Achim Gottscheber
Stefan Enderle
David Obdrzalek (Eds.)

Communications in Computer and Information Science 33

Research and Education in Robotics – EUROBOT 2008

International Conference
Heidelberg, Germany, May 2008
Revised Selected Papers

Springer

Communications
in Computer and Information Science 33

"*This page left intentionally blank.*"

Achim Gottscheber Stefan Enderle
David Obdrzalek (Eds.)

Research and Education in Robotics – EUROBOT 2008

International Conference
Heidelberg, Germany, May 22-24, 2008
Revised Selected Papers

Springer

Volume Editors

Achim Gottscheber
School of Engineering and Architecture
SRH University of Applied Sciences Heidelberg
Heidelberg, Germany
E-mail: achim.gottscheber@fh-heidelberg.de

Stefan Enderle
qfix robotics GmbH
Senden, Germany
E-mail: enderle@qfix.de

David Obdrzalek
Faculty of Mathematics and Physics
Charles University
Prague, Czech Republic
E-mail: david.obdrzalek@mff.cuni.cz

www.eurobot.org/conference

Library of Congress Control Number: 2009934784

CR Subject Classification (1998): I.2.9, I.5, J.2, J.6, J.7

ISSN 1865-0929
ISBN-10 3-642-03557-4 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-03557-9 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12723621 06/3180 5 4 3 2 1 0

Preface

This volume contains the accepted papers presented during the International Conference on Research and Education in Robotics - EUROBOT Conference 2008, held in Heidelberg, Germany, May 22–24, 2008. The EUROBOT Conference 2008 was accompanied by the international amateur robotics contest EUROBOTopen final, edition 2008. The success of both events convinced the EUROBOT Association Executive Committee of the conference's value.

A fundamental aspect of EUROBOT is the promotion of science and technology among young students and researchers. Every year, a new theme for the robotic cup is published. Thus, participants are required to build completely new robots each year.

The theme for 2008 was "Mission to Mars." This resulted in interesting robots as well as interesting papers for the conference. Posters about all competing robots were presented by the teams. There were 54 teams from 26 different countries participating with their robots.

In addition to the paper and poster presentations, there were three invited talks:

- Atsuo Takanishi, Waseda University, Japan, whose talk was about "Humanoid Robotics and Its Applications".
- Oussama Khatib, Stanford University, USA, gave a talk about "Human-Centered Robotics".
- Vice-president of EUROBOT Association Jacques Bally and Secretary David Obdrzalek informed the attendees about "EUROBOT" in general.

The main program consisted of 33 selected regular papers, some of which are published in this Springer CCIS series. The selected papers were carefully reviewed by at least three Program Committee members and the final decision was made by the Session Chairs.

We would like to take this opportunity to thank all the authors for their contributions, the Program Committee members and the additional reviewers for their hard work, which had to be completed in a short period of time, the invited speakers for their participation, and the German local organizers for making the EUROBOT Conference 2008 a successful event.

March 2009

Branislav Borovac
Jean-Daniel Dessimoz
Stefan Enderle
Achim Gottscheber
Julio Pastor Mendoza
David Obdrzalek

Organization

EUROBOT 2008 was organized by the EUROBOT Association and the SRH University of Applied Sciences, Heidelberg, Germany, in cooperation with Planete Sciences.

Executive Committee

Conference Chair	Achim Gottscheber (SRH University of Applied Sciences, Heidelberg, Germany)
Program Chairs	David Obdrzalek (Charles University, Prague, Czech Republic)
	Julio Pastor Mendoza (University of Alcala, Madrid, Spain)
	Jean-Daniel Dessimoz (West Switzerland University of Applied Sciences, Yverdon-les-Bains, Switzerland)
	Stefan Enderle (Isny University of Applied Sciences, Germany)
	Branislav Borovac (University of Novi Sad, Serbia)

Program Committee

Bernd Sommer	DLR, Germany
David Obdrzalek	Charles University, Prague, Czech Republic
Julio Pastor Mendoza	University of Alcala, Madrid, Spain
Giovanni Muscato	University of Catania, Italy
Jean-Daniel Dessimoz	West Switzerland University of Applied Sciences
Stefan Enderle	Isny University of Applied Sciences, Germany
Branislav Borovac	University of Novi Sad, Serbia
Pierluigi Mormino	Eurobot Association, Belgium
David Calkins	Robotics Society of America, USA
Tairo Nomura	Saitama University, Japan
Boualem Kazed	Blida University, Algeria
Fernando Ribeiro	University of Minho, Portugal

Table of Contents

Adaptive Robot to Person Encounter by Motion Patterns

Hans Jørgen Andersen[1], Thomas Bak[2], and Mikael Svenstrup[2]

[1] Department for Media Technology, Aalborg University, 9220 Aalborg, Denmark
hja@cvmt.aau.dk
[2] Department of Electronic Systems, Automation & Control,
Aalborg University, 9220 Aalborg, Denmark
{tba,ms}@es.aau.dk

Abstract. This paper introduces a new method for adaptive control of a robot approaching a person controlled by the person's interest in interaction. For adjustment of the robot behavior a cost function centered in the person is adapted according to an introduced person evaluator method relying on the three variables: the distance between the person and the robot, the relative velocity between the two, and position of the person. The person evaluator method determine the person's interest by evaluating the spatial relationship between robot and person in a Case Based Reasoning (CBR) system that is trained to determine to which degree the person is interested in interaction. The outcome of the CBR system is used to adapt the cost function around the person, so that the robot's behavior is adapted to the expressed interest. The proposed methods are evaluated by a number of physical experiments that demonstrate the effectiveness of the adaptive cost function approach, which allows the robot to locate itself in front of a person who has expressed interest through his or hers spatial motion.

Keywords: Human-robot interaction, Adaptive Control, Social situatedness, Patterns of behavior.

1 Introduction

Technologies such as computing, visual recognition, and wireless connectivity open the door to a new generation of mobile robotic devices that see, hear, touch, manipulate, and interact with humans, potentially creating new interesting human living spaces that are productive, educational, safe and potentially enjoyable. Realizing this vision requires multidisciplinary research involving such areas as psychology, cognitive science, social sciences, computer science, robotics, and engineering. Current themes are discussed in [1].

Interaction between a robot and a human not only rely on the ability to input, output, and process information but also on the spatial relationship between the two actors. As an example most people would run away from a robot approaching at high speeds and most would be very suspicious about a robot approaching

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 1–11, 2009.

from behind. Quality interaction thus requires robot-human coordination in time and space; it requires that the robot can detect the person's interest in interaction and exhibit a behavior with respect for the person's privacy.

Several authors [2,3,4,5] have investigated people's willingness to engage in interaction with robots that exhibit different expressions or follow different spatial behavior schemes. In [6,7] models are reviewed that describe social engagement based on spatial relationships between a robot and a person with emphasis on the spatial movement of the actors. In [8] human-human proxemics distances were discussed and social and intimate zones defined. Social spaces between robots and humans were studied in [9] supporting the use of Hall's proxemics distances also in this context. A method for human-aware navigation using cost functions designed as Gaussian distributions centered around the person, is introduced in [10,11]. Besides the distance between the robot and the person the direction of approach is clearly also important and in [4,12] it is concluded that the preferred approach is from the front right or left and that a frontal and especially a rear approach should be avoided.

The focus of this paper is on determining a persons interest in interaction and adjusting the robot approach accordingly to reach an appropriate position. A person's willingness to engage in interaction is analyzed based on the person's spatial motion and knowledge from previous encounters stored in a case database. The case reasoning is based on [13,14] and facilitate a context aware generation of new cases of interaction. The robot behavior is controlled by a cost function adapted to the determined interest in interaction.

The human robot interaction methodology described in this paper, is supported by a number of experiments that demonstrate the effectiveness of the adaptive cost function approach. The case based reasoning and learning is demonstrated through a simulation study.

2 Materials and Methods

2.1 Evaluating Person Interest

Quality robot person interaction relies on an automatic detection of the person's willingness to interact. To accomplish this a person evaluator based on the motion of the person relative to the robot, is introduced. The philosophy of the evaluator is that if a person is interested in interaction the person will approach the robot whereas if the person is not interested he or she will have a trajectory away from the robot. However, these two extremes clearly have many levels in between where the interest of the person will be difficult to determine. In addition to person motion primitives, the temporal and spatial context of the person may influence the detection. To support this a person evaluator is designed as an adaptive Case Based Reasoning (CBR) system with the following variables as inputs (see Fig. 1):

Distance, $|d_{rp}|$, the distance between the person and the robot
Area, A_{eval}, the area spanned by the directional vector, v_{pers} and d_{rp}
Position, position of the person

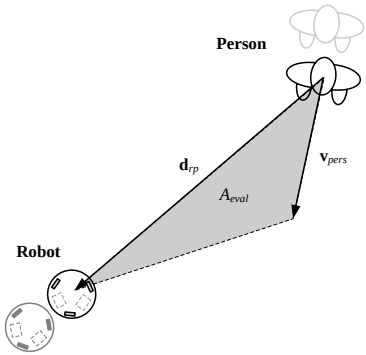


Fig. 1. Illustration of the input variables to the CBR system

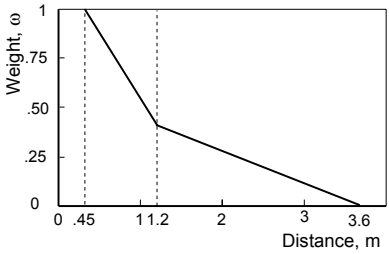


Fig. 2. Weights used for updating of the PI value. The Hall zones [8] are reflected by the 1.2 and 0.45 meter discontinuities.

The distance is used to emphasize that a large area at a large distance is less informative than a a large area at a small distance as indicated in Fig 2. The position is recorded in order for the robot to learn if people exhibiting the same kind of behavior, are most likely to be encountered in certain areas. The output of the CBR is a value $PI \in [0, 1]$, where 0 indicate no interest and 1 is an indication of absolute interest in interaction.

Given that a person has been detected and analyzed it remains to evaluate the persons willingness to interact using a case based approach. The task of specifying a case is a question of determining a distinct and representative set of features connected to the event of a human robot encounter. The outcome of the person detection and analysis, is a natural choice for inclusion in the case description. In addition, including position of the encounter will allow the robot to learn if people exhibiting the same kind of behavior, are most likely to be encountered in certain areas. By recording the time of day at time of detection, the robot may gradually become aware of possible similarities between the solution outcome i.e. whether assistance is needed or not at specific times.

Due to the active participation of the robot in evaluating a person, a number of temporary case lookups are performed, starting at 3.6 m distance and then every 10 cm. Two distinct databases are used. One serving as the main case library holding cases that have been evaluated by explicit expressions of interest, the other is a temporary case library functioning as storage of cases during approach. Each case has an associated PI which store the probability or indication of the detected persons interest to interact. During approach the PI of the cases in the temporary case library are used to control the approach. New cases have a default PI=0.5.

Whenever an outcome of the encounter is known, the temporary cases must be evaluated and afterwards erased. New cases are transferred to the main database for later reasoning. The database access is divided into two, being retrieval/reuse and revision/creation.

Retrieval/Reuse. When looking up in the main case library and no match is found the currently faced case is stored directly into the temporary case database. When a match is found the existing case is copied to the temporary case database, for later alteration of its indication when an outcome is known, i.e. during case revision. When searching for cases in the main database, rules must be set up to support case match. A case matches given that the case fields distance, spanned area, position and time of day all are within specified limits. All cases found to match are returned and stored in the temporary library in ascending order, after mismatch in spanned area.

Revision/Creation. Given that the robot has completed a person evaluation, and that the temporary case library, as a result of the performed case lookups, holds a given amount of cases. Whether the person evaluation has ended because of the person being evaluated as not interested or as a result of conducted communication, the robot should now revise all of the temporary stored cases. Thus, some cases should be created as new cases, while others should be used in updating existing cases.

Either way, the *PI* of the cases in the temporary case library are updated during revision according to the experienced outcome. The spatial relationship between human and robot must be considered when revising and creating new cases. The value of *PI* should naturally be strengthened as she or he is getting closer to the robot. Such weighted alteration has been implemented utilizing the behavioral zones as designated by Hall [8]. The weight as a function of the distance is illustrated in Fig. 2.

When entering the personal zone (1.2 meter) of the detected person, the weight function, w shifts resulting in a radical increase in weight according to distance.

Algorithm 1

```
Initially set all PI = 0.5
if (Interested) then
   PI = PI + wL
   if PI > 1 then
      PI = 1
else if (Not Interested) then
   PI = PI - wL
   if PI < 0 then
      PI = 0
```

Algorithm 1 outlines how *PI* is updated. According to Fig. 2, w is a weight that ensures that observations close to the robot are given a higher weight than observations further away. L is a learning rate factor that controls how much the *PI* is updated due to a new observation, i.e. a new observation should after a while only influence the *PI* value of the cases to a limited degree. Thus, the lower the learning rate, the less effect the weighing will have.

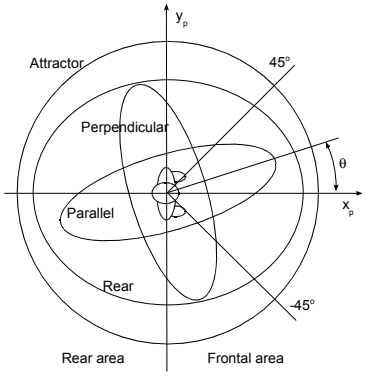


Fig. 3. Illustration of the four Gaussian distributions used for the cost function around the person. The rear area is behind the y_p axis. The frontal area is in front of the y_p axis which is divided in two, one in the interval from $[-45° : 45°]$ and the other in the area outside this interval.

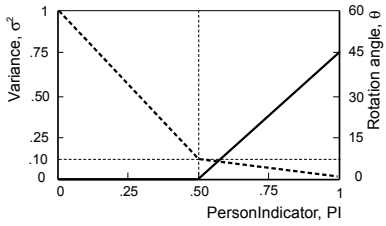


Fig. 4. Relation between the *PI* and variance σ^2 along the minor axis of the *Parallel* and *Perpendicular* distributions (solid line) and rotation angle θ (dashed line)

2.2 Adapting Robot Behavior

Given an indication of interest expressed by the *PI* value from the CBR system, the robot motion must be adjusted accordingly. The robot motion is controlled by an adaptive person centered cost function, which is the weighted sum of four Gaussian distributions which are adapted according to the *PI* value. The four Gaussian distributions are illustrated in Fig. 3 and has the following functions:

Attractor. This distribution is used to attract the robot to the person

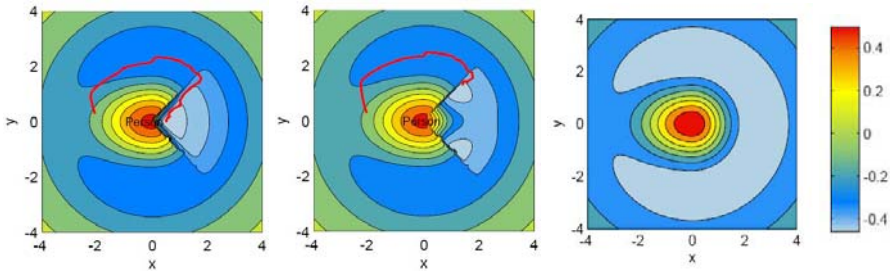
Rear this distribution ensures that the robot does not approach a person from behind.

Parallel. This distribution is initially placed with its major axis parallel to the x_p-axis in the persons coordinate frame and adapting its variances and covariance according to the *PI* value.

Perpendicular. Distribution which initially is placed with its major axis perpendicular to the parallel distribution and adapting its variances and covariance according to the *PI* value.

The four distributions are combined resulting in a cost function landscape. The sum is divided into two areas, respectively, in front and behind the person. Behind only the sum of the *attractor* and *rear* distribution is considered. The values of the covariance for the two distributions are adjusted so the robot stays in Hall's public zone, i.e. 3.6 meters from the person and they are kept constant for all values of *PI*.

The *Parallel* and *Perpendicular* distributions are adapted according to the *PI* value, Hall's proximity distances, and the preferred robot to person encounter



(a) PI=1 (b) PI = 0.5 (c) PI=0

Fig. 5. Shape of the cost function for a person (a) interested, (b) maybe interested, and (c) not interested. Scale of the cost function is plotted to the left. The solid line in (a) and (b) illustrates the robot approach.

reported in [4,12] following the functions illustrated in Fig. 4. The width is adjusted by the value of the variances $\sigma^2_{x,y}$. The rotation θ may be adapted by adjustment of the covariance σ_{xy} according to $tan(2\theta) = \frac{2\sigma_{xy}}{\sigma_x^2 - \sigma_y^2}$.

The result is a change in size and rotation of the *Parallel* and *Perpendicular* distributions which given a *PI* value close to one may guide the robot into a position in front of the person, with an approach angle of approximately 45° given a *PI*. The resulting cost functions for specific values of *PI* are illustrated in Fig. 5.

2.3 Experimental Setup

The proposed methods where implemented on a FESTO Robotino platform. On the platform the software framework Player[15] was installed. The robot is equipped with a URG-04LX line scan laser range finder together with a Creative Live! color Web camera.

The experiments were limited to only involve one person equipped with a pair of red socks in a controlled laboratory environment. For detection of the socks the color blob detection plug-in CMVision [16] was used. All experiments were recorded with a camera mounted in the ceiling and the trajectory of the person and robot were determined for reference.

Training of the CBR system was done by simulation in Stage[15]. For this, the Player plug-in of the VFH+ method [17] together with the wavefront algorithm [15] was implemented to give a virtual person. In the training 20 series where run, 10 with a interested person and 10 with a not interested person.

3 Results

Figures 6 (a) and (b) illustrate the situation where the robot approaches a person from the front. In (a) the person is not interested in interaction whereas in (b) the

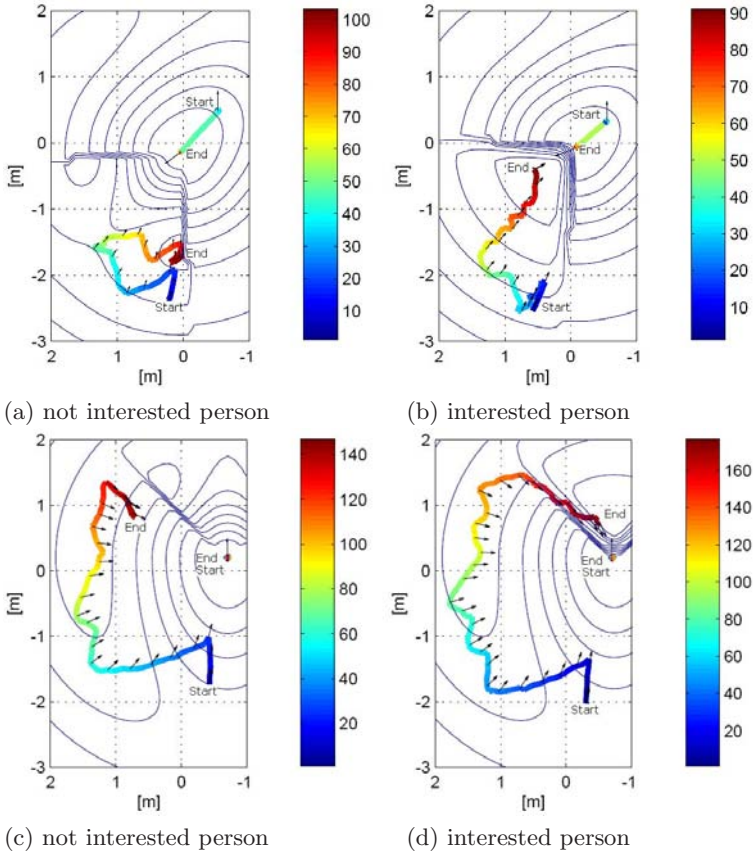
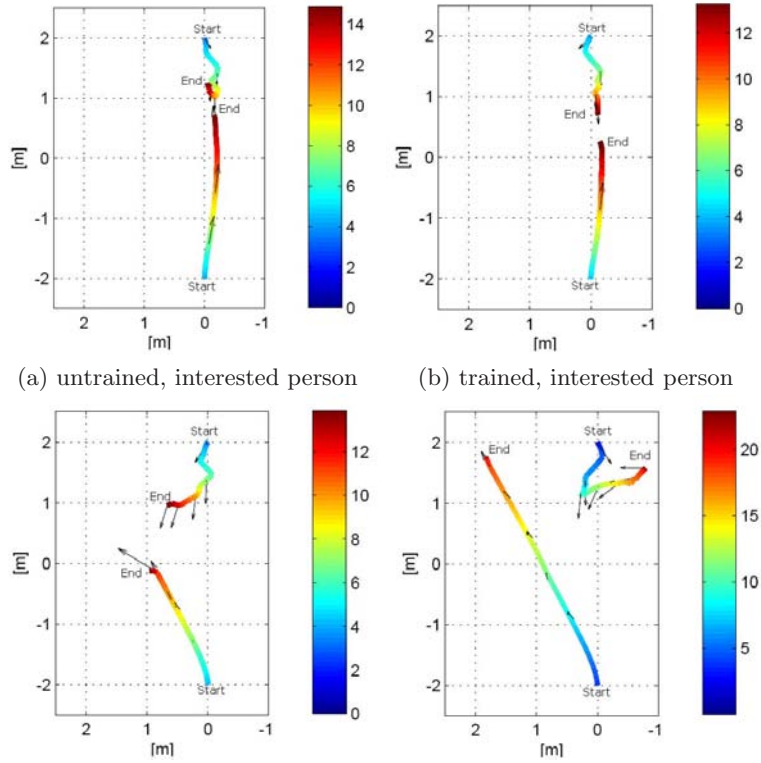


Fig. 6. Trajectories generated from laboratory experiment. The bars indicate the time elapsed and the arrows indicate the heading of the robot and person. The contours illustrates the shape of the cost function at the end of the experiment.

person is. In both cases the robot initially observes the person and determines the person's direction of motion. After 40 seconds the motion of the robot changes according to the person interested in interaction, expressed by the *PI* value from the CBR system. In (a) the robot stops 1.5 meters away from the person and at an angle of 45° relative to the person's direction of motion. In (b) the cost function changes so the robot is allowed to face the person and stops at a distance of 0.7 meters directly in front of the person, ready for interaction.

