A., Vladimirsky, A.: Ordered upwind methods for static hamilton–jacobi equations: Theory and algorithms. SIAM Journal on Numerical Analysis 41(1), 325–363 (2003)
17. Taylor, R.H., Stoianovici, D.: Medical robotics in computer-integrated surgery. IEEE Transactions on Robotics and Automation 19(5), 765–781 (2003)
18. Taylor, Z.A., Cheng, M., Ourselin, S.: High-speed nonlinear finite element analysis for surgical simulation using graphics processing units. IEEE Transactions on Medical Imaging 27(5), 650–663 (2008)
19. Tingelhoff, K., Moral, A.I., Kunkel, M.E., Rilk, M., Wagner, I., Eichhorn, K.W.G., Wahl, F.M., Bootz, F.: Comparison between manual and semi-automatic segmentation of nasal cavity and paranasal sinuses from ct images. In: Proc. 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society EMBS 2007, pp. 5505–5508 (2007)
20. Wilmarth, S.A., Amato, N.M., Stiller, P.F.: Maprm: a probabilistic roadmap planner with sampling on the medial axis of the free space. In: Proc. IEEE International Conference on Robotics and Automation, vol. 2, pp. 1024–1031 (1999)

An Unmanned Helicopter for Autonomous Flights in Urban Terrain

Florian Adolf, Franz Andert, Sven Lorenz, Lukas Goormann, and Jörg Dittrich

Abstract. This work summarizes a multi-disciplinary research project, focusing on key enabling techniques towards true autonomous flight of small, low flying VTOL UAVs. Research activities cover the flying testbed, a simulation and testing environment, as well as integrated components for onboard navigation, perception, planning and control. Promising results and feasibility demonstrations in flight tests underline the successful domain specific enhancements of approaches based on aeronautical engineering, computer science and mobile robotics. The current approaches pave the way towards further research in improved flight control performance and more system autonomy when a-priori mission uncertainties increase.

1 Introduction

Potential mission scenarios for small, low flying UAVs involve agile navigation in urban environments. Today, such UAVs are dependent on interference-prone data links to an operator. Inherent problems like increased latency and reaction time, as well as degraded situational awareness can endanger the success of a mission. Furthermore, a challenging problem is to safely maneuver a UAV in three dimensional autonomous flight, low above the ground and unknown, uncontrolled airspace.

Hence, the project ARTIS (Autonomous Rotorcraft Test bed for Intelligent Systems) aims at maximized onboard information processing and decision-making in cooperation with operators. Thus, the ARTIS project uses a small helicopter platform due to its agile maneuverability, vertical take off and landing capability, and feasible payload capacity for sufficient computational power and necessary sensors.

A number of research projects underline the potential of such helicopter-based VTOL UAVs. Various applications like structure inspection [17], reconnaissance [32], search and rescue [12] or operational aspects [22] have been developed.

Florian Adolf · Franz Andert · Sven Lorenz · Lukas Goormann · Jörg Dittrich
German Aerospace Center (DLR), Institute of Flight Systems,
Lilienthalplatz 7, 38108 Braunschweig, Germany,
e-mail: `{florian.adolf,franz.andert,sven.lorenz,lukas.goormann,joerg.dittrich}@dlr.de`

Advanced flight control [30, 20] as well as onboard perception and decision making [12] are presented in this context. Machine vision in particular, either tightly coupled or loosely coupled to flight control, shows its potential for 3D mapping [17, 30], navigation [8, 24, 11] and target tracking [32]. Furthermore, flight tested results in [28, 30] highlight research directions in reactive obstacle avoidance and global path planning. To our knowledge, only a few papers (e.g. [17]) describe their efforts in testing before flight.

ARTIS is devoted to autonomy in flight by integrating research contributions to the aforementioned challenges. The helicopters are equipped with avionics for onboard sensor fusion and flight control, highlighted in section 2. In section 2.1 a flight control concept is presented. The three dimensional environment perception and situational awareness is shown in 2.2. Approaches for onboard decision making are shown in section 2.3. Furthermore, the methodology for the transition from simulation to flight tests is presented section 3. A research outlook concludes the current project status.

2 The Intelligent Unmanned Aerial System

As presented in [9], the helicopter UAV *midiARTIS* was used for the initial automated flights. Later, the larger helicopter *maxiARTIS* has been developed to improve endurance and payload capacity. Both helicopters (Table 1) follow a modular avionics concept inspired by [10]. Sensory equipment comprises DGPS, inertial sensor and magnetometer for navigation, sonar altimeter for landing, wireless data links,

Table 1 The ARTIS flight test vehicles.

	midiARTIS	maxiARTIS
first flight	2003	2006
rotor diameter	1.9 m	3 m
engine	3.5 kW combustion engine	5 kW turbine engine
endurance	15–20 min	30–45 min
empty weight	6 kg	15 kg
avionics weight	4 kg	4 kg
experimental payload	2 kg	6 kg
flight control computer	Intel P4 mobile 1.4 GHz, QNX OS	
image processing computer	Intel P4 mobile 1.4 GHz, Linux OS	

manual remote control and video data links, a telemetry module and a flight control computer. Additionally, a variety of cameras or laser sensors can be installed and connected to a dedicated image processing computer. The computers and the camera are mounted on a flexible bearing to be exchanged easily.

Its role as a flying testbed enables the evaluation of a variety of approaches to adaptive flight control, vision-based navigation, environment perception and onboard decision making.

2.1 Flight Control

Helicopter dynamics are in general nonlinear, time varying. Together with the modeling uncertainties the state estimation based on light and simple sensor systems and the fundamental changing dynamics between hover and forward flight conditions the control design becomes quite challenging.

Various methods have been developed to focus on the robustness to sensor noise, parameter uncertainties, and unmodeled dynamics while determining the feedback gains. To tackle nonlinear dynamics, presently, a nonlinear control method commonly used in conventional aircraft control is gain scheduling. In this method, linear models are approximated and referenced corresponding to different flight conditions. As a result of changing flight conditions, different feedback gains are calculated to increase robustness and stability. Used together with a standard feedforward control technique adequate closed loop properties can be achieved.

As a consequence of quickly changing flight conditions due to agile maneuvers, the feedback gains must be changed appropriately. Often gains are scaled with respect to specific states, for instance dynamic and barometric pressure. For stability reasons, the gain determining states must not change faster than the dynamics otherwise oscillations may be excited. In addition, a complex feed-forward control is required for sufficient command following.

As a special case of global linearization the method of *Feedback Linearization* [31], also known as Nonlinear Dynamic Inversion (NDI), may be an attractive alternative to the gain-scheduling approach. A state transformation based on the derivation of the output equations leads to a “new” system with linear feedback dynamics. This simplifies the feedback gain determination because only one set of gains is required (Fig. 1). In addition from feedback perspective the approach decouples the dynamics and leads in consequence to simple state or output feedback structures. Unfortunately, this method seems very sensitive to unconsidered actuator dynamics, delays, and parameter uncertainties. Therefore, considerable extensions e. g. Pseudo-Control-Hedging (PCH) [18, 19] were developed.

Based on a simplified linear model [7] and the inversion using Feedback Linearization, respectively, the control law becomes feasible for hover flight conditions. As expected, the dynamics alter with increasing flight velocity and the controller performance decreases considerably as the control law was intended for hover conditions. Adaptive extensions are applicable to compensate rapidly for these changes. The most promising method of on-line adaptation is based on a single hidden layer

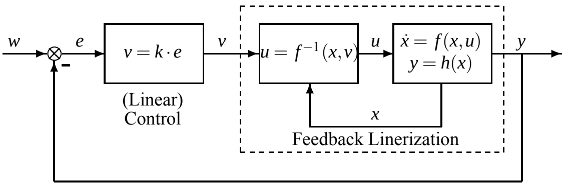


Fig. 1 Principle of Feedback Linearization.

neural network proposed by [25]. Several applications confirm the universal adaption capabilities [20, 16, 23, 15, 21]. On-going research will lead to fault-tolerant control systems for a wide spectrum of applications in highly uncertain environments and hence supports the reliability of autonomous systems.

2.2 Machine Vision

Environmental sensors are used in addition to navigation sensors and they are required for robotic applications like obstacle detection. Cameras are used since they are lightweight, passive and energy-efficient in contrast to laser range finders or radar systems. To process the images on-board in real-time, ARTIS is equipped with a dedicated image processing computer. The cameras are linked to the image processing computer via a IEEE 1394 link and the DCAM programming interface. A large variety of cameras is used, e.g. classical monochrome or color industry cameras or stereo cameras with integrated FPGA-based disparity image generation. The image processing computer is connected to and time-synchronized with the flight control computer for data exchange.

One of the first vision-based applications was the automatic search for a moving ground pattern presented by [4]. An identifiable pattern is fixed on a R/C car to be moved. A downwards-pointing camera is installed at the helicopter and the known pattern is searched in the images. The relative pattern position is estimated by photogrammetric transformations and with that, the helicopter is instructed to fly to a coordinate exactly above the pattern. Flight experiments show successful pattern identification and vehicle following.

2.2.1 GPS-Loss Compensation

Navigation and control is usually based on GPS/INS position and attitude information. If the GPS connection is lost, the position is estimated by integrating acceleration measurements. Without GPS, the position bias increases very fast, and a stabilization or accurate navigation cannot be guaranteed anymore. To cope with that problem, [24] presents a vision-based navigation filter for ARTIS. It is designed for stabilized hovering and navigation with limited GPS connectivity. As in the pattern tracking setup, a camera is pointing downwards. Texture features of an arbitrary planar ground are tracked. By combining the 2D image feature movement with altitude information (e.g. sonar or altimeter-based) and the three attitude angles

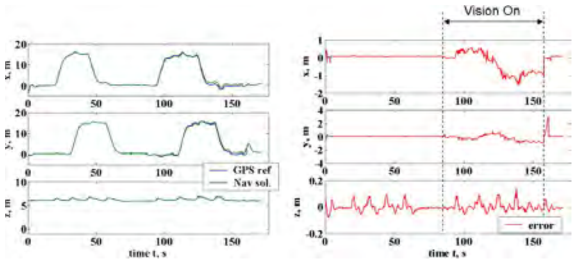


Fig. 2 Flight test results of the vision-based navigation. Comparison of estimated position with unfiltered GPS-based reference (left) and the error between filter output and reference (right).

(IMU-based), all six degrees of freedom in the vehicle pose are obtained when GPS localization is not available. Due to the relative vision-based positioning, drift errors occur like in an inertial measurement system. Nevertheless, the error is significantly reduced. Results (fig. 2) show that it is possible to hover without GPS information and to navigate with an error of only one meter per minute flight time.

2.2.2 Mapping and World Modeling

To plan or update trajectories in unknown urban environments, obstacles must be recognized with sensors and integrated to metric maps. A stereo camera is used for sensing obstacles. It is small, passive and energy-efficient and qualified for distance measurement on a UAV in daylight conditions.

The stereo-based mapping algorithm by [3] combines grid and metric feature maps to use the advantages of both map types. Occupancy grid maps sample the space into occupied and free cells and they are best qualified for sensor fusion, i.e. to reduce noise, integrate multiple images or even multiple cameras or vehicles. They are built incrementally and new sensor data is inserted just in time. Then, clusters of occupied cells are regarded as single obstacles and inserted to the feature map. A global map is built from these features in order to get an output for an obstacle avoidance application. To achieve a compact representation of arbitrary object shapes, the cell clusters are approximated by polygonal structures. In a real scene like a city, objects can have any ground shape from the top view but it is most likely that they have vertical walls. With this assumption, the cell clusters of 3D objects are approximated with one or more bounding prisms. In addition to that, the ground plane height is estimated via simplified Hough transform. Figure 3 shows the helicopter flying in an urban environment and a map created by the described algorithm. The ground is successfully detected. The houses and the gap between them are also recognized.

The resulting prism-based object map is sent to the flight control computer that can calculate a collision-free flight trajectory around the obstacles in the map and instructs the helicopter to fly that path. Map updates are sent regularly so that the trajectory planner can consider previously unknown objects.

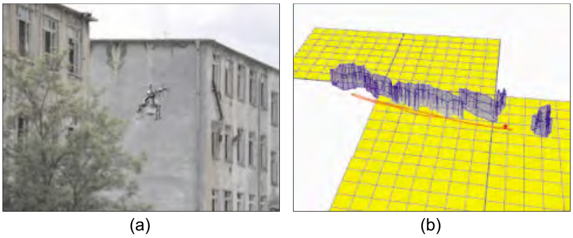


Fig. 3 Example of a created map. Urban outdoor environment (left) and resulting 3-D map of the scenario (right) with ground planes, bounding prisms and the flown trajectory.

2.3 Decisional Autonomy

A mission management system aims on risk and workload reduction of the operator. This section highlights the decision making techniques that assist the operator before and during a mission.

2.3.1 Mission Execution

The work in [2] describes a control architecture that achieves hybrid control by combining the main ideas from the behavior-based paradigm [6, 13] and a three-tier architecture [5]. The former reduces the system modeling complexity for composite maneuvers (e.g. land/take off) as a behavior module that interfaces with the flight controller (Figure 4). The latter has the advantage of different abstraction layers that can be interfaced directly such that each layer represents a level of system autonomy. The Sequence Control System implements the sequencing layer of this architecture. It is open with respect to its interfaces to other onboard components (e.g. the visual world model shown in 2.2.2). Events and commands send by a remote operator or an onboard component can be integrated into the system in a "plug-and fly" fashion. A remote operator can send direct commands on different abstraction levels, ranging from controller supported joystick control towards execution of mission plans. The solutions to more complex mission problems and events (e.g. loss of the data link) are delegated using a sequence of commands specified as a context-free language.

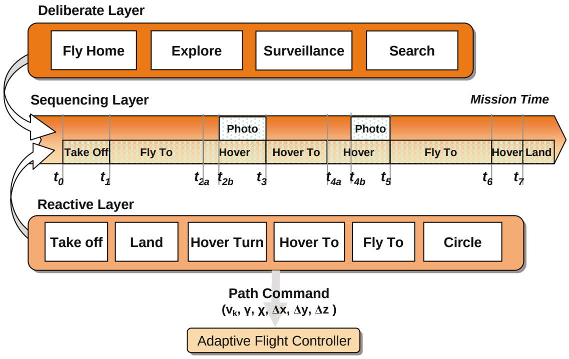

Deliberate Layer
Fly Home
Explore
Surveillance
Search
Sequencing Layer
Mission Time
Photo
Photo
Take Off
Fly To
Hover
Hover To
Hover
Fly To
Hover
Land
t_0
t_1
t_{2a}
t_{2b}
t_3
t_{4a}
t_{4b}
t_5
t_6
t_7
Reactive Layer
Take off
Land
Hover Turn
Hover To
Fly To
Circle
Path Command
$(v_k, \gamma, \chi, \Delta x, \Delta y, \Delta z)$
Adaptive Flight Controller

Fig. 4 Exploration mission shown in context with the control architecture: High-level behaviors uses task specific planners (Deliberate Layer), behaviors are compiled into plans (Sequencing Layer), and movement primitives (Reactive Layer) interface with the flight controller.

More system autonomy increases responsibility for more critical tasks. To assure a safe operation of the Sequence Control System, the system is characterized by multiple techniques. In [29] an event-based decision logic is modeled as a State Chart. It is verified against path-, transition- and state-coverage tests. As shown in [2], the validity of commands (e.g. from ground control station, smart sensors) is validated online with respect to priority and plausibility. As a result, the system allows a robust, deterministic high level control of UAVs. It allows an operator to command a UAV at different levels of autonomy.

2.3.2 3D Path Planning and Task Optimization

UAV operators have an intrinsic need for reduced mission planning complexity when specifying collision free paths and mission tasks (e.g. object search within a given area). Given the onboard control system described in the previous section, it is possible to execute mission plans. This planning system automates the translation of user specified sets of waypoints into a sequence of parameterized behavior commands. The path planner searches collision free paths in obstacle-constrained three dimensional space whereas the task planner determines a near optimal order for a set of tasks. Moreover, a task planner can solve specialized problems, e.g. specifying the actual waypoints for an object search pattern within an area of interest. The mission planner in [1] uses a multi-query strategy, based on a polygonal obstacle representation, a search graph ("Roadmap", [26]) for 3D path planning and Ant Colony Optimization[14,] for the task ordering of heterogeneous agents. In this task planning context two concepts, either minimum spanning-trees or a heuristic solution, are used to resolve potential conflicts before the order of tasks for each UAV is optimized. The synthesis of these planning approaches contribute to the long term research goal of ARTIS, to coordination multiple, heterogeneous agents [9]. Moreover, the work in [27] addresses path optimization of non-linear path geometries (e.g. splines) while global path searches remain graph queries in linear metric space (e.g. visibility graph or road map-based connections). The proposed approach determines which nodes to remove from the effective path ("shortcuts") and where

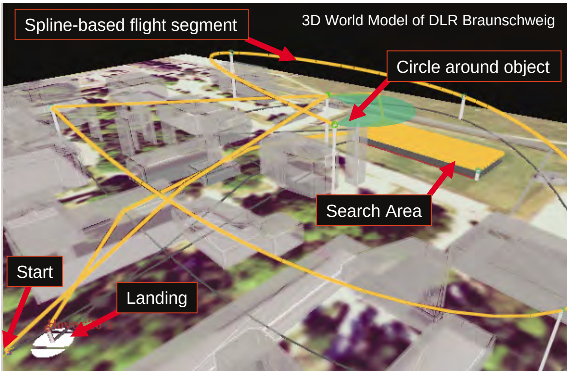

Spline-based flight segment
3D World Model of DLR Braunschweig
Circle around object
Search Area
Start
Landing

Fig. 5 Mission with non-linear flight segments, a search area task and a horizontal inspection circle.

path segments need additional nodes (“smoothing”). As a result the mission planner allows to plan three dimensional, safe and unobstructed paths quickly while optimizing task assignments and task orderings for multiple UAVs (Fig. 5).

3 From Simulation to Outdoor Flights

This section describes the development process for prototype testing in simulation towards flight testing. New approaches to flight control and mission management are developed and tested in a Matlab/Simulink environment . This *Software-in-the-Loop* test simulates the vehicle dynamics, emulates sensor inputs and interfaces to the ground control software via virtual wireless data links. Results from this prototyping phase share the same code basis from simulation to flight test. The integrated hardware and software is tested *Hardware-in-the-Loop* (HITL) before the flight. The UAV and the flight control equipment are exactly the same as it is used later in flight. Sensor inputs and actuator outputs are connected to a real-time environment (Fig. 6) that simulates the aircraft flight mechanics and emulates all sensors. Thus it is possible to test the GNC (Guidance, Navigation and Control) software and the operators control interface under realistic conditions. Furthermore, scenarios can be simulated that currently cannot be performed in real flight tests, e.g. due to law or resource constrains.

The vision behavior and its interaction with the flight control is tested in HITL. The image processing is setup in its own testing rig. Cameras capture images that are generated by virtual 3D scene (Fig. 7) which is projected on one screen for a monocular camera and on two for a stereo camera. The loop is closed by connecting the vision simulation to the flight controller.

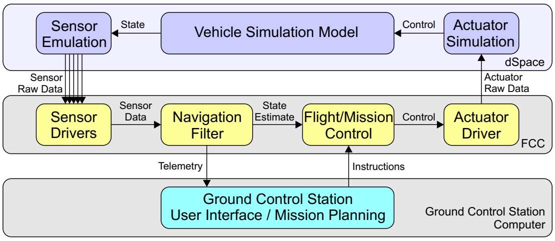

Sensor Emulation
State
Vehicle Simulation Model
Control
Actuator Simulation
dSpace
Sensor Raw Data
Actuator Raw Data
Sensor Drivers
Sensor Data
Navigation Filter
State Estimate
Flight/Mission Control
Control
Actuator Driver
FCC
Telemetry
Instructions
Ground Control Station User Interface / Mission Planning
Ground Control Station Computer

Fig. 6 The Hardware-in-the-Loop Setup.

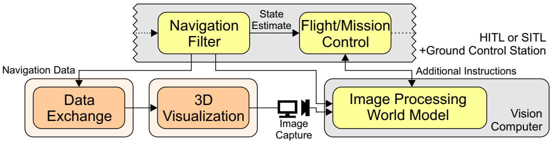

Navigation Filter
State Estimate
Flight/Mission Control
HITL or SITL +Ground Control Station
Navigation Data
Additional Instructions
Data Exchange
3D Visualization
Image Capture
Image Processing World Model
Vision Computer

Fig. 7 Vision Simulation Setup.

Promising results in simulation enable flight tests on one of the ARTIS vehicles introduced in section 2. A safety pilot with manual R/C control, real data links and real sensor inputs make the difference to the setup tested in HITL.

4 Summary

This paper presents the current status of the unmanned helicopter project ARTIS and its two flying test beds. The challenges of autonomous flight near the ground in urban environments are addressed by a set of multi-disciplinary approaches that cover fields in adaptive flight control, vision-based compensation of GPS loss, obstacle mapping, motion planning, and onboard decision making. The project reached a maturity stage in operator-supervised autonomous missions. The development infrastructure allows rapid development and testing of new approaches. It includes software- and hardware-in-the-loop simulations using the original avionics including vision. All presented approaches are flight tested under realistic operational conditions.

In theory, high level control problems in the presented UAV domain, like path planning or performing waypoint navigation maneuvers, cannot be considered to be solved. Integrated architectural concepts can rarely be observed in flight tests. Showing the multi-disciplinary challenges of bridging theory and practice is the goal in this paper and will be an important future research aspect within ARTIS. All research directions covered in this work allow improvement to achieve the vision of fully autonomous mission scenarios. Future approaches need further adaptivity for the flight guidance systems, flexible mapping, obstacle avoidance, online planning capabilities and a specification of more complex mission goals.

References

1. Adolf, F., Langer, A., de Melo Pontes e Silva, L., Thielecke, F.: Probabilistic roadmaps and ant colony optimization for uav mission planning. In: 6th IFAC Symposium on Intelligent Autonomous Vehicles, Toulouse, France (September 2007)
2. Adolf, F., Thielecke, F.: A sequence control system for onboard mission management of an unmanned helicopter. In: AIAA Infotech@Aerospace Conference, Rohnert Park, CA, USA (May 2007)
3. Andert, F., Goormann, L.: A fast and small 3-d obstacle model for autonomous applications. In: IEEE/RSJ IROS, pp. 2883–2889 (2008)
4. Bernatz, A., Thielecke, F.: Navigation of a low flying vtol aircraft with the help of a downwards pointing camera. In: AIAA GN&C Conference (2004)
5. Bonasso, R.P., Firby, R.J., Gat, E., Kortenkamp, D., Miller, D.P., Slack, M.G.: Experiences with an architecture for intelligent, reactive agents. Journal of Experimental & Theoretical Artificial Intelligence 9(2-3), 237–256 (1997)
6. Brooks, R.S.: A robust layered control system for a mobile robot, pp. 204–213. Morgan Kaufmann Publishers Inc., San Francisco (1990)
7. Chowdhary, G., Lorenz, S.: Non-linear Model Identification for a Miniature Rotorcraft, Preliminary Results. AHS Forum 61, Grapevine, Texas USA (2005)

8. Civita, M.L., Papageorgiou, G., Messner, W., Kanade, T.: Design and flight testing of a high-bandwidth h-infinity loop shaping controller for a robotic helicopter. Journal of Guidance, Control, and Dynamics 29(2), 485–494 (2006)
9. Dittrich, J., Bernatz, A., Thielecke, F.: Intelligent systems research using a small autonomous rotorcraft testbed. In: 2nd AIAA Unmanned Unlimited Conference (2003)
10. Dittrich, J., Johnson, E.: Multi-sensor navigation system for an autonomous helicopter. In: 21st Digital Avionics Conference (2002)
11. Doherty, P., Granlund, G., Kuchcinski, K., Sandewall, E., Nordberg, K., Skarman, E., Wiklund, J.: The witas unmanned aerial vehicle project. In: 14th European Conference on Artificial Intelligence, August 2000, pp. 747–755 (2000)
12. Fabiani, P., Fuertes, V., Piquereau, A., Mampey, R., Teichteil-Königsbuch, F.: Autonomous flight and navigation of vtol uavs: from autonomy demonstrations to out-of-sight flights. Aerospace Science and Technology 11, 183–193 (2007)
13. Flanagan, C., Toal, D., Jones, C., Strunz, B.: Subsumption architecture for the control of robots. In: Proceedings Polymodel-16 (1995)
14. Gambardella, L.M., Dorigo, M.: Solving symmetric and asymmetric tsps by ant colonies. In: IEEE ICEC, pp. 622–627 (1996)
15. Holzapfel, F.: Nichtlineare adaptive Regelung eines unbemannten Fluggeräts. PhD thesis, Technische Universität München (2004)
16. Hovakimyan, N., Kim, N., Calise, A., Prasad, J., Corban, J.E.: Adaptive Output Feedback for High-Bandwidth Control of an Unmanned Helicopter. In: AIAA GN&C Conference (2001)
17. Hrabar, S.: 3d path planning and stereo-based obstacle avoidance for rotorcraft uavs. In: IEEE/RSJ IROS, pp. 807–814 (2008)
18. Johnson, E.: Limited Authority Adaptive Flight Control. PhD thesis, Georgia Institute of Technology, Atlanta, GA (November 2000)
19. Johnson, E., Calise, A.: Neural network adaptive control os systems with input saturation. Technical report, Georgia Institute of Technology, Atlanta, Georgia (June 2001)
20. Johnson, E., Calise, A., El-Shirbiny, H., Rysdyk, R.: Feedback linearization with neural network augmentation applied to x-33 attitude control. Technical report, Georgia Institute of Technology, AIAA Guidance Navigation and Control Conference and Exhibit (August 2000)
21. Johnson, E.N., Turbe, M.A.: Modeling, Control, and Flight Testing of a Small Ducted-Fan Aircraft. Journal of Guidance, Control, and Dynamics 29, 769–779 (2006)
22. Jones, H., Frew, E., Woodley, B., Rock, S.: Human-robot interaction for field operation of an autonomous helicopter. In: Mobile Robots XIII (November 1998)
23. Kim, N.: Improved Methods in Neural Network Based Adaptive Output Feedback Control, with Applications to Flight Control. Ph.D thesis, Georgia Institute of Technology (July 2003)
24. Koch, A., Wittich, H., Thielecke, F.: A vision-based navigation algorithm for a vtol uav. In: AIAA GN&C Conference (2006)
25. Lewis, F., Jagannathan, S., Yesildirek, A.: Neural Network Control of Robot Manipulators and Nonlinear Systems. In: British Library Cataloguing in Publication Data. Taylor & Francis, Abington (1999)
26. Pettersson, P.O.: Sampling-based path planning for an autonomous helicopter, Report code: LiU-Tek-Lic-2006:10 (2006)
27. Praxedes, L.G.: Advanced Three-Dimensional Route Planning under Complex Constraints. Instituto Tecnológico de Aeronáutica, São José dos Campos (2007)
28. Scherer, S., Singh, S., Chamberlain, L., Saripalli, S.: Flying fast and low among obstacles. In: IEEE ICRA, pp. 2023–2029 (2007)

29. Schwarzer, R., Adolf, F., Gräbe, H.: Testverfahren für eventgetriebene Software am Beispiel der ARTIS Plattform. Logos Verlag Berlin (June 2007)
30. Shim, D.H., Chung, H., Sastry, S.S.: Conflict-free navigation in unknown urban environments. IEEE Robotics and Automation Magazine 13(3), 27–33 (2006)
31. Slotine, J.-J.E., Li, W.: Applied Nonlinear Control. Prentice Hall, Englewood Cliffs (1991)
32. Watanabe, Y., Calisey, A.J., Johnson, E.N.: Vision-based obstacle avoidance for uavs. In: AIAA GN&C Conference (2007)

Interaction of Altitude Control and Waypoint Navigation of a 4 Rotor Helicopter

Tim Puls, Hannes Winkelmann, Sönke Eilers, Matthias Brucke,
and Andreas Hein

Abstract. The motivation for the investigation of the interaction of a altitude and position resp. waypoint controller is the demand for Unattended Aerial Systems (UAS) capable to fulfill e.g. surveillance task in contaminated or in not accessible areas. This paper deals with the development of such altitude controller and a 2D position controller based on GPS data for Unattended Aerial Vehicles (UAV), which are able to hover above a given destination as well as to navigate between waypoints. In addition, these controllers are designed to let a four rotor UAV fly with the maximum possible speed taking into account that the UAV must not lower the cruising altitude in an uncontrolled manner.