Figures 6 (c) and (d) illustrate the cases where the robot approaches the person from behind. In (c) the person is not interested in interaction whereas in (d) the person is. In both cases the cost function forces the robot to move around the person according to the Hall distance. In (c) the robot stops 1.5 meters from the person and at an angle of 45° relative to the person's direction of motion. In (d) the cost function changes so the robot is allowed to directly face the person at a distance of 0.7 meters.



(a) untrained, interested person (b) trained, interested person

(c) untrained, not interested person (d) trained, not interested person

Fig. 7. Trajectories generated from simulated experiment. The bars indicate the time elapsed and the arrows indicate the heading of the robot and person. Behavior of the robot before and after training of the CBR system. For all experiments the robot starts at $(0, 2)$ and the person in $(0, -2)$. In the interested case the goal position of the robot is set to the person position whereas in not interested case the simulated person is set to goto $(2, 2)$.

To demonstrate the learning abilities, the results of simulated experiments are given in Fig. 7. Results of the robot behavior before and after training are illustrated for a situation where the person is not interested in interaction.

Comparing the behavior of the untrained and trained robot there is a significant change in the robots behavior after training. The motion of the robot does not deviate significantly at large distances due to the lower weighting of observation far from the robot and as a result the PI will have a value of approximately 0.5 for both situations. As the distance between robot and person is reduced as in (a) (at about 2 meters or after 9 seconds), the untrained robot starts to reverse while deviating to the right. This is because the forward part of the cost function is not rotated as the person interest indication is not yet changed from the default value of 0.5, i.e. the cost appears is a in Fig. 5(b). However,after

training the robot evaluates the person as interested and the forward part of the cost function is rotated, allowing the robot to approach the person frontally, i.e. the cost function is adapted to appear as in Fig. 5(a).

In Fig. 7(d) the robot assumes that the person is interested (as this is default) and approaches the person until it prevents the person from continuing towards the goal which is set to $(2, 2)$. However, after training the robot evaluates the behavior of the person as not interested and as a result the PI takes a values of 0 coursing the cost function to appear as in Fig. 5 (c). The result is that the robot is pushed away from the person so the person is able to reach the position $(2, 2)$ without interference by the robot.

4 Discussion

The results above demonstrate how an adaptive Gaussian cost function may be used as the basis for a spatial robot behavior scheme on a planar surface. Given the determination of a person's pose this may be combined with knowledge about the person's interest to reshape the cost function and drive the robot into a position which the designer has determined as appropriate for quality approach and interaction. The adaption may be extended to also include robot speed in regard to the distance.

The method is relying on the pose of the person which clearly limits the method as a person standing still may have a direction frontal to the robot while the robot assumes the worst case and assumes it is approaching from the back. However, a pose detection system based on computer vision could easily be integrated into this approach.

To incorporate experiences from encounters the simulated experiment has demonstrated how a CBR system may be trained and used to automatically adapt the Gaussian cost function. Training was demonstrated to significantly change the robots spatial behavior, clearly generating trajectories that do not interfere with person's not interested in interaction. The case database is set up to support future extensions such as location and time.

Further extensions would naturally include the implementation of multi media information as speech and gesture recognition, all variables that may be included in the CBR-system.

In general the results shows a behavior of the robot as expected. Clearly, the experimental work is quite limited and the final proof of concept involving random person's still needs to be done. Before fully integration of the sensors necessary for such experiments hybrid test as "wizard of oz" or semi controlled experiments where the robot is interactively given the information it is lacking, due to limited sensor support, may be conducted.

5 Conclusion

This paper has described and demonstrated by experimentation a spatial robot behavior scheme that supports elements to ensure quality human-robot interaction. A person centered adaptive cost function based on summation of four

Gaussian distributions adjusted to the person's interested in interaction, is introduced. For adjustment of the cost function a novel person evaluator method based on solely the motion of the person, is introduced. The patterns of behavior is further used to train a CBR-system for learning and adaptation of robot motion to a given situation. The proposed control scheme is evaluated by a number of physical experiments that demonstrate the effectiveness of the adaptive cost function approach, which allows the robot to locate itself in front of a person who has expressed interest through his or hers spatial motion.

References

1. Dautenhahn, K.: Methodology & themes of human-robot interaction: A growing research field. International Journal of Advanced Robotic Systems 4(1), 103–108 (2007)
2. Bruce, A., Nourbakhsh, I., Simmons, R.: The role of expressiveness and attention in human-robot interaction. In: Proceedings, AAAI Fall Symposium (2001)
3. Christensen, H.I., Pacchierotti, E.: Embodied social interaction for robots. In: Dautenhahn, K. (ed.) AISB 2005, Hertsfordshire, April 2005, pp. 40–45 (2005)
4. Dautenhahn, K., Walters, M., Woods, S., Koay, K.L., Sisbot, E.A., Alami, R., Siméon, T.: How i serve you? a robot companion approaching a seated person in a helping context. In: HRI Human Robot Interaction 2006 (HRI 2006), Salt Lake City, Utah, USA (2006)
5. Hanajima, N., Ohta, Y., Hikita, H., Yamashita, M.: Investigation of impressions for approach motion of a mobile robot based on psychophysiological analysis. In: IEEE International Workshop on Robots and Human Interactive Communication ROMAN 2005, August 2005, pp. 79–84 (2005)
6. Michalowski, M., Sabanovic, S., Simmons, R.: A spatial model of engagement for a social robot. In: The 9th International Workshop on Advanced Motion Control, AMC 2006, Istanbul (March 2006)
7. Michalowski, M., Sabanovic, S., DiSalvo, C., Busquets, D., Hiatt, L., Melchior, N., Simmons, R.: Socially distributed perception: Grace plays social tag at aaai 2005. Autonomous Robots 22(4), 385–397 (2007)
8. Hall, E.: The Hidden Dimension. Doubleday (1966)
9. Walters, M.L., Dautenhahn, K., te Boekhorst, R., Koay, K.L., Kaouri, C., Woods, S.N., Lee, D., Werry, I.: The influence of subjects' personality traits on personal spatial zones in a human-robot interaction experiment. In: Proc. IEEE Ro-man, Hashville, August 2005, pp. 347–352 (2005)
10. Sisbot, E.A., Clodic, A., Urias, L., Fontmarty, M., Bréthes, L., Alami, R.: Implementing a human-aware robot system. In: IEEE International Symposium on Robot and Human Interactive Communication 2006 (RO-MAN 2006), Hatfield, United Kingdom (2006)
11. Sisbot, E.A., Alami, R., Siméon, T., Dautenhahn, K., Walters, M., Woods, S., Koay, K.L., Nehaniv, C.: Navigation in the presence of humans. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids 2005), Tsukuba, Japan (December 2005)
12. Woods, S., Walters, M.L., Koay, K., Dautenhahn, K.: Methodological issues in hri: A comparison of live and video-based methods in robot to human approach direction trials. In: The 15th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN 2006), Hatfield, UK, September 2006, pp. 51–58 (2006)

13. Kolodner, J.: Case-based reasoning. Morgan Kaufmann Publishers Inc., San Francisco (1993)
14. Ram, A., Arkin, R.C., Moorman, K., Clark, R.J.: Case-based reactive navigation: A case-based method for on-line selection and adaptation of reactive control parameters in autonomous robotic systems. IEEE Transactions on Systems, Man, and Cybernetics 27, 376–394 (1997)
15. Collett, T.H., MacDonald, B.A., Gerkey, B.P.: Player 2.0: Toward a practical robot programming framework. In: Sammut, C. (ed.) Proceedings of the Australasian Conference on Robotics and Automation (ACRA 2005), Sydney, Australia (December 2005), `http://playerstage.sourceforge.net`
16. Bruce, J., Balch, T., Veloso, M.: Fast and inexpensive color image segmentation for interactive robots. In: Proceedings of IROS 2000, Japan (October 2000)
17. Ulrich, I., Borenstein, J.: Vfh*: Local obstacle avoidance with look-ahead verification. In: IEEE International Conference on Robotics and Automation, San Francisco, April 2000, pp. 2505–2511 (2000)

Design of Multi-segment Humanoid Robot Foot

Branislav Borovac and Sinisa Slavnic

Faculty of Technical Sciences, University of Novi Sad
Trg D, Obradovića 6, 21000 Novi Sad, Serbia
borovac@uns.ns.ac.yu, sinisa_s@uns.ns.ac.yu

Abstract. Most of today's humanoid robots are equipped with flat one segment feet. Only few of them have two-segment feet. Two-segment feet can be realized either with passive or active toe joint. Purpose of development of more complex robotic foot is to enable humanoid robots to walk in more natural way. In this paper will be described basic design of multi-segment humanoid foot, currently under development at the Faculty of Technical Sciences, University of Novi Sad. Paper gives short overview of human foot anatomy, and then, robotic foot design characteristics including actuation and sensing.

Keywords: Humanoid robots, multi-segment robotic foot, actuation, sensing.

1 Introduction

Active research in filed of humanoid robotics and progress of technology in last few decades led to development of number high performances humanoid robots. Current development indicates that the spectrum of robotic activities will expand significantly in the near future, particularly in the area of humanoid robotics. For a long time, robots have not been present only in industrial plants, at the time their traditional workspace, but have been increasingly more engaged in the close living and working environment of humans. This fact inevitably leads to the need of "working coexistence" of man and robot and sharing their common working environment. In near future it can be expected that humanoid robots will replace humans on some tasks which does not request human permanent presence (simple tasks at home, factory or even hospital) or moreover where human presence is not desirable (dangerous environments). For fulfillment of diverse tasks in the environment highly adapted to humans the most promising is "human-like" design. The first step that would enable robots to realize tasks in the manner and with the efficiency similar to those of humans is to make robot's structure close to that of humans. Mechanical design of humanoid robots with increased number of mechanical degrees of freedom improved significantly in past few decades, except of humanoid robot foot which remain almost unchanged in past few decades. Having on mind that human foot is one of the most complicated human body parts concerning number of bones, muscles and nerves lead us to conclusion that robot foot need to be improved in order to improve walk of humanoids.

Most of very sophisticated and excellent robots have one-segment flat foot which area is usually bigger than corresponding human foot would be, for example, Honda

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 12–18, 2009.

ASIMO and P2 robots [1], JOHNNIE [2], HRP-2 [3]). This cause that walk is, to great extent, unnatural.

To improve this, we believe, design of foot have to be much more complex and better suited to the needs of walk. First step in this direction is to make two-segment foot. Several research projects dealt with this issue [4-7]. Two-segment robot foot utilize, for example, BARt-UH II (Bipedal autonomous robot) developed at University of Hanover [4], ROBIAN developed at Laboratoire de Robotique de Paris [5], HRP-2LT developed at Intelligent Systems Research Institute AIST [6] and H6 developed at University of Tokyo [7].

BARt-UH, ROBIAN and HRP-2LR feet are equipped with passive (none actuated) toe joints which enable robot to perform a rolling-like motion of the foot. Additionally, ROBIAN foot adds elasticity to the passive joint between toe and the heel using torsion springs. Motion range of the ROBAIN toe joint is 60 degrees. HRP-2LR also has springs at toe joints. The expected role of the toe springs is to accumulate and release energy during robot running.

H6 foot is two segment foot with actuated toe segment. Motion range of the toe joint is 60 degrees for bending toward tibia and 15 degrees from tibia. Toe joint is actuated with DC motor. Researchers which developed H6 robot reported that walking speed is augmented by 80% by utilizing toe joint. Also, the robot was able to climb on higher stairs if toe joint was used.

2 Human Foot Anatomy

Human foot (Fig. 1) is composed of 26 bones, 33 connections (joints) and more than 100 muscles and ligaments, what is not possible to fully replicate in robotic design. Additionally, very important foot feature is sensing of ground reaction force position and its approximate intensity. Thus, this complex body organ provides the only contact of the humanoid with the ground, ensures appropriate feedback information which enables humanoid motion and permanent realization of dynamic balance.

In Table 1 are given average operational ranges of human ankle, toe and foot fingers [8], but it have to be emphasized that in most of the joints only some slight motions are possible, like rolling and sliding. True significance of these motions has not been investigated to full extent yet.

Table 1. Working angles of standard human

Ankle	pitch	-45 deg to 20 deg
	roll	-20 deg to 30 deg
Toe	pitch	-40 deg to 80 deg
Foot fingers	pitch	-40 deg to 60-80 deg

Ankle joint of humanoid robots is usually designed as 2 DOF joint with perpendicular rotation axes. In humans these axes are not perpendicular. Ankle roll axis is rotated 20 degrees toward inner foot side (counter clockwise in transverse plane) and 40 degrees clockwise in sagittal plane as it is depicted on Fig. 1.

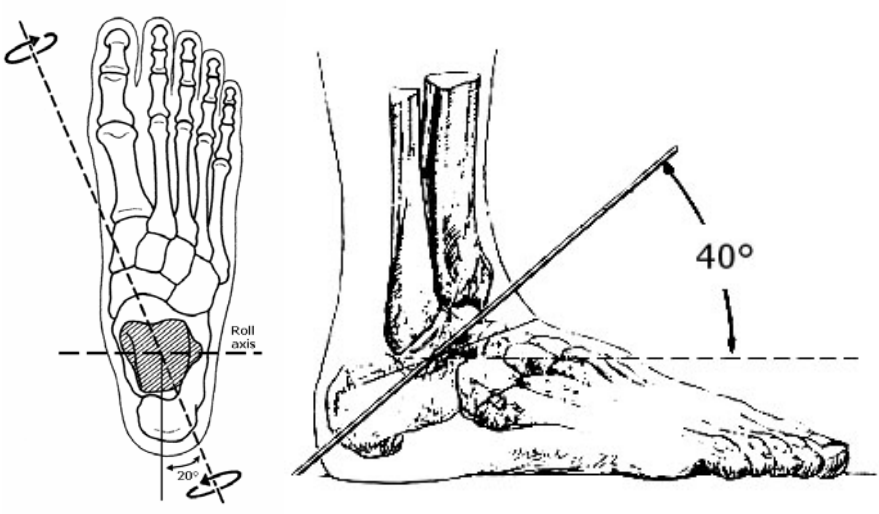


Fig. 1. Orientation of ankle roll axis in humans. Roll axis is rotated 20 degrees toward inner foot side (counter clockwise in transverse plane) and 40 degrees clockwise in sagittal plane.

Multi-segment robot foot needs to satisfy following demands:

- Shape and size of the robot foot should be similar to the shape and size of the adult's human foot,
- Ankle should have 2 DOF joint,
- Fingers need to be actuated,
- Weight of the foot need to be similar to the weight of the adult human foot.

Due to small space available within foot we decided to displace actuation system and to drive foot by driving wires. We also decide to have three fingers instead of five. The decision to keep separate fingers was motivated by wish to avoid all foot fingers to be a single block, but keep fingers as separate "units", to ensure multiple fingers contact with the ground even in case when foot is inclined.

3 Foot Mechanical Design

The robot foot we designed consists of totally 26 parts (Fig. 2). All parts are made of aluminum except of some sliding parts of joints. Thus, weight of the foot itself (without actuation system) has approximate weight of the adult human's foot. Some of the foot parts are designed in more than one version to investigate influence of different designs on foot functionality. One of such parts is ankle joint where we produced two versions (Fig. 3) with different spatial orientations of joint axes. One solution is with perpendicular ankle axes and other is with ankle axes with spatial orientations similar to those of humans. Joint with housing can be simply replaced with another one without any additional change.

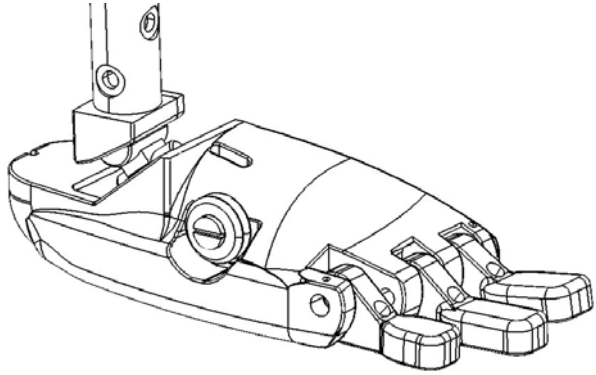


Fig. 2. Picture of the robot foot

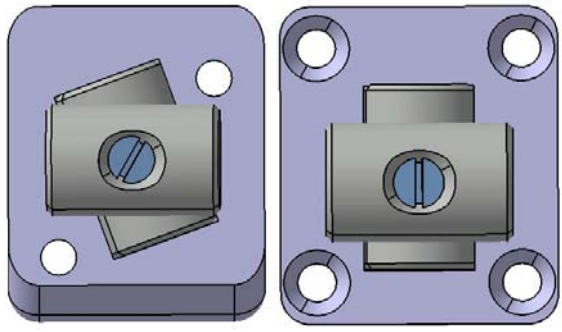


Fig. 3. Two versions of ankle joint that can be used with foot

The foot ankle (Fig. 3) is designed as two sliding cylindrical joints. Rolling bearings are not used to achieve that design would be as compact as possible. Special materials are used for ankle joints to decrease friction between sliding surfaces.

Motion ranges of all joints (ankle and fingers) are same as in humans. Three guide wheels are placed on the foot (Fig. 2) and they will be used for routing fingers driving wires.

4 Actuation

Driving units consisting of a DC motor and a THK LM guide linear actuator will be used for foot actuation (ankle joint). DC motors and a linear actuators shafts are coupled by a timing belts. Therefore, rotational motion of a DC motor is converted to linear motion of the linear actuator's nut block. Due to the high transfer ratio of the linear actuator this system is able to generate high pulling force. Three such driving units will be mounted on each robot's tibia (Fig. 4). Linear actuator's sliding nut will

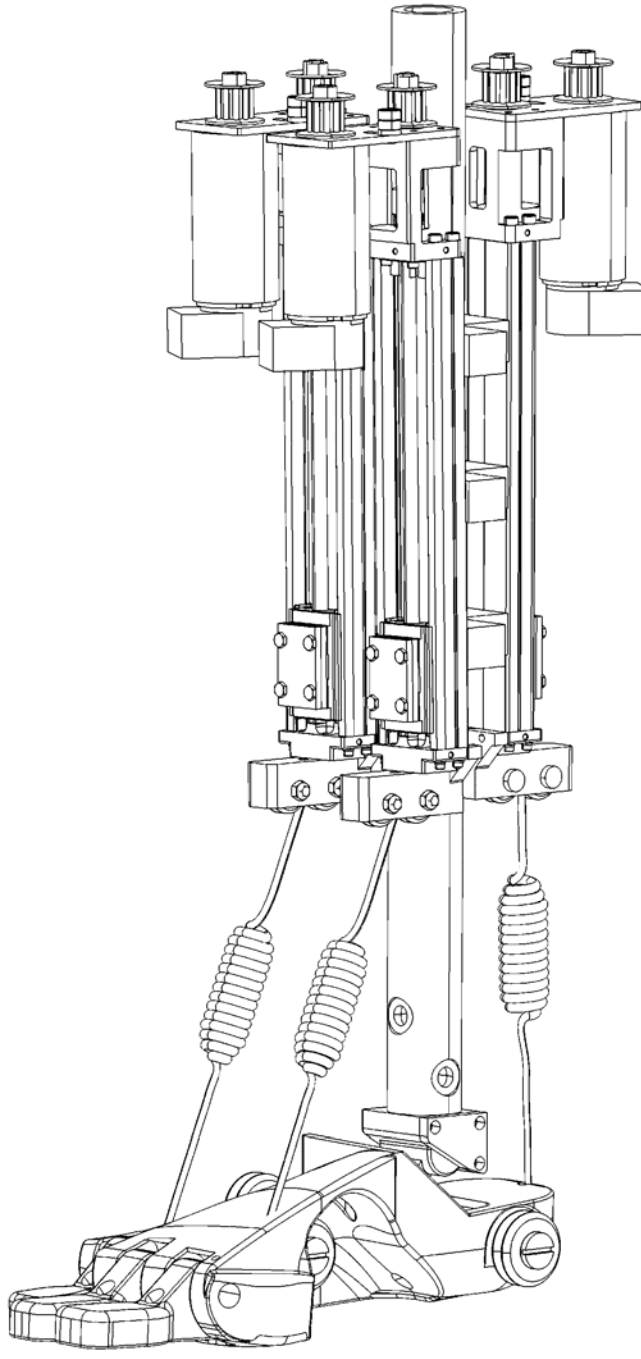


Fig. 4. Robot foot with assembled driving units

be connected with a driving cable to a point on the foot. One driving unit will be used for applying pulling force on the heel and two driving units will be used for applying pulling force on upper front side of the foot. With three driving units it is possible to achieve desired movement in the ankle joints. Using only three driving units has for a consequence slightly complicated control strategy, but weight of the robot leg is significantly reduced. Also, nonlinear springs will be placed on driving wire between linear actuator and the finger in order to introduce elasticity and to enable to control stiffness.

In a first experiments foot fingers will not be actuated. (In Fig. 4 driving wires as well as, actuators for fingers are not shown). Linear springs will be used to introduce elasticity in finger joints. One side of linear springs will be attached on fingers by wires and other side will be connected for tibia by wires too. By bending fingers spring will be extracted. In future experiments intention is to actuate in an active way all tree fingers.

5 Ground Reaction Force Sensing

To enable permanent dynamic balance during walking information about ZMP position is absolutely necessary. By measuring contact pressure between foot and ground or by measuring contact force in several points the ZMP position can be determined[1]. At this moment there are under development several types of sensors appropriate for use at robot foot. They can be classified in two groups: a) miniature, single component sensors which have to be placed at relative large number on the contact surface of the foot (we expect that in this way "profile of the pressure" over foot contact area with the ground can be obtained), and b) bigger sensors which cover larger part of contact foot surface (whole foot may require 3-4 of such sensors). Combination of these two approaches (larger sensors to be placed on foot body, smaller on the fingers) is also possible.