The introduced controllers were implemented in Matlab/Simulink as well as on a target system for testing and verification purposes. Various simulations and experiments are provided explaining and proving the functionality of the described controller.

1 Introduction

It is foreseen that there will be a future market for intelligent surveillance and superring robots, capable of discreetly penetrating confined spaces and maneuvering in those without the assistance of a human pilot tele-operating the vehicle. Thus, the development of small autonomous aerial vehicles for outdoor and urban applications, which are able to perform agile flight maneuvers, is of significant importance. Such vehicles can also be used in environments where direct or remote human assistance is not feasible, e.g. in contaminated areas or in urban search and rescue operations for locating earthquake-victims or establishing ad-hoc networks. Especially the abilities to hover above a given fixed position and to maneuver with high agility at low speed are essential for the mentioned applications. For this reason we decided to use a four rotor vertical takeoff and landing (VTOL) helicopter to cover these applications instead of fixed-wing aircrafts.

Tim Puls · Hannes Winkelmann · Sönke Eilers · Matthias Brucke · Andreas Hein
OFFIS, University of Oldenburg, Escherweg 2, 26121 Oldenburg, Germany,
e-mail: `{tim.puls,hannes.winkelmann,soenke.eilers,`
`matthias.brucke,andreas.hein}@offis.de`

1.1 Problem Description

Besides the challenge of simply keeping such systems in the air due to their inherently unstable and nonlinear behavior, which is described in [1], one of the main problems is the ability to maneuver and hover autonomously without the interaction of a human pilot. Thereby it is desired to have a controller capable of hovering and maneuvering without having to switch between flight modes. Additionally the controller must be capable of handling constant wind as well as heavy blasts. Furthermore it must be taken care of that the UAV does not lower below the commanded altitude. This may occur when the attitude of the UAV is too angular and therefore the maximum thrust of the motors is not sufficient enough to lift the UAV. This effect becomes even more significant when wind attacks it from the direction the UAV is heading to, resulting in an additional loss of altitude. Yet, flying at maximum speed is desired due to the limited endurance of VTOL systems.

2 State of the Art

In [4] and [5] the control and navigation principles of classical fixed wing aircrafts are well described, including problems such as waypoint navigation and zero cross track angle control. But these solutions cannot be directly adapted to 4 rotor helicopters due to the different configuration of 4 rotor helicopters and due to the problem that a four rotor helicopter is an under actuated system (see [1]).

There are many papers regarding modeling and control of 4 rotor helicopters such as [6], [7], [8] but these papers mainly deal with different types of attitude controllers and not with position controllers.

Some work was done in [9] with respect to an attitude and altitude controller for indoor applications. In addition, a PID-position controller was developed and tested that was able to hold an indoor position within a range of 40cm.

In [10] a position controller based on monocular vision has been implemented and tested indoors. The quad rotor helicopter was able to hover above a position within a range of around 1m.

In [2], the ability of combining the Inertial Measurement Unit (IMU) with the Global Positioning System (GPS) to get proper position information is described, and how a VTOL system can be stabilized via GPS and without GPS. Yet, how to navigate between different waypoints is completely left out.

[11] deals with trajectory planning of a 4 rotor helicopter in GPS-denied environments. A Belief Roadmap algorithm was used to plan trajectories in indoor environments.

In [12] a semiautonomous “position hold” algorithm and waypoint navigator is presented. Nevertheless, this controller does not take into account the problems that a UAV could, e.g. be pushed away from a track or can lose height while on transition. This controller is also not designed to fly at the maximum possible speed.

3 Target System

Fig. 1 shows a photograph of the 4 rotor helicopter presented in real flight used to test and verify the developed position controller.

This aerial vehicle consists of an airframe, which is made out of fiber glass, carbon and Kevlar. Its design focuses on low weight, stiffness, and on protecting the environment from the fast rotating and sharp rotor blades. The dimensions of the airframe are 110 cm x 110 cm x 20 cm. Due to the take-off weight of approx. 3 kg the system is equipped with four motor rotor combinations that are able to lift up to 1.7 kg each. Thus, the system is able to carry an additional payload of 1kg and still has enough thrust to maintain its agility. A motor rotor combination consists of a brushless motor, a suitable rotor and a motor controller.

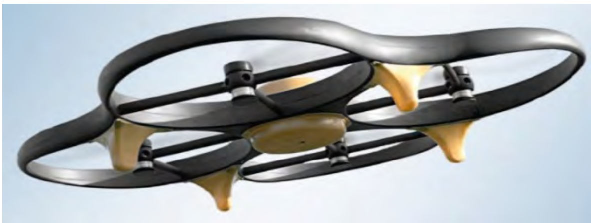


Fig. 1 Aerial robot GUARD® M [14].

3.1 Sensors

The system is equipped with many sensors in order to measure and calculate the current pose of the UAV, particularly three gyroscopes for the angles φ, θ, ψ three accelero-meters for x, y, z and three magnetic field sensors. These sensors are mainly used to detect the attitude of the system. Due to their inherent inaccuracy and drift these sensors are not sufficient to calculate the position in all three dimensions, so that additional sensors are necessary. Hence we use a barometer to correct the calculated height and a GPS receiver to detect the position. Due to the fact that this paper is not about the question of how to detect and calculate a precise geodesic position via GPS (see [2], [3]), it is adequate to use the raw GPS data for test purposes.

3.2 Attitude Controller

The attitude controller developed in [1] for the angles φ (roll), θ (pitch) and ψ (yaw) was implemented on the target system with little changes concerning an additional integral part to handle production tolerance in a proper way. Due to the fact that a four rotor helicopter is an under actuated system this attitude controller is the base of the described position controller because every position change leads to an adjustment of the roll and pitch angles φ and θ.

4 Altitude Control Algorithm

This section focuses on the altitude control algorithm. In Fig. 2, a flow diagram of the algorithm can be seen. Some descriptions in this figure are grayed out. These parts are explained in the Outlook chapter. This control sequence is being calculated with the same frequency as the attitude and position controller. In this case it is 120 Hz. The control algorithm can be divided into 8 parts, which are described in the following sections.

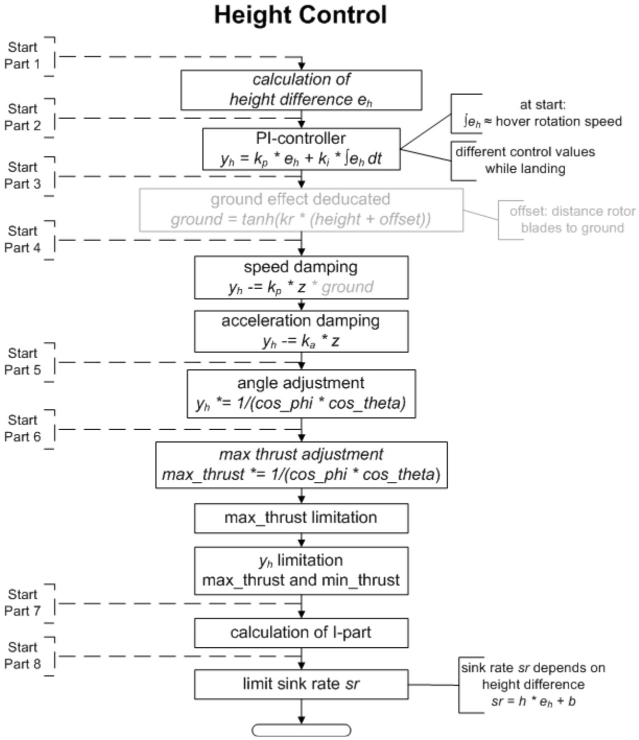

Height Control
Start Part 1
calculation of height difference e_h
Start Part 2
PI-controller
$y_h = k_p * e_h + k_i * \int e_h\,dt$
at start: $\int e_h \approx$ hover rotation speed
different control values while landing
Start Part 3
ground effect deducated
ground = tanh(kr * (height + offset))
offset: distance rotor blades to ground
Start Part 4
speed damping
y_h -= k_p * z * ground
acceleration damping
y_h -= k_a * z
Start Part 5
angle adjustment
y_h *= 1/(cos_phi * cos_theta)
Start Part 6
max thrust adjustment
max_thrust *= 1/(cos_phi * cos_theta)
max_thrust limitation
y_h limitation
max_thrust and min_thrust
Start Part 7
calculation of I-part
Start Part 8
limit sink rate sr
sink rate sr depends on height difference
$sr = h * e_h + b$

Fig. 2 Flow diagram of height controller.

4.1 Part 1: Calculation of Height Difference

In the first part the height difference e_h is calculated, with $e_h = z_d - z_c$, where z_c is the current and z_d the desired height.

4.2 Part 2: PI-Controller

This value e_h is used as input for a slightly adapted PI-controller. The difference to a classic PI-controller lies in the way the integral part is calculated (see Part7). In detail the value y_h is computed with

$$y_h = k_p * e_h + k_i * \int e_h \, dt$$

It is important that at the beginning $\int e_h \, dt$ must be initialized with a value that is sufficient to hover, so that it is not needed to build up the integral from zero.

4.3 Part 3: Ground Effect

To overcome the ground effect of the helicopter it is necessary to add an additional part to the controller which has to change the behavior of the controller while flying near ground (i.e. distance of half rotor diameter). This part was not implemented during the simulations and experiments and therefore it is not described here. *(see Outlook chapter)*

4.4 Part 4: Damping

In this part of the controller two different kinds of damping are added to the prior calculated value y_h. A velocity and an acceleration damping.

$$y_h = y_h - (k_d * \dot{z}_c + k_a * \ddot{z}_c)$$

The velocity damping is essential to provide the UAV from oscillating and the acceleration damping is able to counteract abrupt altitude changes due to blast for example.

4.5 Part 5: Angle Adjustment

In part 5 the current angles φ and θ are taken into account. As soon as φ and θ are not equal zero the thrust vector is not pointing exactly up so that y_h has to be increased to achieve the same lift. For that reason y_h is adjusted in the following way. $y_h = y_h * \frac{1}{\cos\varphi * \cos\theta}$

4.6 Part 6: Thrust Adjustment and Limitation

In this part the prior calculated value y_h is limited to a maximum as well as to a minimum. This is necessary for two reasons. At first a limit is needed due to the fact that there must be enough thrust left in both directions to let the attitude controller work properly and second especially the lower limit is useful to provide the UAV to become instable while sinking too fast due to the downwash of the own rotors. But as described in Part5 the thrust has to be increased while flying with angles not equal zero the maximum limit is increased in the same way.

4.7 Part 7: Calculation of Integral Part

In this part the integral part of the PI-controller is calculated. But in comparison to a classic I-controller the coefficient k_i is not constant. Dependent on $\dot{z}_c$ the coefficient k_i is higher while moving away from z_d and smaller while moving towards z_d. That leads to the fact, that wind changes can be handled fast without the risk of getting an unstable behavior.

4.8 Part 8: Limit Sink Rate

At last the sink rate $\dot{z}_c$ is limited due to the fact that it is not acceptable to lower a given height z_d. But on the other hand it should not take too long to reach the desired height. For that reason the maximum allowed sink rate $\dot{z}_{sr}$ is adjusted concerning the current height difference e_h in the following way. $\dot{z}_{sr} = h * e_h + b$, where h and b are parts of a linear equation, with $h < 0 \; and \; b > 0$. The maximum and minimum sink rate has to be limited to proper values.

As soon the sink rate $\dot{z}_c$ lowers $\dot{z}_{sr}$, y_h will be increased with $y_h = y_h - ((\dot{z}_c + \dot{z}_{sr}) * k_{sr})$.

5 Position Control Algorithm

This section focuses on the 2D position control algorithm. In Fig. 3, a flow diagram of the algorithm can be seen, which is able to cover all requirements described above.

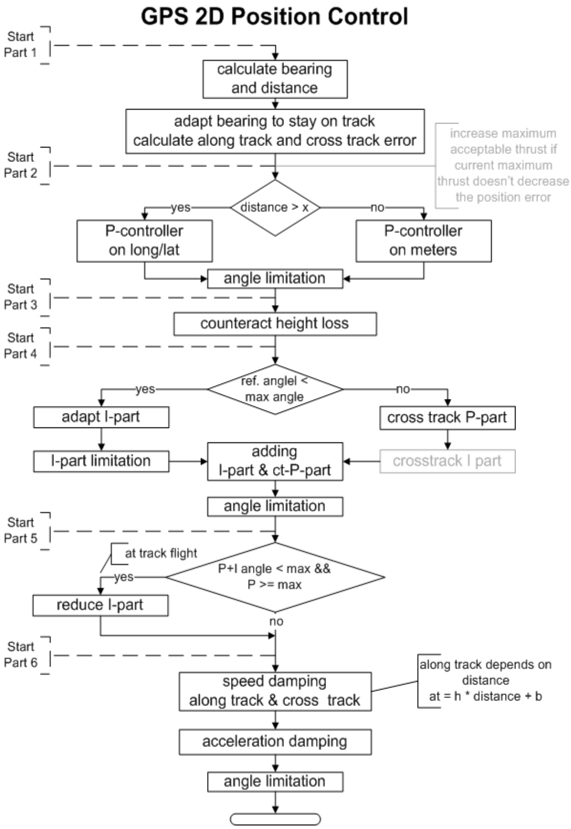

GPS 2D Position Control
Start Part 1
calculate bearing and distance
adapt bearing to stay on track calculate along track and cross track error
increase maximum acceptable thrust if current maximum thrust doesn't decrease the position error
Start Part 2
distance > x
yes
no
P-controller on long/lat
P-controller on meters
angle limitation
Start Part 3
counteract height loss
Start Part 4
ref. anglel < max angle
yes
no
adapt I-part
cross track P-part
I-part limitation
adding I-part & ct-P-part
crosstrack I part
angle limitation
Start Part 5
at track flight
yes
P+I angle < max && P >= max
reduce I-part
no
Start Part 6
speed damping along track & cross track
along track depends on distance at = h * distance + b
acceleration damping
angle limitation

Fig. 3 Flow diagram of position controller.

As described above the maximum thrust that is allowed in order to regulate the altitude is limited, so that there is still a thrust reserve for the attitude controller to be able to work properly. From there on, this maximum allowed thrust determines the maximum velocity the UAV can achieve or the wind the UAV can withstand, along with the maximum allowed attitude angles.

In the following description of the controller the current yaw angle (azimuth angle) of the helicopter has been disregarded. Nevertheless the current yaw angle has to be taken into account when computing the new roll and pitch angles.

The control algorithm can be divided into 6 parts. In this paper only the relevant parts concerning track flight and height control are described. A full description of the controller can be found in [15].

5.1 Part 1: Bearing and Distance Calculation

In the first part the bearing angle ω (with respect to the geographic north) and the distance d (in meters) from the current position to the destination are being calculated. These two values are directly computed from longitudes and latitudes provided by the GPS receiver (see [13]).

5.2 Part 2: Proportional Controller

In the second part the new desired roll and pitch angles φ and θ are calculated. The attitude is computed proportionally out of the distance with respect to bearing. This is the normal way of determining the attitude and was used during test flights.

The last step of Part 2 is the limitation of the attitude angles. The fact that the roll and pitch angles do not exceed a determined maximum value, is a fact that has to be taken care of. If the angles exceed this value they will be scaled down proportionally to the maximum allowed value under preservation of bearing. This maximum value determines the maximum velocity of the UAV but if the maximum angle is too large the available thrust is not sufficient enough to keep the altitude, so a suitable tradeoff has to be found.

5.3 Part 3: Counteract Height Loss

In part 3 a possible height loss is being counteracted. If the current altitude drops below a determined height z_{hl}, the roll and pitch angles, calculated by the proportional part in part 1, are reduced simultaneously with the same factor. The factor increases proportionally regarding the negative altitude error. The result is that the orientation of the UAV becomes more and more horizontal and the altitude increases again. Another effect is that the velocity decreases while adjusting the height error. That leads to the fact that during simulcasts position and height changes the height difference has to be handled preferentially.

It has to be mentioned that the procedure to counteract height loss is done before the integral part is added to the new angles. It can happen that due to wind the integral part was build up significantly. If now the UAV should change its height the integral part should be preserved otherwise the UAV would be moved with the wind until the desired height has been reached.

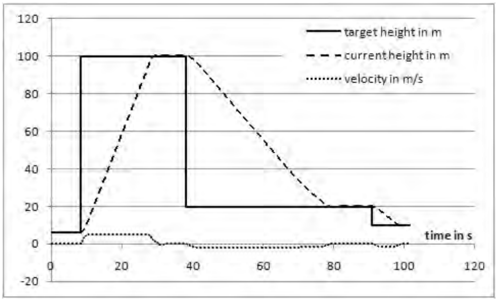
6 Simulations

Several simulations on Matlab/Simulink have been performed prior real flight tests. This chapter presents the results of these simulations. The flight dynamics of the 4- rotor-helicopter has been implemented using Matlab/Simulink [1]. The C-code of the presented control loop was embedded into this simulation environment, so that after a successful validation the microcontroller could be programmed with only minor modification left.

6.1 Altitude Control

In Fig. 4 the result of the height controller without position modification is shown.

Fig. 4 Simulation - altitude control - 100m.



It can be seen that due to the sink rate limitation it takes longer to lose height than to gain height and that the sink rate reduces in the end of descending. In addition the reduction of the sink rate leads to no overshot.

6.2 Track Flight with Altitude Control

In Fig. 5 a simulated flight of a 320 m track is demonstrated.

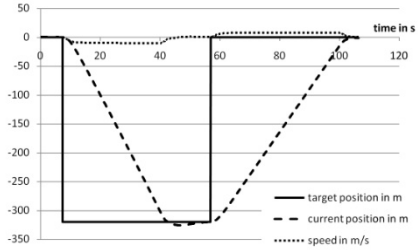


Fig. 5 simulation – 320 m track flight – position.

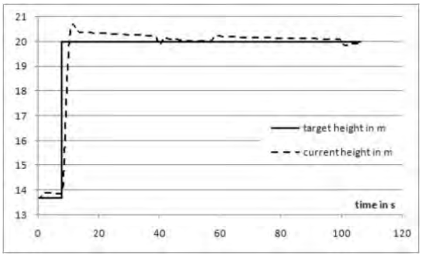


Fig. 6 simulation - 320 m track flight – height.

This simulation was done once before the associated experiment (Fig. 9) to figure out whether the controller would work in general. With a post- inclusion of the weather conditions on the day the experiment took place, a better comparison between simulation and experiment has been won. In Fig. 5 one can now see the simulation within the proper wind conditions. The UAV was programmed to change height and simultaneously to fly 320 m directly southbound and back to the starting point with a constant wind from the north at a speed of 2-3 m/s. As expected the wind leads to increased velocity when flying southwards and to the fact that the UAV overshoots the destination in the south more due to the faster velocity and the wind. In Fig. 6 the corresponding altitude course is demonstrated. It can be seen how the integral part of the controller works and that even during angle modifications the current altitude is nearly constant.

6.3 Track Flight with Simultaneous Altitude Change

In Fig. 7 the resultant trajectory of a simultaneous position and altitude modification is illustrated.

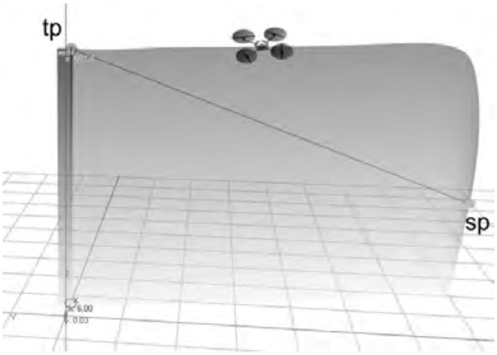


Fig. 7 Simulation - track flight with simultaneous altitude change.

The UAV was supposed to fly from the starting position *sp* (height: 20m) to the target position *tp* (height: 50m and 70m away from *sp*). The behavior that the UAV does not fly directly (white line) to *tp* is grounded in part 3 of the position controller. Here one can see that it is the preferred task of the UAV to hold resp. reach the desired altitude before heading towards *tp*.

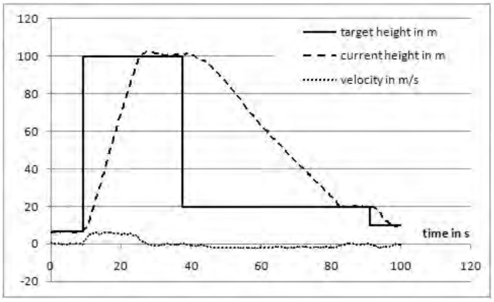
7 Experiments

This chapter provides the results of the experiment. To do so, the already implemented controller in Matlab/Simulink was ported to the ARM7 controller on the UAV and the same test scenarios were flown, as during the simulations. However it must be said that due to the fact that the wind conditions in higher altitudes can badly be measured properly in all places, the actual wind conditions were measured on ground.

7.1 Altitude Control

In Fig. 8 an experiment of the above mentioned 100m altitude simulation is demonstrated.

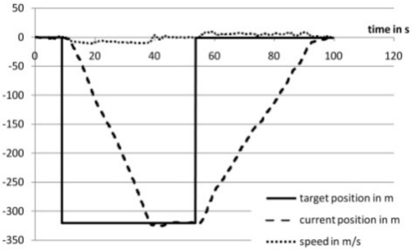
Fig. 8 Experiment - altitude control - 100m.



It can be seen that the result of the experiment matches nearly the result of the simulation. The following experiments were based on this altitude controller.

7.2 Track Flight with Altitude Control

In Fig. 9 the real progress of the same 320 m track flight as described above is shown.



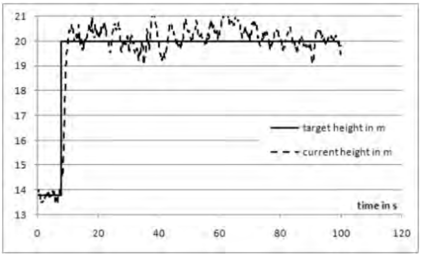


Fig. 9 Experiment – 320 m track flight – position.

Fig. 10 Experiment - 320 m track flight – height.

The chronological sequence of the experiment in comparison to the simulation is nearly the same. The only difference can be found in the fluctuating velocity and height while flying form one waypoint to the other. This is first grounded in blasts and second due to the angle of approach the UAV must take to change position the UAV is pushed down by the airstream.

7.3 Track Flight with Simultaneous Altitude Change

In Fig. 11 the experimental trajectory of a simultaneously altitude and position modification is shown.

As in the simulation it can be seen that the UAV first tries to gain the target height before heading toward the target position. Due to the measured tail wind during the experiment of approximately 3m/s the dent in the trajectory as well as the greater overshooting at the target position compared to the simulation can be explained.

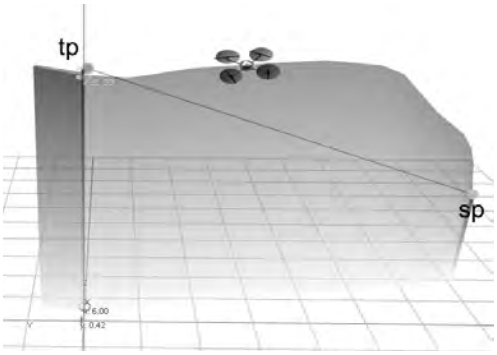


Fig. 11 Experiment - track flight with simultaneous altitude change.

8 Conclusion

In this paper a control algorithm has been introduced that is able to cover all the requirements that are necessary to operate a VTOL UAV. The developed controller were simulated and tested successfully in outdoor experiments. With these controllers the 4 rotor helicopter is able to fly with the maximum possible speed without the risk of losing height during track flight.

The experiments were executed based on the data of the GPS receiver. Nevertheless this algorithm is sensor-agnostic and can be operated with any sensor providing the required data.

9 Outlook

During the simulation and experiments we became aware of some improvements that are already marked as gray descriptions in Fig. 2.

As described in part 3 of the height controller the ground effect has to be handled properly when flying with distances under half rotor diameter to ground. Under this circumstance it is required to add out the increased thrust due to the ground effect to prevent instable behavior.

Another problem could occur if the UAV should simultaneously change its altitude and position and the wind changes significantly during the ascending. Due to part 3 of the position controller the UAV would be moved away with the wind then. But this problem has never become important during experiments and could be counteracted by divide the simultaneous altitude and position modification into two sequent phases.

Acknowledgment. This research activity was performed in cooperation with Rheinmetall Defence Electronics GmbH, Bremen and partly sponsored by the CART (competitive aerial robot technologies) research cluster in Bremen, Germany.

References

1. Kemper, M.: Development of an Indoor Attitude Control and Indoor Navigation System for 4-Rotor-Micro-Helicopter, Dissertation, University of Oldenburg, Germany, February 02 (2007)
2. Wendel, J.: Integrierte Navigationssysteme, Oldenbourg Wissenschaftsverlag GmbH (March 2007)
3. U-blox, A.G.: Essentials of Satellite Navigation, Compendium Doc ID: GPS-X-02007-C (April 2007), `http://www.ublox.com/customersupport/docs/GPS_Compendium GPS-X-02007.pdf`
4. Brockhaus, R.: Flugregelung. Springer, Berlin, 2. Auflage, neubearb. A (July 2001)
5. Rüther-Kindel, W.: Flugmechanik – Grundlagen und stationäre Flugzustände, script ws 2006/2007, `http://www.tfh-wildau.de/ll05/3.Semester/Flugmechanik/`
6. Bouabdallah, S., Murrieri, P., Siegwart, R.: Design and Control of Indoor Micro Quadrotor. In: IEEE International Conference on Robotics and Automation 2004, New Orleans, USA, pp. 4393–4398 (2004)
7. Bouabdallah, S., Noth, A., Siegwart, R.: PID vs LQ Techniques Applied to an Indoor Micro Quadrotor. In : IEEE/RSj -International Conference in Intelligent Robots and Systems, September 28-October 2, Sendai, Japan (2004)
8. Bouabdallah, S., Siegwart, R.: Backstepping and Sliding-mode Techniques Applied to Indoor Micro Quadrotor. In: IEEE-International Conference on Robotics and Automation, Barcelona, Spain (April 2005)
9. Hoffmann, G.M., Huang, H., Waslander, S.L., Tomlin, C.J.: Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment. In: AIAA Guidance, Navigation and Control Conference and Exhibit, 20-23, Hilton Head, South Carolina, AIAA 2007-6461 (August 2007)
10. Tournier, G.P., Valenti, M., How, J.P.: Estimation and Control of a Quadrotor Vehicle Using Monocular Vision and Moiré Patterns. In: AIAA Guidance, Navigation and Control Conference and Exhibit, 21-24 August 2006, Keystone, Colorado, AIAA 2006-6711 (2006)
11. He, R., Prentice, S., Roy, N.: Planning in Information Space for a Quadrotor Helicopter in a GPS-denied Environment. In: IEEE International Conference on Robotics and Automation (ICRA 2008), 19-23 May 2008. Pasadena Conference Center, Pasadena (2008)
12. Meister, O., Mönikes, R., Wendel, J., Frietsch, N., Schlaile, C., Trommer, G.F.: Development of a GPS/INS/MAG navigation system and waypoint navigator for a VTOL UAV. In: SPIE Unmanned Systems Technology IX, Orlando, FL, USA, April 9-12, 2007, vol. 6561, p. 65611D (2007)
13. Hofmann-Wellenhof, B., Wieser, M., Legat, K.: Navigation: Principles of Positioning and Guidance. Springer, Wien (2003)
14. Küke, R., Vyshnevskyy, M.: private communication, Rheinmetall Defence Electronics GmbH, Bremen, Germany
15. Puls, T., Winkelmann, H., Eilers, S., Brucke, M., Hein, A.: Position Control and Waypoint Navigation of a 4 Rotor Helicopter via GPS. In: IEEE International Conference on Robotics and Automation, Kobe, Japan (April 2009) (in review)

Cooperative Autonomous Helicopters for Load Transportation and Environment Perception

Konstantin Kondak, Markus Bernard, Fernando Caballero, Ivan Maza,
and Anibal Ollero

Abstract. This paper summarizes research of the authors on cooperation of autonomous helicopters. Particularly, the paper presents results of the joint load transportation by means of multiple autonomous helicopters that was experimentally demonstrated for the first time. Furthermore, the paper presents the developed cooperative perception system and its experimental validation by tracking of a fireman The paper also includes the description of decision making and mission planning architecture for execution of complex tasks and its validation in experiment with sensor node deployment by using two autonomous helicopters.

1 Introduction

The interest in small-scale UAVs has been steadily increasing for the last ten years. The ability to operate in dangerous environments and reduced system and operation costs are two main reasons for that. Small-scale autonomous helicopters attract attention due to their vertical take off, landing and hovering capabilities. They have excellent manoeuvrability and can get very close to the objects. Due to this characteristic there is a wide range of possible applications from close range observations to the environment manipulation, e.g.: filming and taking still pictures, installation of the communication equipment or sensor nodes, taking probes of the soil. The increased interest to small-scale autonomous helicopters led to several more or less mature one-helicopter-systems in research labs and on the market, see e.g. [13].

Konstantin Kondak · Markus Bernard
Institut für Technische Informatik und Mikroelektronik, Technische Universität Berlin,
Einsteinufer 17, 10587 Berlin, Germany,
e-mail: {kondak,bernard}@cs.tu-berlin.de

Fernando Caballero · Ivan Maza · Anibal Ollero
Robotics, Vision and Control Group, University of Seville, Camino de los Descubrimientos,
41092 Seville, Spain,
e-mail: {caba,imaza,aollero}@cartuja.us.es
Anibal Ollero is also with the Center for Advanced Aerospace Technology (CATEC)

The usage of multiple cooperative autonomous helicopters can have important advantages for the applications and should be considered as a next step in the development of technologies for small-scale UAVs. These advantages result from the possibility to take sensor data readings from different positions at the same time and to increase the load transportation capability by using multiple coupled helicopters.

This overview paper is devoted to main technologies for control, coordination and environment perception for systems composed of multiple autonomous small-scale helicopters. These technologies were developed and implemented by the authors with the focus on disaster managing and filming scenario. The development has been conducted in cooperation between TU Berlin (TUB), AICIA, a tecnology transfer association of the University of Seville, and the Center for Advanced Aerospace Technologies (CATEC) in Spain. The developed technologies have been integrated into a system, which can be configured by the operator for different tasks based on environment perception and load transportation, e.g. sensor node deployment, objects and persons recognition/tracking, close range filming. The number of used helicopters can be configured depending on the mission. This system will be a part of the AWARE-platform [1] which has been developing in the framework of AWARE project founded by European Commission.

In this paper we present main concepts and algorithms as well as the results of real integrated flight experiments. Section 2 is devoted to the new application field of autonomous helicopters: load transportation. In Sect. 3 the cooperative perception is considered. A general framework based on stochastic information filter for perception by multiple sensors of different types is presented. The usage of the framework for perception with multiple autonomous helicopters opens the way for implementation of intelligent sensor moving strategies in order to increase the recognition probability and the state estimation precision of the tracking objects. The decision making and mission planning architecture for execution of complex tasks and its validation in experiment with sensor node deployment by using two autonomous helicopters are presented in Sec. 4. In Sec. 5 conclusions are made.

2 Load Transportation

We have been developing an automatic control system which allows the usage of one or multiple small size helicopters for slung load transportation. The number of helicopters should be configurabl depending on helicopters capabilities and the load to be transported. At the beginning of our research the small size helicopters were considered only as experimental platform to study the slung load transportation problem and to verify the algorithms in real flight experiments before adapting them to full size helicopters. Later, it was realized that even for a slung transportation system using small size helicopters many real world applications exist, e.g. taking a sample of an unknown (possibly toxic or contaminated) fluid and installation of sensor networks in disaster areas.

Theoretical research on load transportation was concentrated on two configurations, using one [8, 15] and two coupled helicopters (so called twin lift configuration) [12]. For the systems with more than two helicopters almost nothing has been published; the authors found only one reference [5] where the modeling of such system is studied.

Due to the complexity of a slung load transportation system and unavoidable simplifications in modeling and system analysis, the validation of the developed control algorithms in real flight experiments is a very important, but unfortunately also a very difficult task. The authors only have knowledge of two (semi) autonomous slung load systems using full size helicopters: The Kaman K-MAX helicopter was modified for fully autonomous operation and used for slung load transportation [15], and a guidance system called "iSLD-IVC" (iMAR Slung Load Damping based on inertial stabilized vision control) developed by iMAR GmbH and the German Aerospace Center DLR. In the last system, the control loop is not directly closed, but the output of the controller is presented to the pilot using an artificial horizon instrument.

For small size UAVs, the group of Aalborg University did interesting research on modeling and control of slung load systems, see e.g. [6, 4]. The work was verified using a single small size helicopter. In their approach a camera is used to observe the motion of the load. Regarding coupled autonomous slung load transportation using two or more full/small size helicopters (at least to the knowledge of the authors) no experiments have been performed by other groups.

2.1 Modeling and Controller Design

For modeling we consider the general case where n helicopters are connected to one load. To get the dynamical equations of motion we used the Kane-method. The appropriate definition of generalized coordinates, speeds and motion constraints allows to generate the equations for systems with an arbitrary number of helicopters. Each additional helicopter adds one motion constraint (the length of the rope does not change during the time). The resulting equations are used for control design as well as for simulation. For more on modeling see [10, 3]. The aerodynamics of the load and ropes are currently neglected, but will be considered in future work for flights with high velocity.

The general scheme of the proposed control algorithm for one or several helicopters coupled with the load is composed of two loops: the outer loop for translation control and inner loop to control the orientation of each helicopter, see Fig. 1.

The input of the control scheme in Fig. 1 is the desired trajectory $\mathbf{x}^*(t)$ for helicopters or for the load. The translational motion of each helicopter is controlled in

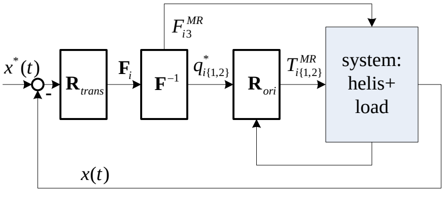


Fig. 1 General control scheme for control of coupled helicopters.

the outer loop by controller $\mathbf{R}_{trans}$. Using the deviations from the desired trajectories the controller $\mathbf{R}_{trans}$ calculates for each helicopter i the forces $\mathbf{F}_i$ which should be generated by its rotors. The helicopter i can realize the force $\mathbf{F}_i$ by adjusting the absolute value of the main rotor lifting force F_{i3}^{MR} and adjusting the orientation of the main rotor plane or fuselage $q^*_{i\{1,2\}}$. The desired orientation of the main rotor plane is controlled in the inner loop by controller $\mathbf{R}_{ori}$. The values F_{i3}^{MR} and $q^*_{i\{1,2\}}$ are calculated using algebraic relations in block $\mathbf{F}^{-1}$.

In practical cases the rope can not be connected to the center of mass of the helicopter and the force in the rope produces torques imposed on the helicopter fuselage. Therefore the influence from the load or from the other helicopters transmitted to the helicopter fuselage by the rope, makes the orientation control a challenging task [9, 10]. The developed approach for control of coupled helicopters is based on following three ideas:

- The design of the orientation controller (for each helicopter) accounts for the complete dynamics (translational and rotational) of the whole system, all coupled helicopters and the load. This is required due to the strong mutual coupling between translation and rotation of each helicopter and the load.
- The design of the translation controller is based on simplified model and accounts only for the translational dynamics of the whole system. In this simplified model the helicopters are modeled as mass points which can produce forces in direction perpendicular to current orientation of the main rotor plain.
- The usage of the force sensor in the ropes simplifies the design of orientation controller and makes it robust against variations of system parameters and disturbances.

The usage of the force sensor in the rope is a key issue in the proposed controller design. The force measured in the rope can be easily recalculated into resultant force and torque acting on the fuselage from the rest of the system (the orientation of the rope is measured by two encoders in the cardan-joint, see Sec. 2.2). These two values are used in the feedback loop of orientation controller. The usage of the force sensor signal in the feedback loop has three main advantages:

1. the closed loop system becomes very robust against variation of system parameters and disturbances
2. the orientation controller for such complicated systems becomes quite simple
3. the orientation controller does not depend on the number of helicopters connected to the load.

As we have shown in [10], the rotation dynamics for a single helicopter, represented by two rigid bodies for the fuselage and the main rotor, can be expressed by the following equations:

$$T_1^{MR} + K_{12}u_2 + K_{11}\dot{u}_1 = 0 \quad (1)$$

$$T_2^{MR} + K_{21}u_1 + K_{22}\dot{u}_2 = 0 \quad (2)$$

Where $T_{1,2}^{MR}$ are the torques generated around the longitudinal and lateral axes of the fuselage, $u_{1,2}$ are rotation speeds of the fuselage and the coefficient K_{xx} are constant parameters of the helicopter and of the main rotor speed. We assume here that the influence of the rotation around the vertical axis (rotation speed u_3) on $u_{1,2}$ is small and can be considered as disturbance. This assumption is true if u_3 is hold on zero or on constant values using additional tail rotor controller with a time constant much smaller compared to the time constant of the orientation controller for $u_{1,2}$. Equations (1, 2) are coupled through $u_{1,2}$. This coupling leads to oscillations (for the parameters of typical small size helicopter) once the system has been stimulated. The scheme for the control of roll and pitch angles $q_{1,2}$ for a single helicopter is shown in Fig. 2, where the gray block denotes the model of helicopter rotational dynamics and kinematics. The controller is composed of blocks $\mathbf{Q}^{-1}$, $\mathbf{D}$ and two feedback loops with gains K_u, K_q for rotation speeds $u_{1,2}$ and orientation angles $q_{1,2}$ respectively. The rotation dynamics described by Eqs. (1), (2) are represented in Fig. 2 by the block $\mathbf{W}$. The block $\mathbf{D}$ of the controller is used to decouple the plant between $T_{1,2}^{MR}$ and $u_{1,2}$. The decoupling can be performed by means of known techniques from linear control theory, e.g. using matrix composed of compensating transfer functions. This orientation controller shows a good performance and robustness in simulation and real flight experiments with different types of helicopters as we have shown in [9, 10].

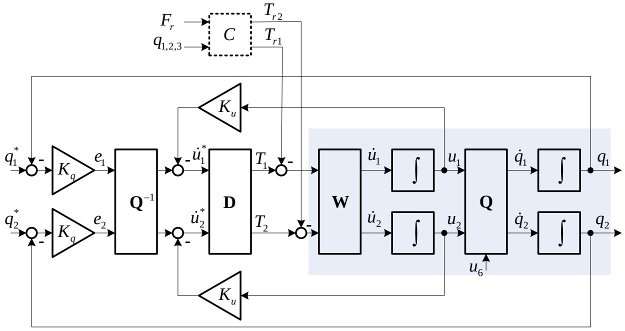


Fig. 2 Scheme for the orientation control.

The rotational dynamics of a helicopter coupled to the load by means of a rope are strongly influenced by the motion of the whole system. To account for this influence block $\mathbf{D}$ should be replaced by the inverse rotational dynamics $\tilde{\mathbf{D}}$ not of a single helicopter, but of the whole system (considering both, the rotation and translation of each helicopter). With this new block $\tilde{\mathbf{D}}$, the orientation controller for helicopter coupled to the load shows equal performance as the orientation controller with block $\mathbf{D}$ for a single helicopter without load in case for nominal values of all system parameters. The simulation experiments have shown that, unlike in the case of a single helicopter, the orientation controller with inversion block $\tilde{\mathbf{D}}$ for coupled helicopter is very sensitive to variation of the system parameters ($5-10\%$ variation could be critical). To overcome this problem, we proposed to use a force sensor in the rope. The force $\mathbf{F}_r$, measured in the rope, will be used to calculate the influence on the rotational dynamics of the coupled helicopters from the remainder of the system. This influence is expressed by means of torque $\mathbf{T}_r = \mathbf{F}_r \times \mathbf{p}_{r-cm}$, where $\mathbf{p}_{r-cm}$ is the position vector connecting rope attaching point r and helicopter CoM. The resulting orientation controller is composed of the orientation controller for a single helicopter and the compensator block **C**, see Fig. 2, where $\mathbf{T}_r$ is calculated and subtracted from torques calculated in **D**. The usage of the compensator **C** allows us to decouple the control of each helicopter from the rest of the system and to use the same controller independent of the number of helicopters coupled together via the ropes. More details of the presented controllers can be found in [2, 3, 9, 10].

2.2 Experimental Results

The load transportation system has been used several times in different experiments, as stand alone system as well as a part of AWARE platform. In Fig. 3 two load transportation tasks are shown. The picture on the left with one helicopter transporting the load was taken in Utrera (Spain), April 2008. The first successful experiment was conducted by the authors in Berlin, November 2007 (http://www.youtube.com/watch?v=J3gmzn0bSa4). The picture on the right with three helicopters transporting the load was taken on our test ground near Berlin, December 2007 (http://www.youtube.com/watch?v=tl6DYWNe9ac). For these flight experiments three identical helicopters were used: take off mass 12.5 kg, 1.8 m main rotor diameter, operated at 1300 rpm. The helicopters are equipped with a multi-UAV modular autopilot system developed in TUB group. The rope is attached to the helicopter by means of the Load Deployment Device (LDD) which is mounted between the landing skids and equipped with two encoders integrated in the cardan-joint for measurement of the rope angles as well as with force sensor to measure the

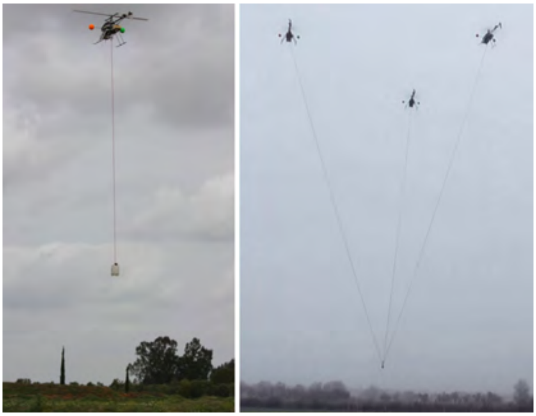


Fig. 3 Single and multiple UAV slung load transportation.

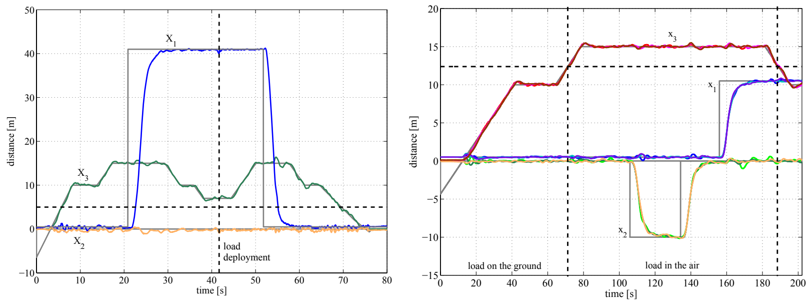


Fig. 4 Motion of the load (1 kg) during transportation using one helicopter (left) and motion of three helicopters transporting one load of 4 kg (right).

rope tension. Due to the equipment attached to the helicopters the pay load of each machine is approx. 1.5 kg.

In the first experiment one helicopter transported a load of 1 kg using a rope of 5 m length. Fig. 4 (left) shows part of the system state during this experiment. All coordinates are given in an ground fixed frame with the origin at the helicopter take-off position. Coordinate x_3 represents the helicopter altitude. Until the load was dropped the coordinates $x_{1,2}$ show the position of the load and after the load dropping, the position of the helicopter itself. The dropping is indicated by a vertical slashed line. The gray line denotes the desired position given to the controller. We consider the performance of the controller as quite good, despite the stormy weather conditions on that day. Even with steady wind of 30 km/h and wind gusts up to 40 km/h the controller was able to stabilize the helicopter and damp upcoming oscillation of the load.

In the second experiment a load of 4 kg was transported by means of three helicopters. In this experiment ropes with a length of 13 m were used. The helicopters were arranged as a equilateral triangle on the ground, with a distance of 8 m between the helicopters. In Fig. 4 (right) the coordinates of all three helicopters during the whole flight are shown. The coordinates of the helicopters are shown in different ground fixed frames, which have the same orientation, but different origins (take-off position of each helicopter), therefore there are no offsets between the helicopter trajectories. The load was lifted when the helicopters reached approximate 12.4 meters. The weight of the load was not considered in the controller and therefore a small disturbance in the x_3 trajectories can be observed at the moment the load was lifted from the ground as well as during strong acceleration in $x_{1,2}$-direction. A position error of each helicopter in hovering was approx. ±0.3 m During the whole flight the triangular formation of the helicopters with a precision of about ±0.3 meters and the load was moved very smoothly.

3 Cooperative Perception

The main purpose of the Perception System (PS) is to build and update a consistent representation of the environment. A fully distributed probabilistic framework has been developed in order to achieve detection and tracking of events using the

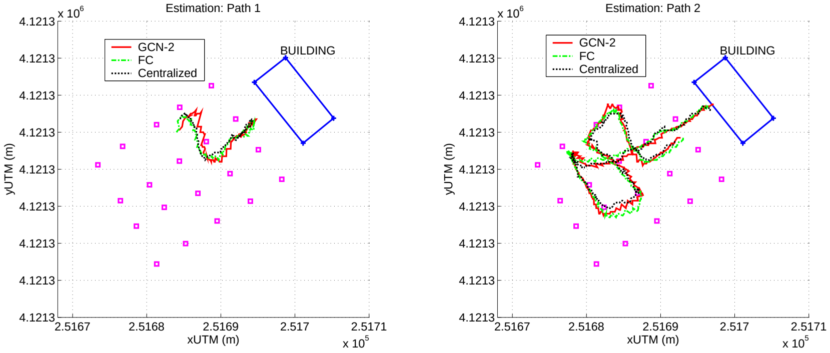


Fig. 5 XY estimation provided by the ground camera (GCN-2) and the helicopter (FC), and centralized estimation. The path is divided in two parts, going (left) and return (right).

sensors provided by the AWARE platform: visual and infrared images from UAVs and ground cameras, scalar measures like temperature, humidity, CO or node signal strength from the Wireless Sensor Network (WSN). It allows reducing the network bandwidth requirements on data transmission and dividing the processing load among different perception nodes. As a result, the scalability of the whole system will be improved. In addition, the ability of separately process the information increases the robustness of the architecture.

Then, the whole PS system is divided in several software instances called perception sub-systems (PSS), each attached to an AWARE platform element such as UAV-cameras (see later Fig. 6), ground cameras, and WSN. Each of them processes locally the environment information (images, sensors, ...) in order to reduce the amount of data transferred through the network. All the PSSs share their beliefs about specific events. The variety of events to be detected and tracked by the PSS is large but a common representation based on a probabilistic representation has been considered [14, 11]. The events are related with the real world by means of their position and velocity in a global coordinate frame. This information is always accompanied by the error estimation represented as an information matrix. In order to disambiguate among different events located in the same place (or close), general information about the event is also included: mean color, histogram, intensity and received signal strength indication (RSSI) when available. Finally, in order to fuse the different estimations, an Information Filter has been applied due to its properties for distributed implementation. Please note that there is no centralized instance for information processing in the system.

3.1 Experimental Results on Cooperative Fireman Tracking with UAVs and Ground Cameras

A fireman tracking experiment was performed using a helicopter and a ground camera. In Fig. 5 the evolution of the estimation provided by each PSS entity, helicopter and ground camera, is shown. Notice how both estimations converge to the centralized computation. Moreover, the estimated standard deviation with respect to the centralized estimation have similar values, so the decentralized approach is working properly.

A further analysis of the data shows an offset of approximately $1m$ in the estimations in both, centralized and decentralized filters. This error is mainly generated by uncertainties in the orientation of the camera in the helicopter. Although the yaw and roll angles given by helicopter are accurately estimated, the pitch angle was manually estimated by using an inclinometer with errors around 3 degrees. These errors are directly translated into errors in the fireman position estimation that leads to the previously mentioned offset.