6 Conclusion

In this paper was presented first results of new "human-like" robot foot design. The most of the parts described are already manufactured, but complete assembly and functionality testing has not been performed yet.

We believe the new multi-segment "human-like" robot foot will improve walking capabilities of humanoid robots by achieving better (more vesatile) contact between foot and ground surface. Introducing separate fingers should improve transition from double to single-support phase (push-off phase of gait). As a consequence this will enable more "natural" walking of humanoid robot. We are planning a number of experiments to perform with this foot. We expect the experiments will provide to us additional experience in the course how the design can be further improved.

Our next task will be design of knee and hip joints with appropriate actuation.

[1] Position of the ZMP is same as position of ground reaction force acting point while robot is dynamically balanced.

References

1. Hirai, K., Hirose, M., Haikawa, Y., Takenaka, T.: The Development of Honda Humanoid Robot. In: Proc. of the IEEE ICRA, pp. 1321–1326 (1998)
2. Pfeiffer, F., Löfflerr, K., Gienger, M.: The Concept of Jogging JOHNNIE. In: Proc. of the IEEE ICRA, pp. 3129–3135 (2002)
3. Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., Hirata, M., Akachi, K., Isozumi, T.: Humanoid Robot HRP-2. In: Proceedings of the IEEE ICRA, pp. 1083–1090 (2004)
4. Gerecke, M., Albert, A., Hofschulte, J., Strasser, R., Gerth, W.: Towards an autonomous, bipedal service robot. Tagungsband 10 Jahre Fraunhofer IFF, Magdeburg (2002)
5. Konno, A., Sellaouti, R., Amar, F.B., Ouezdou, F.B.: Design and Development of the Biped Prototype ROBIAN. In: Proceedings of the IEEE ICRA, pp. 1384–1389 (2002)
6. Kajita, S., Kaneko, K., Morisawa, M., Nakaoka, S., Hirukawa, H.: ZMP-based Running Enhanced by Toe Springs. In: Proc. of the IEEE ICRA, pp. 3963–3969 (2007)
7. Nishiwaki, K., Kagami, S., Kuniyoshi, Y., Inaba, M., Inoue, H.: Toe Joints that Enhance Bipedal and Fullbody Motion of Humanoid Robots. In: Proceedings of the IEEE ICRA, pp. 3105–3110 (2002)
8. Krajcinovic, J.: Foot and ankle surgery, textbook (in Serbian). University of Novi Sad, Medical School (1995)

Software-Hardware Mapping in a Robot Design

Pavol Jusko, David Obdrzalek, and Tomas Petrusek

Faculty of Mathematics and Physics, Charles University, Prague
Malostranske namesti 25, 118 00 Praha 1, Czech Republic
pavol.jusko@gmail.com, david.obdrzalek@mff.cuni.cz,
petrusek@gmail.com

Abstract. In this paper[1] we present a way how to change the design of a small robot for Eurobot contest from a design with a lot of hardcoded, hard to maintain and hard to extend functionality to a more universal design with much better maintainability and upgradability by use of software-hardware mapping. In the process, we show how the change of communication topology and software design rework helped to achieve the goal.

Keywords: Autonomous robot design, layered software design, software-hardware mapping.

1 Introduction

Autonomous robot design is a complex process covering many aspects and containing many decisions. This paper describes second-year experience of MART (Mat-phys Robot Team) – a student team based at Faculty of Mathematics and Physics (authors of this paper are MART members) [1].

Our team in its current composition took part in Eurobot autonomous robot contest [2] first in 2007. After this participation we carefully re-evaluated our design in the light of our experiences and observations gained during the work on our robot and during the contest itself.

In this paper, we present the way how we have changed the design and implementation of our robot to be more robust, scalable and simpler to maintain, upgrade and further develop. In our paper we focus on the mapping between hardware and software and leave aside other tasks like mechanical construction, high-level algorithms, overall robot "intelligence" and others.

2 Original Design

Our robot was designed for Eurobot 2007 contest as the first robot of a renewed team. It was build from scratch without reusing older pieces of hardware or previously written software. The core controlling part used standard PC based on mini-ITX

[1] The work was partly supported by the project 1ET100300419 of the Information Society Program of the National Research Program of the Czech Republic.

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 19–28, 2009.

motherboard with Linux operating system, and all mechanical parts were constructed on purely amateur level.

The task we were facing was to design simple, yet robust mechanism of message passing between sensors and effectors at physical layer and their corresponding hardware abstraction layer in the software. Our old hardware communicated using well known RS-232 interface, which is common on personal computers and is easy to program in POSIX environment. The first implementation connected all the peripherals to one microcontroller (MCU) which provided all communications with the PC using its internal serial communication module. To transfer the data, we developed a simple packet oriented protocol.

Since the number of peripherals increased, it became necessary to add another controller. Therefore we had to split the RS-232 link to multiple MCUs (using a simple hardware splitter), and start to use packet addressing. The packets were created in the main control program. Upon reception, the MCU firmware decomposed the packet, checked its consistency, took appropriate action, and sent back an answer message, which contained complex sensor information. Since the MCU response was obligatory and the format of such message was fixed (= full information from the sensors), it can be seen that the protocol complexity gradually raised to an unacceptable level and was no longer simple.

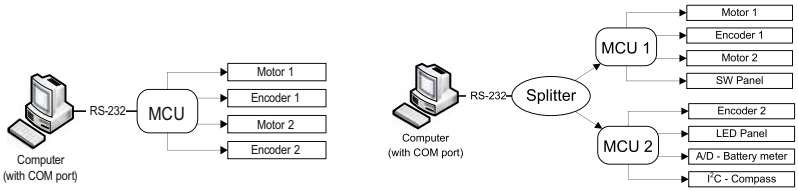


Fig. 1. Original communication design before (left) and after (right) adding new peripherals

As we mentioned above, we wanted to develop universal software for a universal robot which uses any kind of hardware. To achieve this, we decided to implement a layered design (see also [3]). The layers should separate used HW and higher logic of the robot. There were always doubts, how many layers do we need to cover this abstraction and yet not to suffer from too complex design. Too few layers would make the design messy; too many layers would make the design unnecessarily complicated. With this on mind, we have created 3 layers named ”Communication layer“, ”HW abstraction layer“ and ”Smart layer”. From top to bottom (see Fig. 2), the layer functions are:

– The Communication layer handles the RS-232 link and takes care of the packet representation. Its name depicts the functionality from the topmost layer point of view and does not necessarily mean that actual communication with real hardware modules must be performed from this layer (as will be seen in Chapter 4).
– The HW abstraction layer performs two main middle layer tasks: it controls the motors (and generally any actuator) and maintains localization data (and generally any other environmental data). To fulfil these tasks, it is composed of two objects, the Driver and the Localizer. The Driver object handles the motors by setting their

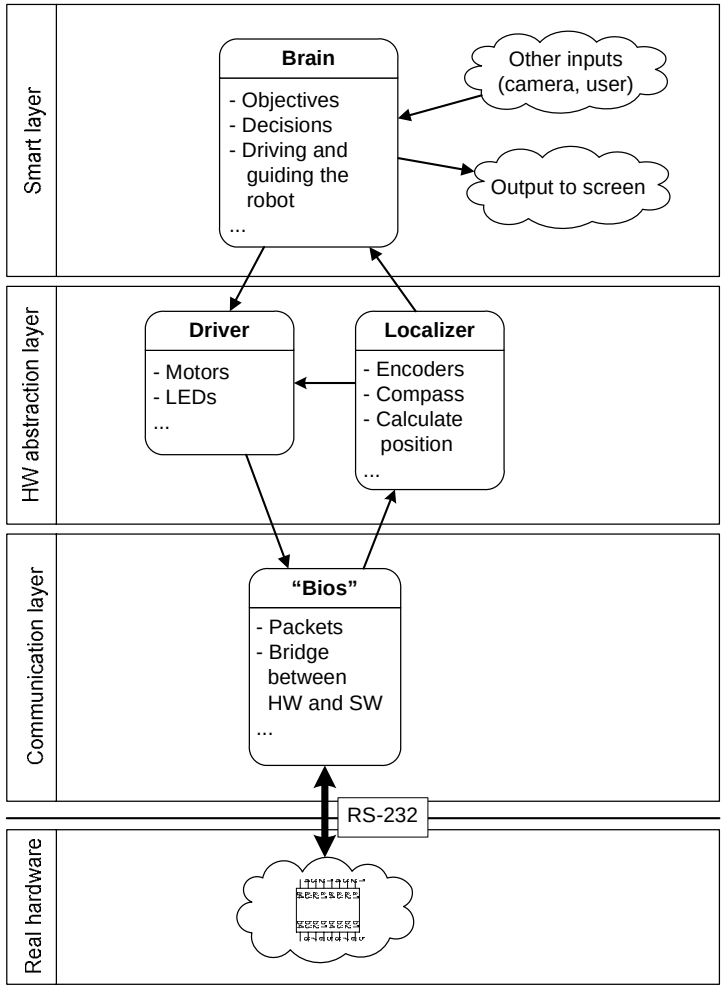


Fig. 2. Original layered software design

speed and braking. It provides the interface to steer easily. The Localizer object reads the information from all devices which can help robot positioning. It computes the exact location and let other objects to read and exploit it.

– The Smart layer contains, as the uppermost level, "The Brain" of our robot. The Brain is the most important part of this layer so, we usually use this name for simplicity to address the whole layer. The Brain itself is implemented as a finite state machine. It tries to achieve all the given objectives, drives the robot and takes care of all the other things which can happen around the robot and are significant for its work. This layer can also use any other peripherals connected by any other way than using our RS-232-based link, for example the USB-connected web-camera or a display.

The communication is the key of successful robot behaviour. For example, the Driver needs some information about the real speed of the wheels to correct the trajectory (e.g. when we want the robot to go straight but due to any reason it diverts slightly), or the Brain reads the position from the Localizer and gives orders to the Driver to achieve its high-level goals (e.g. to navigate to a specific place).

This design fulfilled its goal to be easily adjustable for other tasks. In 2007, we used the robot in two contests: Eurobot [2] and Robotour [4]. They both focus on autonomous mobile robots, but they differ a lot:

- Eurobot is an indoor contest played on a relatively small playing field – a planar table 2x3m, Robotour is an outdoor contest taking place in a huge park with routes made of paver blocks and tarmac.
- Robots in Eurobot contest must meet size limits (maximum height of 35 cm and perimeter of 120 cm). These restrictions do not apply in Robotour, but in this contest the robot can gain bonus point for the ability to carry a 5 kg ballast load, whereas Eurobot robots are not permitted to carry any unnecessary ballast at all.
- During Eurobot contest, two robots compete on the playing field in a 90 sec match and must avoid collisions with the opponent; in Robotour, individual robots start for their 1km long journey in 5 minute intervals and usually travel without any interaction with other robots.
- For navigation during Eurobot contest, the robot can use beacons placed on fixed position around the playing field, but there is no such equivalent in Robotour.

From the start, we designed our robot not to be limited exactly to Eurobot contest. We wanted to create a more flexible platform and it has proved that we have reached this goal. The necessary modifications needed for the robot to be able to switch between Eurobot and Robotour consisted of necessary changes to the traction and required almost no software modifications except the highest level application logic due to different overall contest setup.

In contrast with the fact the robot was successfully used in the two mentioned contests, we have realized severe drawbacks in our design of the hardware and the lowest software layer (Communication layer). Firstly the hardcoded solution of packet handling is not maintainable, nor expandable. For example, when we wanted to add a new peripherals (LED panel, compass etc.), we had to make a lot of changes in the protocol handler (the packet anatomy) and, worse, to make changes even in the hardware, namely to add a new MCU. Further, the use of RS-232 implies certain bitrate, which could cause unacceptably low data bandwidth (or latency) if higher number of devices is connected or if more data transfers are needed.

In our case, we were not able to enjoy the qualities of fast encoders because the maximum packet transfer rate would be exceeded. This situation did not have a solution without complete protocol change, for which we did not have time in the tight contest schedule.

The Eurobot contests rules are changing every year. Also, different contests have different rules and so need different robot implementations. To prepare the robot for a new contest edition or another contest at all, it is obvious the mechanical part of the robot need a change. We also wanted to develop and add new devices to improve the cognition and mobility of the robot and therefore changes in software were necessary too. We used this as an opportunity to improve the software design as well, and the next edition of Eurobot contest was the right impulse to start.

3 Hardware Changes

In order to address the disadvantages we removed the one-to-one concept and implemented new communication based on bus topology. The layered design allowed us to change quite a lot of hardware-related parts with only small changes to the software (see [5]). We have chosen the well known I²C high speed bus [6] with SMBus-based packet protocol [7] as the new transport link. This allowed us to use full I²C high speed transfers while gaining from the SMBus protocol comfort. Our devices are connected to the computer using 3rd party USB to I²C bridge [8]. In Linux, the preprogrammed kernel I²C support allows its easy use (e.g. it handles packet transfers and performs the SMBus checksumming). On the application level, the I²C-connected devices are represented by regular files with regular read/write operations on those files. On the system level, the read/write I²C transfers are implemented using a set of callbacks which convert the binary application data to the SMBus protocol compliant packets and vice versa.

The individual modules for motor control were implemented as a "HBmotor board" (see Fig. 5 or team homepage [1]). The board consists of one MCU controlling the board main functions (Atmel AVR-based microcontroller, in our case), H-Bridge

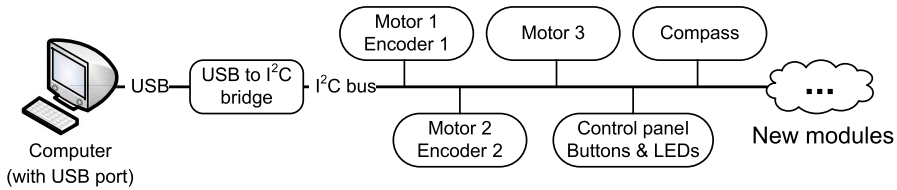


Fig. 3. New communication design

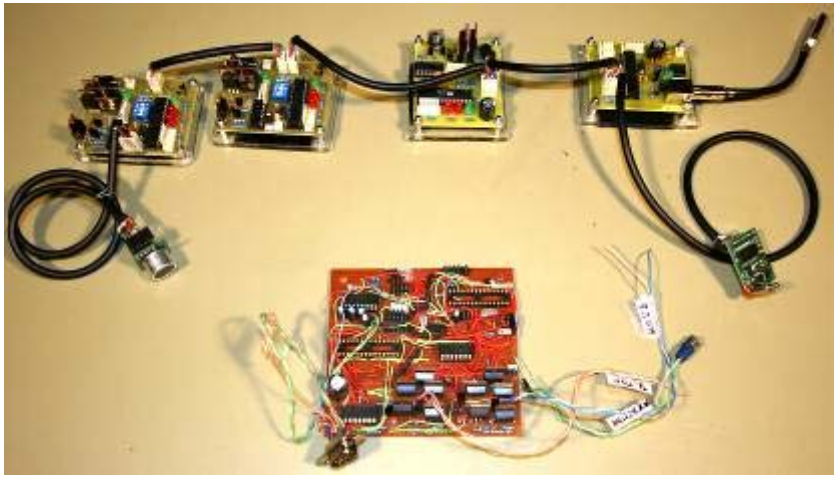


Fig. 4. Comparison of peripheral modules: new I²C version (top), old RS-232 version (bottom)

for motor power drive (standard MOSFET circuit driven by PWM - Pulse Width modulation) and interface for encoders and switches (digital inputs). The MCU firmware implements following tasks:

- control of the H-Bridge operation: enable/disable rotation, direction and PWM signaling for motor rotation control
- motor feedback: input from encoders giving odometry information for the control software
- SMBus slave function for communication between the main control PC and this module.

Even the HBmotor board was designed for motor control, its MCU firmware is reusable for other sensor and effectors boards, for which purpose the hardware specific code must be changed but the core can remain unchanged.

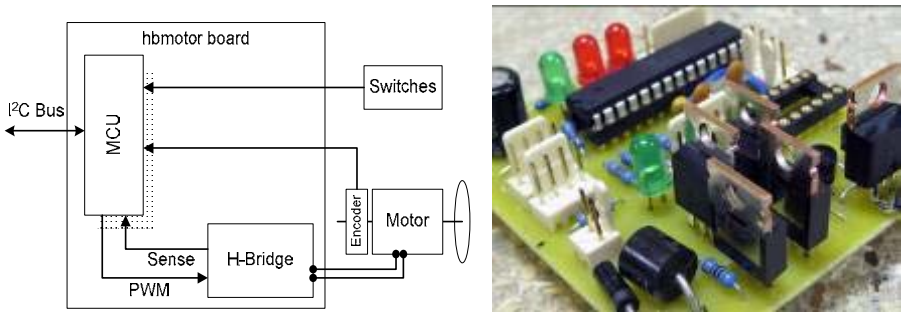


Fig. 5. HBmotor board schematic diagram and photo of implemented board

4 Software Changes

Due to the new hardware design, the modules changed to be independently communicating with the PC. It allowed us to use this behaviour also in the object-oriented software design. The old packet handler called ”Bios“ in the Communication layer became unnecessary and was replaced by independent objects which represent real devices. These objects create a mapping between the software and the hardware. Consequently, more classes appeared at the HW abstraction layer. Their purpose is to provide more information for the Brain and also for other objects in the same layer to increase their power. The diagram (see Fig. 6) may now look bigger, but the design consists of bigger number of smaller independent modules whose size does not reflect code size (in fact, the total size of the code remained roughly the same).

To separate the physical layer implementation (in our case SMBus communication) from the upper level software, we have created Linux kernel modules for handling the devices. These modules use standard Linux virtual filesystem (sysfs) I^2C implementation, which is based on directories and files: each directory represents one device and a file in a directory represents certain ability of the respective device. Our simple approach binds each file to one MCU SMBus register. This design decision separates the hardware implementation (MCU code, choosen bus technology etc.) from the application which uses

this hardware. Besides other advantages, such approach easily allows different people to work on these separate parts (which was also the case of our team).

Every device is now handled by an object used for the communication (via read/write operation upon I^2C interface files). The objects provide the interface to access all needed low-level device data from other objects. On the HW abstraction layer, new classes may be implemented to raise the functionality of the raw devices so that they can be considered "smarter" by the Brain or other objects in the HW abstraction layer. For example, such class could implement moving-average or a sample memory for a range-finder device which standardly provides only a single value representing the range finder actual measurement. Also, with the use of object-oriented programming methods it is now easy to use more devices of the same kind or of the same type origin as their software abstraction will be created using simple instantiation and inheritance.

After changing the communication to asynchronous model, straightforward implementation could easily lead to every single device being handled by a separate thread. However, for certain cases this is not the right way. For example, the encoder information from the two driving motors should be read synchronously to eliminate readout differences and jitter. Using independent threads, the load on the I^2C bus could also cause unwanted harmful peaks because of the unpredictable process scheduling performed by the operating system. To better benefit from the asynchronous communication, we decided to implement a very simple scheduler instead of a completely new threading model. This scheduler helped us to shorten the waiting period between individual cycles of communication with a specific devices (e.g. motors with encoders, in our case via the HBmotor board). This decision lead to improved reaction time of the Driver and the information topicality in the Localizer (see Chapter 2).

As the communication is one of the most important issues to solve to reach the overall goals, we gave special attention to its design and implementation. At the same time with the completeness, we also aimed for simplicity. In our design, the layers communicate strictly with only adjacent layers in a standard way. From top to bottom, the process is following (see also Fig. 6):

- The Brain is implemented as a finite state machine. It uses information from the HW abstraction layer to accomplish its top-level objectives – it reads the robot's position by calling the Localizer (getPosition()) and guides the robot by giving orders to the Driver (gotoXY(position)).
- The orders are propagated from the HW abstraction layer to the Communication layer by setting the right speed for each individual motor. The Driver receives the information about desired direction and the speed from the Brain and regulates the individual motors on each powered wheel using standard PID (proportional–integral–derivative) regulation. The PID regulation needs data from the motor encoders, which is provided by the Communication layer objects via the Localizer.
- Information needed by the Brain for navigation is provided by the Localizer and Range finder objects from the HW abstraction layer. These two objects encapsulate the individual modules information and thus the robot may be easily supplemented by new hardware providing nearly any other type of data. In such a case, the data for the Brain would be processed by the Localizer first and so can be of the same format as before.