Finally, it could be interesting to compare these results with the estimations produced by the system when UAVs are not included into the equation. Particularly, we have compared with the estimations provided from the RSSI of the nodes of a wireless sensor network when the fireman carry a node and several nodes are located near the path followed by the fireman. We could see that the experiments with the UAV had smaller errors and the main reason could be the characteristics of the information provided by the UAV with respect to the WSN estimations. In general, if the targets are visible, UAVs will always provide more information than a WSN because, on one hand, UAVs can measure the object from several point of views increasing the triangulation capability and, on the other hand, the information provided by the sensor network, although well bounded, is affected by a considerable noise that can degrade the global estimation. However, WSN can be used when the target is not visible from the cameras, as for example, when the firemen carrying a node enters into a building in which a wireless sensor network has been installed.

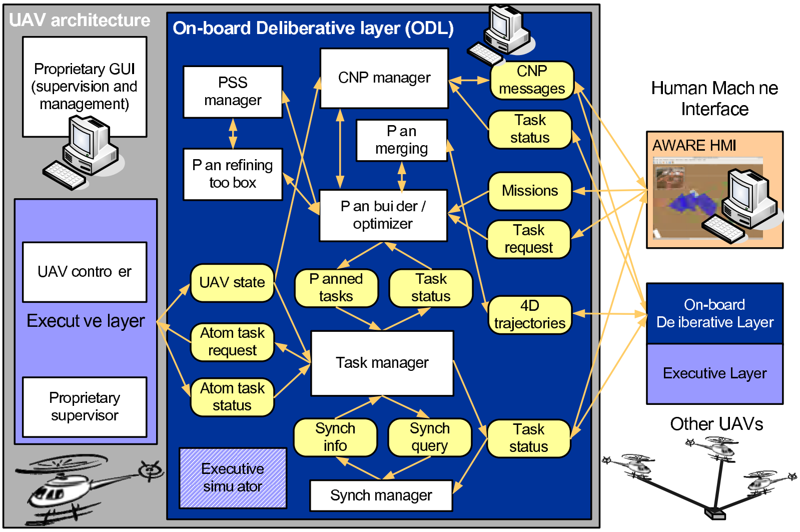

UAV architecture
Proprietary GUI (supervision and management)
UAV controller
Executive layer
Proprietary supervisor
On-board Deliberative layer (ODL)
PSS manager
CNP manager
Plan merging
Plan refining toolbox
Plan builder / optimizer
CNP messages
Task status
Missions
Task request
UAV state
Planned tasks
Task status
4D trajectories
Atom task request
Atom task status
Task manager
Synch info
Synch query
Task status
Executive simulator
Synch manager
Human Machine Interface
AWARE HMI
On-board Deliberative Layer
Executive Layer
Other UAVs

Fig. 6 Global overview of the distributed UAV system architecture.

4 Decision Making Architecture

The AWARE subsystems will have to coordinate and to cooperate to achieve a global mission. The tasks that will be achieved by the AWARE platform involve coordination, mainly for sharing space, and cooperation, for example for the surveillance at different altitudes or from different viewpoints of a same area, or when an UAV plays the role of a radio re-transmitter from/to other UAVs and the central station.

The global picture of the AWARE distributed UAV system based on proposed architecture is shown in Fig. 6. In each UAV, there are two main layers: the On-board Deliberative Layer (ODL) and the proprietary Executive Layer. The former deals with high-level distributed decision-making whereas the latter is in charge of the execution of the tasks. In the interface between both layers, the ODL sends task requests and receives the execution state of each task and the UAV state. For distributed decision-making purposes, interactions among the ODLs of different UAVs are required. Finally, the Human Machine Interface (HMI) software allows the user to specify the missions and tasks to be executed by the platform, and also to monitor the execution state of the tasks and the status of the different sub-systems. It should be mentioned that distributed task allocation is supported by the platform. This feature might be necessary in situations where large numbers of entities have to interact, where direct communication with a central station is not possible, or where the local dynamics of the situation require timely reaction of the UAVs involved in it.

4.1 Multi-UAV Mission Involving Deployment of Sensor Nodes: Experimental Results

The objective of the experiment was to deploy one sensor node from one helicopter (`UAV_2`) in a given location to repair the connectivity in a wireless sensor network [7], whereas a second helicopter (`UAV_1`) supervised the operation with an on-board camera (http://www.aware-project.net/videos/videos.shtml). Then, the AWARE platform user specified the following tasks to be executed:

- `task_1`: deploy a node in the waypoint `wp_1` after `task_2` is completed (the goal was to repair the connectivity of the network deploying a node in `wp_1`).
- `task_2`: visit waypoint `wp_2` to monitor the wireless sensor network area.
- `task_3`: During the node deployment, this operation should be also monitored with the camera from `wp_2`.
- `task_4`: After node deployment, the network area should be monitored again from `wp_2`.

As the user did not allocate those tasks manually, the distributed task allocation process was started from the HMI software. The negotiation involved the CNP manager modules (see Fig. 6) of `UAV_1` and `UAV_2`, and due to the different devices on-board each UAV, `task_1` was allocated to `UAV_2` (bids with infinite cost for the other tasks) whereas the rest of tasks were allocated to `UAV_1` (bids with infinite cost for `task_1`). The CNP manager module operation is based on the Contract Net Protocol, the implementation details for the algorithms are presented in [16].

During the mission, the interface with the executive layer was the task manager module (see Fig. 6) which sent particular tasks to execution. Moreover, the dependencies between tasks of different UAVs are also handled by the task manager with the assistance of the synchronization module. On the other hand, the plan merging modules did not detect any conflict between the planned 4D trajectories of the UAVs.

Finally, it should be mentioned that this experiment represents the first test of the AWARE platform integrating the helicopters and the wireless sensor network in a node deployment mission.

5 Conclusions

The cooperation of multiple autonomous helicopters has many possible applications for detection and monitoring activities as well as for environment manipulation. This paper presents a challenging application: the transportation of a load by means of cooperation of multiple autonomous helicopters. For the first time the load transportation with three coupled helicopters could be demonstrated. Furthermore, the paper presents a fully distributed cooperative perception method based on the application of the Information Filter to integrate measurements from different sensors, e.g. cameras on-board helicopters, cameras on ground and a wireless sensor network. The above skills are integrated into the decision making and planning architecture of the AWARE platform. The application of the proposed methods in a multi-UAV mission involving the autonomous deploying of sensor nodes is also presented including experimental results. On-going work work will be concentrated on increasing on-board intelligence for dealing with unforeseeable events from environment. The new functionalities integrated into AWARE platform will be demonstrated next May 2009 in Utrera, near Seville (Spain).

Acknowledgements. We would like to thank Professor Hommel for supporting this work over all the years.

References

1. AWARE: Platform for autonomous self-deploying and operation of wireless sensor-actuator networks cooperating with aerial objects (2006), `http://www.aware-project.net`
2. Bernard, M., Kondak, K.: Generic slung load transportation system using small size helicopters. In: Proceedings of the IEEE International Conference on Robotics and Automation (accepted for publication, 2009)
3. Bernard, M., Kondak, K., Hommel, G.: Load transportation system based on autonomous small size helicopters. In: 23rd Int. Unmanned Air Vehicle Systems Conf. (2008)

4. Bisgaard, M.: Modeling, estimation, and control ofhelicopter slung load system. Ph.D. thesis, Department of Electronic Systems Section for Automation and Control Aalborg University (2007)
5. Bisgaard, M., Bendtsen, J., Cour-Harbo, A.L.: Modelling of generic slung load system. In: AIAA Modeling and Simulation Technologies Conference and Exhibit (2006)
6. Bisgaard, M., la Cour-Harbo, A., Johnson, E.N., Bendtsen, J.D.: Vision aided state estimator for helicopter slung load system. In: 17th IFAC Symposiumon Automatic Control in Aerospace (2007)
7. Corke, P., Hrabar, S., Peterson, R., Rus, D., Saripalli, S., Sukhatme, G.: Autonomous deployment and repair of a sensor network using an unmanned aerial vehicle. In: Proc. IEEE Int. Conf. Robotics and Automation, New Orleans, April 2004, pp. 3602–3608 (2004)
8. Faille, D., van der Weiden, A.: Robust regulation of a flying crane. In: Proceedings of the 4th IEEE Conference on Control Applications (1995)
9. Kondak, K., Bernard, M., Losse, N., Hommel, G.: Elaborated modeling and control for autonomous small size helicopters. In: ISR/ROBOTIK 2006 Joint conference on robotics (2006)
10. Kondak, K., Bernard, M., Meyer, N., Hommel, G.: Autonomously flying VTOL-robots: Modeling and control. In: IEEE Int. Conf. on Robotics and Automation, pp. 2375–2380 (2007)
11. Merino, L.: A cooperative perception system for multiple unmanned aerial vehicles. Application to the cooperative detection, localization and monitoring of forest fires. Ph.D. thesis, University of Seville (2007)
12. Mittal, M., Prasad, J.V.R.: Three-dimensional modeling and control of a twin-lift helicopter system. Journal of Guidance, Control, and Dynamics 16, 86–95 (1993)
13. Scherer, S., Singh, S., Chamberlain, L., Elgersma, M.: Flying fast and low among obstacles: Methodology and experiments. The International Journal of Robotics Research 27, 549–574 (2008)
14. Sukkarieh, S., Nettleton, E., Kim, J.H., Ridley, M., Goktogan, A., Durrant-Whyte, H.: The ANSER Project: Data Fusion Across Multiple Uninhabited Air Vehicles. The International Journal of Robotics Research 22(7-8), 505–539 (2003)
15. Theodore, C.R., Tischler, M.B., Colbourne, J.D.: Rapid frequency-domain modeling methods for unmanned aerial vehicle flight control applications. Journal of Aircraft 41, 735–743 (2004)
16. Viguria, A., Maza, I., Ollero, A.: SET: An algorithm for distributed multirobot task allocation with dynamic negotiation based on task subsets. In: Proceedings of the IEEE International Conference on Robotics and Automation, Rome, Italy, pp. 3339–3344 (2007)

Robot on the Leash — An Intuitive Inexpensive Interface for Robots Using the Nintendo Wii Remote

Sven Olufs and Markus Vincze

Abstract. This paper presents an easy-to-use interface that takes the robot on a virtual leash: using the Nintendo Wii Remote the user can go towards target places while pointing at the robot. Using the inbuilt infrared camera and accelerometers and a couple of LEDs on the robot, the robot will follow the user. We show how a Particle Filter and an IMM Kalman can be configured such that simple hand gestures with the Wii make the robot follow the user's intention. The robot leash interface has been tested with 12 volunteers who are interested in new technology but have never controlled a robot. The result is that most users could within a few minutes show the robot the first three places in a home environment. Given the little cost of the interface (about $50) the proposed robot leash is a promising human robot interface.

1 Introduction

In this paper we present an inexpensive interface (total hardware costs less than $50) for mobile robots that is easy-to-use and fun-to-use. Instead of considering the robot as a mere piece of technology, we consider the robot as a pet. For instance, we control the robot like a dog, i.e. by a virtual leash to guide it and by using simple gestures. Commands for a robot can be ”stay”, ”follow”, ”bring xy” etc. In practice the recognition of spoken words or the visual detection of the user & gestures is quite hard. We equip the user with dedicated hardware that simplifies the recognition process: the Nintendo Wii Remote.

Sven Olufs · Markus Vincze
Automation and Control Institute, Technische Universität Wien,
Gußhausstraße 27–29 / E376, 1040 Wien, Austria,
e-mail: `{olufs,vincze}@acin.tuwien.ac.at`

The Wii Remote (approx $40) is a quite interesting device for human robot interfaces, because it is inexpensive and equipped with an infrared camera. The main idea is to equip the robot with patterns of infrared beacons (custom build approx $10) that are used to calculate the relative pose to the user. An additional sensor in the Wii allows also tracking the pose when no beacons are visible. If the robot has a basic understanding of the environment and its pose, we can also extend the interface with ”pointing to things” for manipulation.

This paper is organised as follows: The next Section gives a brief overview of the Related Work. This is followed by a description of the Concept of the Interface. In section 4 the Nintendo Wii is presented. Sections 5 and 6 present the technical details of our implementation as well as the Experimental Setup. Experimental results are presented in Section 7. Finally, we discuss the results and the approach in Section 8.

2 Related Work

Our interface is based on the idea that for interaction we identify the position of the user and the direction he/she is pointing. The related literature propose many approaches, for example vision based ones: Waldherr et al. [1] proposed an vision based gesture interface that is able to detect and recognize up to four gestures (Left, Right, Turn, Stop, etc). It is based on colour tracking for human identification and a neuronal network for gesture detection. The user must stand face to face with the robot. The direction of pointing is indirectly obtained using the gestures. Instead of neuronal networks, Isard and Blake [2] used active contours to track hand gestures and the human silhouette with a multi hypothesis system in camera images. In contrast to [1] it is able to detect the orientation of the fingertips, but not to detect the arm of the user.

Another approach is the usage of external devices. The easiest device are buttons or touch screens. In this case, the pose of the user is not important. Recent robot projects like Minerva [3] have shown that such interfaces are suboptimal: Due to their technical nature they were not understood by most of the audiences. Kemp et al. [4] used a laser pointer to point to things that the robot shall grasp. The robot is equipped with an omnidirectional camera with an appropriate filter for the wavelength of laser i.e. only the laser beam is visible in the image. An additional stereo camera detects grasping points on the objects. Corrales [5] used an opportunistic approach like [6]: the user wears a jump suit that is equipped with gyroscopes and accelerometers. The position of the user is known through an external localisation system.

The use of the Wii as robot interface has become quite popular in the last years. Rehm et al. [7] uses the accelerometers for gesture recognition, while Connolly [8] used the accelerometers for robot arm control.Please note that the use of IR-Cameras and beacon is quite popular [9] in the Augmented Reality Community. The method is vice-versa to our approach: Multiple external ir-cameras are used and the user is equipped with beacons. The matching of the beacons is commonly done with the well known 3 point or 4 point algorithm.

3 Concept of the Interface

The main idea of the interface is to guide the robot through the environment on a leash like a dog, i.e. to show it around in an unknown environment: the robot is guided to a position through straining at the leach. The use of the Wii Interface and robot is easy: The user takes a position in front of the robot and points the Wii to the centre of the robot. The relative pose of the user is detected using the infrared beacons on the robot and the Wii's infrared camera. The robot tries to keep a constant distance (1.2m) to the Wii by moving forward or backwards. For instance, a distance of 0.8m will result in a backward motion of the robot (pushing). The robot also tries to keep the angular orientation to Wii. That is, the robot will turn left when the user moves the Wii to the left. This allows the user to guide the robot through the environment by walking "in front" of the robot. Once the environment is learned (or a-priori known) we can assign nodes in the environment and send the robot to the nodes through gestures. The nodes are also learned through gestures. Using nodes simplifies the use of gestures due to the fact that the user does not have to do "precise" pointing all time. Now we use can use another application of the interface i.e. "point to things" that the robot shall manipulate. This is possible because the robot knows its position in the environment and can detect and track the user by means of the Wii remote. Please note that manipulating does not always mean to grasp objects: For instance, if the user points the remote at a lamp, it can be switched on/off, or pointing at TV set can be used to change the channel. To enable this, the robot would dispose of a device to remotely control appropriately euipped power plugs or be integrated into a home automation system.

4 Nintendo Wii Remote

The Nintendo Wii Remote is a commercial product that is used as wireless Bluetooth input device or "game pad" for the Nintendo Wii video game console. Figure1 depicts the Wii Remote. It has been available since spring 2007 worldwide and it is inexpensive. It assumes a one-handed remote control-based design instead of the traditional gamepad controllers. The remote has extra sensors for measuring the relative movement and rotation of the controller in 3D Space via 3-axis accelerometers. The sensor is located in the centre of the remote. Nintendo and AiLive Inc. offer a gestures

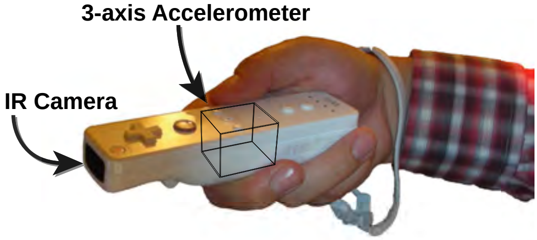


Fig. 1 Wii Remote Sensors.

recognition software development kit that uses SVN and neuronal networks. The remote also features a simple actuator: the ”rumble kit” for tactile feedback. Furthermore, it contains a CMOS ”PixArt Multi-Object Tracking” infrared camera that is able to simultaneously track up to four infrared beacons. Nintendo uses two active infrared beacons to initialise the 3D pose of the controller in a 2D Plane (the beacons are usually placed below the screen connected to the Wii game console). The sensor has a resolution of 80x60 pixels and is able to detect blobs at this resolution.

5 Technical Implementation

Our technical implementation is inspired by the Nintendo Wii video game console: It is based on the concept that the user actually points to the desired position on the screen with the input device itself instead of controlling the console via a joystick or mouse. This is done using pose estimation via triangulation using two active a-priori known beacons. We use a different setup to obtain the position of the controller in 3D by extending the beacon setup itself to 3D. We equip the robot with a-priori known beacons with specific unique pattern on each side of the robot that is easy to distinguish (see Figure 5). We use two kind of infrared beacons: directional light source beacon and quasi point like light source beacon with a angle of beam of 30 degree and 180 degree respectively. Both types of beacons are made of five infrared diodes that are aligned in quasi star like orientation. The quasi point like beacons uses an additional prism to extend the angle of beam. We use this two kinds of beacons due to the Remote can only detect four light sources and a typical infrared diode has a angle of beam approx. 16 degree. Experiments have shown that a mixture of both beacons types improves the overall robustness of the system up to 80 percent.

The pose is estimated by applying a variant of Markov Chain Monte Carlo filters to the data: the Monte Carlo Localisation (MCL). The position of the robot is represented by a set of n weighted particles. Each particle contain a ”believed” position with an assigned probability π. The pose of the robot is estimated by using the observations as a likelihood function of the believed poses/states while MCL attempts to maximise the likelihood of the beliefs. MCL uses the sensor readings of the infrared camera, accelerometers readings and the a-priori known visibility information of each beacons to estimate the pose of the Wii. It is assumed that even a point-like

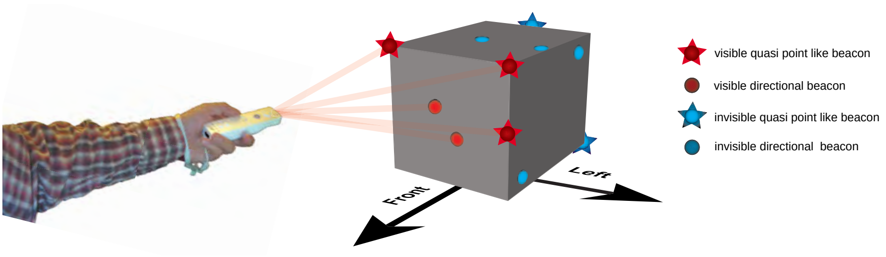

Front
Left
visible quasi point like beacon
visible directional beacon
invisible quasi point like beacon
invisible directional beacon

Fig. 2 Technical Implementation of our Approach.

beacon is not visible through the robot body itself. The MCL framework provides a framework that is able to deal with outliers and provides also a framework for tracking using sensor and motion models i.e. even if no sensor data is available. The Wii remote is able to track and detect up to 4 light sources i.e. so the beacons and other light sources that emits light in the infrared spectrum (normal lamps does). Due to this limitation it is not feasible to use the well known 3 or 4 point algorithm or RANSAC for pose estimation. Experiments have shown that approx 2.1 beacons are visible using the Wii interface; In 31% no (true) beacon was visible in the fiel of view of the camera.

5.1 Monte Carlo Localisation

As all Bayesian filters, MCL methods address the problem of estimating the state x [10] of a *dynamic system*. The implementation requires two things: the motion model and the sensor model. The motion model is used to integrate the *actions u* to the current pose/state while the sensor model integrates the observations. The usual MCL algorithm works recursive in four different stages: (1) first, in the *prediction* stage the motion model is used to integrate the *actions u* to all particles e.g. the particles are simply moved. In the following stage (2) the observations are used to *update* the weight π of the particles. Next (3) the weight of all particles is normalized to one. At last (4) the particles are *resampled* to get the posterior distribution. Technically the resampling discards particles with low weights and moves it to a specific (random) particle with a high weight. In our implementation we move to position of the new ”offspring” particle in respect to the weight of the parent particle i.e. a low weight of the parent particle will result in a relative high translation. The next section gives the used models that are needed to implement MCL.

5.2 Motion Model

The motion model $p(x_t|x_{t-1},u_{t-1})$ represents the effects of action u_{t-1} on the Wii's pose (here x_t) in respect to the pose in the last time step. We expect that the Wii is moved by rotation and translation. Our kinematic model assumes the rotational center in the elbow joint of the user, see Figure 3. This assumption enables us to

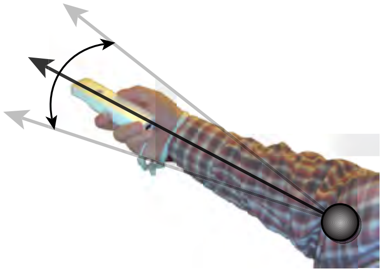


Fig. 3 Wii Remote Kineamtic model.

obtain to rotational speeds of roll, pitch and yawn from the accolometers. In experiments we figured out that this assumption ussally holds true: In 90 % of all cases the remote was used properly. To obtain the rotational and translative speeds some addinal work is needed.

5.2.1 Accelerometer Data Processing

An accelerometer is measuring acceleration *and* gravity induced reaction forces. One can see that the acceleration can be easily obtained if the influence of the gravity is known. In the case the Nintendo Wii the influence of gravity on the remote is not known due to missing sensors which measures rotational speeds e.g. gyroscopes. We use the particle filter system itself to compensate this: The *believed* pose (including orientation) of each hypothesis is used to guess to *guess* the gravity individually for each particle. Due to the theory of particle filter particles [11] with a proper pose (and orientation) will survive in the sample set and populate the resample set. In theory the set will convert to "good" values after several (>3) time iterations if beacon are detected. Finally the acceleration can be obtained simple vector algebra. As next we have to extract the rotational and translative part from the processed acceleration of the Wii. The problem here is that a direct mapping is not possible from a three dimensional vector. For instance the absolute angles of pitch and roll can be easily obtained if the remote is not moved i.e. using the accelerometers output as gravity vector. In the case of a (fast) moving Wii we can only obtain the rotational and translative parts with a certain probability. We use the length of the 3-axis vector **axel** of the raw accelerometers data to calculate the probability τ with a weight function $\tau = gaussianFunction(a|1g - |\mathbf{axel}||, \mu, \sigma)$. The parameters a, μ, σ are learned with ground truth data. Due to spread of standard factory models the learning process is necessary for all new (not yet unused) Wii remotes. Finally we translate the angles of pitch and roll into rotational speed values.

5.2.2 IMM Kalman

We use the standard Interacting multiple model (IMM) Kalman [12] filter as framework for our motion model. The IMM framework provides for multiple models by allowing distinct process noise models for each of the underlying motion models. The extension to extended Kalman filters with IMM is straightforward. We use two independent models in our implementation: a model for a *moving* remote and a model for a *non-moving* Wii, both first order for translation and rotation. Each particle uses own individual IMM Kalman filters instead of a global IMM filter for all particles.

5.3 Sensor Model

The *Sensor Model* is the heart of the MCL. It reflects the probability of measuring y_t if we assume that the state of the Wii is x_t. In our case y_t is the data obtained from the camera sensor. Let m be the number of observations coming from the camera $\Psi = \{\Psi_1, ..., \Psi_m\}$ and k be the number of the a-priori known beacons $\Omega = \{\Omega_1, ..., \Omega_k\}$. Each Ψ reflects a possible sensor reading of Ω so first a data association is needed. Unfortenatly the problem of a optimal association is NP-hard with known solutions like combinatorial optimization approach e.g. MHT (Multiple Hypothesis Tracking) [13, 14] or sequential Bayesian approach e.g. JPDAF (Joint Probabilistic Data Association Filter) [15]. Due to the fact that we use a multi hypothesis tracking system above, it is not needed to track all data associations over time. We use an exhaustive search algorithm ($O(n^2)$) in each time step instead. The algorithm works as follows: First a $\Psi \times \Omega$ matrix of the Euclidian error of all possible combinations of $\Psi_{1...m}$ and $\Omega_{1...k}$ is calculated. The Euclidian error is the minimal distance of the projected view ray of $\Psi_i|x_t$ to the point $\Omega_l|x_t$. Note that $\Omega_l|x_t$ is ∞ if the beacon is not visible from x_t due to occlusion of the robot itself or angle of beam of the beacon itself. Now the algorithm searches the smallest error in the matrix and assigns the sensor data i exclusively to the beacon j: Ψ_i^j. We use the following model to calculate the likelihood: $p(y_t|x_t) = \sum_{i=0}^{m} p(\Psi_i^j|x_t)p(\Omega_j|x_t)$. Here it is also assumed that all observations are independent from each other. Here $p(\Psi_i|x_t)$ calculates the probability of every observation with respect to the believed pose (of the particles). In contrast to the approach in literature [16] we are summing up the probabilities instead of multiplying them. This enables a lower sensitivity of (few) ”bad” observations to the probability. $p(\Omega_j|x_t)$ reflects the probability that the beacon Ω_j is visible from an specific angle i.e. x_t. We use a gaussian weight function using the angle depending on the type of beacon (quasi point or directional). The likelihood function $p(\Psi_i|x_t)$ is defined as $p(\Psi_i|x_t) = \frac{c^2}{c^2+B(\Psi_i^j,x_t)^2}$. The function is very similar to the squared error function for errors $B(\Psi_i^j, x_t) \leq c$ and is bounded above by a constant for larger errors, thus the influence of outliers onto the estimate is bounded. We do not use a squared error function due to it is not robust with respect to outliers. In our implementation we choose $c \approx 50$. B is a function which returns the Euclidian error from the matrix of Ψ_i^j.

5.4 Initialisation and Pose Estimation

For initialisation the user has to point the Wii remote steady to the centre of the robot. This ensures that sensor data is available and reduces the state space during initialisation. Yaw is zero and roll and pitch can be read out of the accelerometers. The initialisation of the system itself is done in good old Monte Carlo fashion: All hypothesis are randomly distributed in the state space on the x/y/z axis. In analogy to traditional Monte Carlo approaches our pose is obtained by building the weighted average of all particles. In the case of no available sensor information of the camera the motion model is only applied to the particles. Here the particles are just moved and not resampled. Due to drift of the sensors we use an additional damping factor of 95% for the motion model if no camera data is available.

6 Experimental Setup

For our experiments we use a quasi-holonomic mobile robot (Fig. 4) ”James”. With a payload of 150 kg, the platform has a maximum speed of 5km/h. We mounted four quasi point-like ir- beacons and three directional ir- beacons on James. All beacons are visible if the remote is places in front of the robot and points to its centre. The quasi point-like on top of the setup are 170 degrees visible from the front of the robot. This is due to construction issues of the robot itself. The other quasi points like beacons on bottom are almost visible from all sides of the robot. The MCL prototype is implemented in Matlab on a Laptop with 1.6Ghz Pentium M. and runs with 5Hz with 40 particles in the normal use and 200 particles in the initialisation phase.

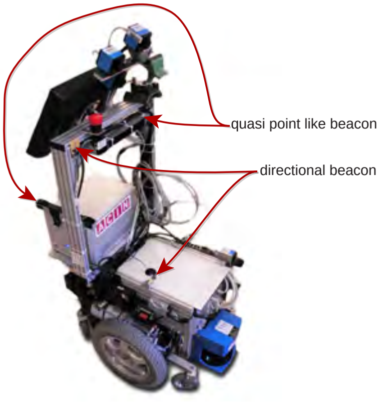


Fig. 4 Implemention of the interface on our robot James.

7 User Trails and Results

The main concept of the evaluation is to let a volunteer to guide the robot to three specific spots in a test living room (Fig. 5) in a specific order (1 to 2, 2 to 3 and 3 to 1) as fast as possible using the Wii interface. After a short instruction the volunteer had to guide the robot without any help of the instructor. For the sake of simplification we place the robot at spot no 1 as start spot for all trails. The environment itself and spots are identical for all volunteers. The test environment consists at of two rooms with a driveable area of $15m^2$. The room on the left-hand side is a typical living room with at two seating-accommodation; The right room is a workroom. The room layout offers enough space for three non overlapping ”spots of interest”: 1) A start spot that is easy to access[1]. 2) A second spot[2] that is difficult to reach and to leave. 3) A third spot[3] that is close to solid furniture and provides a (partial) different type of ground surface e.g. carpet.

---

[1] min 1.5m to the margins.

[2] 0.2m - 0.3m to the margins left and right, min 1.5m to the front.

[3] max 0.3m to the furniture in front, 0.5m-1m left and right.

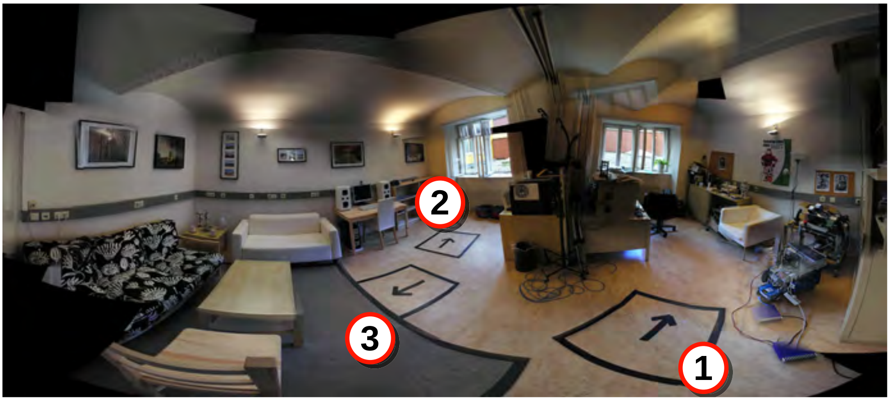


Fig. 5 Panoramic view of the used test environment.

The different margins of each spot are defined on purpose. The first spot is easy to give the volunteer a chance to get a ”feeling” for the interface and robot. The second spot requires advanced navigation skills to move the robot through corridor. The narrowness of the corridor has also another purpose: It is too tight to fit a human and robot at the same time. This prevents the volunteer to pass the robot while it’s moving though the corridor. As a side effect the volunteer has only two chances1 to leave the place with the robot: The robot must be turned 180 degree or ”pushed back”. The idea behind spot three is similar to the previous one: The place does not offer enough space in front of the robot for the volunteer to reach the spot. At least 1.2m free space to the furniture would be needed to reach the spot. Please note that the user must be in front of the robot. It is on purpose that the user has to ”move around / move behind” the furniture first. In our setup we used a small coffee table on a carpet. The alignment of the carpet relative to the spot is again on purpose: The robot has to turn to the left to reach the start position; it is likely that at least one wheel1 runs on the carpet while the other wheels still run on the regular ground. In our setup the carpet provides a much better grip than the (slippery) normal ground.

We asked a group of 12 volunteers to use the Wii interface for evaluation. The mean age of the group is 30 years old (i.e. 21-38). Almost all volunteers consider themselves ”open to new technologies”. First, every volunteer was instructed who to use the interface and the robot, by the typical learning-by-doing approach. Then the user had to control the robot to all spots in the specific order 1 to 2,2 to 3 and 3 to 1 with no help from the instructor. All volunteers were able to guide the robot through the environment with ”almost” no touching of the environment during the first run. Table 1 shows the time for the fastest run per volunteer. The time for a touch panel and joystick interface are provided for the sake of completeness. One can see that the average time is in good company with the touch panel and joystick interface. As expected the corridor of spot no. 2 was the most difficult part of the environment. No user tried to ”push back” the robot to leave the spot and move to

Table 1 Results of the User Trails. The fastest run per volunteer is counted.

User No	time (s)	User No	time (s)	User No	time (s)	User No	time (s)
1	574	4	267	7	130	10	177
2	153	5	118	8	129	11	92
3	188	6	255	9	80	12	86

Wii Interface Average time	187s
Touchpanel Interface Average time	120s
Joystick Interface Average time	192s

spot no 3. All tried to turn the robot ”on-spot”. The two different types of ground was no problem for the most users. In fact all users were surprised that the robot reacts total different with two types of ground with different grip (see above). Two volunteers underestimated the speed of the robot and hit slightly the corridor of spot no. 2. 10 of 12 liked to interface, the rest was unsure. The interface was well accepted by 11 of 12 users.

8 Conclusions

In this paper we presented an approach for an inexpensive interface for Robots using the Nintendo Wii Remote. It is build from inexpensive hardware i.e. the Nintendo Wii Controller and a couple of infrared LEDs. The needed computational power for computation is manageable and adjustable trough the number of hypothesis. First implementations of the leach (the robot & Wii was connected through a leach) have shown that a real leach is inefficient: The leach was almost wrapping the robot after a short amount of time. It turned out that a virtual-leach is much easer to handle, but it is difficult to understand. One user complained the missing haptics-feedback of the virtual leach compared to a real leach which is a clear drawback of our approach. Another drawback is the limited precision of the accelerometers inside the Wii. Slow motions ($< 1.5\frac{cm}{s}$) are not recognized by these sensors. We plan to extend our approach by the usage of cyclic switch on/off ir-beacons running at different a-priori known frequencies e.g. 1Hz and 0.33Hz to improve the robustness to false positives.

Acknowledgments. The research leading to these results has received funding from the European Community’s Sixth Framework Programme (FP6/2003-2006) under grant agreement FP6-2006-IST-6-045350 (robotshome).

References

1. Waldherr, S., Romero, R., Thrun, S.: A gesture based interface for human-robot interaction. Autonomous Robots 9, 151–173 (2000)
2. Isard, M., Blake, A.: ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework. In: Burkhardt, H.-J., Neumann, B. (eds.) ECCV 1998. LNCS, vol. 1406, pp. 893–908. Springer, Heidelberg (1998)

3. Thrun, S., Bennewitz, M., Burgard, W., Cremers, A., Dellaert, F., Fox, F., Hähnel, D., Lakemeyer, G., Rosenberg, C., Roy, N., Schulte, J., Schulz, D., Steiner, W.: Experiences with two deployed interactive tour-guide robots. In: Proceedings of the International Conference on Field and Service Robotics, Pittsburgh, PA (1999)
4. Kemp, C., Anderson, C., Nguyen, H., Trevor, A., Xu, Z.: A point-and-click interface for the real world: laser designation of objects for mobile manipulation. In: Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction, HRI (2008)
5. Corrales, J., Candelas, F., Torres, F.: Hybrid tracking of human operators using imu/uwb data fusion by a kalman filter. In: Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction, HRI (2008)
6. Mayol, W., Davison, A.J., Tordoff, B., Molton, N., Murray, D.W.: Interaction between hand and wearable camera in 2d and 3d environments. In: Proceedings of British Machine Vision Conference, BMVC (2004)
7. Rehm, M., Bee, N., André, E.: Wave like an egyptian – accelerometer based gesture recognition for culture specific interactions. In: Procedings of HCI 2008 Culture, Creativity, Interaction (2008)
8. Connolly, C.: Kuka robotics open architecture allows wireless control. Industrial Robot: An International Journal 35, 12–15 (2008)
9. Pintaric, T., Kaufmann, H.: Affordable infrared-optical pose tracking for virtual and augmented reality. In: IEEE VR Workshop on Trends and Issues in Tracking for Virtual Environments (2007)
10. Thrun, S., Fox, D., Burgard, W., Dellaert, F.: Robust monte carlo localization for mobile robots. Artificial Intelligence 128, 99–141 (2000)
11. Crisan, D.: Particle filters - a theoretical persperctive. In: Doucet, A., Freitag, N., Gordon, N. (eds.) Sequential Monte Carlo Methods in Parctise, pp. 17–41. Springer, Heidelberg (2001)
12. Blom, H., Bar-Shalom, Y.: The interacting multiple model algorithm for systems with markovian switching coefficients. IEEE Transactions on Automatic Control 33, 780–783 (1988)
13. Reid, D.: An algorithm for tracking multiple targets. IEEE Transaction on Automatic Control 24(6), 843–854 (1979)
14. Bar-Shalom, Y., Fortmann, T.: Tracking and data association, 1st edn., Boston. Mathematics in science and engineering, vol. 179 (1988)
15. Schulz, D., Burgard, W., Fox, D., Cremers, A.: People tracking with a mobile robot using sample-based joint probabilistic data association filters. International Journal of Robotics Research, IJRR (2003)
16. Thrun, S., Fox, D., Burgard, W.: Monte carlo localization with mixture proposal distribution. In: AAAI/IAAI, pp. 859–865 (2000)

Robot Basketball: Ball Dribbling — A Modified Juggling Task

Georg Bätz, Marion Sobotka, Dirk Wollherr, and Martin Buss

Abstract. Ball dribbling is a central element of basketball. One main challenge for realizing basketball robots is to achieve the stability of the periodic dribbling task. We show that the dribbling problem is closely related to robot juggling problems that are well-studied. To this end, the paper introduces a hybrid (discrete-continuous) dynamical model for ball dribbling that provides the basis for the design of reference trajectories and controllers. Furthermore the paper discusses local stability and parameter sensitivity in particular in comparison to juggling. Theoretical results are experimentally validated for ball dribbling using an industrial robot. Force/torque-based tracking and vision-based tracking are compared. For both tracking approaches, dribbling for multiple cycles is achieved. The vision-based approach performs better as compared to the force/torque-based approach, in particular for imprecise estimates of the coefficient of restitution.

1 Introduction

The control of rhythmic tasks has been a very active research area over the last 25 years. Besides robotic walking, there are also numerous works that investigated other rhythmic tasks, which are commonly summarized with the term *juggling*. Buehler *et al.* [1] coined the term for tasks that require interaction with an object or multiple objects that would otherwise fall freely in the earths gravitational field. These tasks generally constitute nontrivial dynamic environments. Their dynamic properties change intermittently subject to excitation from the robot. For juggling tasks, the continuous motion of the actuator is used to control the continuous motion of the ball through an intermittent contact.

Although the dribbling task is, to some extent, comparable to the extensively studied *classic* juggling task, there are significant differences concerning feasible

---

Georg Bätz · Marion Sobotka · Dirk Wollherr · Martin Buss
Institute of Automatic Control Engineering, Technische Universität München,
Theresienstraße 90, 80290 Munich, Germany,
e-mail: `{georg.baetz,marion.sobotka,dw,mb}@tum.de`

period-one motions, local stability, sensitivity towards parameter variations, and the dribbling task is an interesting example of a rhythmic task to be studied. The remainder of Sec. 1 gives an overview on research in robotic juggling and summarizes the contributions of this paper.

Holmes was the first to investigate the motion of a ball bouncing vertically on a sinusoidally vibrating table [2]. He found that for sufficiently large excitation velocities and a coefficient of restitution close to one, the system exhibits a variety of irregular non-periodic motions in addition to the expected harmonic motions. Bapat and coworkers [3] followed up the work of Holmes, focusing on the sinusoidal motions which led to stable periodic collisions. Their results showed that with sensorless sinusoidal actuation, the actuator acceleration at impact has to be *negative* in order to obtain a stable period-one orbit.

Buehler *et al.* investigated the stabilization of juggling tasks [1, 4, 5]. In their work they considered the trajectory control of a puck, which is constrained to lie on a plane turned into the earth's gravitational field. As control input they used a bar which is actuated around a revolute joint. They proposed and experimentally verified the mirror algorithm to control the system [1]. It should be noted that the mirror algorithm requires permanent tracking of the ball. Rizzi and Koditschek successfully applied the mirror law to spatial juggling of two balls with a three DoF robot [6, 7, 8]. Following the work of Buehler *et al.*, Lynch and Brogliato have proposed stabilizing feedback laws for the planar juggler [9, 10].

Atkeson, Schaal, and coworkers investigated robotic juggling with emphasis on the aspect of *learning* control [11, 12, 13] and as benchmark model for neuroscience studies [14, 15]. Atkeson and Aboaf demonstrated in [11] that task-level learning improves the performance of a juggling robot. The system practiced the task, monitored its own performance and adjusted its aim to perform the task for more cycles. Schaal and Atkeson presented an open-loop sinusoidal actuation to obtain stable fixed points in the juggling task [12]. In their experimental setup, they used a one-joint robot with pantograph linkage to maintain a horizontal paddle orientation. In [13, 14], Schaal and Atkeson further explored the implementation of robot learning for the juggling task. Based on their earlier results they successfully predicted the human behavior for the task [15].

Sepulchre and Gerard [16] introduced the stabilization problem of dynamical billiard in the plane similar to the one of Lehtitet and Miller [17]. Dynamical billiards are part of the mathematical theory of particle trajectories within a closed reflective boundary. The system consists of a point mass moving in the plane under the influence of a constant gravitational force and colliding elastically with two intersecting edges which can be actuated to rotate around their intersection point. Ronsse and Sepulchre studied different types of actuation and investigated the importance of timing the control feedback [18, 19, 20]. They also investigated feedback control strategies for the bouncing ball system and discussed the influence of the impact acceleration on the robustness of the system to parameter uncertainties [21].

An example of robot dribbling was presented by Shiokata *et al.* for experimental evaluation of a high-speed vision system [22]. For the experiment, a ping-pong ball and an actuator with four DoF were used. In their experiments, the dribbling motion could only be maintained for a few cycles (appr. 1 s).

Introducing dribbling as a modified juggling task, this paper presents the following contributions: First, a hybrid system model for the dribbling task is derived. The model provides the basis for the theoretical analysis of the task. Second, the control design and the selection of feasible reference trajectories to generate a periodic dribbling motion are discussed. The discussion is based on the existing solutions for the juggling task. Third, a comparison between the dribbling task and the *classic* juggling task is presented. The comparison considers feasible period-one motions, local stability and sensitivity to parameter variations. Fourth, an experimental evaluation of the control design three dimensional space is presented. For the experiments, two different ball tracking modalities have been considered.

The remainder of the paper is organized as follows: In Sec. 2, the hybrid system model is derived. The control design and the selection of reference trajectories are discussed in Sec. 3. The comparison between the dribbling task and the *classic* juggling task is presented in Sec. 4. The experimental evaluation of the dribbling task is shown in Sec. 5. Finally, a conclusion and outlook can be found in Sec. 6.

2 System Model

This section introduces the model of the dribbling system. The model will provide the basis for the theoretical analysis in Sec. 4. As a reference, Fig. 1 shows the notations which are used in the paper. For the derivation of the system model the following assumptions were made: First, the ball is considered as a mass point. Second, an inelastic collision model with instantaneous transitions, described by the coefficient of restitution, is used [23]. Third, rotational ball velocity is negligible. Fourth, the model considers only motion along the vertical z-axis. For modeling, the hybrid state model (HSM) approach is used as it offers a framework for systems with discontinuous trajectories and discrete states [24]. The hybrid state vector $\varsigma(t)$ is composed of the continous state $\boldsymbol{x}(t) \in \mathbb{R}^n$ and the discrete state $x_d(t) \in \mathbb{Z}$,

$$\varsigma(t) = \big(\boldsymbol{x}(t), x_d(t)\big)^T \in \mathbb{R}^n \times \mathbb{Z}. \tag{1}$$

External inputs are divided into continous $\boldsymbol{u}(t) \in \mathbb{R}^m$ and discrete $u_d(t) \in \mathbb{Z}$ control inputs. For a constant discrete state x_d, the system is modeled by ordinary differential equations $\dot{\boldsymbol{x}} = f_{x_d}(\boldsymbol{x}, \boldsymbol{u}, t)$. Transition surfaces $s_i\left(\boldsymbol{x}, \boldsymbol{u}, x_d\right) = 0$ determine the occurence of a discrete event and jump maps $\boldsymbol{\varphi}_i\left(\boldsymbol{x}, \boldsymbol{u}, x_d, t^-\right)$ determine

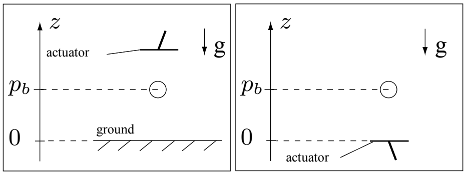


Fig. 1 Notations and coordinate axis for the dribbling task (left) and the *classic* juggling task (right).

the hybrid state ς^+ immediately after the event. A detailed description of this hybrid modeling framework can be found in [25].

The hybrid state model is now used to derive the system model for the dribbling task. The state $\boldsymbol{x}(t) = (p_b, v_b)^T$ summarizes position and velocity of the ball. The state of the actuator $\boldsymbol{u}(t) = (p_a, v_a, a_a)^T$ summarizes position, velocity and acceleration of the actuator. The actuator state is denoted by $\boldsymbol{u}(t)$ as it will later be considered as control input. Due to the assumption of instantaneous impacts, there is only one discrete state $x_d(t) = 0$ denoting that the ball is in free flight. Collisions with the actuator (ground) occur if the corresponding transition equation s_a (s_g) has a zero crossing. Then, the velocity and the discrete state are reset

$$\begin{aligned} s_a(\boldsymbol{x}, \boldsymbol{u}) &= p_b - p_a \qquad & \varsigma^+ &= \boldsymbol{\varphi}_a\left(\varsigma^-, \boldsymbol{u}\right) = \left(p_b^-,\, -c_{r,a} v_b^- + (1 + c_{r,a})\, v_a,\, 0\right)^T \\ s_g(\boldsymbol{x}) &= p_b & \varsigma^+ &= \boldsymbol{\varphi}_g\left(\varsigma^-\right) = \left(p_b^-,\, -c_{r,g} v_b^-,\, 0\right)^T \end{aligned} \tag{2}$$

where $c_{r,a}$ denotes the coefficient of restitution for the actuator impact and $c_{r,g}$ the coefficient of restitution for the ground impact. For the free flight, the continuous dynamics are described by the gravitational acceleration $g = 9.81$ m/s^2 with

$$\dot{\boldsymbol{x}} = \boldsymbol{f}_0(\boldsymbol{x}) = (v_b, -g)^T . \tag{3}$$

The transition graph for the ball dribbling setup is depicted in Fig. 2. For the *classic* juggling task, the continous dynamics are also described by (3). In contrast to the dribbling task there is only the transition map $\boldsymbol{\varphi}_a\left(\varsigma^-, \boldsymbol{u}\right)$ which describes impacts with the actuator, see (2). Based on the model of the dribbling task, the first-return map of the system can be derived. The state of the ball immediately after the impact with the actuator $\boldsymbol{x}(t_{k+1}) = \boldsymbol{x}[k+1]$ as a function of the state in the previous period $\boldsymbol{x}(t_k) = \boldsymbol{x}[k]$ and the impact times $t[k]$ and $t[k+1]$ is given by

$$\begin{aligned} p_b[k+1] &= p_b[k] + v_{b,gi}\left(t[k+1] - t_{li}\right) - \frac{g}{2}\left(t[k+1] - t_{li}\right)^2 \\ v_b[k+1] &= -c_{r,a}\left(v_{b,gi} - g\left(t[k+1] - t_{gi}\right)\right) + (1 + c_{r,a})\, v_a[k+1] \end{aligned} \tag{4}$$

with the abbreviations $v_{b,gi}$ for the ball velocity immediately after the ground impact and t_{gi} for the time of the ground impact

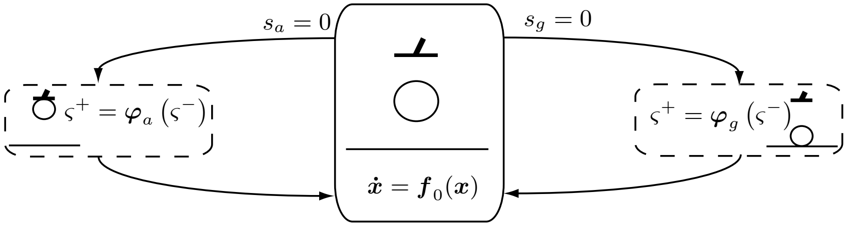

$s_a = 0$
$s_g = 0$
$\varsigma^+ = \boldsymbol{\varphi}_a\left(\varsigma^-\right)$
$\varsigma^+ = \boldsymbol{\varphi}_g\left(\varsigma^-\right)$
$\dot{\boldsymbol{x}} = \boldsymbol{f}_0(\boldsymbol{x})$

Fig. 2 Transition graph for the ball dribbling setup: The solid box represents the discrete state *free flight*, the dashed boxes represent the jump maps for the collisions with the actuator and the ground respectively.

$$v_{b,gi} = c_{r,g}\sqrt{v_b^2[k] + 2gp_b[k]}, \quad t_{gi} = t[k] + \frac{1}{g}\left(v_b[k] + \sqrt{v_b^2[k] + 2gp_b[k]}\right). \quad (5)$$

It should be noted that impact time $t[k+1]$, position $p_b[k+1]$ and velocity $v_b[k+1]$ are influenced by the control law. The first-return map (4) provides the basis for the theoretical analysis in Sec. 4.

3 Control Design

The control structure has two main modules: ball tracking and robot trajectory generation. In the first module, tracking of the ball is performed either force/torque or vision based, see [26]. In the second module, a continuous time reference trajectory for the actuator is generated based on the tracked ball position. For successful dribbling, the generated trajectory has to result in a periodic dribbling motion. Such periodic solutions are characterized by fixed points in the state space of the first-return map (4).

For a period-one motion, the steady-state solutions of the system (4) are characterized by a constant ball position p_b^* and velocity v_b^* after impact with the actuator and a constant period T^* between two actuator impacts:

$$p_b^* = p_b[k+1] = p_b[k] = h^* \qquad v_b^* = v_b[k+1] = v_b[k] \quad (6)$$

where $h^* = p_b^*$ is the dribbling height.

For the classic juggling task, different solutions have been reported in literature to realize stable period-one motions: A sinusoidal trajectory

$$p_a(t) = A\sin(\omega t + \phi_0) + p_0 \quad (7)$$

has been proposed, which does not require any sensory feedback [2, 12]. A mirror law which requires permanent tracking of the ball,

$$p_a(t) = (-(1-c_r)/(1+c_r) - \kappa\,(E^* - E(t)))\,p_b(t) \quad (8)$$

has been discussed in [1, 5, 6, 7, 8]. Here, E^* ($E(t)$) denotes the desired (actual) ball energy. The mirror law is not dynamically feasible for the dribbling task as the discontinuity of the ball velocity at ground impact would require infinitely large joint torques. A trajectory based on the state-feedback

$$p_a(t) = \ p_a[k+1] + v_a[k+1]\,(t - t[k+1]) + \frac{a_a[k+1]}{2}\,(t - t[k+1])^2 \quad (9)$$

for $t[k] < t < t[k+1]$ has been proposed in [20, 27]. For the three controls, end effector position p_a^* and velocity v_a^* at impact are determined by the desired period-one motion. The end effector acceleration, in contrast, is a free parameter influencing the local stability, see Sec. 4.2. A reference trajectory analog to (9) is used for the dribbling task as it represents a dynamically feasible reference trajectory and offers a direct adjustment for a_a.