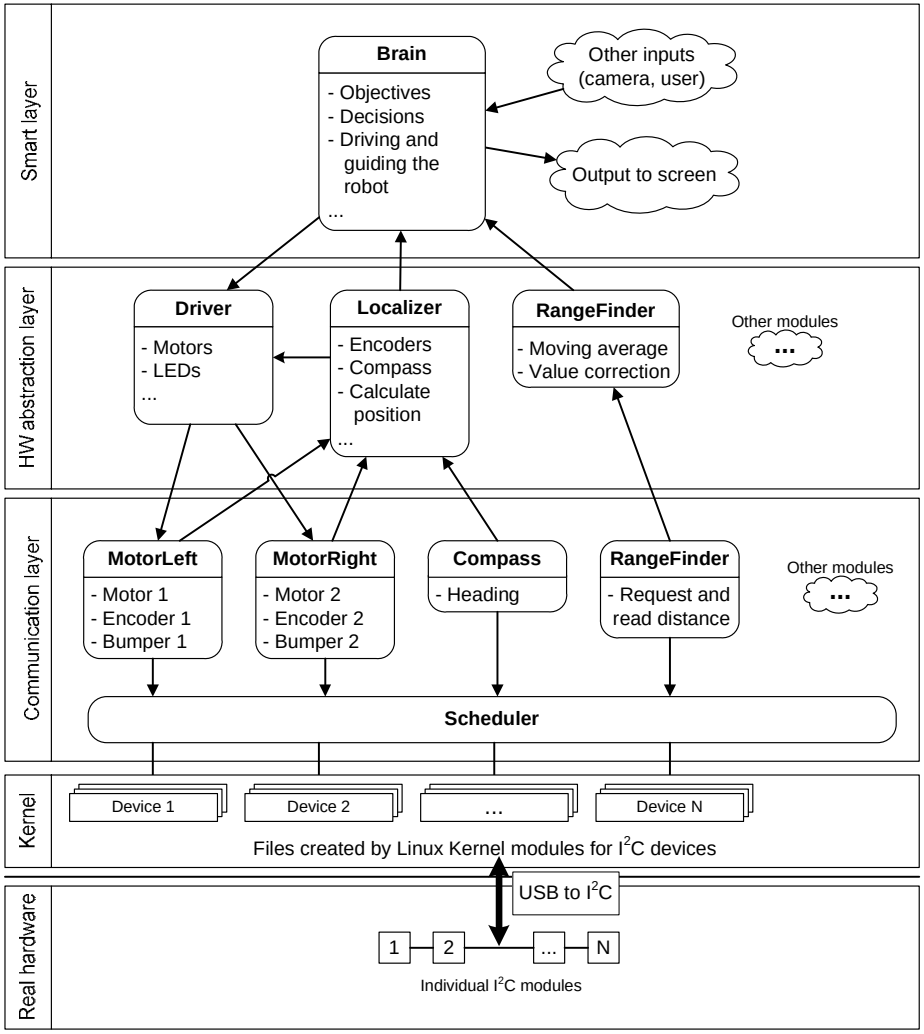


Fig. 6. New layered software design

- Basic task for the objects in the Communication layer is to provide interface between the HW abstraction layer and the operating system Kernel modules (files). Here, the Scheduler takes care of balanced communication.
- To reduce the unnecessary communication and to improve the reaction speed, we decided to implement certain simpler functionality in the (rather low positioned) Communication layer. This functionality can be depicted as a parallel to reflexes, which do not have to be controlled from the high level control modules. For example, without the need to process inputs and give orders by the Brain, the motors are able to work in a simple self-control mode. When the Brain switches the motor to this “goUntilBumper” mode, the motor (or precisely the instance of the motor object) takes care of the necessary action: using the set speed, the real motor is driven

until its bumper (every HBmotor board is equipped with a bumper input) signalizes its activation. Then, the motor is automatically stopped and the information is propagated to upper layers. Such event can be used by the Brain for qualified decision about further robot activity. This approach seems to have a lot of benefits, because it is easily usable by higher layers and at the same time upper layer does not have to deal with it. Furthermore, such automatic reaction is immediate and is not burdened with communication between layers.

5 The Results

With the new design, the maintainability and upgradability of our robot improved a lot. To add any new device now we need only to:

1. Attach it to the I^2C bus and assure it can communicate using SMBus protocol.
2. Provide the kernel module a function responsible for translation between SMBus-transferred data and their corresponding virtual data files for this device.
3. Create a communication and logic class, which knows how to interpret the transferred data. This class may either provide the data "as is" or may perform quite complex data processing to simulate the device capabilities and thus increase its usability.
4. Use the new module from the Smart layer of the software or from other device objects.

For the three first steps, we have prepared (and successfully used during work on our robot) basic prototype code which may be easily adapted with specific new device attributes:

1. A MCU firmware prototype, which provides the SMBus functionality and leaves space for specific device handling.
2. A stub kernel module, which transfers data written to the virtual file onto the I^2C bus and data received from the I^2C bus from its dedicated device into the virtual file buffer.
3. A simple class, which encapsulates the file data and allows standard read / write operations. This class may be arbitrarily extended to provide more complex functionality as mentioned earlier.
4. The fourth step is using the data in the highest layer which obviously cannot have a prototype implementation.

During the depicted process, a lot of hardcoded stuff has been removed and replaced by new code which follows the proposed methods and design decisions. This code is simpler and easier to read and the new design allows improving the robot by adding new devices with only minimal effort needed.

It is now possible to add a wide variety of new devices, e.g. the infrared range-finder, acceleration meter, GPS receiver as new sensors, or new motors, tool handlers or other various actuators and manipulators. Even the change of the transport link from I^2C with SMBus to other standard transport (e.g. CANopen) is easily possible. We plan to add such peripherals to our robot so that it can be used for new contest editions and even for other contests at all. Thanks to the new design, any such single change will not affect other parts of the robot or the software.

6 Conclusion

In our paper, we have shown a new design of a robot as a result of a change from a design with a lot of hardcoded, hard to maintain and hard to extend functionality to a more universal design with much better maintainability and upgradability. The new design has proven its qualities; it provides the functionality of the old design and at the same time it gives much better opportunities for the future.

There are not so many books about designing software for robots. During the adaptation depicted in this paper we mainly used knowledge gained in other areas: we used for example ideas for middleware like those shown by Britton & Bye in [9] or by Tannenbaum & van Steen in [3]. We got a good inspiration for software architecture from [10] by Alexandrescu, and about tier systems from [3]. Furthermore, we have used design patterns from [5] by the famous "Gang of Four" – Gamma, Helm, Johnson, Vlissides; concepts presented in this book has not only helped us a lot but it really pleased us to do so. After we have managed to split the implementation into several separate parts (phases), we were able to try, test and use different programming styles like feature driven programming, test driven programming, pair programming and, when the deadline approached, also extreme programming [11]. Looking back, we can clearly say all of this was a very good lecture for us. Not just from this particular Eurobot project view, but more from the view of general learning process as we exploited all those methods, which we have heard about in a classroom, during the work on a real project.

Acknowledgements

The work was partly supported by the project 1ET100300419 of the Information Society Program of the National Research Program of the Czech Republic. As our work was part of a bigger team project, we would also like to thank other MART team members for cooperation, as well as people from Department of software engineering of Faculty of Mathematics and Physics, Charles University, Prague for their support.

References

1. Homepage of MART team, `http://mart.matfyz.cz`
2. Eurobot Autonomous robot contest, `http://www.eurobot.org`
3. Tannenbaum, A., van Steen, M.: Distributed systems: principles and paradigms. Prentice-Hall, Englewood Cliffs (2007)
4. Robotour competition, `http://robotika.cz/competitions/en`
5. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns: Elements of Reusable Object-oriented Software. Addison-Wesley, USA (1994)
6. I^2C-bus specification, `http://www.nxp.com/i2c`
7. System Management Bus (SMBus) Specification, `http://smbus.org`
8. i2c-tiny-usb, `http://www.harbaum.org/till/i2c_tiny_usb/index.shtml`
9. Britton, C., Bye, P.: IT Architectures and Middleware: Strategies for Building Large, Integrated Systems. Addison-Wesley, USA (2004)
10. Alexandrescu, A.: Modern C++ Design: Generic Programming and Design Patterns Applied. Addison-Wesley, USA (2004)
11. Kadlec, V.: Agilní programování. Computer Press, Praha (2004)

The Huggable Robot Probo, a Multi-disciplinary Research Platform

Kristof Goris*, Jelle Saldien, Innes Vanderniepen, and Dirk Lefeber

Vrije Universiteit Brussel, Robotics & Multibody Mechanics Research Group,
Pleinlaan 2, 1050 Brussel, Belgium
{kristof.goris,jelle.saldien,innes.vanderniepen,
dirk.lefeber}@vub.ac.be
http://anty.vub.ac.be

Abstract. The concept of the huggable robot Probo is a result of the desire to improve the living conditions of children in hospital environment. These children need distraction and lots of information. In this paper we present the concept of this new robot. The robot will be employed in hospitals, as a tele-interface for entertainment, communication and medical assistance. To communicate according to social rules, the robot needs the ability to show facial expressions. Using a well defined set of Action Units (AU) it's possible to express some basic emotions. A prototype of the robot's head, capable of showing these basic emotions is presented. In order to express emotions, an emotional interface is developed. The emotions, represented as a vector in an 2D emotion space, are mapped to the DOF used in the robot. A graphical user interface to control the virtual and real prototype is also presented.

Keywords: Emotional interface, human-robot interaction, huggable robot, multi-disciplinary research platform.

1 Introduction

A hospitalization can have serious physical and mental influences, especially on children. It confronts them with situations that are completely different from the these at home. In hospital, a child's experiences are more limited due to the closed and protective environment, which leads to many difficulties [1]. The social robot Probo will assist in providing information and moral support.

The development of the social robot Probo is part of the ANTY project, of which the main objective is to offer solutions to some specific needs of hospitalized children.

Another aspect of the ANTY project is the creation of a multi-disciplinary research community. The first prototype will be used as a test bed to investigate future possibilities and approaches to anticipate on arising social problems in Probo's work environment. Therefore, collaboration with pediatricians, sociologists and psychologists is a must. New opportunities, such as: Human-Robot Interaction (HRI) and Robot-Assisted Therapy (RAT), will be investigated.

* This work was supported by the Brussels-Capital Region.

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 29–41, 2009.

Besides the development of the prototype and the set up of a multi-disciplinary research community, the project also aims at being an educational stimulant for technological innovation by collaborations with other research groups and (high)schools.

This paper focuses on the conceptual ideas behind the robot Probo and some results of preliminary examinations that lead us to the actual prototype of an actuated robot head, capable of showing facial expressions.

2 Concept Probo

2.1 Operational Goals

In hospital, children need to be distracted from the scary and unfortunate hospital life e.g. by getting in contact with their family and friends. Furthermore, they require moral support and they have specific needs for relevant information about their illness, the hospital environment, medical investigations, etc. [1]. Several projects already exist that aim to use Information and Communication Technologies (ICT) like internet and webcams to allow hospitalized children to stay in contact with their parents, to virtually attend lectures at school and to provide information [2]. However, these ICT applications are usually computer animations displayed on PC, television screens or laptops. Moreover, people are used to interact with embodied creatures and have evolved communication skills, which both need a body for expression [3].

Recently Animal Assisted Therapy (AAT) and Animal Assisted Activities (AAA) are used in specific hospitals [4]. These kind of therapies and activities make use of animals to distract and to comfort the patients. AAT and AAA are expected to have useful psychological, physiological and social effects. However these animals are unpredictable and they can carry diseases. Very recently there were some experiments where robots were used in stead of animals for these kind of therapies. Using these social pet robots for therapeutic purposes has more advantages and a better chance of being allowed in the hospitals. For example, the seal robot Paro [5], [6], Sony's dog robot Aibo [7] and Philips' cat robot iCat [8] are being tested for Robot Assisted Therapy (RAT).

Bearing this in mind, the development of a 3D social robot, called Probo, has started. Communication will be the first focus of this robot. Probo is about 70 cm tall and equipped with a fully actuated head, an active vision system, an affective nonsense speech and a touch screen to comfort, inform and address children in a playful manner. The main goal for the robot Probo is to become a friend for the children.

2.2 Huggable Imaginary Robot Animal

Probo has to be seen as an imaginary animal based on the ancient mammoths. Its name is derived from the word Proboscidea, the order of animals with a proboscis, including the species of the elephant-like mammoths. The basic design of the robot is based on an imaginary animal, so that there is no exact similarity with well-known creatures. That way there are no or less expectations compared with a real animal. For instance, if a robot dog is presented, the children may be expected that the robot dog will bark like a real dog.

(a) Internal mechanics covered with soft materials, like foam

(b) Computerized 3D representation

(c) Removable fur jacket

Fig. 1. Concept of the huggable robot Probo

Probo's huggable and soft appearance, intriguing trunk, and interactive belly-screen are striking. To communicate with the children, the robot is equipped with a fully actuated head, with actuated eyes, eyelids, eyebrows, ears, trunk, mouth and neck. The internal mechanics will be covered with soft and flexible materials, like foam, and on top of it a removable fur-jacket. Figure 1 shows a 3D computer model of the imaginary robot animal, Probo, and the way how the internal mechanics will be covered.

In [9], the relationship between color and emotion was tested, whereas the color green attained the highest number of positive responses (95.9%), followed by the color yellow (93.9%). The majority of emotional responses for the green color indicated the feelings of relaxation and calmness, followed by happiness, comfort, peace, hope, and excitement.

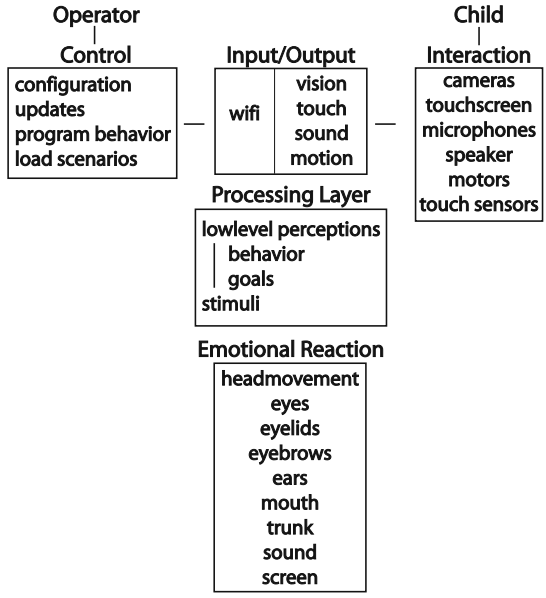


Fig. 2. Probo as a Robotic User Interface

The green color was associated with nature and trees, and thus creating feelings of comfort and soothing emotions. For these reasons Probo's color is chosen green.

2.3 Multi-disciplinary Robotic User Interface

Probo will function as a Robotic User Interface (RUI) between an operator and a child as shown in Figure 2. First, the robot will be controlled by an operator (caregivers, researchers, medical staff, etc.) who wants to communicate with the child. The robot functions as an interface that performs preprogrammed scenarios and reacts on basic input stimuli. The input stimuli, coming from low-level perceptions, are derived from vision analysis, audio analysis and touch analysis. Those stimuli will influence the attention-system and emotion-system, used to set the robot's point of attention, current mood and corresponding facial expression. The vision analysis includes the detection of faces, objects and facial features. Audio analysis includes detecting the direction and intensity of sounds and the recognition of emotions in speech.

A specific behavior-based framework is being developed to process these input stimuli. The framework is based on earlier work of Ortony, Norman and Revelle [10], who focus on the interplay of affect, motivation and cognition in controlling behavior. Each is considered at three levels of information processing: *the reactive level* is primarily hard-wired and has to assure the quick responses of the robot to make it look alive; *the routine level* provides unconscious, un-interpreted scenarios and automotive activity; and the *reflective level* supports higher-order cognitive functions, including behavioral structures and full-fledged emotions. Starting with a *social interface*, the reactive and routine level are being implemented. Currently, there is a shared control between the operator, configuring behavior, emotions and scenarios, and the robot, having basic autonomous reactions.

The development of such a social robot is very multi-disciplinary as shown in Figure 3. From the engineering point of view, the challenge is to build a safe and user friendly robot. For safe and soft interaction the joints need to be flexible, which can

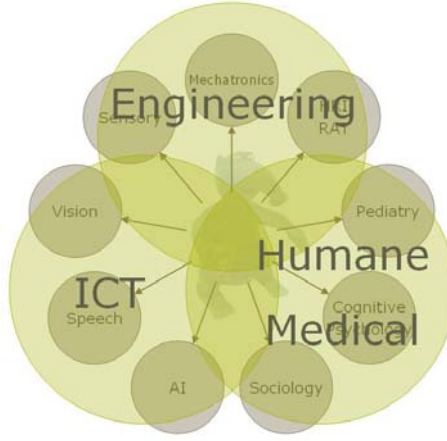


Fig. 3. Multi-disciplinary

be obtained by incorporating compliant actuation. Compliant actuators are gaining interest in the robotic community. Pneumatic artificial muscles [11](such as McKibben muscles, Festo muscles, PPAM [12]), electric compliant actuators (such as VIA [13], AMASC [14] and MACCEPA [15]) and voice coil actuators [16] are some examples of compliant actuators. While some of them exhibit adaptable compliance, so that the stiffness of the actuated joint can be changed, it is not required in the Probo robot. Therefore, compliance is introduced by placing elastic elements between the motor and the actuated robot joint. In this way the external forces on the joint will be dissipated by the elastic elements.

The use of transferable mechanical and electronic components leads to an effective development and realization of the robot prototype. A modular mechanical and system architecture simplifies assemblage and maintenance. To realize a full-body sense of touch, a sensitive skin will be used. A good example is being developed (by Stiehl et al. [17]) for a therapeutic robotic companion named: The Huggable. In another approach, research has started for the use of photonic crystal fibers [18] which will be implemented in some parts of Probo, such as the trunk. Software for these sensors and for vision and speech analysis is developed as components. Component based software engineering emphasizes on decomposition of the engineered systems into functional or logical components with well defined interfaces used for communication across the components. Components are considered to be a higher level of abstraction than objects and as such they do not share state and communicate by exchanging messages carrying data.

3 Actuated Robot Head

3.1 Expression of Emotions

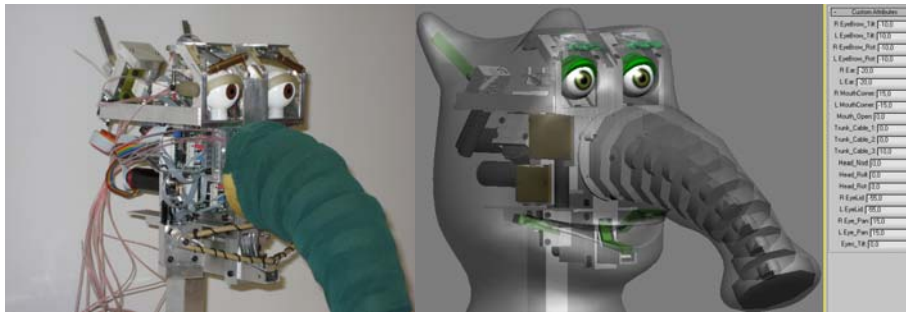
In order to communicate and interact with humans following social rules, Probo needs the ability to express emotions. Therefore, a fully actuated head, for facial expressions, has been developed. In [19] the importance of facial expressions in human face-to-face communication is described. For the display of emotions most of the degrees of freedom (DOF) in its face are based on the Action Units (AU) defined by the Facial Action Coding System (FACS) developed by Ekman and Friesen [20]. AU express a motion of mimic muscles as 44 kinds of basic operation, with 14 well defined AU it becomes possible to express the emotions of anger, disgust, fear, joy, sorrow, and surprise. These emotions are often supported as being the 6 basic emotions from evolutionary, developmental, and cross-cultural studies [21]. Table 1 shows the different DOF compared with other actuated robot heads used for facial expressions. Figure 4a shows the prototype of the actuated robot head, with a total of 17 DOF [22]. Three additional DOF will enable the head to bend, nod and rotate.

3.2 Active Vision System

Besides the role of the eyes to show some facial expressions, there are other reasons to equip a social robot with actuated eyes. The phenomenon that occurs when two people cross their gaze is called eye contact [23]. People use eye-gaze to determine what interests each other. The same phenomenon will be used between robot and child to

Table 1. DOF and ranges of the actuated joints of Probo's head in comparison with other non-humanoid robot heads

Kismet	Eddie	iCat	**Probo**		
(DOF)				**Range [°]**	
Eyes (3)	Eyes (3)	Eyes (3)	**Eyes (3)**	**Pan** **Tilt**	**100** **80**
Eyelids (2)	Eyelids (4)	Eyelids (2)	**Eyelids (2)**		**150**
Brows (4)	Brows (4)	Brows (2)	**Brows (4)**		**45**
Ears (4)	Ears (4)		**Ears (2)**		**90**
Yaw (1)	Yaw (1)		**Mouth (3)**	**Yaw**	**45**
Lips (4)	Lips (4)	Lips (4)		**Lipcorners**	**60**
Neck (3)		Neck (2)	**Neck (3)**	**Rotate** **Nod** **Bend**	**120** **70** **70**
	Crown (1)		**Trunk (3)**		**360**



(a) Prototype of the robotic head (b) Virtual model of the robotic head

Fig. 4. Probo's actuated head with 17 DOF: eyes (3 DOF), eyelids (2 DOF), eyebrows (4 DOF), ears (2 DOF), trunk (3 DOF) and mouth (3 DOF)

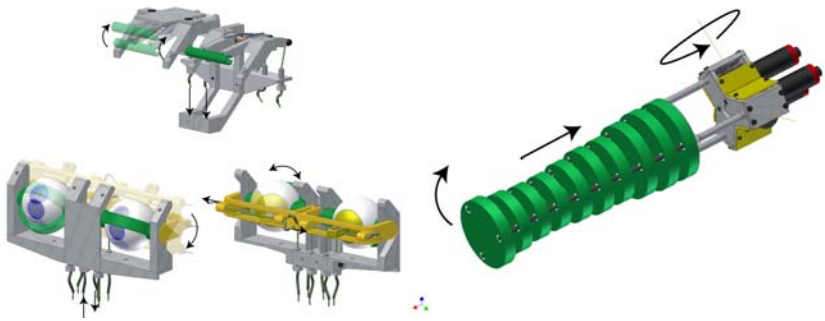
encourage human robot interaction. By focussing the robot's gaze to a visual target, the person that interacts with the robot can use the robot's gaze as an indicator of its intentions. This facilitates the interpretation and readability of the robot's behavior, as the robot reacts specifically to what it is looking at [24]. This *visual target* will be referred to as the robot's *point of attention* (POA). Furthermore, when a robot is intended to interact with people, it requires an active vision system that can fulfill both a perceptual and a communicative function. An active vision system is able to interact with its environment by altering its viewpoint rather than passively observing it. Therefore, the designed eyes are hollow and contain small cameras. As these cameras can move, the range of the visual scene is not restricted to that of the static view.