For the experiments, the spatial character of the task is considered: Based on the tracked ball position $\boldsymbol{p}_{b,act}(t)$, the xy-position $p_{x,a}(t)$, $p_{y,a}(t)$ and the orientation $\boldsymbol{o}_a(t)$ of the actuator are adjusted to compensate deviations $e_x(t)$ and $e_y(t)$ from the desired horizontal position [26].

4 A comparison of the Classic Juggling and the Dribbling Task

The comparison of the classic juggling and the dribbling task considers feasible period-one motions (Sec. 4.1), local stability (Sec. 4.2), and sensitivity to parameter variations (Sec. 4.3) for both tasks.

4.1 Feasible Period-One Motions

For the juggling task, the actuation impact occurs at $z = 0$ whereas for the dribbling task, the actuation impact occurs at $z = h$ and an additional, autonomous impact occurs at $z = 0$. Consequently, for the juggling task, the ball trajectory of a period-one motion is unambiguously defined by its energy level E^*. For the dribbling task, in contrast, an infinite amount of feasible period-one motions exist for a given energy level E^* because of the additional second impact. By varying the dribbling height h, the ratio between potential and kinetic energy at impact with the actuator can be changed and can thus influence the resulting motion of a certain energy level. This characteristic is illustrated in Fig. 3 where feasible period-one motions for the two tasks are depicted for a fixed energy level. Due to the energy loss on the ground, a period-one motion for the dribbling task is characterized by two energy levels: $E^{*\prime}$ and E^*. However, as these energies are related through $E^* = c_{r,g}^2 E^{*\prime}$, one energy suffices to describe the energy level of the motion. In the following, the energy after the ground impact E^* will be used.

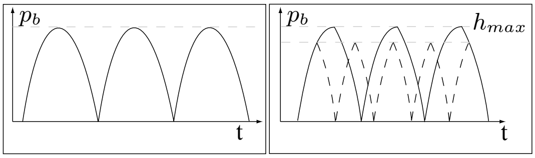


Fig. 3 Period-one motions of the ball for a fixed energy level E^*: the unique solution for the juggling task (left) and two feasible solutions for the dribbling task (right).

4.2 Local Stability

The stability of periodic orbits through impact control can be rephrased as the fixed point discrete-time stabilization of the corresponding Poincaré map. For the dribbling task, the Poincaré map is given in (4). The difference equation has an asymptotically stable fixed point $\boldsymbol{x}^*$ if all eigenvalues of the linearization are strictly within the unit circle [28]. A quantitative comparsion of periodic solutions at different energy levels can be realized through topologic orbital equivalence (TOE) [15]. The scaling relation

$$d := \left\{p_b = e^2 p_b'; v_b = e v_b'; v_a = e v_a'\right\} \tag{10}$$

fulfills the requirements of TOE for (4) (e $>$ 0). With this equivalence, it is sufficient to compare the local stability of the tasks for one common energy level E^*. This leads to a distinct result for the juggling, whereas the stability of the dribbling also depends on the chosen dribbling height h^*.

Fig. 4 shows the magnitude of the absolute maximum eigenvalue as a function of the impact acceleration for values of c_r in the range $c_r \in \{0.6, 0.8\}$, see [3, 12]. For the analysis of the dribbling task, it was assumed that $c_r = c_{r,g} = c_{r,a}$. Fig. 5(a) shows the magnitude of the largest eigenvalue as a function of the impact acceleration for values of c_r in the range $c_r \in \{0.6, 0.8\}$ for a fixed dribbling height h^*. Fig. 5(b) shows the magnitude of the absolute maximum eigenvalue as a

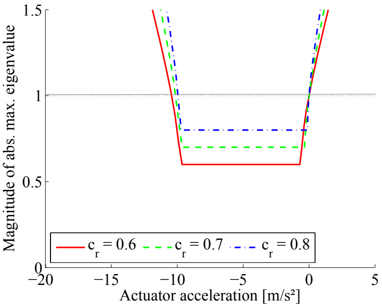
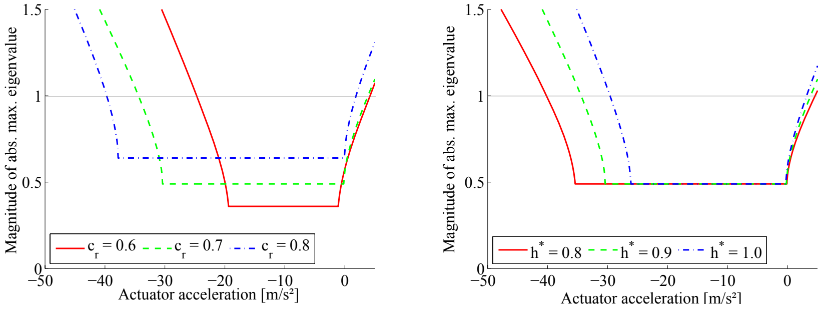


Fig. 4 Classic juggling task: Magnitude of the absolute maximum eigenvalue of the period-one motion as a function of the actuator acceleration for the juggling task with E^*.



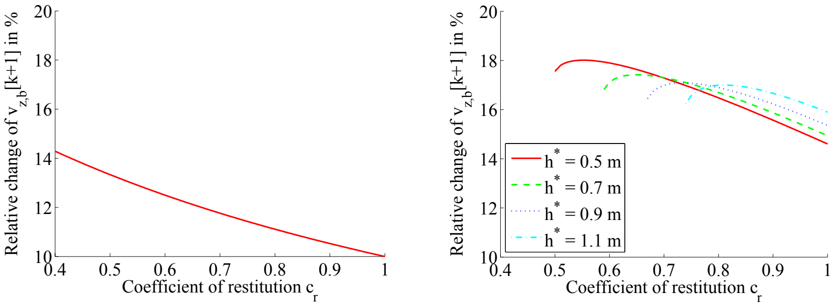
(a) Different values of c_r (b) Different values of h^*

Fig. 5 Dribbling task: Magnitude of the absolute maximum eigenvalue as a function of the actuator acceleration for different values of (a) c_r respectively (b) h^* at a given energy level E^*.

function of the impact acceleration for values of h^* in the range $h^* \in \{0.8\text{ m}, 1.0\text{ m}\}$ for $c_r = 0.7$. For both the dribbling and the juggling task, the actuator acceleration at impact has no influence on the shape of the period-one motion (see Sec. 3). However, the acceleration of the actuator has a significant effect on the stability of the system. For both tasks, there exists a (negative) acceleration range in which the largest eigenvalue is constant. In addition, the results for the dribbling task imply that for a given energy level E^*, local stability can be increased by reducing the dribbling height h^*.

4.3 Sensitivity to Parameter Variations

For the comparison of parameter sensitivity, a variation of the coefficient of restitution c_r has been chosen. Such a variation could reflect a rotational velocity of the ball, which has been neglected in the model. In Fig. 6, the relative change of the ball velocity $v_b[k+1]$ at the impact point is shown if the coefficient of restitution for the k-th impact varies by 0.1. Fig. 6(a) shows this effect for the classic juggling task, Fig. 6(b) for the dribbling task. In both plots, the relative change is given as a function of c_r. As can be seen, the dribbling task has a higher sensitivity to parameter variations. In addition, the sensitivity is a function of the steady-state impact height $h^* = p_b^*$, which is not the case for the juggling task. And, more important, only for the classic juggling a return to the impact point p_b^* is guaranteed.



(a) Juggling Task (b) Dribbling Task

Fig. 6 Relative change of $v_b[k+1]$ in percent as a function of c_r when c_r varies by $\varDelta c_r = 0.1$.

5 Experimental Evaluation

For the experimental evaluation, a six DoF Stäubli RX90B industrial robot with open control architecture is used. The robot is equipped with a six DoF force/torque sensor and a circular plate (r = 0.17 m) at the end effector. In order to implement own control concepts, a PC-based controller works in parallel with the robot's own

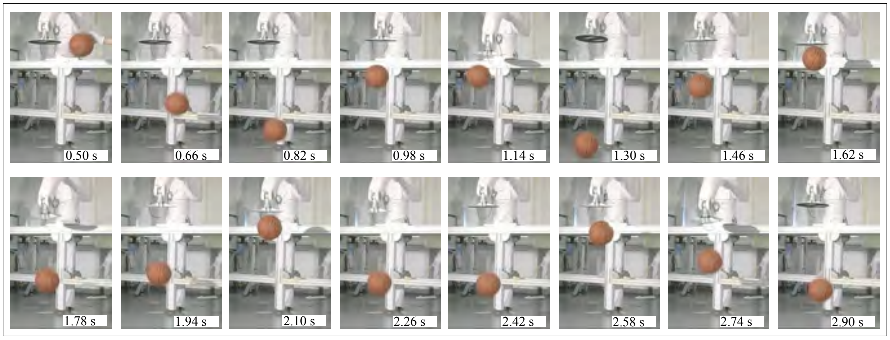


Fig. 7 Experimental snapshots of vision-based dribbling with initialization by human operator (sequence from 0.5 s to 2.9 s, time duration between two snapshots 0.16 s).

control unit. The configuration calibration, and supervising of the robot is still performed with the original architecture. The control, however, is passed over to the additional PC, which runs Matlab/Simulink in a real-time Linux (RTAI) environment. For the motion control of the robot, a computed torque feedforward controller in combination with a decentralized PD controller is used. The design of the reference trajectory for the actuator in z-direction is based on the results of Sec. 4: a trajectory based on state-feedback analog to (9) with negative acceleration at impact is chosen. The dribbling height is set to $h^* = 0.9$ m at an energy level that would allow $h^*_{max} = 1.2$ m.

With the force/torque sensor, dribbling for 5 to 10 cycles is achieved. The performance strongly depends on a precise estimate of the coefficient of restitution. As the end effector itself is moving and vibrating during the impact, the measured relative forces/torques are noisy which complicates the estimation of $\boldsymbol{v}_b[k]$. Another decline in performance arises from the fact that the desired end effector orientation is calculated based on the xy-position of the last impact. Consequently, an increase of the horizontal position error during the current dribbling cycle is not considered.

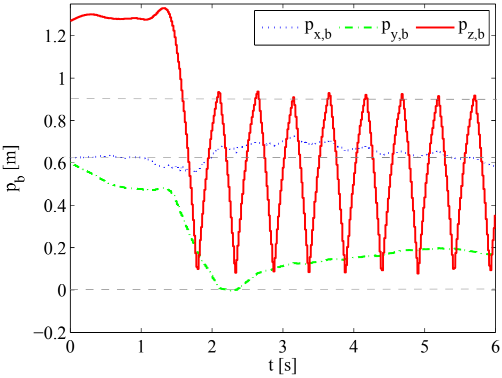


Fig. 8 Tracked ball position $\boldsymbol{p}_b(t)$ (vision-based) for the desired impact position $\boldsymbol{p}^*_b = (0.62\text{ m}, 0.00\text{ m}, 0.90\text{ m})^T$.

With the vision system, dribbling for multiple cycles (> 15) is achieved. To initialize the task, the ball is held in a human's hand and then dropped into a specified area of the robot workspace. Fig. 7 shows a sequence of snapshots, Fig. 8 the tracked ball position during the first seconds of a dribbling task.

6 Conclusion and Future Work

This paper discussed a robotic dribbling task. To this end, an overview on research in robotic juggling was presented. Based on the hybrid system model and the discrete dynamics of the dribbling task, local stability of period-one motions and the sensitivity to parameter variations was analyzed. The dribbling task was compared to the classic juggling task: As a conceptual difference between the two, it was shown that for a given energy level E^*, there exist an infinite amount of period-one motions for the dribbling task whereas the period-one motion of the juggling task is uniquely defined. It was shown that the dribbling task has a higher sensitivity to parameter variations than the classic juggling task. Also, for a varying coefficient of restitution, the return to the fixed point $\boldsymbol{p}_b^*$ is not guaranteed for the dribbling task which is in contrast to the classic juggling task. A state-feedback control was experimentally evaluated based on two different sensor modalities. For both approaches, dribbling for multiple cycles was achieved. For the vision based approach, the coefficient of restitution is adapted based on the tracked ball position $\boldsymbol{p}_{b,act}(t)$. For the force/torque-based approach, an experimentally determined, fixed value for the coefficient of restitution was used. Hence, this approach is more sensitive to parameter variations.

For the goal of a robotic basketball player, ball dribbling has to be realized for changing environmental conditions. To this end, future work will extend the system model with a more detailed impact model including a rotational ball velocity. Also, the next step in the theoretical evaluation of feasible control laws, a region of attraction analysis, will be investigated. In addition, experiments will be conducted evaluating dexterous dribbling tasks with varying height and varying horizontal position.

Acknowledgements. The first author gratefully thanks the German National Academic Foundation for their support.

References

1. Buhler, M., Koditschek, D., Kindlmann, P.: A family of robot control strategies for intermittent dynamical environments. IEEE Control Systems Magazine 10, 16–22 (1990)
2. Holmes, P.J.: The dynamics of repeated impacts with a sinusoidally vibrating table. Journal of Sound and Vibration 84(2), 173–189 (1982)
3. Bapat, C.N., Sankar, S., Popplewell, N.: Repeated impacts on a sinusoidally vibrating table reappraised. Journal of Sound and Vibration 108(1), 99–115 (1986)

4. Buehler, M., Koditschek, D., Kindlmann, P.: A one degree of freedom juggler in a two degree of freedom environment. In: Proc. IEEE International Workshop on Intelligent Robots, October 31 –November 2, 1988, pp. 91–97 (1988)
5. Buehler, M., Koditschek, D., Kindlmann, P.: Planning and control of robotic juggling and catching tasks. The International Journal of Robotics Research 13(2), 101–118 (1994)
6. Rizzi, A., Koditschek, D.: Progress in spatial robot juggling. In: Proc. IEEE International Conference on Robotics and Automation, May 12–14, pp. 775–780 (1992)
7. Rizzi, A., Koditschek, D.: Further progress in robot juggling: the spatial two-juggle. In: Proc. IEEE International Conference on Robotics and Automation, May 2-6, 1993, pp. 919–924 (1993)
8. Rizzi, A., Koditschek, D.: Further progress in robot juggling: solvable mirror laws. In: Proc. IEEE International Conference on Robotics and Automation, May 8–13, 1994, pp. 2935–2940 (1994)
9. Lynch, K.M., Black, C.K.: Recurrence, controllability and stability of juggling. IEEE Transaction on Robotics and Automation 17(2), 113–124 (2001)
10. Brogliato, B., Zavalo-Rio, A.: On the control of complementary-slackness juggling mechanical systems. IEEE Transactions on Automatic Control 45(2), 235–246 (2000)
11. Aboaf, E., Drucker, S., Atkeson, C.: Task-level robot learning: Juggling a tennis ball more accurately. In: Proc. of IEEE Int. Conf. on Robotics and Automation, pp. 1290–1295 (1989)
12. Schaal, S., Atkeson, C.: Open loop stable control strategies for robot juggling. In: Proc. IEEE International Conference on Robotics and Automation, May 2-6, pp. 913–918 (1993)
13. Schaal, S., Atkeson, C.: Robot juggling: implementation of memory-based learning. IEEE Control Systems Magazine 14, 57–71 (1994)
14. Schaal, S., Atkeson, C.: Memory-based robot learning. In: Proc. IEEE International Conference on Robotics and Automation, May 8-13, 1994, pp. 2928–2933 (1994)
15. Schaal, S., Sternad, D., Atkeson, C.: One-handed juggling: A dynamical approach to a rhythmic movement task. Journal of Motor Behavior 28(2), 165–183 (1996)
16. Sepulchre, R., Gerard, M.: Stabilization of periodic orbits in a wedge billiard. In: Proc. 42nd IEEE Conference on Decision and Control, December 9–12, vol. 2, pp. 1568–1573 (2003)
17. Lehtitet, H.E., Miller, B.N.: Numerical study of a billiard in a gravitational field. Physica 21(1), 93–104 (1986)
18. Ronsse, R., Lefevre, P., Sepulchre, R.: Open-loop stabilization of 2d impact juggling. In: Proc. IFAC NOLCOS, Stuttgart, Germany, pp. 1157–1162 (2004)
19. Ronsse, R., Lefevre, P., Sepulchre, R.: Timing feedback control of a rhythmic system. In: Proc. of 44th IEEE Conference on Decision and Control and the European Control Conference CDC-ECC 2005, pp. 6146–6151 (2005)
20. Ronsse, R., Lefevre, P., Sepulchre, R.: Sensorless stabilization of bounce juggling. IEEE Transactions on Robotics 22(1), 147–159 (2006)
21. Ronsse, R., Sepulchre, R.: Feedback control of impact dynamics: the bouncing ball revisited. In: Proc. 45th IEEE Conference on Decision and Control, pp. 4807–4812 (2006)
22. Shiokata, D., Namiki, A., Ishikawa, M.: Robot dribbling using a high-speed multifingered hand and a high-speed vision system. In: Proc. IEEE International Conference on Intelligent Robots and Systems IROS 2005 (2005)
23. Domenech, A.: A classical experiment revisited: The bounce of balls and superballs in three dimensions. American Journal of Physics 1, 28–36 (2005)
24. Buss, M.: Hybrid control of mechatronic systems. Journal on Systems, Control and Information; Institute of Systems, Control and Information Engineers, ISCIE 46(3), 126–137 (2002)

25. Sobotka, M.: Hybrid Dynamical System Methods for Legged Robot Locomotion with Variable Ground Contact. PhD thesis, Technische Universität München (2007)
26. Bätz, G., Lee, K.-K., Wollherr, D., Buss, M.: Robot basketball: A comparison of ball dribbling with visual and force/torque feedback. In: Proc. IEEE International Conference on Robotics and Automation ICRA 2009 (2009)
27. Ronsse, R., Lefevre, P., Sepulchre, R.: Rhythmic feedback control of a blind planar juggler. IEEE Transactions on Robotics 23(4), 790–802 (2007)
28. Parker, T., Chua, L.: Practical Numerical Algorithms for Chaotic Systems. Springer, Heidelberg (1989)

Research on Exoskeletons at the TU Berlin

Christian Fleischer, Konstantin Kondak, Andreas Wege, and Ingo Kossyk

Abstract. We present an overview of the work devoted to exoskeletons which has been performed for the last seven years at TU Berlin. Three different types of exoskeleton devices have been developed: hand and finger exoskeletons for rehabilitation, a leg exoskeleton for motion support, and an arm exoskeleton for multimodal human-computer interaction. The research has been focused on different types of control strategies and algorithms as well as on the implementation of applications. The main part of this work is devoted to using electrical muscle signals for the control systems. We present the concepts for design and application, review the main algorithms and show experimental results and first experience with applications.

1 Introduction

Exoskeletons are actuated mechanical constructions attached to the human body. The possibility to influence the human motion makes this kind of robots fascinating research objects with a huge potential for practical applications. The range of possible applications is wide: motion of human limbs along predefined trajectories (e.g. locomotion of fully paralyzed patients), pure force enhancement, intelligent assistant (e.g. assembly assistance) as well as immersion into virtual reality [16, 5, 13, 21].

The level of control intelligence required for an exoskeleton for the above mentioned applications is very high: Due to the direct contact between the mechanical construction and the human body the system should have the same level of ”motion intelligence” as the human. Therefore, the development of a universal controller for such devices is difficult due to lack of basic approaches. Even the control along fixed predefined trajectories is complicated because human limbs have to be modelled if a high performance control with high gains is required. Furthermore, mechanical

---

Christian Fleischer · Konstantin Kondak · Andreas Wege · Ingo Kossyk
Institut für Technische Informatik und Mikroelektronik, Technische Universität Berlin,
Einsteinufer 17, 10587 Berlin, Germany,
e-mail: {`fleischer,kondak,awege,kossyki`}`@cs.tu-berlin.de`

constructions become complicated for non-restricted applications (support for the whole body). The realistic approach to further developments is to choose particular applications and to develop specialized mechanical constructions and controllers. Most controllers for clinical applications are based on the evaluation of muscle signals to incorporate human activity directly into the control loop with different approaches on how to evaluate those signals [2, 4, 11, 29]. Most other applications (e.g. force support for factory workers, rescue teams, soldiers) use dynamic models of interaction forces [12, 20] or input from additional devices and sensors [17].

For our research three fields of application were chosen: a hand exoskeleton for rehabilitation support after hand surgeries, a leg exoskeleton with an actuated knee joint for support/force enhancement and an arm exoskeleton to be used as a haptic device with a large workspace for immersion into virtual reality. We have laid the focus on the development of the controllers and have chosen constructions with some restrictions on the human motion to be able to assemble the devices in a short time and at moderate costs. Despite the motion limitations the constructions allow us to investigate the desired applications without considerable drawbacks.

The paper is organized as follows: In section 2 the hand exoskeleton for rehabilitation support after hand surgeries is presented. The purpose of this device is to move the fingers of a patient in order to restore motor capabilities of the hand after surgery and to avoid post-surgery complications. One important issue of the controller development was the realization of high gains in a system with unknown variable parameters and big unmodelled disturbances caused by human interaction. The successfully realized approach is based on the application of sliding-mode control. In section 3 the exoskeleton for the lower extremities with an actuated knee joint is presented. This device is supposed to be used for force enhancement of elderly or support for disabled people. The research with this device was focused on the application of electrical muscle signals (EMG) for control. The main issue of the realized controller was the usage of those signals directly in the control loop to allow flexible and spontaneous movements. In section 4 a perspective on the ongoing work related to the arm exoskeleton for immersion into virtual reality with haptic feedback is presented. In section 5 the work is concluded.

2 Exoskeletons for the Hand

Two devices were developed. Both were designed with the focus on the support of rehabilitation and diagnostics after hand surgeries or strokes. The first device – the hand exoskeleton – is equipped with many sensors and utilizes a complex linkage to exert forces on joints of the hand. It was mainly used for the development of control algorithms. The second device – called finger exoskeleton – supports only a single finger, but features an improved mechanical construction and was developed with focus on safety and usability during clinical trials. Figure 1 shows both devices.

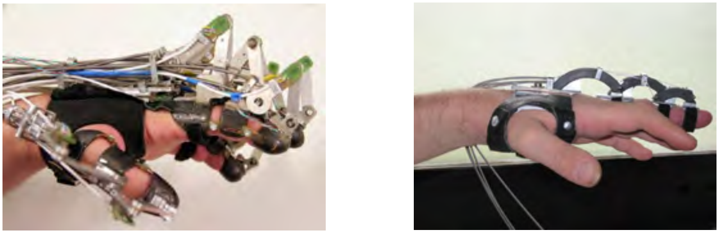


Fig. 1 Developed exoskeleton devices. Left: hand exoskeleton with 20 degrees of freedom. Right: finger exoskeleton with improved mechanical construction and 3 degrees of freedom.

2.1 The Hand Exoskeleton

The hand exoskeleton supports flexion, extension, abduction and adduction of all major joints of each finger, resulting in 20 supported degrees of freedom. The palm is free of mechanical elements to allow interaction with the environment. The fingers are moved by a construction of levers actuated through pull cables guided by flexible sheaths and driven by DC motors. Pulleys at the levers allow bidirectional movement. Finger joint angles are measured by Hall sensors which are integrated into the mechanical construction. Angles of the axes at the motor units measured by optical encoders correspond to the angles measured by the Hall sensors. Because of varying tension in the connecting cables both values for joint angles deviate. Five force sensors integrated between the levers and finger attachments measure forces during flexion and extension at the finger tip. Surface EMG electrodes measure muscle activity at ten locations at the forearm.

2.2 The Finger Exoskeleton

The finger exoskeleton supports three degrees of freedom for one finger with a range of motion of 90 degrees for the proximal finger joints. The improved mechanics with circular arc joint is more compact and allows a smooth and ergonomic movement of the finger without additional supporting joints at the sides of the finger joints. The device was designed with the focus on reliability and safety for the use during clinical trials. The safety measures include a medical power supply, slip clutches to limit the motor torque and a watchdog monitoring the microcontroller responsible for the control loop. The safety of the system is certified and clinical trials are scheduled.

The circular arc joint joints are composed of several arcs which can move against each other. They work similar to a telescopic joint with the difference that the guide is along a circular arc and not along a line. The arc joint is placed above the finger joint so that the center point of the circular arc is in one place with the revolving joint of the finger. The movement of the joint occurs in both directions by using a cable driven mechanism with two cables. When combining three circular joints for the finger exoskeleton, it is possible to get along with four pull cables, similar as for the human hand: one for extension and three for the flexion.

Compared to others, this construction is simple and robust. A similar construction was used for the proximal joint force support for an extravehicular glove [28]. In contrast to us they utilize an ultrasonic motor attached to the hand base which moves a steel belt along a circular guide. But by using our cable driven construction it becomes possible to use other actuators and control the distal joints.

2.3 Control Methods for the Hand and Finger Exoskeletons

The hand exoskeleton supports three control modes: trajectory control, force control, and EMG control. Due to the lack of sensors the finger exoskeleton supports only trajectory control. The first control mode allows following trajectories determined by the therapist. This allows repeating exercises reliably with a high accuracy. The challenge was here to achieve the required motion accuracy despite strong disturbances caused by patient interaction. The developed controller was based on application of a sliding-mode high gain controller described in [15]. The second control mode uses force sensors to determine the motion of the exoskeleton. The force control uses force sensor readings to calculate the desired motion with an open-loop admittance control scheme. During this control mode the exoskeleton follows the motions of the hand. But the force sensors cannot distinguish between internal and external contact forces. Therefore, they are insufficient to measure the human motion intention during contact with the environment (e.g. grasping), which is common during rehabilitation. Integrating more force sensors to distinguish between internal and external is not practicable for the hand exoskeleton due to the constricted space. Details of construction, sensors and control algorithms are described in [26].