The five DOF eyes module, used as active vision system in Probo, exists of two hollow eyeballs each mounted in an oribt as shown in Figure 5a. According to the chosen DOF based on the AU mentioned earlier; the eyes can pan separately and tilt together,

each eye can be covered by an upper eyelid and the eyelids can blink separately. The eyebrows module fits on top of the eyes module. Each eyebrow has two DOF meaning that both the vertical position and the angle of each eyebrow can be set independently. Nine servomotors, together with a Bowden cable mechanism are used to power the eyes, eyelids and eyebrows. Axial springs and the usage of flexible cables both introduce compliance. Using flexible Bowden cables creates the opportunity to group and isolate the different servos and to place them anywhere in the robot. That way heat and noise dissipation can be controlled and the head can be held light-weighted, both resulting in a safe design.

3.3 Attractive Trunk

The trunk is a special part in contrast with most other robotic heads (eg. [3], [8], [25]) that use eyes, eyelids, eyebrows and a mouth for facial expressions. The proboscis or trunk of our robot appears to be the most intriguing element according to a small survey amongst children aged 10-13. It is used for grabbing and maintaining the child's attention. When the child's attention is focused on the trunk, the child's face fits within the range of the on board eye cameras. This simplifies the recognition of children's mood or emotional status. In this way the robot can react properly to different situations and it intensifies certain emotional expressions and it increases interactivity.



(a) CAD model of the Eyes and Eyebrows (b) CAD model of the Trunk

Fig. 5. Probo's eyes and trunk

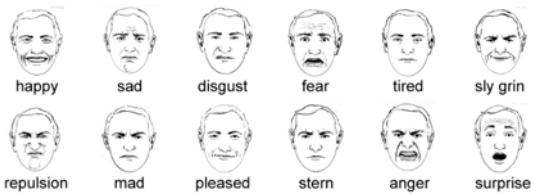
The three DOF trunk, as shown in Figure 5b, consists of a foam core with segmented extension discs. Axial to the centerline, three flexible cables are guided through the discs and attached to the front disc. The end of each cable is attached to a wind-up pulley resulting in a motion of the entire trunk. The motion of the trunk depends on; the number of discs, the dimensions of the discs and the core, the flexibility of the cables and the composition of the foam. A high compliance and durability of the trunk is ensured by using a foam material actuated by flexible cables. Interaction with this trunk will be safe both for the child, that can not get hurt, and for the motors, that can not be broken.

3.4 Recognition Tests

A virtual model (Figure 4b) of Probo has been created to evaluate the design. The model combines the mechanical designs with the visual exterior of the robot, represented by the skin, attached to the mechanical moving parts. The mechanical parts are linked together to obtain kinematical movements for realistic visual motions of the model. The movements can be controlled with sliders to set the desired angles for the DOF and simulating actuation of the parts. To test the recognition of facial expression, the virtual model was used in a preliminary user-study. The study was based on a survey performed by Cynthia Breazeal evaluating the expressive behavior of Kismet [3]. We asked the subjects to compare renderings of the virtual model (Figure 6b) with a series of line drawings of human expressions (Figure 6a).

Twenty-five subjects (6 - 8 years of age) filled out the questionnaire. The children were presented an image of our virtual model representing one of the 8 emotions. For each of those images they had to choose the best matching sketch representing human emotions. The results are shown in Table 2. The results from the test show that the intended emotions *surprise, fear* and *happy* have a low similarity with the sketches. Because the sketches contain also a drawing stating a *pleased* emotion, the low result for *happy* can be explained. Combing the two gives even a 90% similarity between the *happy* emotion and a *happy* or *pleased* human face. The image expressing fear was often related to *sorrow* and *pleased*. There is a strong resemblance between the images representing *fear* and *sorrow* (15%). This can partly be explained because our model lacks lower eyelids resulting in a smaller difference in eye-opening. The lowest similarity was found with the *surprise* emotion, where slightly more children linked the *surprise* image with the *fear* sketch (29%). During the test, the observation was made that the children were really seeking for a visual resemblance without recognizing the underlying emotions.

When performing the same test on fifteen adult people (20 - 35 years of age) the results in Table 3 were similar with the exception of *surprise*. Where the children had difficulties identifying the emotion of surprise most of the adults (81%) had a positive match. We also observed that some of the adults, first try to recognize the underlying emotions rather than just look for a graphical similarity, resulting in better matches.



(a) The sketches used in the evaluation, copied from Kismets survey, adapted from (Faigin 1990) [3]

(b) The 6 basic emotions (anger, disgust, fear, happy, sad and surprise) on the left and the expressions tired and neutral on the right

Fig. 6. Facial expressions used in preliminary user-study

Table 2. The result of the comparison test with children shown in percentage match

% match	happy	sad	disgust	mad	fear	tired	surprise	neutral
happy	**54**	0	7	0	0	0	18	0
sad	0	**74**	9	7	15	2	0	0
disgust	0	4	**62**	4	3	0	0	4
mad	1	2	2	**66**	3	9	0	16
fear	0	0	0	0	**48**	0	29	0
tired	0	4	5	2	0	**87**	3	4
surprise	0	0	0	0	9	0	**28**	0
sly grin	5	0	2	11	5	0	0	0
stern	0	12	9	0	2	0	0	40
anger	2	0	0	3	0	0	7	4
repulsion	2	4	0	7	3	0	0	0
pleased	36	0	4	0	12	2	15	32

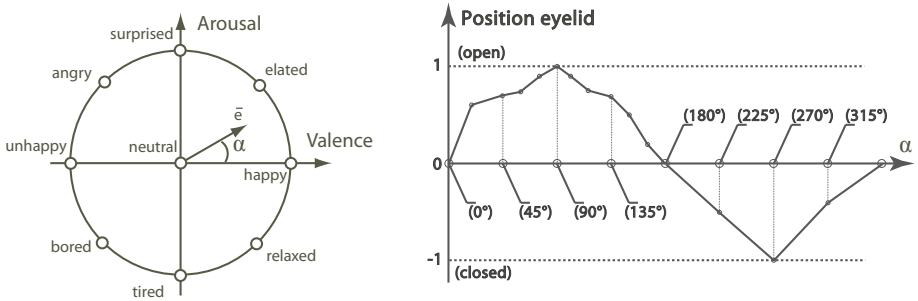
Table 3. The result of the comparison test with adults shown in percentage match

% match	happy	sad	disgust	mad	fear	tired	surprise	neutral
happy	**56**	0	0	0	6	0	13	0
sad	0	**88**	0	0	44	13	0	6
disgust	0	6	**63**	0	0	0	0	0
mad	0	0	6	**69**	0	0	0	6
fear	0	0	0	0	**44**	0	0	6
tired	0	0	6	6	0	**81**	0	44
surprise	0	0	0	0	0	0	**81**	6
sly grin	19	0	6	0	0	0	0	0
stern	0	6	19	19	6	0	0	19
anger	0	0	0	6	0	0	0	0
repulsion	0	0	0	0	0	0	0	0
pleased	25	0	0	0	0	6	6	13

4 Emotional Interface

To realize a translation from emotions into facial expressions, emotions need to be parameterized. In [3], Kismet's facial expressions are generated using an interpolation based technique over a three-dimensional, componential *affect space* (arousal, valence, and stance). Cf. [3] our model has two dimensions; valence and arousal to construct an emotion space, based on the circumplex model of affect defined by Russell [26].

Figure 7a shows the emotion space of Probo. The x-coordinate represents the valence and the y-coordinate the arousal, consequently each emotion $e(v, a)$ corresponds to a point in the valence-arousal-plane (Figure 7a). In this way we can specify basic emotions on a unit circle, placing the neutral emotion $e(0, 0)$ in the origin of the coordinate system. Now each emotion can also be represented as a vector with the origin of the coordinate system as initial point and the corresponding arousal-valence values as



(a) Emotional space based on Russells circomplex model of affect

(b) Adjustable interface for defining the values of the DOF, in this case an eyelid, for each emotion

Fig. 7. Display of emotions

the terminal point. The direction α of each vector defines the specific emotion whereas the magnitude defines the intensity of the emotion. The intensity i can vary from 0 to 1, interpolating the existing emotion $i = 1$ with the neutral emotion $i = 0$. Each DOF that influences the facial expression is related to the current angle α of the emotion vector. An adjustable interface is developed to define the desired value for each angle $(0° - 360°)$ of the different DOF. By selecting one degree of freedom, we set a value for each basic emotion on the unit circle and use linear interpolation to obtain a contiguous relation. By adding more (optional) points or values the curve can be tuned to achieve smooth, natural transitions between different emotions. An example for the degree of freedom controlling the eyelid is shown in Figure 7b.

A graphical user interface (GUI) (Figure 8) has been developed wherein the user can fully configure the facial expressions and use the emotion space to test the different emotions and transitions. The user will obtain visual feedback from a virtual model of the robot. In addition to the facial expression this interface has been extended with a component controlling the point of attention. This component controls the eyes and neck motion according to a specific point in the three dimensional space. The respective coordinates of that point can be altered in real time and will be represented as a red cube in the virtual space. This coordinate is translated into rotation angles for the 4 DOF controlling the eyes (pan/tilt) and the head (pan/tilt). As part from the vision analysis, a face recognition component is developed using Intel®'s OpenCV library. This component uses a webcam to capture the images and then calculates the center of the face as a cartesian coordinate. This coordinate can then be used to control the point of attention in the virtual space. Another component in this interface gives the user the ability to create animations, store, edit and play them. Each animation consists of different key frames, which hold the values of the DOF at a given time. There is a linear interpolation between the different key frames resulting in a contiguous animation. The emotional interface can be used to easily insert emotions at a certain point in an animation. The different animations are stored in a database and will be employed later to build scenarios for the robot.

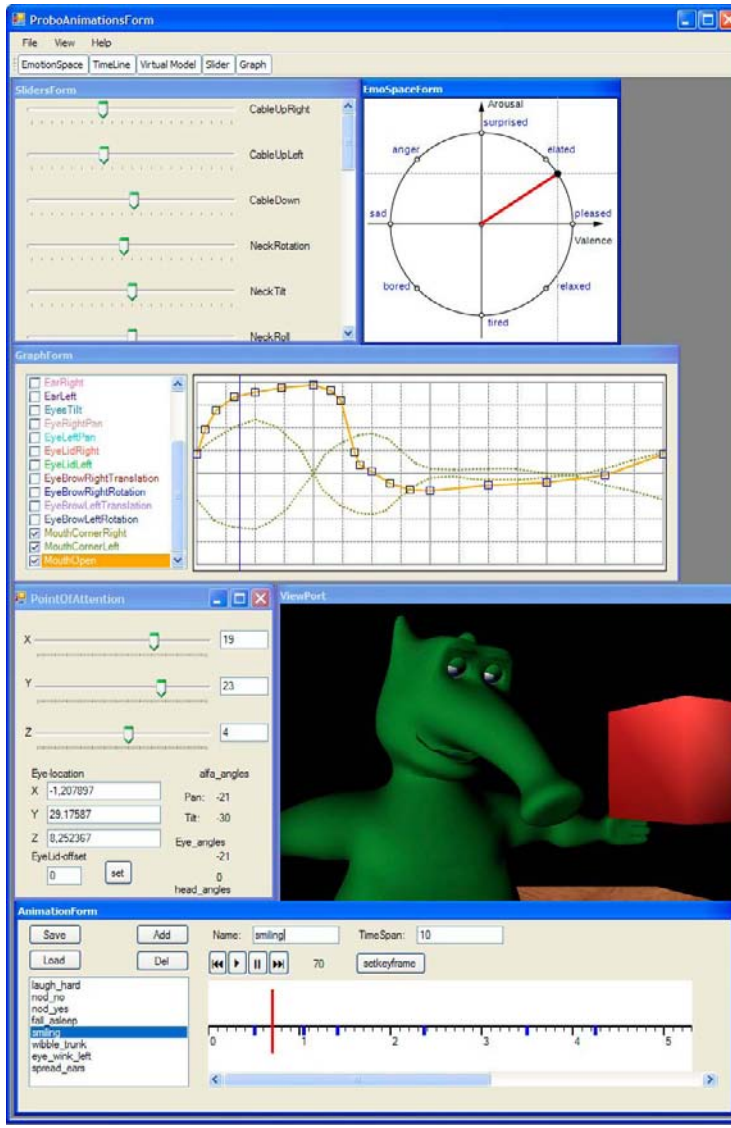


Fig. 8. Graphical User Interface

5 Conclusion

To inform and comfort hospitalized children in a playful way, the development of a new social robot, Probo is started. Probo's main goal is to communicate with the children. Therefor a fully actuated robot head, capable of showing facial expressions, is designed. The degrees of freedom in the head are based on the action units defined in the facial action coding system. Compared with other non-humanoid robot heads Probo has an

intriguing trunk. By us of soft and flexible materials together with compliant actuators, the design is safe and as a consequence safe human robot interaction is assured. With this multidisciplinary test-bed, new opportunities like robot assisted therapy will be explored in collaboration with pediatricians, sociologists and psychologists. To control the robot, an emotional graphical user interface has been developed. With this interface an operator can set all degrees of freedom of the virtual and at the same time linked real model. Different emotions can be showed by moving a cursor in a 2D emotion space.

References

1. Vanderfaeillie, J., Vandenplas, Y.: Zieke kinderen en jongeren. In: Handboek orthopedagogische hulpverlening. Deel 1. Een orthopedagogisch perspectief op kinderen en jongeren met problemen, pp. 199–237. Leuven/Voorburg: Acco (2005)
2. Fels, D., Shrimpton, B., Roberston, M.: Kids in hospital, kids in school. In: Lassner, D., McNaught, C. (eds.) Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2003, Honolulu, Hawaii, USA, AACE, pp. 2358–2363 (2003)
3. Breazeal, C.: Designing Sociable Robots. Mit Pr (2002)
4. Burch, M.: Animal-assisted therapy and crack babies: A new frontier. Pet Partners Program: A Delta Society Newsletter (1991)
5. Shibata, T., Mitsui, T., Wada, K., Touda, A., Kumasaka, T., Tagami, K., Tanie, K.: Mental commit robot and its application to therapy of children. In: Proceedings of 2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, vol. 2 (2001)
6. Shibata, T., Wada, K., Saito, T., Tanie, K.: Robot Assisted Activity for Senior People at Day Service Center. In: Proc. of Int'l Conf. on Information Technology in Mechatronics, pp. 71–76 (2001)
7. Tamura, T., Yonemitsu, S., Itoh, A., Oikawa, D., Kawakami, A., Higashi, Y., Fujimooto, T., Nakajima, K.: Is an Entertainment Robot Useful in the Care of Elderly People With Severe Dementia? Journals of Gerontology Series A: Biological and Medical Sciences 59(1), 83–85 (2004)
8. van Breemen, A.: iCat: Experimenting with Animabotics. In: Proceedings, AISB 2005 Creative Robotics Symposium (2005)
9. Kaya, N., Epps, H.: Relationship between Color and Emotion: A Study of College Students. College Student Journal 38(3), 396–406 (2004)
10. Ortony, A., Norman, D., Revelle, W.: 7 Affect and Proto-Affect in Effective Functioning. Who Needs Emotions? The Brain Meets the Robot (2005)
11. Daerden, F., Lefeber, D.: Pneumatic artificial muscles: actuators for robotics and automation. European Journal of Mechanical and Environmental Engineering 47(1), 11–21 (2002)
12. Verrelst, B., Van Ham, R., Vanderborght, B., Lefeber, D., Daerden, F., Van Damme, M.: Second generation pleated pneumatic artificial muscle and its robotic applications. Advanced Robotics 20(7), 783–805 (2006)
13. Tonietti, G., Schiavi, R., Bicchi, A.: Design and control of a variable stiffness actuator for safe and fast physical human/robot interaction. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 528–533 (2000)
14. Hurst, J., Chestnutt, J., Rizzi, A.: An actuator with physically variable stiffness for highly dynamic legged locomotion. In: Proceedings of 2004 IEEE International Conference on Robotics and Automation. ICRA 2004, p. 5 (2004)
15. Van Ham, R., Vanderborght, B., Van Damme, M., Verrelst, B., Lefeber, D.: MACCEPA, the mechanically adjustable compliance and controllable equilibrium position actuator: Design and implementation in a biped robot. Robotics and Autonomous Systems 55(10), 761–768 (2007)

16. McBean, J., Breazeal, C.: Voice Coil Actuators for Human-Robot Interaction. In: IEEE. RSJ International Conference on Intelligent Robots and Systems (IROS 2004), Sendai, Japan (2004)
17. Stiehl, W., Lieberman, J., Breazeal, C., Basel, L., Lalla, L., Wolf, M.: Design of a therapeutic robotic companion for relational, affective touch. In: IEEE International Workshop on Robot and Human Interactive Communication, 2005. ROMAN 2005, pp. 408–415 (2005)
18. Urbanczyk, W., Martynkien, T., Szpulak, M., Statkiewicz, G., Olszewski, J., Golojuch, G., Wojcik, J., Mergo, P., Makara, M., Nasilowski, T.: Photonic crystal fibers: new opportunities for sensing. In: Proceedings of SPIE, vol. 6619, p. 66190G (2007)
19. Mehrabian, A.: Communication without words. Psychology Today 2(4), 53–56 (1968)
20. Ekman, P., Friesen, W.: Facial Action Coding System. Consulting Psychologists Press (1978)
21. Ekman, P.: Are there basic emotions? Psychological review 99(3), 550–553 (1992)
22. Goris, K., Saldien, J., Vanderborght, B., Lefeber, D.: The Huggable Robot Probo: Design of the Robotic Head. In: AISB symposium (accepted) (2008)
23. Breazeal, C.: Regulation and Entrainment in Human-Robot Interaction. Experimental Robotics VII (2001)
24. Miyauchi, D., Sakurai, A., Nakamura, A., Kuno, Y.: Human-robot eye contact through observations and actions. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004, vol. 4 (2004)
25. Beira, R., Lopes, M., Praça, M., Santos-Victor, J., Bernardino, A., Metta, G., Becchi, F., Saltaren, R.: Design of the Robot-Cub (iCub) Head. In: Proc. IEEE International Conference on Robotics and Automation, ICRA, Orlando (May 2006)
26. Posner, J., Russell, J., Peterson, B.: The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology. Development and Psychopathology 17(03), 715–734 (2005)

Comparison of PID and Fuzzy Logic Controllers in Humanoid Robot Control of Small Disturbances

Miomir Vukobratović[1], Branislav Borovac[2], and Mirko Raković[2]

[1] Institute Mihajlo Pupin, Volgina 15, 11000-Belgrade, Serbia
vuk@robot.imp.bg.ac.yu
[2] University of Novi Sad, Faculty of Technical Sciences, 21000-Novi Sad, Trg Dositeja Obradovića 6, Serbia
borovac@uns.ns.ac.yu, rakovicm@uns.ns.ac.yu

Abstract. Presence of permanent perturbations during walk requires simultaneous control of dynamic balance preservation and correction of internal synergy to bring it as close as possible to reference one. It is not answered yet in full extent how control for each of these tasks have to be synthesized. In this paper is discussed use of PID and fuzzy control for these purposes. In simulation experiment we applied for dynamic balance preservation only PID controller, but for internal synergy compensation two different controllers (PID and fuzzy) were applied. Obtained results were compared and discussed.

Keywords: Humanoid robots, ZMP (Zero Moment Point), dynamic balance, fuzzy control.

1 Introduction

Although the problem of bipedal gait has been in the focus of researchers for almost forty years, not all aspects of the control synthesis have been satisfactorily addressed yet. From humanoid robot control system is required to ensure simultaneous realization of a coordinated and functional motion of the joints (realization of the given gait type) and constant preservation of dynamic balance (preventing the robot's overturning during the walk).

All robot's movements are controlled in joint state space (internal synergy), whereas the verification of the realization efficiency is based on the robot's behavior in the external (Cartesian) coordinates (external synergy). In order to have the control task realized in one and verified in another state space it is necessary to have a unique relationship between these two spaces. During the regular gait, this unique relationship is ensured by fulfilling the requirement for preserving dynamic balance, which is manifested as the requirement that at least terminal link of the kinematical chain of the supporting leg (or of both legs in the case of double-support phase) is immobile with respect to the ground. This requirement is fulfilled if ZMP [1-6] is permanently within support area [7,8], either in single or double support phase.

Each deviation in the motion of the joints (deviation of internal synergy from the reference one) causes a deviation of the ZMP from its reference position, jeopardizing

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 42–53, 2009.

thus dynamic balance [5,6,8]. It is well known that the disturbances can be compensated for in different ways, depending of their type, complexity of the humanoid's mechanical structure (number of DOFs and their disposition) and, of course, disturbance intensity[1] [9]. Corrective actions can be planned and undertaken with the aim of preserving dynamic balance or bringing closer internal synergy to the reference one. It may easily happen that the compensation action aimed at bringing closer internal synergy to the reference produces as a side effect additional increase in deviation of the ZMP from the reference, jeopardizing thus dynamic balance. Thus, it can be stated that additional task of the control system part aimed to bring closer internal synergy to the reference one is not to jeopardize dynamic balance "too much". In "fine tuning" of the control system for internal synergy recovery priority should not be given to fast decrease of system deviation from reference motion (deviation decrease can be realized in one or even more half-steps), but priority should be "not-jeopardizing" of dynamic balance.

Since there is no use to perfectly realize the internal synergy while the humanoid is falling, we hope that it is quite straightforward that the priority must be given to preventing the humanoid's falling, i.e. to compensating for the ZMP deviation. Only then when the humanoid is not explicitly endangered (there is no direct threat of falling) the compensation activity can be "shared" between tasks of the following internal synergy and minimization of the ZMP deviation from its reference position.

In our previous paper [9] we already investigated control structure of PID regulator applied in each of these tasks. In this paper, for compensation of the ZMP deviation was designed PID regulator with variable feedback gains. For internal synergy compensation were compared two approaches: again PID regulator and fuzzy regulator. Obtained results are compared.