Details: EMG Control for the Hand Exoskeleton

EMG sensor data can be used to control the hand exoskeleton. Those signals represent the human intention better than using only force sensors. However, there are several difficulties in their application. One problem is that only a subset of all muscles responsible for the hand motion can be measured by surface EMG sensors. Therefore, it is not possible to use the EMG signals alone to control arbitrary motions in all supported degrees of freedom. Force contributions of muscles that are not measured are derived from the other muscle signals by assuming a specific hand movement. The second problem is that captured EMG signals are superimposed by the signals of nearby muscles. This problem can be reduced by utilizing blind source separation algorithms. Figure 2 depicts the several computing steps of the EMG control scheme used for the hand exoskeleton.

Due to the limitation of space only the extrinsic muscles of the forearm are measured. The following sensor locations were selected for measurement: *M. flexor digitorum superficialis* near wrist (two electrodes), *M. flexor policis longus* (one electrode), *M. flexor digitorum superficialis* middle section of forearm (one electrode),

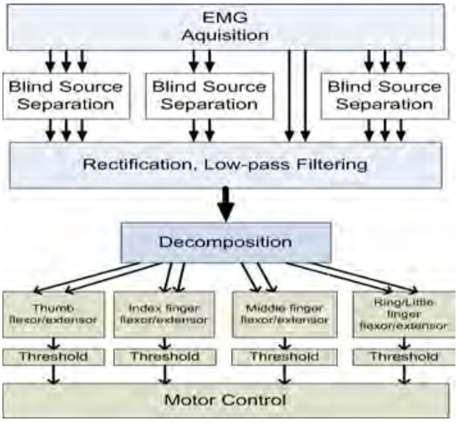

EMG
Aquisition
Blind Source Separation
Blind Source Separation
Blind Source Separation
Rectification, Low-pass Filtering
Decomposition
Thumb flexor/extensor
Index finger flexor/extensor
Middle finger flexor/extensor
Ring/Little finger flexor/extensor
Threshold
Threshold
Threshold
Threshold
Motor Control

Fig. 2 Scheme of the EMG control for the hand exoskeleton device. After EMG signal acquisition blind source separation is used to partially separate the superimposed EMG signals. After rectification and low-pass filtering an additional decomposition step is used to assign the signals to different degrees of freedom. After applying a threshold the signals are translated to forces which are used to generate a trajectory executed by the underlying position control.

and M. flexor digitorum superficialis upper section of forearm (one electrode) at the palmar side. *M. Extensor digitorum* (three electrodes)*, M. Extensor pollicis longus* (one electrode)*, and M. Extensor indicis* (one electrode) at the dorsal side.

The blind source separation algorithm used was chosen for low latency, to prevent problematic delays. The signals are filtered by a weighted low-pass differential filter. The inverse demixing matrix is then approximated by an iterative algorithm described in [10] (EASI). By using these algorithms a separation of about 1.5dB for neighboring sensors can be achieved. After the blind source separation the signals are still partially superimposed.

Additional filtering steps are: rectification, low-pass filtering and another decomposition step. This step is implemented to achieve muscle signal decomposition for the different degrees of freedom. The goal is to achieve a one degree of freedom control for each finger. To accomplish this a single extensor and flexor signal for each degree of freedom are separated by a simple linear demixing algorithm. During calibration several movements for each finger have to be performed to determine a demixing matrix. During early experiments it turned out, that it is problematic to separate the muscle signals from the ring and little finger: therefore movement of these fingers is expected to be coupled. Another limitation is that the wrist should not be moved as the muscle signals change for different hand postures.

As only a few muscles responsible for the motions of the hand are measured an accurate biomechanical model to translate the muscle activation signals is not possible. Instead a very simple model is implemented, where each finger is in its relaxed position when no muscle activation is measured. Depending on the muscle activation a linear force is calculated and the fingers are moved as if acting against a constant friction. The generated trajectory is then executed by the underlying position controller which is based on sliding mode control and therefore very robust for parameter variations and varying loads. By using the EMG control algorithm it was possible to control the hand exoskeleton in four degrees of freedom, more details and results of this algorithms are shown in [27].

3 Lower Extremities

The focus of the research with the lower extremity exoskeleton is to investigate control schemes which allow the exoskeleton to recognize and support the intended movements of the operator. The human-machine interfaces which the schemes depend on are designed to cope with the abilities and limitations of certain categories of users. We have investigated the following schemes:

1. The *torque amplifying scheme* evaluates muscle activities of the operator and adds a linear amount of torque to the supported joint through the actuator. The main means of information transportation are electrical muscle signals (EMG).
2. The *model-based approach* is designed to allow the system to control the postural stability of the operator with the exoskeleton. The interface is based on evaluation of muscle signals and information about the movement, posture and environment.

The exoskeleton used for those experiments covers thigh, shank, and foot of the right leg (refer to figure 3). The knee joint is driven by a linear actuator consisting of a DC motor and a ball-screw. The ratio of the transmission is chosen in such a way that the power is sufficient to give support with up to 150Nm in the knee joint and allows slow walking [6]. Six electrodes are embedded in the thigh brace and record signals from knee flexor and extensor muscles from the skin. Those signals together with the knee angle are sent to a single board computer which the operator is carrying on his back. All signals are digitized with 1kHz and downsampled to 100Hz for model computation. The resulting values are passed to a low-level control loop which runs with 1kHz to provide sufficient control signals for the actuation.

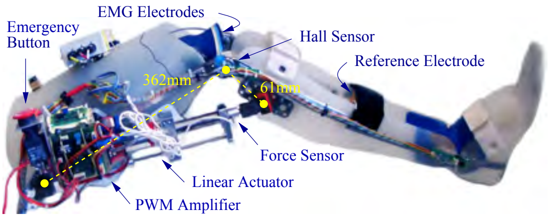
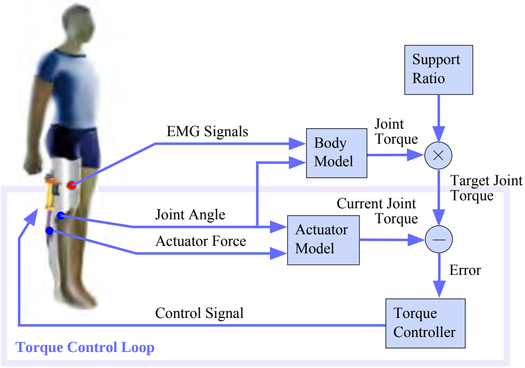


Fig. 3 Leg exoskeleton (without the portable PC): The knee joint is driven by a linear actuator. Knee and ankle angles are measured with Hall sensors and EMG sensors are embedded in the thigh brace to measure muscle activity.

3.1 EMG-Based Torque Amplifier

This control scheme is based on the evaluation of electrical signals which are emitted by the muscles upon their activation as shown in figure 4: Inside the body model a Hill-type muscle model derives the neural activation of the muscle from its EMG signal, converts this activation into a muscle fiber force and finally into the torque the muscle is producing around the knee joint [8]. Among other details, this muscle model includes the computation of the neural activation based on [23], the active

Fig. 4 The EMG signals are recorded with electrodes attached to the thigh and are evaluated through a body model that computes an estimate of the operator's torque contribution. The result is multiplied by a support ratio and passed to the torque controller that sets the signals for the actuator appropriately. This system acts as an amplifier of the operator's torque.



and passive force components as a function of the muscle fiber length and the muscle and tendon pathway along the skeleton as described in [30, 3]. This is performed for all six recorded muscles. The resulting sum of the torques gives an estimation of the operator's own torque contribution to the movement. This value is multiplied by a support ratio and used as the target value for the PID torque controller of the actuator.

In addition to the control scheme, we have developed a new calibration algorithm for the model parameters which is described in detail in [6]. This calibration has to be performed whenever the exoskeleton is put on, because some characteristics of the operator depend on the current body state (e.g. moisture on skin).

The *torque amplifying scheme* puts the user in full control of the exoskeleton, allows flexible movements and spontaneous reactions to circumstances in the real world. But the user has to be able to react to the movement that results from the combined effort of his / her own muscles and the actuator torques in a coordinated manner. The operator has to be in full control of his / her own muscles, although they may be too weak to fulfill the intended task. Intended applications are for example: carrying aids for factory workers or force support for elderly people.

3.2 Model-Based Controller

This approach is based on computation of a rigid body model of the human with two legs and a torso. The joint angles and velocities of the corresponding limbs are measured with the aid of a second, light-weight orthosis for the other leg and the torso. The sensor values are fed into the inverse computation of the model which gives an estimate of the torques the operator is producing with his muscles in the joints, except for the supported knee joint where the movement is defined through the actuator. In this joint, the torque of the operator is derived from EMG signals [7]. The biomechanical model then predicts the movement that will result in the next period of time (10–20ms in advance, refer to figure 5). The stability controller [14] applies

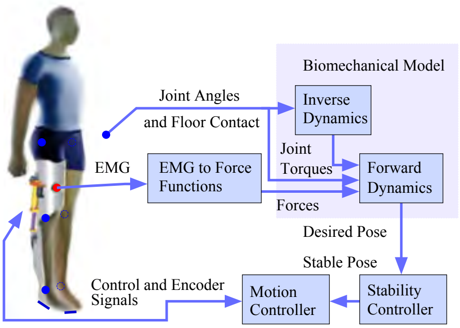

Biomechanical Model
Joint Angles
and Floor Contact
Inverse Dynamics
Joint Torques
Forward Dynamics
EMG
EMG to Force Functions
Forces
Desired Pose
Stable Pose
Control and Encoder Signals
Motion Controller
Stability Controller

Fig. 5 The model-based control scheme: The biomechanical model predicts the motion by evaluating the movement and muscle activity of the operator. This prediction is modified to ensure postural stability before the result is passed to the PID position controller of the actuation.

torques to the joints so that the zero moment point does not leave the footprint of the operator's foot that has floor contact (similar to control of biped walking robots).

The focus of the *model-based approach* is to give support to extremely handicapped people who are not able to balance themselves, but have a certain degree of control over their own limbs. The exoskeleton should be used during rehabilitation or at home. While it is not possible to control postural stability with an exoskeleton that supports only the knee joint, this algorithm nevertheless gives interesting insights in problems related to this approach.

3.3 Experiments and Discussion

The most successful experiments have been performed with the *force amplifying* scheme. For the first time, a lower extremity exoskeleton has been controlled with

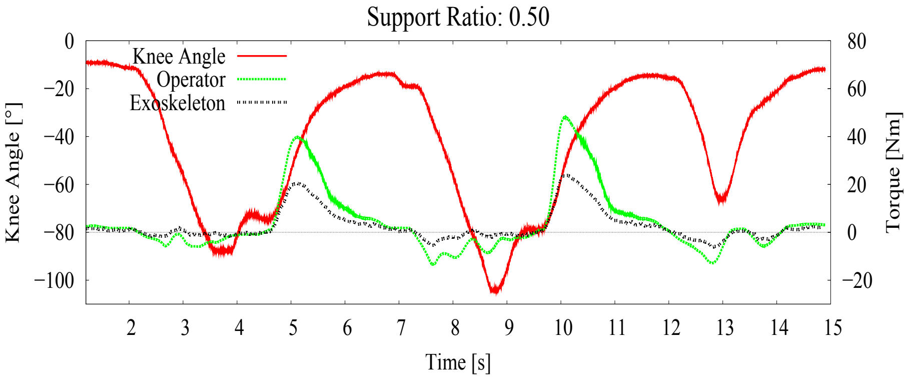


Fig. 6 Stair climbing experiment with a support ratio of 0.5: The movement can be performed smoothly and the actuator contributes half of the estimated operator's torque (0°: straight leg, negative angles: knee flexion). At $t \approx 3.8$s the foot is raised over the first step and at $t \approx 4.5$s lowered onto this step. After that the not-supported leg climbs the second step and at $t \approx 8.8$s the foot of the supported leg is raised over the third step, before it is put down again at $t \approx 9.5$s.

a sophisticated biomechanical model evaluating EMG signals which fuses results from different international research groups. Such an experiment is shown in figure 6, where the stair climbing movement is supported with a ratio of 0.5. It can be seen that the resulting movement is smooth while the exoskeleton is contributing a significant amount of torque. More experiments are presented in detail in [6, 8].

The *model-based approach* revealed some fundamental problems which could not be solved up to now: The model prediction is too inaccurate to allow the actuator to contribute significant amounts of torque in a robust manner, preventing similar real-life experiments as detailed in [7]. A much more complex sensor setup with information about the operator and the environment would be necessary but this contradicts the idea of a flexible and mobile system.

4 Work in Progress: Arm Exoskeletons for Multimodal Human-Computer Interaction

The ongoing work in our laboratory for multimodal human-computer interaction is focused on the development of a system for virtual reality with haptic feedback based on light weight arm exoskeletons. The key features of this system are a high degree of immersion into the computer generated virtual environment and a large working volume. Existing solutions for mobile haptic displays combine a grounded haptic device like the Sensable Phantom with a mobile robotic platform and control strategies to place the platform in an optimal manner to the user in order to maximize the workspace. These solutions suffer of drawbacks like the limited reactiontime and speed of the mobile platform and the control strategies [22, 9]. Our approach aims at a wearable haptic device and thus avoids these drawbacks. The high degree of immersion will be achieved by multimodal human-exoskeleton interaction based on haptic effects, audio and three dimensional visualization. The large working volume will be achieved by a lightweight wearable construction which can be carried on the back of the user. The first prototype of the mechanical construction is shown in figure 7.

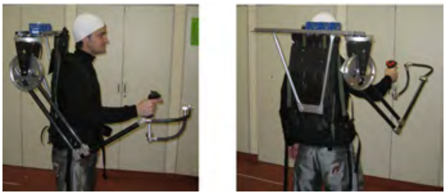


Fig. 7 First prototype for arm exoskeleton with three actuated degrees of freedom. The design objectives were: the reduction of the weight and increasing of the working volume.

By reducing the weight of the arm exoskeleton in order to make it wearable, the degree of stiffness of the mechanical constructions is getting limited. This will also limit the stiffness of the contact forces in the haptic modality. Due to the reduction of the devices weight it is also very difficult to actuate all degrees of freedom of human arm motion with a wearable mechanical construction. Therefore the arm exoskeleton will only have three actuated degrees of freedom. Lack of high stiffness as well as non-actuated degrees of freedom make it difficult to reproduce interactions with rigid objects in virtual environment realistically. The drawbacks of bad stiffness of a lightweight wearable mechanical construction can be compensated by elaborated control to some extent. The integration of additional actuated degrees of freedom into the exoskeleton arms would increase weight and reduce the working space of the mechanical construction.

In our research work we plan to achieve a compensation of the above mentioned drawbacks by utilizing the subjective perception of the user. Earlier research has shown that there is a great potential for this approach to improve the degree of immersion in a virtual environment [24, 19, 1, 25, 18]. The elaborated control of three actuated degrees of freedom in each arm should create the subjective feeling of high fidelity interaction. This interaction will be enriched by sound and visual effects. Development of algorithms for appropriate combination of these modalities for reproducing different interaction effects is the first of three main research fields of our laboratory for multimodal human-computer interaction. The second research field is related to the control of mechanical constructions with bad stiffness in haptic applications. Efficient algorithms for visualization, collision detection and force rendering in large three dimensional virtual environments is the third research field. The research conducted in these three fields is focusing on specific applications. At the moment we are developing particular effects for a game and for putting a virtual production line into operation.

5 Summary and Conclusion

During the last seven years we have performed research with promising results in the area of exoskeleton devices. Especially the developments for human assistance (load carrying, haptic guidance) and rehabilitation offer an interesting perspective for future research and applications.

Acknowledgements. We would like to thank Professor Hommel for supporting this work over all the years.

References

1. A Behavioral Adaptation Approach to Identifying Visual Dependence of Haptic Perception. IEEE Computer Society, Washington (2007), `http://dx.doi.org/10.1109/WHC.2007.2`
2. Andreasen, D., Allen, S., Backus, D.: Exoskeleton with EMG Based Active Assistance for Rehabilitation. In: 9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005, pp. 333–336 (2005)

3. Delp, S., Loan, J., Hoy, M., Zajac, F., Topp, E., Rosen, J.: An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures. IEEE Transactions on Biomedical Engineering 37(8), 757–767 (1990)
4. DiCicco, M., Lucas, L., Matsuoka, Y.: Comparison of control strategies for an EMG controlled orthotic exoskeleton for the hand. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation, 2004 (ICRA), vol. 2, pp. 1622–1627 (2004)
5. Ferris, D., Gordon, K., Sawicki, G., Peethambaran, A.: An improved powered ankle-foot orthosis using proportional myoelectric control. Gait & Posture 23, 425–428 (2006)
6. Fleischer, C.: Controlling Exoskeletons with EMG signals and a Biomechanical Body Model. Dissertation, Technische Universitt Berlin, Institut fr Technische Informatik und Mikroelektronik (2007)
7. Fleischer, C., Hommel, G.: Predicting the intended motion with EMG signals for an exoskeleton orthosis controller. In: Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (2005)
8. Fleischer, C., Hommel, G.: A Human–Exoskeleton Interface Utilizing Electromyography. IEEE Transactions on Robotics 24(4), 872–882 (2008)
9. Formaglio, A., Prattichizzo, D., Barbagli, F., Giannitrapani, A.: Dynamic performance of mobile haptic interfaces. IEEE Transactions on Robotics 24(3), 559–575 (2008)
10. Jutten, C., Herault, J.: Blind separation of sources, part 1: an adaptive algorithm based on neuromimetic architecture. Signal Process. 24(1), 1–10 (1991)
11. Kawai, S., Yokoi, H., Naruse, K., Kakazu, Y.: Study for control of a power assist device. In: Kawai, S., Yokoi, H., Naruse, K., Kakazu, Y. (eds.) Proceedings of the 2004 IEEE/RSJ International Conference on Development of an EMG based controller considering a human model. Intelligent Robots and Systems, 2004 (IROS), vol. 3 (2004)
12. Kazerooni, H.: Exoskeletons for human power augmentation. In: Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005 (IROS), pp. 3459–3464 (2005)
13. Kiguchi, K., Iwami, K., Yasuda, M., Watanabe, K., Fukuda, T.: An exoskeletal robot for human shoulder joint motion assist. IEEE/ASME Transactions on Mechatronics 8(1), 125–135 (2003)
14. Kondak, K., Hommel, G.: Control and online computation of stable movement for biped robots. In: Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003 (IROS), vol. 1 (2003)
15. Kondak, K., Wege, A., Stanczyk, B., Buss, M., Hommel, G.: Robust motion control for fully actuated robots using sliding mode. at-Automatisierungstechnik 7, 342–352 (2006)
16. Lee, S., Sankai, Y.: Power assist control for leg with HAL-3 based on virtual torque and impedance adjustment. In: 2002 IEEE International Conference on Systems, Man and Cybernetics, vol. 4, p. 6 (2002)
17. Liu, X., Low, K.: Development and preliminary study of the NTU lower extremity exoskeleton. In: 2004 IEEE Conference on Cybernetics and Intelligent Systems, vol. 2 (2004)
18. Magnusson, C., Gröhn, K.R.: Audio haptic tools for navigation in non visual environments. In: ENACTIVE 2005, the 2nd International Conference on Enactive Interfaces (2005)
19. van Mensvoort, I.M.K.: What you see is what you feel: exploiting the dominance of the visual over the haptic domain to simulate force-feedback with cursor displacements. In: DIS 2002: Proceedings of the 4th conference on Designing interactive systems, pp. 345–348. ACM, New York (2002)
20. Nakamura, T., Wang, Z., Saito, K., Kosuge, K.: Wearable antigravity muscle support system utilizing human body dynamics. Advanced Robotics 20(11), 1237–1256 (2006)

21. Naruse, K., Kawai, S., Yokoi, H., Kakazu, Y.: Development of wearable exoskeleton power assist system for lower back support. In: Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003 (IROS), vol. 4 (2003)
22. Peer, A., Komoguchi, Y., Buss, M.: Towards a mobile haptic interface for bimanual manipulations. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007, pp. 384–391 (2007), doi:10.1109/IROS.2007.4399008
23. Potvin, J., Norman, R., McGill, S.: Mechanically corrected EMG for the continuous estimation of erector spinae muscle loading during repetitive lifting. European Journal of Applied Physiology and Occupational Physiology 74, 119–132 (1996)
24. Rock, I., Victor, J.: Vision and touch: An experimentally created conflict between the two senses. Science 143(3606), 594–596 (1964), `http://dx.doi.org/10.1126/science.143.3606.594`
25. Treviranus, J.: Adding haptics and sound to spatial curriculum. In: 2000 IEEE International Conference on Systems, Man, and Cybernetics, vol. 1, pp. 588–592 (2000), doi:10.1109/ICSMC.2000.885057
26. Wege, A., Kondak, K., Hommel, G.: Mechanical design and motion control of a hand exoskeleton for rehabilitation. In: Proc. IEEE ICMA 2005 (Int. Conf. on Mechatronics and Automation), pp. 155–159 (2005)
27. Wege, A., Zimmermann, A.: Electromyography sensor based control for a hand exoskeleton. In: Proc. of IEEE Int. Conf. on Robotics and Biomimetics (ROBIO 2007), pp. 1470–1475 (2007)
28. Yamada, Y., Morizono, T., Sato, S., Shimohira, T., Umetani, Y., Yoshida, T., Aoki, S.: Proposal of a skilmate finger for eva gloves. In: IEEE International Conference on Robotics and Automation, 2001. Proceedings 2001 ICRA, vol. 2, pp. 1406–1412 (2001)
29. Yang, C., Niu, B., Chen, Y.: Adaptive neuro-fuzzy control based development of a wearable exoskeleton leg for human walking power augmentation. In: Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, pp. 467–472 (2005)
30. Zajac, F.: Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control. Crit. Rev. Biomed. Eng. 17(4), 359–411 (1989)

Hard- and Software Architecture of a Small Autonomous Underwater Vehicle for Environmental Monitoring Tasks

Christoph Osterloh, Marek Litza, and Erik Maehle

Abstract. This paper presents the design of the hardware and software architecture, the selected sensors, sensor data processing and first experimental results for a small autonomous underwater vehicle (AUV). The robot was designed as an inexpensive autonomous platform for collecting environmental data in inshore and coastal waters, stand-alone or in swarms. The utilised thrusters allow the AUV a high range of speed, and its light sensor and echo sounder a detection of obstacles. A microcontroller is used as the main processing unit. The software is distributed over specialised tasks scheduled by a small Real Time Multitasking Kernel. Based on the integrated sensors an avoid and submergence behavior are presented.

1 Introduction

The quality of waterbodies is essential for the health of people living in its neighbourhood. In the EU Water Framework Directive (WFD 2000/60/EC) the European countries commit to achieve ’good quality’ of all EU inshore and coastal waters by 2015 [1]. To control the quality, suitable monitoring methods are required. Here automatic systems can save costs compared to e.g. divers collecting data or human operators working on remotely operated vehicles (ROV) [2]. Beyond stationary sensor networks AUVs are used for such environmental monitoring tasks.

In the majority of cases only a single vehicle is used, thus it takes a long time to scan a large area. A large coordinated swarm of AUVs could solve this task in a more time efficient way. Furthermore, the exact localisation of a single AUV can be improved due to informations from its swarm neighbors [3]. The natural redundancy of a swarm allows higher fault-tolerance, even in case of a complete breakdown of one vehicle. Otherwise a swarm can collect a lot of redundant data and thereby improve its quality.

Christoph Osterloh · Marek Litza · Erik Maehle
Institute of Computer Engineering, University of Lübeck,
Ratzeburger Allee 160, 23538 Lübeck, Germany,
e-mail: {`osterloh,litza,maehle`}`@iti.uni-luebeck.de`

But even small of-the-shelf AUVs like [4] are relatively expensive and still unprepared for cooperation in large swarms or teams. For this purpose we present in this work a small inexpensive AUV MONSUN I (MONitoring System and Underwater Navigation Robot) specialized for coordinated environmental monitoring and suitable for beeing used in a team.

The expandable design allows it to carry different sensor sets and adjust to different tasks. In contrast to the Serafina project [5] we will equip our AUV also with a ballast diving system for static diving. Thus the robot will be able to lay on the ground and collect data with minimal power consumption. This and the a high bandwidth of possible speeds allows the vehicle to achieve different tasks in an energy efficient way.

Apart from recognizing obstacles by photosensors, a camera for identifying team members is integrated. Studies and simulations in [6] and [7] have shown that a basic swarm behavior like flocking are already possible without explicit communication only by the recognition of other robots and obstacles.

The following section contains information about the hardware components and architecture of the AUV. Section 3 describes the robots software architecture and its behaviour. The experiments and its results are shown in Section 4 followed by the conclusions.

2 Hardware Architecture

2.1 Mechanical and Electrical Systems

The hull of the AUV has a length of 55cm and a width of $10\,cm$ (without motors). It is made of glass fibre reinforced plastic(GRP), and is divded in three parts (Fig. 1).

The front part has a viewing panel for the integrated camera. Five ultrabright white LEDs ($25000\,mcd$) are arranged around the viewing panel, which are used for distance measurement by the light-sensor.

Fig. 2 shows the boards of MONSUN, which are mounted on an aluminium frame, one above the other. Underneath the CMUCAM 3 camera module (Fig. 2(a) 1.), which is placed on top of the boards, the communication- and navigation board (Fig. 2(a) 3.) is located, including the motor controller for the thrusters (Fig. 2(a) 2.), the Bluetooth-module (Fig. 2(b) 7.) and the gyroscope cube (Fig. 2(b) 8.). Among the main processing unit (Fig. 2(a) 4.), a board for the servo-motor controller for the fins, the accelerometers and an SD card slot are arranged (Fig. 2(a) 5.). The bottommost board (Fig. 2(a) 6.) contains a second microcontroller for the echo sounder as well as power converter and a bus converter for the light sensor that is attached behind the viewing panel. Additionally there is a further SD card slot for the camera. Above the SD card slot, a small board for the echo sounder is mounted.

This arrangement was chosen to prevent long cable route and many connectors. The frame containing the boards can be assembled outside the AUV and fit in as one part. At the backside of the retainer a programming interface as well as a serial interface is located to reprogram the main processing unit while the boards are still incorporated within the hull.

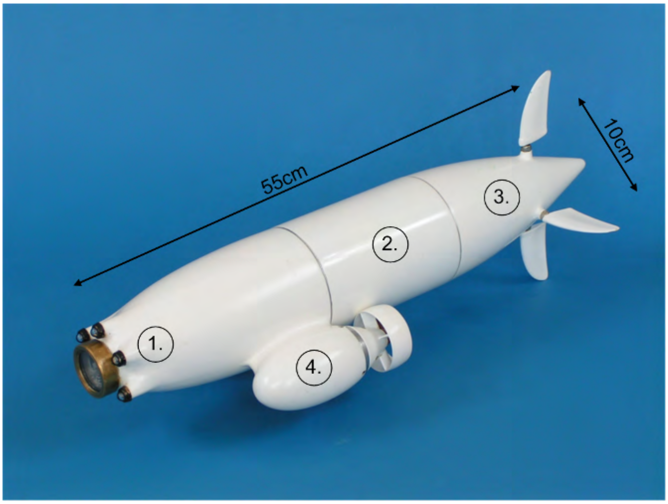
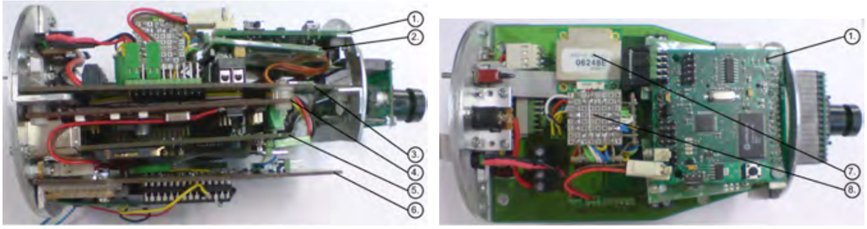


Fig. 1 The AUV MONSUN has a length of 55 *cm* and a width of 10 *cm* (without the thrusters). Part (1) contains the boards and the camera, part (2) the batteries and balast for balancing and the stern part (3) the servo motors for the rudders and elevators. The motors are housed in part (4).



(a) Boards of MONSUN in the side view.
1. The CMUcam 3
2. Motor controller for the thrusters
3. Communication and navigation board
4. Ethernut board (main processing unit)
5. Logging, SD20 and navigation board
6. Fishfinder board

(b) Boards of MONSUN in the plan view.
1. The CMUcam 3
7. Bluetooth modul
8. Gyroscope cube

Fig. 2 Side and plan view of the boards of MONSUN.

The power supply unit with accumulators and power converters is located in the 15 *cm* long middle part. Additionally there is place for two piston tanks for static diving, which momentarily is used to store balast for balancing.

In the stern part there are three servo-motors for the four movable fins. Furthermore, there are antagonistic weights to counteract the high weight of The AUVs front section. By connecting these three part by aluminium bayonet-mounts, the hull is pressure-resistant up to a depth of $10\,m$.

The main thrusters of the AUV are reversible electric motors with 43mm propellers, which generate $7\,N$ of thrust and high range of speed. Therefore fast rotations with a small turning radius are possible, beside the commonly used forward or backward motion.

To steer the course and for dynamic diving we mounted four movable fins at the stern hull of the AUV. While the two rudders are connected by a linkage and are steered by one servo-motor, the two elevators are separately controllable. The servo-motors have a torsional moment of $30\,NCM$ and a control speed of $0.16s/60°$. Using the rudders it is possible to adjust small course changes and work against drift. The elevators allow dynamic diving as well as adjusting of possible inclinations in the water.

In order to avoid the generation of PWM signals for the motor and servo contol, we use a $SD20$-servo-controller, which is connected to the control-platform by the Inter-Integrated Circuit Bus (I^2C). This allows 126 discrete positions in both directions for each servo-motor, as well as 126 diskrete speed steps in both directions for the thrusters.

The electronic is supplied by two series-connected $7.4V$ LiPo-batteries with $5.4Ah$. Supply voltages for the controller platform and sensors of $8V$, $5V$ and $3V$

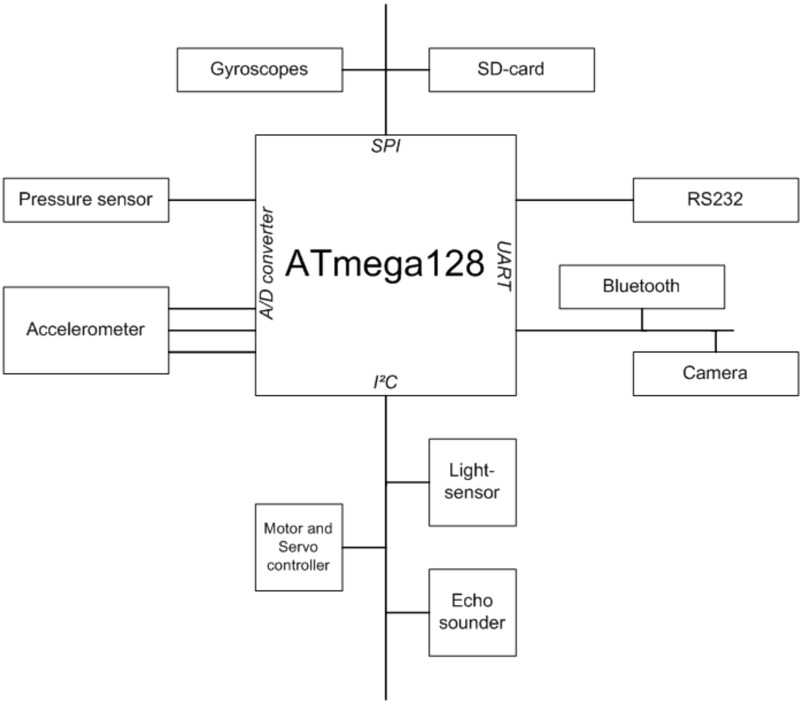

Gyroscopes
SD-card
SPI
Pressure sensor
RS232
A/D converter
ATmega128
UART
Bluetooth
Accelerometer
Camera
I²C
Light-sensor
Motor and Servo controller
Echo sounder

Fig. 3 Busnetwork connection hardware components to the ATmega128.

are provided by potential transformers. The power supply is switched on in a contactless way with a solenoid in the rear part of the AUV.

As controller platform we use an *Ethernut-Board 2.1B* [8]. This board applies an $8\,Bit$ ATmega128 running at $14.7456\,MHz$ with $128\,KB$ flash memory and $4\,KB$ SRAM and provides another $512\,KB$ banked SRAM and $512\,KB$ serial flash memory. Fig. 3 shows the connections of the different components and sensors to the ATmega128.

2.2 Sensors

The AUV MONSUN contains various sensors which can be divided into (1) positioning and navigation of the AUV and (2) assessment and storage of data.

For navigation unter water it is important to know the position, orientation, pitch and roll of the AUV. Moreover, to measure the distance to the ground and to detect obstacles are crucial features.

To track the position, orientation, pitch and roll of the AUV, we use a small Inertial Measurement Unit composed of a cube, on which three inexpensive *ADIS16100* gyroscopes are located. These gyroscopes have a resolution of $0.24\,^{\circ}/s$, a full-scale range of $\pm 300\,^{\circ}/s$ and a bandwidth of $40\,Hz$. Each gyroscope is equipped with a temperature sensor with a resolution of $0.14\,K/LSB$. The gyroscopes are connected to the main processing unit by the serial peripheral interface (SPI). Additionally, we use a three-axis *LIS3L02AS4* accelerometer, which has a user selectable full-scale of $2\,g$ or $6\,g$. It is capable of measuring accelerations over a maximum bandwidth of $4.0\,KHz$ for the X and Y axis and $2.5\,KHz$ for the Z axis. The sensor is adjusted to 2g and the outgoing voltage is sampled and digitized with the internal $10\,Bit$ Analog/Digital converter of the ATmega128 with a frequency of $20\,Hz$.

To measure depth, we use a *Freescale MPXH6250A* series pressure sensor with high temperatur accurancy. It has a range from 3.0 to $36\,psi$. The voltage output of the pressure sensor is connected to the A/D-converter of the ATmega128. The resolution of $10\,Bit$ enables an accuracy in depth measurement to $2.5\,cm$.

The recognition of obstacles underwater is more difficult than in the air due to the characteristics of the water. In [10] some light-to-frequency sensors are employed. To solve this problem we use a *TAOS TSL2561* light-to-digital converter that transform light intensity into a digital signal output equipped with an I^2C interface. The integrated A/D-converter has a resolution of $16\,Bit$. The light-sensor ist combined with the five ultrabright white LEDs ($25000\,mcd$) in front of the AUV.

Because of the short range of the light-sensor of about $60\,cm$ we integrated an echo sounder for distances above this value. Therefore we use a *Fishfinder Humminbird SMARTCAST RF45E*. This module has a sonar coverage of $90\,^{\circ}$, and is able to measure depth from $60\,cm$ up to $30\,m$. The measuring accuracy amounts about $4\,cm$. Additionally it is possible to estimate the size of the objects by the length of the returning signal.

The piezo of the fishfinder is mounted in the front part of the AUV in an angle of $45\,^{\circ}$ to the longitudinal axis. This allows a detection of obstacles in front of or under

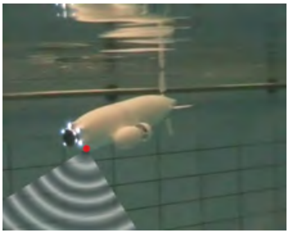


Fig. 4 Position and opening angle of the Echo Sounder in MONSUN.

the AUV (Fig. 4). Ordinarily, the fishfinder is a wireless sensor, which sends his data up to 30 m to the receiver. In order to process the data in the AUV, we removed the antenna and tap the sensor data before the transmission unit to compute the distance. This is done by a separate microcontroller (ATmega168), which acts as a slave and is connected to the main processor by the I^2C Bus.

With the integrated CMUCAM 3 camera it is possible to take pictures from the AUV point of view. The filenames of the pictures contain the system time of the main processing unit, so pictures can be synchronized with the logging data. The camera is controlled by the main processing unit over a serial interface.

During a mission collected data concerning the position, behaviour and visualization need to be stored. For this purpose a SD card was integrated into the AUV. Data are stored in a text file such that the records can be read out later. Additionally it is possible to send the records via Bluetooth to a receiver. Fig. 5 shows the structure of a log file entry. Every five seconds the stack consumption of the individual tasks is written. Subsequently, the data of the acceleration sensors, the gyroscopes, the light-sensor, as well as the echo-sounder follow. Furthermore, actual instructions for the camera, the system time and the system status are dumped.

Data storage on a SD card is inflexible and does not allow to interfere with the AUV. At the water surface or close to it, it is possible to circumvent this aspect via a Bluetooth connection to a base station. MONSUN uses a Class 1 Bluetooth *ESD110* module with a range of approx. 100 m and transmits data with up to 230 $Kbps$.

```
STACK: wdog=110, ins=17, cons=6, tele=66, brain=4,
       sd=74, clock=2, light=28, echo=4
DATA
CAM !0
TIME #0:0:15
ANGLES: -25,-41,1
GYROS: -16,-17,-1
LIGHT: 0,0,0,255
ECHO: 255,65535
STATUS: 3
```

Fig. 5 Section of the logged data. The first two lines show the stack consumption of the running tasks. The next lines contains the instructions for the camera and the global system time. Among this, data of different sensors as well as the actual system state are stored.

3 Software Architecture

The Software Architecture of MONSUN can be divided into six major units. The first part is the sensor unit, including the accelerometer and the gyroscopes, the light-sensor, echo sonar and pressure sensor for measuring depth and detecting obstacles. Although the sensors belong to one group, every sensor has his own task. The second unit are the actors. The servo and motor controller are part of this group. The third unit consists of the Inertial Navigation System (INS). The communication modules with the RS232 and Bluetooth interface represent the fourth part. The fifth unit is responsible for logging the sensory data for eventual analyses. The last unit is the high level control block of the AUV, including the main control unit and the behavior unit.

Each of these tasks and sensors has different requirements in view of execution time and periodicity. Therefore, AvrX, a Real Time Multitasking Kernel written for the Atmel AVR series of micro controllers, is applied [9]. The Kernel is written in assembly language and has a size of approximately 700 words and is compatible with Gnu GCC and IAR compilers. AvrX contains approximately 40 functions for Tasking Management, Semaphores, Timer Management, Message Queues, Single Step Debugging support an FIFO support with synchronization.

4 Behaviour

MONSUN possesses different autonomous behaviors to avoid and submerge obstacles. Fig 7 shows a simple avoid behavior. When detecting an obstacle by its light sensor, MONSUN accelerates backwards and turns thereby around clockwise for about 90° and then dives on.

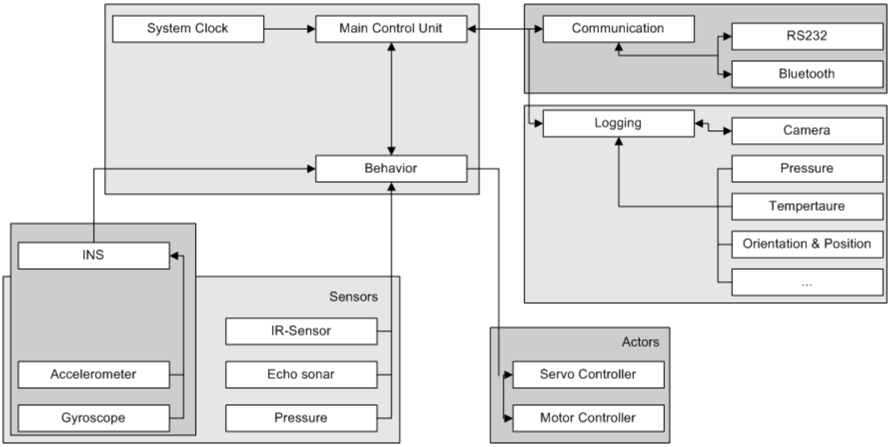

System Clock
Main Control Unit
Communication
RS232
Bluetooth
Logging
Camera
Pressure
Tempertaure
Orientation & Position
...
Behavior
INS
Sensors
IR-Sensor
Echo sonar
Pressure
Accelerometer
Gyroscope
Actors
Servo Controller
Motor Controller

Fig. 6 The six parts of the Software Architecture of MONSUN.

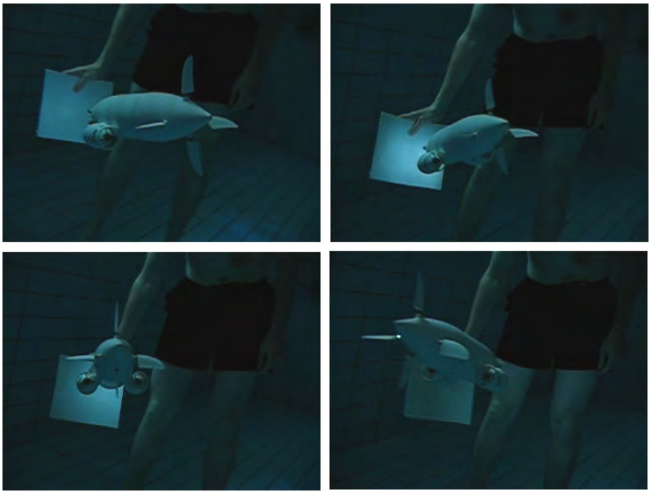


Fig. 7 The simple avoid behaviour of MONSUN. After detecting an obstacle, MONSUN accerelates backwards, turns around and dives on.

Fig. 9 shows a submerging of a recognised obstacle. Since currently MONSUN can submerge only dynamically, it accelerates backwards, also in order to prevent a collision with the obstacle while drifting. Afterwards, MONSUN dives by setting the fin and a strong acceleration forwards. If the lower end of the obstacle is recognised, the fins are set straight. After passing the object the emerging process begins. Its length depends on the time of submerging, in order to achieve the desired submerged depth again.

To verify the scenario shown in fig. 8, we performed a test run. The sequence was filmed and tracked afterwards on the computer.

Fig. 9(a) shows the distance covered over time. As one can see, the obstacle was recorgnised at about second 3.25 and MONSUN accelerates backwards. At second 4.5 the thrusters change their direction and accelerate forwards to submerge the object.

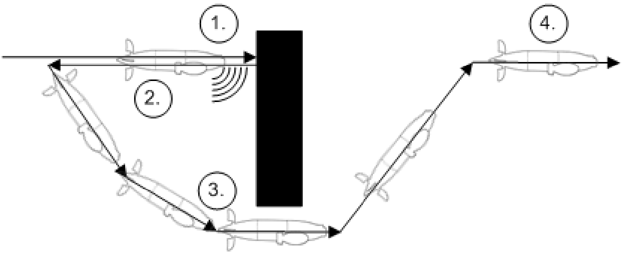
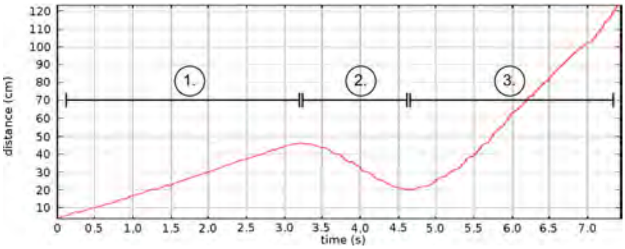
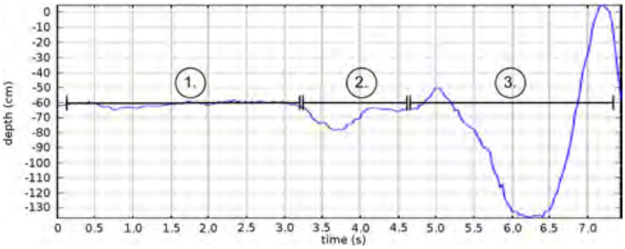


Fig. 8 Abstract representation of the programmed submergence scenario. MONSUN recognises the obstacle (1), steers backwards (2) and dives under (3) it to the original depth (4).



(a) Distance covered over time of a submergerce scenario. Phase (1) shows the forward motion and regognition of the obstacale, phase (2) the backward motion and phase (3) the submergence.



(b) Depth covered over time of a submergerce process. Phase (1) shows the forward motion and regognition of the obstacale, phase (2) the backward motion and phase (3) the submergence.

Fig. 9 Submergence scenario of MONSUN. Data were assessed by tracking the movement of MONSUN.

Fig. 9(b) shows the depth covered over time. After detecting the obstacle, the fins are placed upwards. This causes the dip of the front part in the interval of time 3.25-3.75*s*. Because of the changing direction of the thrusters the front part of the AUV turns up briefly at time 5.0*s* before submerging the object.

5 Conclusions

In this paper we presented the basic of a swarm competend AUV: MONSUN I. We have reported the hardware and software architecture and have shown, that MONSUN is capable to react to its environment by avoiding and submerging objects and obstacles. The high range of speed allows the fast reaching of a location as well as slow speed mobility for investigations. By recognising other team members it is (generally) possible to manage simple tasks without an explicit communication in the swarm.

The actual disadvantage of MONSUN I are mainly the choise of the some of its sensors and its hardware packaging. Some analog sensors, as for example the gyroscopes and acceleration sensors are too inaccurate for an exactly determination of the current position. The used hardware packaging makes it very difficult to replace individual components.

6 Future Works

To avoid the drawbacks of MONSUN I, we currently redesign our AUV. The next generation *MONSUN II* incorporates an improved hardware and software architecture based on more accurate sensors and a modular packaging. More sensors for environmental measurement will be integrated as well as a piston tank for being able to collect data laying on the bottom of waters without using the thrusters.

For a better identification of the team members an improved camera unit is used. By additional light signals at the side and the rear end of MONSUN II, a better recognition of the team members and a relative positioning will be possible. In order to allow explicit communication between the team members, we are developing a small underwater modem. By this the robots will be able to exchange necessary environmental information or status messages among themselves coordinating the swarm more effectively.

References

1. The EU Water Framework Directive - integrated river basin management for Europe, `http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2000:327:0001:0072:EN:PDF`
2. Listak, M., Martin, G., Pugal, D., Aabloo, A., Kruusmaa, M.: Design of a semiautonomous biomimetic underwater vehicle for environmental monitoring. In: 6th IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA 2005), Espoo, Finland, June 27-30, pp. 9–14. IEEE, Los Alamitos (2005)
3. Kottege, N., Zimmer, U.R.: Relative localisation for AUV swarms. In: IEEE International Symposium on Underwater Technology 2007 (SUT 2007), Tokyo, Japan, pp. 588–593 (2007)
4. REMUS 100 description, `http://www.hydroidinc.com/remus100.html`
5. The Serafina project, `http://users.rsise.anu.edu.au/~serafina/`
6. Arkin, R.C.: Cooperation without Communication: Multi-agent Schema Based Robot Navigation. Journal of Robotic Systems 9(3), 351–364 (1992)
7. Lee, G., Chong, N.Y.: Adaptive self-configurable robot swarms based on local interactions. In: Proceedings of the 2007 IEEE7RSJ Interantional Conference on Intelligent Robots and Systems, San Diego, CA, USA, October 29 - November 2, 2007, pp. 4182–4187 (2007)
8. Ethernut Board: Technical specification and datasheet, `http://www.ethernut.de/pdf/ethernut21b_ehwm_2_0.pdf`
9. AvrX Real Time Kernel specification, `http://www.barello.net/avrx/`
10. Albiez, J., Kerdels, J., Fechner, S., Kirchner, F.: Sensor Processing and Behaviour Control of a Small AUV 20. In: Fachgespräch Autonome Mobile Systeme AMS 2007 Kaiserslautern, Kaiserslautern, Germany, pp. 327–333. Springer, Heidelberg (2007)

Author Index

Adolf, Florian 275
Albers, Albert 169
Andert, Franz 275
Atwal, Pradeep 89

Barner, Simon 135
Bätz, Georg 323
Bernard, Markus 299
Berns, Karsten 179
Blank, Sebastian 179
Bootz, Friedrich 263
Borst, Christoph 203
Brown, Martin 191
Brucke, Matthias 287
Burgard, Wolfram 11
Buss, Martin 1, 323
Bücken, Arno 43

Caballero, Fernando 299
Callaghan, Dean J. 65
Coyle, Eugene 65

Dallali, Houman 191
Deiterding, Jan 31
Dittrich, Jörg 275

Eichhorn, Klaus W.G. 263
Eidenberger, Robert 89
Eilers, Sönke 287
Elkmann, Norbert 123

Farrell, Gerald 65
Feiten, Wendelin 89
Fischer, Markus 111
Fleischer, Christian 335

Gecks, Thorsten 53
Geisinger, Michael 135
Goormann, Lukas 275
Gorges, Nicolas 227
Gösling, Thomas 251
Grundmann, Thilo 89

Hein, Andreas 287
Henrich, Dominik 31, 53, 111, 147
Herder, Just L. 21
Hirzinger, Gerd 203
Hüfner, Tobias 251

Iser, René 239

Klasing, Klaas 1
Knoll, Alois 135
Kondak, Konstantin 299, 335
Kossyk, Ingo 335
Krahwinkler, Petra 43
Krettek, Christian 251
Kubus, Daniel 239
Kuntze, Helge-Björn 215

Litza, Marek 347
Lorenz, Sven 275
Luksch, Tobias 179

Maehle, Erik 347
Maza, Ivan 299
McGrath, Mark M. 65
Milighetti, Giulio 215

Ollero, Anibal 299
Olufs, Sven 311
Osterloh, Christoph 347
Oszwald, Markus 251
Ott, Lionel 11
Ottnad, Jens 169

Penzlin, Felix 123
Puls, Tim 287

Rajan, Ginu 65
Richtsfeld, Andreas 99
Rilk, Markus 263
Ritter, Moritz 215
Roßmann, Jürgen 43, 159
Ruf, Henning 159

Sander, Christian 169
Schlette, Christian 159
Schulz, Hannes 11
Semenova, Yuliya 65
Sobotka, Marion 323
Soller, Katharina 147
Sturm, Jürgen 11
Suchý, Jozef 77

Vanderborght, Bram 191
Vincze, Markus 99, 311

Wagner, Ingo 263
Wahl, Friedrich M. 239, 251, 263
Wahl, Thomas 179
Walter, Christoph 123
Wege, Andreas 335
Westphal, Ralf 251
Wijk, Volkert van der 21
Winkelbach, Simon 239, 251
Winkelmann, Hannes 287
Winkler, Alexander 77
Wojtczyk, Martin 135
Wollherr, Dirk 1, 323
Wörn, Heinz 227

Zacharias, Franziska 203