2 Description of Mechanical Structures of the Mechanism

In this section we describe the kinematical schemes of robot's mechanical configuration [9] that was used in the present work. The basis for deriving the mechanism's mathematical model is the software for forming the dynamic model of a branched, open or closed, kinematical chain whose links are interconnected with joints having only one DOF. The structure of the mechanism having 54 DOFs, used in the present work, is shown in Fig. 1. The first kinematic chain represents the legs (links 1-27), the second chain extends from the pelvis and comprises the trunk and the right hand (links 28-51), and the third chain (links 52-54) forms the left shoulder and arm. The multi-DOF joints were modeled as a set of "fictitious" links (massless links of zero-length) interconnected with the joints having one DOF. For example, the hip joints, which are in reality spherical joints with three DOFs, are modeled as sets of three one-DOF joints whose axes are mutually orthogonal. Thus the right hip is modeled by

[1] Small disturbances are defined as disturbances that can be compensated for during the realization of internal synergy (the deviations of internal synergy are constantly diminishing), whereas in the case of large disturbances tracking of internal synergy is temporarily abandoned, compensation action realized with the aim to avoid falling down (e.g. by stepping aside), and, when dynamic balance is re-established, the tracking of internal synergy is continued. In this paper only small disturbances are considered.

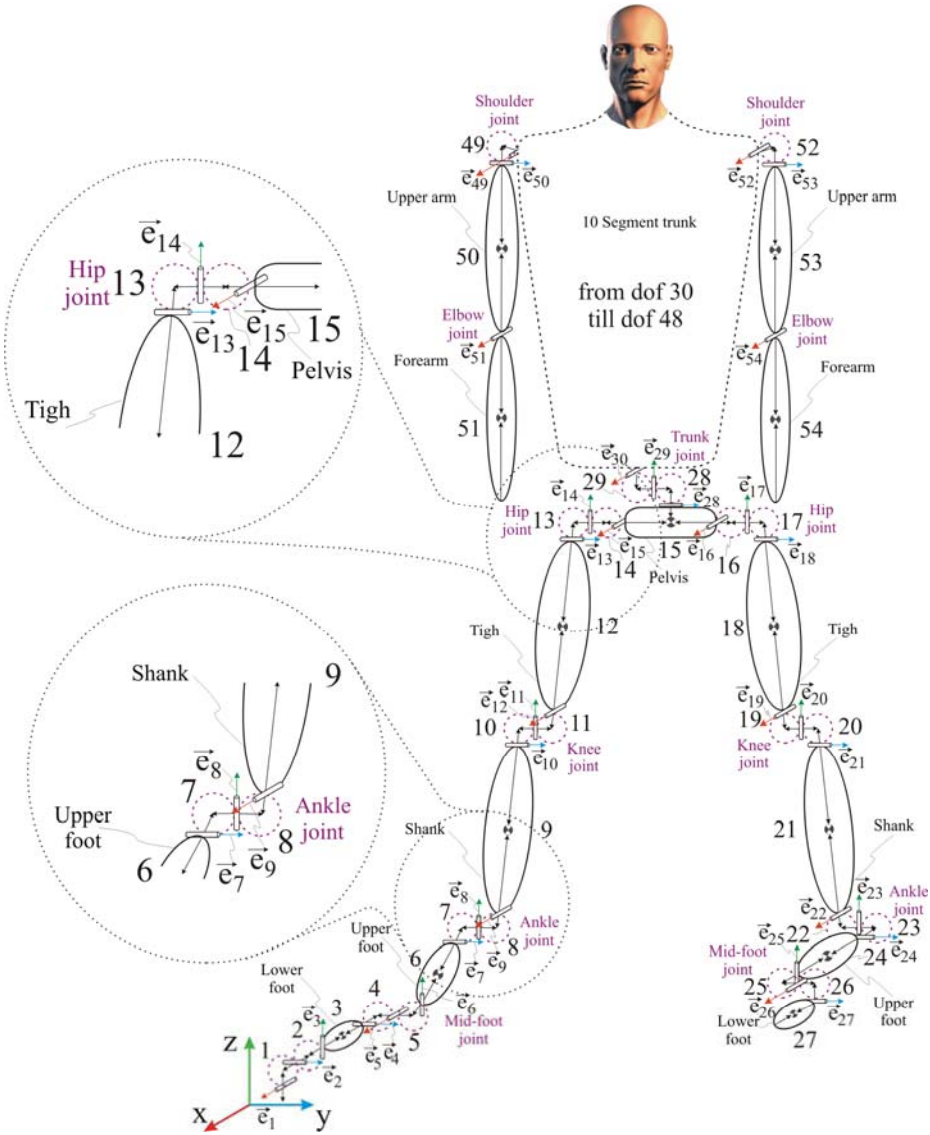


Fig. 1. Schematic of the robot's basic mechanical configuration having 54 DOFs

a set of simple joints 13, 14 and 15 (with the unit vectors of rotation axes $\vec{e}_{13}$, $\vec{e}_{14}$ and $\vec{e}_{15}$), and the left hip by the set of joints 16, 17 and 18 (the unit vectors $\vec{e}_{16}$, $\vec{e}_{17}$ and $\vec{e}_{18}$). The links connecting these joints (for the right hip the links 13 and 14, and for the left links 16 and 17) were needed only to satisfy the mathematical formalism of modeling a kinematic chain. The other links (those that are not part of the joints with more DOF's) whose characteristics correspond to the links of an average human body (link 9 corresponds to the shank, link 12 to the thigh, etc.) are presented by solid lines in Fig. 1. In the same figure, the links that were needed only for modeling "complex"

joints with more DOFs (having no mass and with the moment of inertia and length being equal zero) are presented by dashed lines, to indicate their "fictitious" nature.

Of special importance is the way of modeling the foot-ground contact in order to determine the exact position of the ZMP during the motion and observe the moment when the mechanism is out of dynamic balance. The loss of dynamic balance means that the mechanism collapses by rotating about one of the edges of the supporting foot, and this situation, obviously, has to be prevented. The contact of the mechanism with the ground is modeled by two rotational joints, determined with the unit vectors $\boldsymbol{e_1}$ and $\boldsymbol{e_2}$ (Fig. 1), mutually perpendicular. At the ZMP for dynamically balanced motion, it is constantly ensured that $\mathbf{M_Y} = 0$ $((\mathbf{M_X} \perp \mathbf{M_Y}) \wedge (\mathbf{M_X}, \mathbf{M_Y} \in \mathrm{XoY}))$. It should be especially emphasized that the mechanism feet were modeled as the two-link ones. In Fig. 1 the anterior part of the right foot (toes) is presented by link 3, and its main part (foot body) by link 6. The toes of the left foot are presented by link 27 and the foot body by link 24. The trunk is divided into several links (in this work it was modeled as being composed of 10 links) interconnected with the joints having two DOFs each (rotation about the y-axis (inclining forward-backward) and rotation about the x-axis (inclining left-right)).

The motion of all the links of the locomotion system was determined on the basis of the semi-inverse method using predescribed motion of the legs and predefined trajectory of ZMP shown on Fig. 2. for one half step.

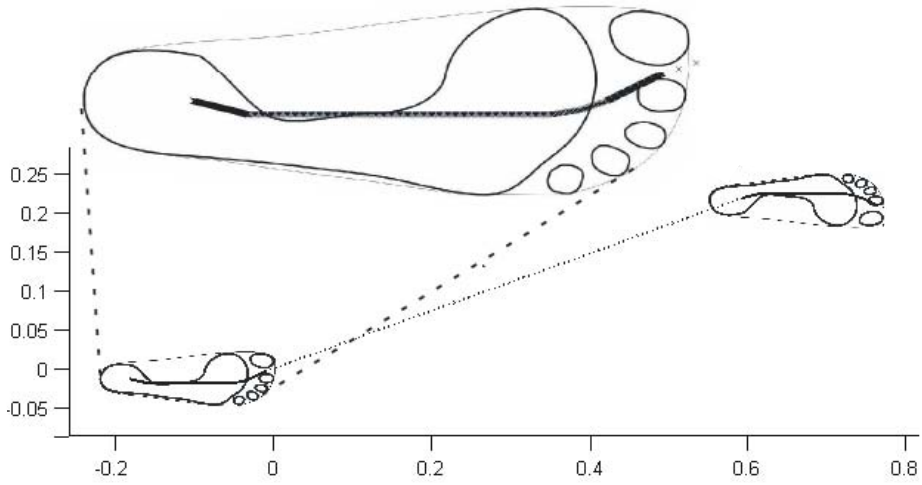


Fig. 2. Reference trajectory of ZMP for one-half step

In this way we obtained the reference motion (motion without any disturbance) of the mechanism.

3 Simulation Experiment

In this section control structure is adopted and simulation experiment are described.

3.1 Description of Experiment and Control System for Dynamic Balance

In simulation experiment deviation in joints reference angles (perturbation) were introduced at the beginning of the simulated motion. In hip and ankle (joints with unit vectors $\boldsymbol{e_9}$, $\boldsymbol{e_{15}}$, $\boldsymbol{e_{16}}$ and $\boldsymbol{e_{22}}$) were added $+5^o$ or -5^o to obtain system posture as shown in Fig. 3. As a consequence of change in system posture deviation in ZMP position will also appear. Simulation lasted one half-step.

Control aim was to eliminate angular deviations in system joints while walking. This means that two tasks have to be performed simultaneously: elimination of deviation in joint angles and preservation of dynamic balance.

Accordingly, in each time instant t_i, the total control input at each joint consists of two parts: reference control (obtained for the motion without disturbances) and correction part, which depend of the disturbance intensity. In other words,

$$u_{n_i}^{(t_i)} = u_{\text{reference } n_i}^{(t_i)} + \Delta u_{\text{at joint } n_i}^{(t_i)} . \tag{1}$$

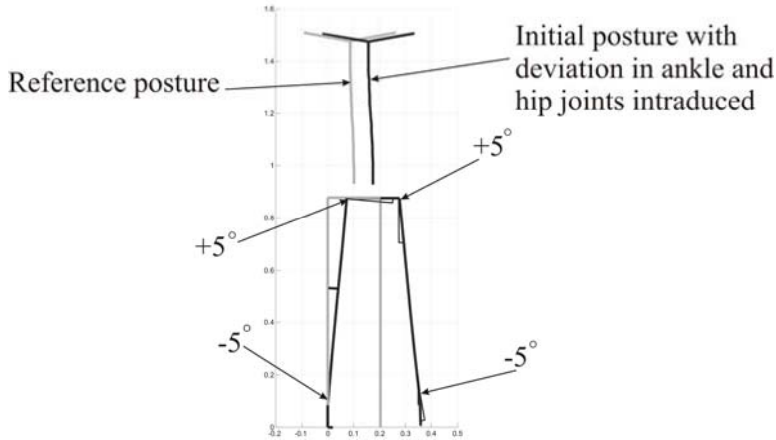


Fig. 3. Angular deviations at joints at the beginning of the simulated motion

The corrective part $\Delta u_{\text{at joint } n_i}^{(t_i)}$ consists also of two parts:

- The most important task is to prevent system to overturn. Thus, one part of the corrective control serves to minimize the ZMP position deviation from the reference $\Delta u_{\text{ZMP at joint } n_i}^{(t_i)}$, i.e. this control should preserve dynamic balance. This task can be allocated to one or more mechanism's joints.
- The other part of the corrective control ($\Delta u_{\text{local at joint } n_i}^{(t_i)}$) should ensure minimization of the deviation of the actual synergy from the reference one at each joint. This part of the corrective control we call local control, as the regulators involved act at the individual joints (locally).

The task of compensating for ZMP deviation should not be assigned to every joint at the same time, but may be allocated only to certain joints. Therefore, if we want to compensate for the ZMP deviations in the x-direction with the aid of the joint n_i, the control law will be of the form:

$$\Delta u_{ZMP\,at\,joint\,n_i}^{(t_i)} = k_{p\,ZMP\,n_i} \cdot \Delta ZMP_x^{(t_i)} + k_{i\,ZMP\,n_i} \cdot \sum_{i=1}^{t_i} \Delta ZMP_x^{(i)} + \\ + k_{d\,ZMP\,n_i} \cdot \left(\Delta ZMP_x^{(t_i)} - \Delta ZMP_x^{(t_i-1)}\right) \tag{2}$$

where k_{pZMPn_i} is the position feedback gain at the joint n_i, for the compensation of ZMP deviation; k_{iZMPn_i} and k_{dZMPn_i} are integral and derivative feedback gains at the joint n_i, also for compensating the ZMP deviation; $\Delta ZMP_x^{(t_i)}$ stands for the deviation of ZMP in the direction of the x-axis at a time instant t_i. Analogously to (2) corrective control for ZMP deviation in y-direction can be obtained in form:

$$\Delta u_{ZMP\,at\,joint\,n_i}^{(t_i)} = k_{p\,ZMP\,n_i} \cdot \Delta ZMP_y^{(t_i)} + k_{i\,ZMP\,n_i} \cdot \sum_{i=1}^{t_i} \Delta ZMP_y^{(i)} \\ + k_{d\,ZMP\,n_i} \cdot \left(\Delta ZMP_y^{(t_i)} - \Delta ZMP_y^{(t_i-1)}\right) . \tag{3}$$

It should be borne in mind that the axis of the joint performing compensation should be perpendicular to the direction of ZMP deviation. Thus, Eq. (2) is applied in joints 7 (ankle), 13 (hip) and 28 (trunk), while Eq. (3) is applied in joints 9 (ankle), 15 (hip) and 30 (trunk). At joints of the leg in swing phase compensation obtained for leg in support phase has been applied. For example, compensation obtained for joint 15 (hip of supporting leg), is also applied at joint 16 (hip of the leg in swing phase).

Table 1. Feedback gains for ZMP compensation and the corresponding increments and decrements

Feedback gain	Joint	Min.	Max.	Increment	Decrement
	7 and 13	1	9	0.1	0.01
k_{pZMP}	9 and 15	5	13	0.1	0.01
	28 and 30	2	10	0.1	0.01
k_{iZMP}	all joints	3	3.08	0.0001	0.00001
k_{dZMP}	all joints	2	2.8	0.001	0.0001

To ensure priority of the preservation of the dynamic balance over internal synergy compensation coefficients in Eqs. (2) and (3) are not constant, but depend on ZMP deviation intensity in x and y direction. If ZMP is in 5mm wide zone with respect to ZMP reference position coefficient is set to its minimal value. As ZMP exit out of the 5 mm zone feedback gains start increase by increments specified in Table 1 up to its maximal value. If ZMP return to 5 mm zone it decrease by decrement from Table 1 up to its minimal value. In Table 1 are, for all joints involved in dynamic balance preservation, specified minimal and maximal feedback gains, as well as corresponding increments and decrements.

3.2 PID and Fuzzy Regulator

Regulators for internal synergy compensation were applied at same joints as regulators for ZMP deviation compensation, i.e. at joints 7, 9, 13, 15, 28 and 30. Also, compensation obtained for joints of the leg in support phase was applied at joints of the leg in swing phase. For all other joints only reference control was applied.

Two approaches were applied in internal synergy compensation: PID regulator and fuzzy logic regulator. In both cased "additional amount of control" ($\Delta u^{(t_i)}_{\text{local at joint } n_i}$) was added to reference control.

PID regulator for internal synergy compensation is defined in following way:

$$\Delta u^{(t_i)}_{\text{local i at joint } n_i} = k_{p\, n_i\, \text{local}} \cdot \Delta q^{(t_i)}_{n_i} + k_{i\, n_i\, \text{local}} \cdot \sum_{i=1}^{t_i} \Delta q^{(i)}_{n_i} + k_{d\, n_i\, \text{local}} \cdot \left(\Delta q^{(t_i)}_{n_i} - \Delta q^{(t_i-1)}_{n_i} \right). \qquad (4)$$

Coefficient in (4) are not constant, but also depend on ZMP position: if ZMP is within 5mm zone coefficients increase by defined increment up to maximal values, but if ZMP is out of the 5mm zone coefficients decrease by defined decrement up to minimal values (Table 2). In this way is ensured that when dynamic balance is not endangered priority is given to achieving reference internal synergy. Otherwise, priority is given to fall prevention.

Another approach applied was to design fuzzy controller. Block diagram of fuzzy controller is shown on Fig. 4. Controller is of mamdani type and has two inputs and one output. Inputs for controller are the error in joints and error derivation in joints 7, 9, 13, 15, 28 and 30:

Table 2. Feedback gains for local regulators and the corresponding increments and decrements

Feedback gain	Joint	Min.	Max.	Increment	Decrement
$k_{p\ \text{local}}$	All joints	10	200	0.1	10
$k_{i\ \text{local}}$	All joints	2	3	0.0005	0.05
$k_{d\ \text{local}}$	All joints	3	8	0.002	0.2

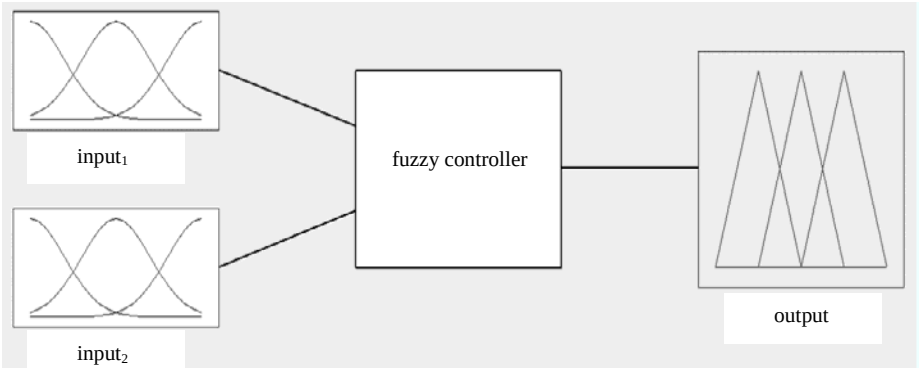


Fig. 4. Block diagram of fuzzy controller

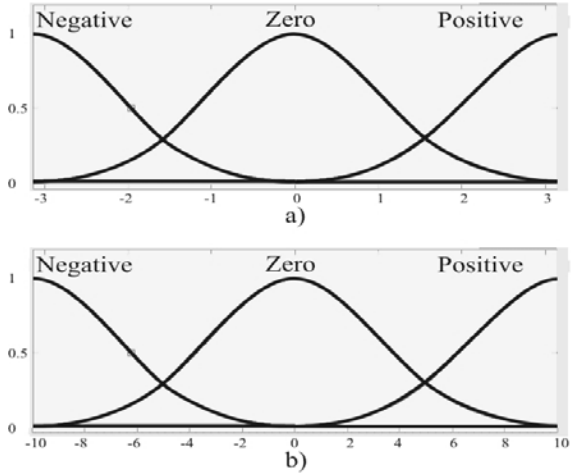


Fig. 5. Membership functions for $input_1$ a), $input_2$ b**)**

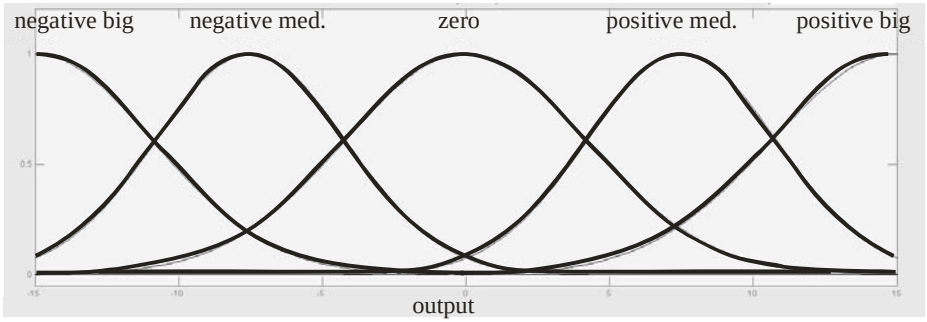


Fig. 6. Membership functions for output

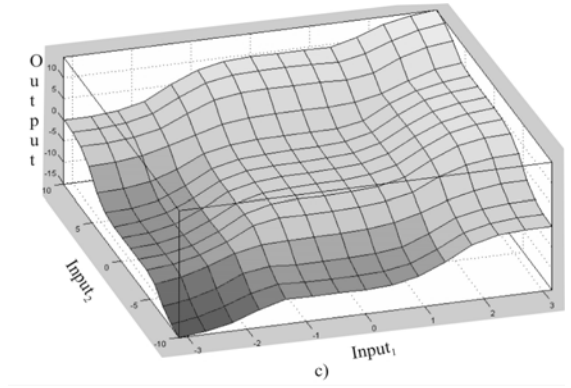


Fig. 7. Transfer surface for adopted fuzzy controller

$$\text{input}_1^{(t_i)} = \Delta q_{n_i}^{(t_i)} \quad \text{and} \quad \text{input}_2^{(t_i)} = \left(\Delta q_{n_i}^{(t_i)} - \Delta q_{n_i}^{(t_i-1)}\right). \tag{5}$$

Membership functions for inputs are shown in Fig. 5. In Fig. 6. are shown membership functions for output. All membership functions (for inputs and output) are Gaussian curve membership function.

The following "IF-THEN" rules are introduced in to controller:

IF input_1 is negative	and input_2 is negative	THEN output is negative big
IF input_1 is negative	and input_2 is zero	THEN output is negative med.
IF input_1 is negative	and input_2 is positive	THEN output is zero
IF input_1 is zero	and input_2 is negative	THEN output is negative med.
IF input_1 is zero	and input_2 is zero	THEN output is zero
IF input_1 is zero	and input_2 is positive	THEN output is positive med.
IF input_1 is positive	and input_2 is negative	THEN output is zero
IF input_1 is positive	and input_2 is zero	THEN output is positive med.
IF input_1 is positive	and input_2 is positive	THEN output is positive big

As a result of introduced "IF-THEN" rules and input and output membership functions we can generate a transfer surface (Fig. 7.).

When fuzzy controller is used corrective control $\Delta u_{\text{local at joint } n_i}^{(t_i)}$ is defined as:

$$\Delta u_{\text{local at joint } n_i}^{(t_i)} = k_{\text{fuzzy}} * \text{output}^{(t_i)} \quad . \tag{6}$$

where $\text{output}^{(t_i)}$ is fuzzy controller output, k_{fuzzy} is coefficient whose intensity depend on ZMP deviation from its reference position and it changes from 0.1 to 1. When ZMP is out of 5 mm zone k_{fuzzy} decrease up to its minimal value by decrement 0.05, but in case ZMP deviation is less than 5mm, k_{fuzzy} increase till 1 by increment 0.025.

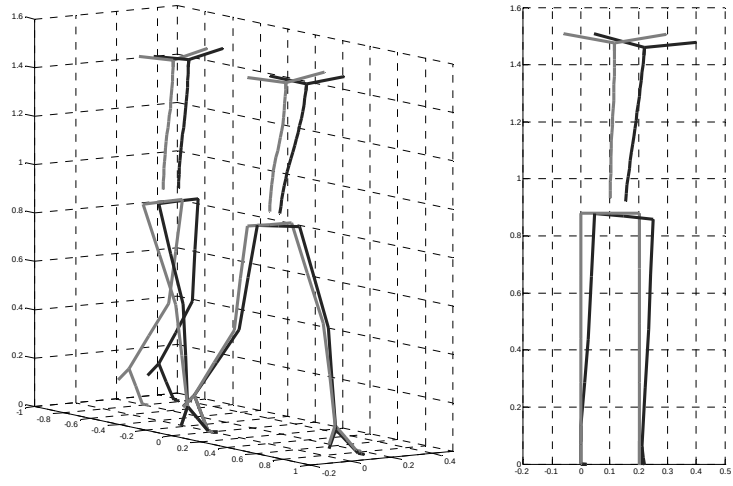


Fig. 8. Reference and actual biped postures at the beginning and at the end of half-step. For internal synergy compensation was used PID regulator. a) perspective view, b) sagittal plane.

3.3 Results

In Fig. 8. is shown stick diagram of the locomotion system on the beginning and the end of the simulated half step if for internal synergy deviation compensation is used PID regulator. In Fig. 9 is shown same situation, but in case for internal synergy deviation compensation is used fuzzy regulator.

Due to completeness in Figs. 10 and 11 are shown reference and actual ZMP positions in case as local regulator are used PID and fuzzy regulator.

From Figs. 10 and 11 can be seen ZMP trajectory during simulated period, as well as feet position during single-support phase and at the beginning of double-support phase. It is clear that both approaches ensure preservation of dynamic balance what is

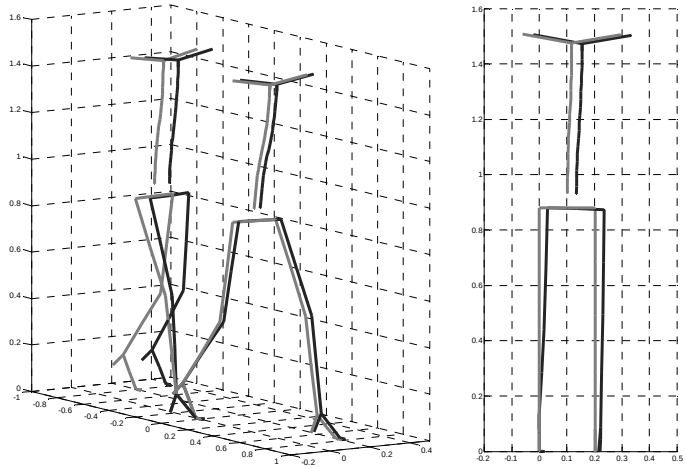


Fig. 9. Reference and actual biped postures at the beginning and at the end of half-step. For internal synergy compensation was used fuzzy regulator: a) perspective view, b) sagittal plane.

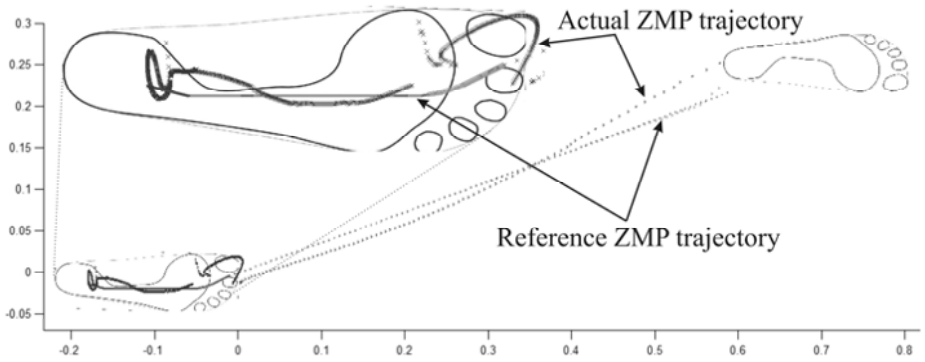


Fig. 10. Reference and actual ZMP position during one-half step; PID local regulator was used

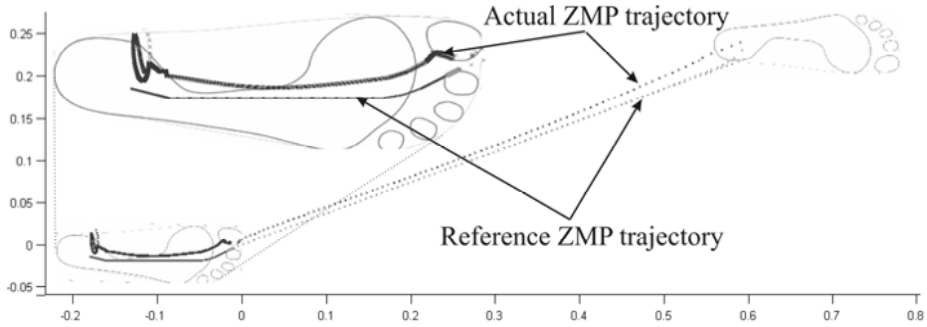


Fig. 11. Reference and actual ZMP position during one-half step; fuzzy local regulator was used

the most important requirement in humanoid robots control synthesis. But, from Figs. 8 and 9 it can be seen that at the end of half step posture in case fuzzy regulator was applied is closer to reference posture. In addition, in Figs. 10 and 11 is shown that that ZMP trajectory during half-step is much more "calm", as a consequence of more “smooth” compensation motion of internal synergy i.e. with smaller intensity of forces induced in the process of compensation. This is very good example that by more appropriate compensation of internal synergy as a side effect preservation of dynamic balance can be improved.

4 Conclusion

In this paper comparison of use of PID and fuzzy controller in compensation of small disturbances of internal synergy was performed. Both controllers were applied in parallel with controller whose task was to preserve dynamic balance, i.e. to prevent mechanism overturn during walk. In both cases simulation experiment was successful. But, fuzzy controller showed somewhat more promising characteristics in sense of more smooth compensation (movements without rapid accelerations) what reflects on the ZMP trajectory under foot. Thus, some more detailed investigation in this direction is planned in the future for example use of fuzzy controller for dynamic balance or use of neural network based controllers.

References

1. Juričić, D., Vukobratović, M.: Mathematical Modeling of Biped Walking Systems, ASME Publ. 72-WA/BHF-13 (1972)
2. Vukobratović, M.: How to Control the Artificial Anthropomorphic Systems. IEEE Trans. on System, Man and Cybernetics SMC-3, 497–507 (1973)
3. Vukobratović, M.: Legged Locomotion Systems and Anthropomorphic Mechanisms, Mihajlo Pupin Institute, Belgrade (1975); also published in Japanese, Nikkan Shimbun Ltd. Tokyo, in Russian, “MIR”, Moscow, 1976, in Chinese, Beijing (1983)

4. Vukobratović, M., Borovac, B., Surla, D., Stokić, D.: Biped Locomotion – Dynamics, Stability, Control and Application. Springer, Berlin (1990)
5. Vukobratović, M., Borovac, B.: Zero-Moment Point- Thirty Five Years of its Life. Int. Jour. of Humanoid Robotics 1(1), 157–173 (2004)
6. Vukobratović, M., Borovac, B.: Note on the Article, Zero-Moment Point-Thirty Five Years of its Life. Int. Jour. of Humanoid Robotics 2(2), 225–227 (2005)
7. Vukobratović, M., Borovac, B., Potkonjak, V.: Towards a Unified Understanding of Basic Notions and Terms in Humanoid Robotics. Robotica 25, 87–101 (2007)
8. Vukobratović, M., Borovac, B., Potkonjak, V.: ZMP: A Review of Some Basic Misunderstanings. Int. Jour. of Humanoid Robotics 3(2), 153–176 (2006)
9. Vukobratović, M., Borovac, B., Raković, M., Potkonjak, V., Milinović, M.: On some aspects of humanoid robots gait synthesis and control at small disturbances. Int. Jour. of Humanoid Robotics (accepted for publication) (2008)

A Java-Based Framework for the Programming of Distributed Systems for Mobile Robots

Daniel Westhoff and Hagen Stanek

genRob GmbH, Aidlingen, Germany
{westhoff,stanek}@genRob.com

Abstract. We propose a novel concept for the programming of distributed systems for mobile robots. A software architecture is presented that eases the development of applications for mobile robots. This software architecture is based upon the Roblet-Technology, which is a powerful medium for robots. It introduces the possibility to develop, compile and execute a distributed application on one workstation. The fundamental paradigm of the Roblet-Technology is the strong use of mobile code. Using mobile code an application distributes parts of itself through the network and builds up a distributed application. Since the Roblet-Technology uses Java the development is independent of the operation system. With the feature of running programs as a distributed software, the framework allows running algorithms which need great computation power on different machines which provide this power. In this way, it greatly improves programming and testing of applications in service robotics. We provide several examples of complex applications which were developed using our framework. They all have in common that they use the Roblet-Technology to combine several independently developed software components.

1 Introduction

Robotic systems are becoming more and more complex. The number of constitutional parts that make up current robotic research platforms is increasing. A multitude of sensors can be found in these robots: tactile sensors from basic bumper switches to force and torque sensors, range measuring systems like infrared, ultra-sonic, radar and laser based sensors or vision systems including cameras as different as low-cost web-cams and high-dynamic-range cameras. On the actuator side one finds mobile robot platforms with a variety of drive systems, walking or climbing robots, robot arms with different degrees of freedom or complex robotic hands with multiple fingers. In service robotics all these are combined in autonomous mobile manipulators that accomplish tasks in a diversity of applications.

The field of service robotics has seen a lot of advances over the last years, but still lacks usability and robustness. We think that one reason for this is the absence of a unifying software architecture that handles the miscellaneous challenges which the software engineers encounter. These challenges vary from

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 54–67, 2009.

the development of distributed applications to the handling of the diversities of different hardware platforms present in service robotics.

Over the last years, the research community has come to realise that the ambitious objectives of robotic research can only be reached based on solid software architectures. These architectures must support the requirements of the heterogeneous modern robot systems. Briefly summarised, the main requirements are: hardware abstraction, extendability, scalability, limited run-time overhead, actuator control, modularisation, support for networked computing, simplicity, consistency, completeness, support for multiple operating systems. [1] and [2] have conducted surveys and evaluations of existing software systems. They provide a good elaboration on the merits and demerits of these architectures.

In this paper we propose a framework that meets these challenges and enables a programmer to develop advanced applications for service robots. A main feature of the framework is the ability to integrate existing solutions to specific robotic problems. We will show that it is possible to encapsulate libraries for motion control for manipulators as well as for mobile robots. A variety of hardware devices connected to a service robot will be integrated into the architecture. A layer of abstraction will generalise the access to these devices. Thus, developed applications can be transferred to other robotic systems without changes.

The remainder of this paper is organised as follows: In section 2 an overview of existing software architectures in robotics is given. This is followed by a discussion of the merits and demerits of the existing software developments. Motivated by this, section 4 introduces our software architecture and how hardware is encapsulated by the proposed framework. In Section 6 applications are presented where the framework was applied successfully. Section 7 gives a conclusion and an outlook on future work.

2 Related Research

This section gives an overview of existing software architectures for service robots. Recently, a workshop during the 2004 conference on Intelligent Robots and Systems (IROS) tried to list the various research activities in the field of robotic middleware [3]. One year later a similar workshop was held at the 2005 Conference on Robotics and Automation [4]. The outcome of the second workshop is collected in [5]. In the following, some of the activities in the field of robotic software environments are discussed. Besides, further related research projects are stated.

The *OROCOS* project started in 2000 as a free software project due to the lack of reliable commercial robot control software [6]. It is divided into two decoupled sub-projects: *Open Real-time Control Services* and *Open Robot Control Software*. The first one is a real-time software framework for applications for machine control. The second one is a set of libraries and an application framework including generic functionality mainly for manipulators. Support of mobile robots is still in its early stages.

In 2004 the *Orca* project emerged from the OROCOS project [7]. It adopts a component-based software engineering approach using *Ice* [8] for communication

and the description of interfaces. The project's goals are to enable and to simplify software reuse and to provide a generic repository of components. The use of different middleware packages for inter-component communicating is extensively discussed on the project's home page. Beside writing custom middleware, the use of CORBA and XML-based technologies is compared to Ice. Orca is available for various operating systems and compiles natively.

[9] introduces the *Player/Stage* project, a client-server framework to enable research in robot and sensor systems. It provides a network interface to a variety of robot and sensor hardware and to simulators for multiple robots. Multiple concurrent client connections to the servers are allowed. Client applications connect over TCP sockets. The project's server software and the simulators are limited to Unix-like operating systems.

In [10] MARIE is presented, a design tool for mobile and autonomous robot applications. It is mainly implemented in C++ and it uses the *ADAPTIVE Communication Environment* (ACE) [11] for communication and process management.

In 2002 *Evolution Robotics* introduced the *Evolution Robotics Software Platform* (ERSP) for mobile robots [12]. It is a behaviour-based, modular and extensible software available for Linux and Windows systems. The main components that are included are vision, navigation and interaction.

In December 2006 Microsoft released the first stable version oft their robot software development kit *Microsoft Robotics Studio*[13]. The kit features a visual programming language to create software for robot systems, a 3D simulated environment and a runtime to execute the programs on the robot hardware. The use of the software is restricted to several versions of Microsoft Windows including Windows CE for mobile applications. It is strongly based on Microsoft's .NET framework.

In [14] a service robot for a biotechnological pilot laboratory is presented. The mobile platform of this robot is equal to parts of TASER which is presented in this paper. A seven degrees-of-freedom arm is mounted on top of the mobile platform. The system is designed to take samples from a sampling device, handle a centrifuge, a fridge and other biotechnological equipment and fulfil the complete process of sample management. It relieves the personal of the laboratory of monotonous time consuming tasks. Nevertheless it operates in a standard laboratory with standard equipment. An easy-to-use script language is proposed to define high-level work sequences. The scripts are parsed by the robot's control software and the robot fulfils the defined task. This encourages the idea of simplifying the programming of robots but lacks the flexibility of a widespread programming language including network programming for distributed systems.

3 Discussion of Existing Software Frameworks

The main contradiction a programmer of a service robot has to deal with is the trade-off between easy operability and full flexible utilisation. Easy operability means the system should be easily programmable by non-experts. To allow this,

current research investigates man-machine interfaces for natural instruction and communication. Unfortunately, most of these interfaces are only available as prototypes. But, commercial implementations of these techniques will not be available for some years to come. One compromise was to use script languages as in [14]. These languages are easy to learn and reduce the amount of knowledge needed to operate the robot. They abstract low-level details of the robot control and provide mid-level functionality. This mid-level functionality is then used to implement high-level task-oriented applications.

On the other hand script languages have serious limitations. They only allow alteration of certain parameters and sequences up to a certain point. Modifications or new tasks can only be implemented if they do not require more functionality than offered by the script language. Anything that is not possible within the functionality of the mid-level script language is therefore not possible at the task-oriented level. Such modifications would require access to low level details, but are intentionally hidden by the script languages.

As a general conclusion a framework for mobile robots should provide an easy-to-learn high-level interface for non-expert personnel, while also providing access to low-level details. This allows adding functionality that is missing in the high-level interface.

Most of the presented frameworks can be seen as component-oriented architectures and thereby try to address the above mentioned problems. The Roblet-Technology presented in section 4 is a novel component based architecture. Since its basis is one programming language the developers benefit of a unified programming environment which eases the exchange of knowledge within a project. Especially concerning the maintenance of complex robotic systems and the frequent changes of developers in scientific projects and institutions we think this will be an immense advantage over other robotic development environments.

4 Software Architecture

In this section we propose a novel software architecture to ease the development of high-level programs combining the functionality of robotic subsystems.

Many service robot systems are developed for a specific task like mail delivery, hospital service or laboratory services. In order to keep the software maintainable the low-level details of the system are hidden by a hardware abstraction layer (HAL). The task-level programs implementing the service are then based on this HAL. Using the robot for a different service often can not be done by simply writing a new task-level program, but requires additional low-level changes as well. In existing systems this cannot be done while the robot operates. We will explain how our architecture allows easy task-oriented programming by providing high-level functionality as well as access to parts of the low-level architecture. Otherwise, adding new functionality to perform new or even only slightly changed tasks would not be possible.

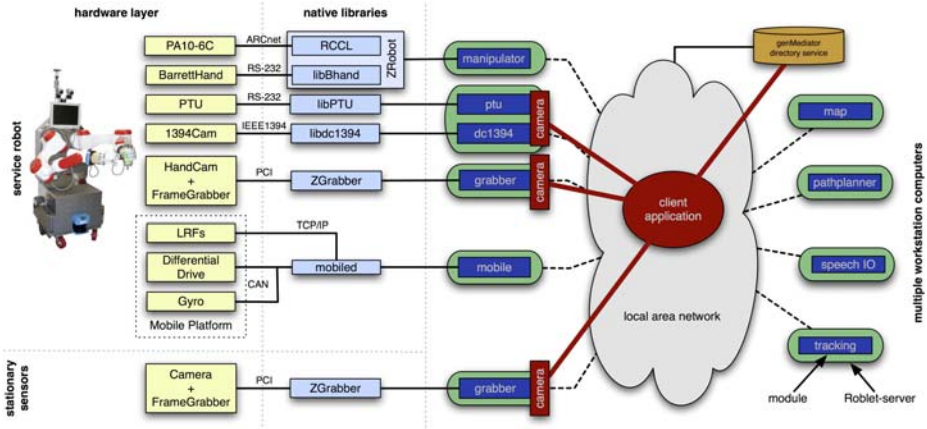


Fig. 1. The software architecture of the robot TASER at the University of Hamburg: Roblet-servers (RS) are used to provide a hardware abstraction layer. Generalisation is realised by this hardware abstraction. Distributed applications are independent of the particular hardware of the robot system. Some Roblet-servers and connections are left out for clarity. The client application is an example on how different hardware is encapsulated. The client application uses only the unit *camera* which unifies the access to cameras. The actual type of hardware is not known to the client application.

4.1 Roblets

The basics of the proposed framework are realised with Java and use Roblet-Technology, a concept firstly introduced in [15]. Roblet-Technology is a client-server architecture where clients can send parts of themselves, referred to as Roblets, to a server. The server, referred to as Roblet-server, then executes the Roblets with well-defined behaviour in case of malfunctions. Notice that not only data is transmitted between the client and server but complete executable programs. This can be compared to Java Applets but with the difference that Roblets are not downloaded but sent. Complex setups can consist of multiple client applications and Roblet-servers. A Roblet terminates if the execution of its code finishes normally or throws an exception. Exceptions are sent back to the client application. In addition, a Roblet can be terminated by a client application remotely or by the Roblet-server directly. After a Roblet terminates, the Roblet-server resets itself to a well defined state.

In section 5 we give a short example how a simple distributed application looks like when it is programmed with our framework.

Roblet-Technology is applicable to all kinds of distributed systems but it has several features that make its integration into robotic applications useful. In general, high-level applications in service robotics are mostly distributed systems. Besides one or multiple mobile robots, there are visualisation- and control applications that run on workstations in a local area network. Sometimes there is no direct access to the robot systems via keyboard, mouse and monitor but only through a wireless network. Roblet-Technology introduces the possibility to

develop, compile and execute an application on one workstation. When the application is executed it will send parts of itself to available servers and spread in the local network. Roblets may send parts of themselves to other servers as well. The network communication is hidden from the programmer by the Roblet library, which simplifies the overall development. That means, the network is transparent and developing distributed applications based on Roblet-Technology is like developing one application for one workstation. Access to the remote servers is encapsulated in a client library, reducing the execution of a Roblet on the remote system to one method call.

4.2 Modules

For robotic applications we propose *modules* to extend the basic Roblet-server provided by the Roblet framework. A module is loaded when the Roblet server is started. It is meant to encapsulate a class of similar functionality.

For the robot TASER of the University of Hamburg we developed several modules. A more detailed explanation of this robot is given in section 6. One module merges the functionality of the mobile platform, a second module wraps the manipulator system including the robot arms and the hands. There are modules for the different vision systems, the pan-tilt unit, a speech module and other parts of the interaction subsystem. Figure 1 gives an overview of the main parts of the current software architecture for TASER. The system incorporates several smaller Roblet-servers and multiple client applications not shown in the figure for clarity. Notice that the map server and the path-planning server don't run on the robot's control computer but on a workstation in the local network. This allows the integration of information gathered by multiple robots. For example, in the case of dynamic map adjustment this relieves the robot's on-board computer of some computationally expensive tasks which need no real-time capabilities.

4.3 Units

Modules are further divided in *units*. Units are Java interfaces that are implemented within the modules. Units build the hardware abstraction layer in our framework. For example, a module encapsulates the localisation subsystem of a mobile robot and a Roblet wants to query the current *pose* estimate[1] of the robot. Then the module would implement a unit which defines a method to get the pose. On another robot there may be another localisation system encapsulated by another module. But, if the module implements the same unit, the same Roblet can be executed on both robots and works without changes. Nonetheless, special features of a subsystem are made available to Roblets if module-specific units, e.g. to change special parameters of a subsystem, are implemented. Therefore, a Roblet has only access to units, it does not know anything about a module and a module's implementation of the interface. The whole concept is strictly object-oriented.

[1] A *pose* is the triple of 2D position coordinates and the robot's orientation.

By introducing units, the framework is able to generalise access to similar classes of subsystems without loosing access to their special features. Additionally, units introduce a possibility of versioning into the system. If new features are integrated into a module then new units will be introduced. As long as older units are still available, all client applications and their Roblets using these old units still work. This has proved to be of great use since complex applications often consist of dozens of client applications and Roblet-servers. A transition to new units can be accomplished step by step for each client application.

Figure 2 shows a chart of the structure of a Roblet-server.

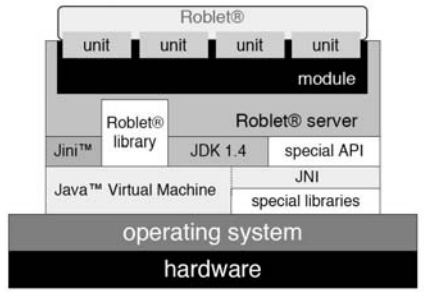


Fig. 2. The chart shows the structure of a Roblet-server and how it hides the hardware from a Roblet

4.4 Platform Independence

There were several reasons to use Java to implement the concept of Roblet-Technology: First of all, Java virtual machines and compilers are available for a variety of different platforms from embedded system over PDAs to workstation computers. All these different systems can be found in the field of robotics. Since Java source code is compiled into bytecode, the programs can be compiled on any of these systems and executed on another system without change. Besides, Java provides a vast standard library available on all of these systems. The standard libraries include techniques for network communication like RMI or Jini used within the Roblet framework. These well-tested libraries ensure reliable operation of the framework since they are used in millions of Internet applications as well.

For the developer of a client application the view of the system is unified. He does not need any knowledge about the heterogeneous network structure, the differences between operating systems and so on. All he has to be familiar with is Java and programming becomes like programming on one single machine.

In contrast, using other programming languages like C/C++ would require the compilation of the source code for each target machine. Additional libraries, e.g. CORBA, Ice or ACE, are required for network communication, which demand additional knowledge of the programmer. Further on, these libraries may sometimes be only available for a subset of systems present in a robotic scenario.

In future, the .NET framework from Microsoft may become an alternative to Java since it also compiles source code into a bytecode first. Nonetheless, to date .NET is only available for Windows platforms. The open-source projects implementing .NET for other platforms do not provide full support yet.

Since Java has no real-time capabilities, programs written within a Roblet are not intended to contain real-time control loops. There exists a specification for a real-time java virtual machine but at present no implementation [16]. The Roblet framework allows less skilled programmers to design and develop robotic applications without in-depth knowledge about the used subsystems. First experiences using the Roblet framework in lectures for graduate students have proved this.

Nonetheless, the developers of modules still must have knowledge about specific technologies they want to use. For example, if we want to encapsulate a C/C++-library that controls a hardware component the module developer will have to write a wrapper for that library using the *Java Native Interface* (JNI). That requires at least knowledge about C/C++ and Java. But, we think in future the number of developers of client applications will be much greater than that of module developers.

5 Roblet Application

In this section we will give a short example how software components are used in our Roblet-Framework. We present two Java classes. The first class illustrates how Roblets are sent from the client application to a server. The second class includes the Roblet code which is executed on the server side.

Our example explains how path planning capabilities are integrated into a robotic application. We need the basic Roblet-server from the Roblet-Framework and the path planning module for our robot TASER which is loaded when the server starts running. For our client application we use the client library of the Roblet-Framework. With the client library we send Roblets to the server.

Listing 1 shows the Java class `Pathplanner` that can be used for path planning on the client side. An object of this class is created with a string containing the IP address and the port of the Roblet server with the path planning module. Instead of the IP address a hostname can be used. We could start the server on the same machine where we develop the client application. If we use port 8000 the parameter string for the constructor is `localhost:8000`.

Each time the method `plan()` is called an instance of the class `PathPlanning-Roblet` is created, marshaled and sent to a slot of the server. A slot is a sandbox environment the server provides. This sandbox is a security layer that restricts the access of the Roblet code to the underlying system. Listing 2 shows the Java code of the class `PathPlanningRoblet`. A Roblet has to implement the Java interface `Roblet` which only specifies the method `execute(Robot robot)`. In addition, we implement the interface `Serializable` since the object will be serialised by the client library to send it over a network. On the server the method `execute()` of the Roblet is called.

```
import genRob.genControl.client.Client;
import genRob.genControl.client.Server;

public class Pathplanner
{
  private final Server server;

  public Pathplanner (final Client client,
                      final String serverName)
  {
    server = client.getServer (serverName);
  }

  public Path plan (final RobotProperties properties,
                    final Point start,
                    final Point end)
  {
    return (Path) server.getSlot ().run
      (new PathPlanningRoblet (properties, start, end));
  }
}
```

Listing 1. This Java code example shows a class that can be used within a client application to question a Roblet server which provides path planning capabilities. Each time a path is requested a Roblet is send to the server. The Roblet invokes a path planning algorithm on the server and returns the answer to the client application. Some imports as well as appropriate exception handlingoutines are left out for clarity.

First, the Roblet code queries the server for path planning capabilities. The path planning module of the server provides an implementation of the Unit `Planner` which can be received by calling `getUnit()` on the `Robot` parameter. If the path planning module was not loaded the return value of the `getUnit()` call equals `null`. Then, we throw an exception that ends the Roblet. The presented exception handling is very basic to keep the example simple.

If the Roblet gained access to the path planning unit it computes a path from a start to a target point. Internally the path planning algorithm contacts the map server to query the current information about the environment. The start and target point were parameters of the constructor of the Roblet as well as some properties of the robot that will drive along the path. These member variables were serialised when we sent the Roblet to the server. Therefore, they are now usable on the server side of this distributed system. If we instantiate other objects which are not present on the servers classpath, the server notifies the client application. Then, the client application will provide the class descriptions to the server.

The path from the start to the goal point is returned to the client application. A `Path` object is return by the `execute()` method. The server sends this object to the client. On the client side the path is returned as the result of the method call `run()` on the `Slot` object.

How the path is computed depends on the properties of the robot and is out of the scope of this simple example. If there is no path that the robot can drive this is encapsulated in the `Path` object and can be queried.

```
import org.roblet.Roblet;

public class PathPlannerRoblet
  implements Roblet, Serializable
{
  private final RobotProperties properties;
  private final Point start;
  private final Point end;

  public PathPlanningRoblet (final RobotProperties props,
                             final Point start,
                             final Point end)
  {
    this.properties = props;
    this.start      = start;
    this.end        = end;
  }

  public Object execute (Robot robot)
    throws Exception
  {
    final Planner planner
      = (Planner) robot.getUnit (Planner.class);

    if (planner != null)
      return planner.plan (properties, start, end);
    else
      throw new Exception ("Planner unit not provided.");
  }
}
```

Listing 2. The above class implements the Roblet interface. It is instantiated on a client machine. The instance is serialized and send to a server which calls the method `execute()`. The Roblet requests an implementation of the unit `Planner` to calculate an appropriate path for the robot. Some imports as well as appropriate exception handling routines are left out for clarity.

This elementary example gives an insight into the Roblet-Framework. Writing two similar classes for the module that controls the mobile robot gives the developer all that he needs to drive a robot safely in our office environment. On the other side with only four small classes of Java code he creates a distributed system that connects a client application with a robot, a path planning service and a map database. The structure of such a system is visualised in figure 3.

A Roblet can establish more advanced network communication channels between the server and the client. It may start threads which open sockets or use the Remote Method Invocation provided by Java's standard libraries. A Roblet ends when the method `execute()` finishes. If a thread is started by a Roblet the thread will stay alive in the slot on the server until the thread ends. Therefore, Roblets may only be needed to distribute code onto a number of different servers when the application starts. These Roblets establish network connections to the client application. Roblets can send Roblets themselves to other servers and thereby create complex distributed system structures. As a result, each client application can create the communication network that is most appropriate. One may choose to use XML data for communication, another may compress all data before transmission and a third one may implement special encryption algorithms to increase security.

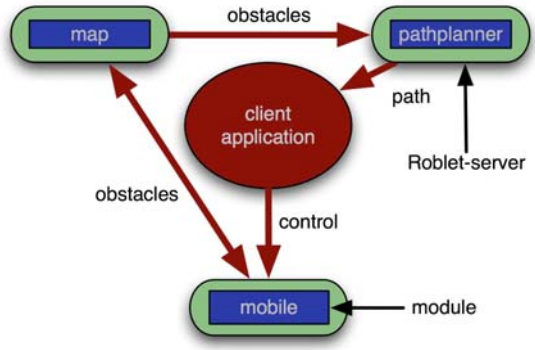


Fig. 3. The flow chart visualises the structure of the distributed system described in appendix 5. A roblet is sent by the client application to the path planner. The pathplanner contacts the server with the map data and receives the obstacles currently stored in the map. After that the pathplanner calculates a path and sends the answer back to the client application. Then, the client application could contact the mobile robot to control its motions. The mobile robot uses the map for localisation purposes. In addition, it sends information about newly detected obstacles to the map server.

6 Applications in Service Robotics

In this section we will describe two applications which emphasise the capabilities of the proposed architecture. The applications show TASER when it accomplishes high-level tasks using a combination of its various components. TASER is a multi-modal service robot located at the institute TAMS of the University of Hamburg.

6.1 Interaction with the Environment

The first example is a combination of localisation, planning of paths, object manipulation and interaction where the robot is instructed to operate a light switch. An operator chooses a light switch and commands the robot via speech commands or an interactive dialogue to operate it. The application uses various Roblet-servers shown in figure 1.

First, a Roblet on the Roblet-server for the speech IO informs the client application that a light switch is to be operated by the robot. Then the position of the light switch which is stored as a point of interest in a map is requested from a map server. This Roblet-server encapsulates a database in which map elements like obstacles and points of interest are stored. Multiple applications can get, alter or add map elements of the database concurrently through this Roblet-server. Then, a Roblet is sent to the path-planning server to get a path to the light switch. The Roblet sends a new Roblet to the robot. There, the new one drives the robot to a suitable position at the light switch, so that the arm can reach the switch. The position of the arm relative to the switch is obtained form a

method call to the arm-operations library which is provided by the corresponding Roblet-server. By solving the kinematic chain, the robot computes a position and orientation suitable to operate the switch. After this position has been reached by the robot arm, a Roblet tries to fine-position the manipulator in front of the light switch with the hand camera by visual servoing. A separate Roblet-server for the hand camera provides positioning errors calculated on the observed images. When the arm is centred in front of the switch, an approach move is made by the arm which is force controlled by sensor input of the BarrettHand. The sensors of the hand are precise enough to stop the movement of the arm when the finger touches the switch. In the final step the finger operates the switch. By using a final movement of individual fingers, even switches like a double-switch can be operated independently.

6.2 Grasping and Transportation

The second example is given by the task of object grasping and transport. The user can advise the robot to fetch and carry objects lying on a table via an interactive dialogue. Each source of information about humans interacting with the robot is encapsulated into its own Roblet-server and can thereby be employed by Roblets. The robot plans its path to the object using the Roblet-server for path-planning. After reaching a position suitable for object grasping, the robot tries to identify the object by means of object recognition. In case of ambiguities the interaction system is used with other Roblet-servers to resolve the situation. For example, if the robot cannot distinguish objects on the table, it uses the active vision system to recognise pointing gestures and gaze to resolve the position the manipulator of the robot is intended to move to. Additionally, the user can teach the robot new grasping motions and grasps [17].

When the object is successfully recognised, the robot selects a suitable grasp for the object from an internal grasp database and executes it. After grasping the object the robot moves the manipulator back into a safe transporting position. If the transport position has influence on the security outline around the robot, this outline is modified and a path to where the object is to be placed will be calculated based on the new outline. When the final position has been reached the robot will set down the grasped object and is available for new tasks again.

7 Conclusion

The presented software architecture enables the building of high-level applications for service robots using standard components for robots as well as specialised hard- or software. We proved is with the application of our framework to the service robot TASER at the University of Hamburg. The software architecture of the robot based on the Roblet-Technology is a powerful medium for robots. The feature of running client programs as a distributed software offers the possibility to run algorithms which need great computation power on different machines which provide this power. The type of communication, e.g.

encrypted or compressed communication, can be changed during runtime. Each client application can use its individual and appropriate type of communication.

The convenience of used proved in several lectures we gave where student were able to build applications for our robot. Some of these application are used in our daily routines when we work with the robot.

One next step will be to implement further software to improve the usability of the robot system and create a toolbox of reusable program parts. In this step the variety of the high-level functions like object grasping and multi-modal interaction will be increased. Furthermore, the possibilities of autonomous navigation and map building will be extended.

Another step will be the port of some modules to other robot platforms. With this we can show that the hardware abstraction provided by *units* is reliable, when we do not need changes in the client applications.

Additionally we try to integrate components of other robotic software environments like these of section 2. Our framework will help to connect the various great efforts that are carried out in all these projects.

References

1. Orebäck, A., Christensen, H.I.: Evaluation of Architectures for Mobile Robotics. Autonomous Robots 14(1), 33–49 (2003)
2. Kramer, J., Scheutz, M.: Development Environments for Autonomous Mobile Robots: A Survey. Autonomous Robots 22(2), 101–132 (2007)
3. Workshop on Robot Middleware towards Standards. In: International Conference on Intelligent Robots and System (IROS 2004), Sendai, Japan (2004), `http://www.is.aist.go.jp/rt/events/20040928IROS.html`
4. Workshop on Principles and Practice of Software Development in Robotics. In: IEEE International Conference on Robotics and Automation (ICRA 2005), Barcelona, Spain (2005)
5. Brugali, D. (ed.): Software engineering for experimental robotics. Springer tracts in advanced robotics. Springer, Berlin (2007)
6. Bruyninckx, H.: Open robot control software: the OROCOS project. In: Procedings of the IEEE 2001 International Conference on Robotics and Automation (ICRA 2001), Seoul, Korea, vol. 3, pp. 2523–2528 (2001), `http://www.orocos.org`
7. Brooks, A., Kaupp, T., Makarenko, A., Orebäck, A., Williams, S.: Towards Component-Based Robotics. In: Proceedings ot the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), Alberta, Canada (2005), `http://orca-robotics.sourceforge.net`
8. Henning, M.: A new approach to object-oriented middleware. Internet Computing 4(1), 66–75 (2004)
9. Gerkey, B.P., Vaughn, R.T., Stoy, K., Howard, A., Sukhatme, G.S., Mataric, M.J.: Most Valuable Player: A Robot Device Server for Distributed Control. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2001), Wailea, Hawaii, pp. 1226–1231 (2001)
10. Cote, C., Letourneau, D., Michaud, F., Valin, J.-M., Brousseau, Y., Raievsky, C., Lemay, M., Tran, V.: Code Reusability Tools for Programming Mobile Robots. In: Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004), Senda, Japan, pp. 1820–1825 (2004)

11. Schmidt, D.C., Box, D.F., Suda, T.: ADAPTIVE — A Dynamically Assembled Protocol Transformation, Integration and eValuation Environment. Concurrency: Practice and Experience 5(4), 269–286 (1993)
12. Karlsson, N., Munich, M.E., Goncalves, L., Ostrowski, J., Di Bernado, E., Pirjanian, P.: Core Tehnologies for service Robotics. In: Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004), Senda, Japan (2004)
13. Microsoft Robotics Studio, `http://msdn.microsoft.com/robotics/`
14. Scherer, T.: A mobile service robot for automisation of sample taking and sample management in a biotechnological pilot laboratory, University of Bielefeld, Ph.D Thesis (2005), `http://bieson.ub.uni-bielefeld.de/volltexte/2005/775/`
15. Westhoff, D., Stanek, H., Scherer, T., Zhang, J., Knoll, A.: A flexible framework for task-oriented programming of service robots. In: Robotik 2004, VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik, VDI-Berichte, Munich, Germany (2004) ISBN 3-18-091841-1
16. The Real-Time Java™ Expert Group: The Real-Time Specification for Java (RTSJ) (2002), `http://rtsj.dev.java.net`
17. Hüser, M., Baier, T., Zhang, J.: Learning of demonstrated Grasping Skills by stereoscopic tracking of human hand configuration. To Appear, IEEE International Conference on Robotics and Automation, Orlando, Florida (May 2006)

Wheeeler – Hypermobile Robot*

Grzegorz Granosik[1], Krzysztof Mianowski[2], and Michał Pytasz[1]

[1] Technical University of Lodz, Institute of Automatic Control, Stefanowskiego 18/22, 90-924 Lodz, Poland
granosik@p.lodz.pl, mpytasz@wpk.p.lodz.pl
[2] Warsaw University of Technology, The Faculty of Power and Aeronautical Engineering, Nowowiejska 24, 00-665 Warsaw, Poland
kmianowski@meil.pw.edu.pl

Abstract. We have designed and built the prototype of hyper redundant, articulated mobile robot propelled on wheels and therefore called - Wheeeler. In this paper we present progress in our project, focusing on modeling and prototyping phase. We show model of the robot built and verified in 3D simulator and proof-of-concept 3-segment device. Wheeeler is designed to operate in a rough terrain and fulfill tasks such as climbing up or down the stairs, going through trenches, avoiding or climbing over obstacles, operating in narrow, limited spaces like ventilation shafts. The major difficulty of control of hypermobile robots is synchronization of multiple actuators. Design of the high level control system, which can help human operator to intuitively steer this robot is the main goal of our project. In the further part of this paper we introduce communication and control architecture.

Keywords: Hypermobile robot, teleoperation.

1 Introduction

Hypermobile robots are a group of articulated body, serpentine robots; however, in comparison to them they introduce actuated wheels, legs or tracks. Such machines can travel in rough terrain and overcome obstacles much higher than robot itself. In addition to high mobility, due to its slender body, robot can crawl into narrow spaces or pipes for inspection or search and rescue actions. These extended capabilities of hypermobile robots caught attention of researchers in a few laboratories, comparing to the vast number of laboratories working on mobile robots in general. In spite of the fact that design of hypermobile robots is difficult and resource consuming (many identical segments and joints have to be built), there are several working robots and a few practical applications of these robots shown already [4].

The first practical realization of a hypermobile robot, called KR-I, was introduced by Hirose and Morishima from Tokyo Institute of Technology [5] and later improved with version KR-II [6]. This first serpentine robot was large and heavy, had a train-like

* This work was supported by The Ministry of Science and Higher Education under Grant No. 3 T11A 024 30.

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 68–83, 2009.

appearance comprising of multiple vertical cylindrical segments on powered wheels (tracks in KR-I).

There are a few projects of sewer inspection robots realized in Germany [9, 16, 17]. Robots are usually richly equipped with sensors: infrared distance sensors, torque sensors, tilt sensor, angle sensors in every segment, and a video camera in the head segment. Some of them present semi-automatic or automatic driving abilities. Recently developed MAKROplus robot is commercially available [7].

In Japan, country the most suffering from earthquakes, the International Rescue System Institute was established in 2002. One of the goals of this organization is promoting and supporting developments of search and rescue robots. Among many designs, four hypermobile robots have been presented to date. They are Souryu and IRS Souryu, first introduced by Takayama and Hirose [18], 4-segment pneumatically driven Moira [12] and 8-segment Kohga designed by Kamegawa et al. [8].

Researchers from the Mobile Robotics Lab. at the University of Michigan designed the whole family of hyper mobile robots called Omnis [3]. In the OmniPede, the first one, they introduced three innovative functional elements: (1) propulsion elements (here: legs) evenly located around the perimeter of each segment; (2) pneumatic power for joint actuation; and (3) a single so called “drive shaft spine” that transfers mechanical power to all segments from a single drive motor [10]. Further study led to the development of the far more practical OmniTread, which offers two fundamentally important advantages over its predecessor and, in fact, over all other serpentine robots described in the scientific literature to date. These features are: maximal coverage of all sides of all segments with propulsion elements and joint actuation with pneumatic bellows [2].

The newest construction from NREC (National Robotics Engineering Center) is Pipeline Explorer – robot designed and built for inspection of live gas pipelines [15]. This robot has a symmetric seven-element articulated body containing: locomotor modules, battery carrying modules, and support modules, with a computing and electronics module in the middle. It is fully untethered (battery powered, wirelessly controlled) and can be used in explosive underground natural gas distribution pipelines.

An example of reconfigurable hypermobile robot was developed by Zhang et al. [19]. The JL-I system consists of three identical modules having a form of crawlers with skid-steering ability. Each module is an entire robotic system that can perform distributed activities but a docking mechanism enables adjacent modules to connect or disconnect flexibly and automatically. This system with several identical modules which can work separately or simultaneously when assembled, uses hierarchical software, based on the multi-agent behavior-based concept. Robot showed ability to climb steps, span gaps and recover from any rollover situation.

2 Wheeeler Design

Our project of Wheeeler is focused on design of a high level control for hypermobile robots. As we observed from the literature review, most of the hyper mobile robots presented to date lack the autonomy or intuitive teleoperation. Although, every robot has some control system but in most cases they employ multi degree-of-freedom (DOF) joysticks [12] or sophisticated user interfaces [1], or require more then one

human operator. Our goal is to simplify teleoperation of these robots and increase their applicability. We start with precise modeling of Wheeeler and designing the most intuitive user interface to control it. Then we show some mechanical details of suspension system and proof-of-concept prototype containing 3 identical segments built with many off-the-shelf components used in RC models technology.

2.1 Simulation

For the modeling phase of the project, the Webots 5 PRO simulator was selected [11]. It is a mobile robot modeling environment which uses VRML model description, interpreted during simulation by ODE physics and OpenGL presentation layers. The modeling scene may consist of multiple elements which can be general solid objects, lights and robots. Each robot can have own controller implemented using C, C++ or Java programming language. Communication between robots is possible through robot nodes simulating wireless communication devices with predefined link parameters.

2.2 Concept of Wheeeler

Structure of described robot is modular. It consists of seven, geometrically identical segments shown in Fig. 1. Each segment has an actuated axle with two wheels and a passive suspension. On each of two ends of a segment, there is a 1DOF actuated joint, to be connected to the following segment, or in case of robot ends – to attach a camera. Assembled robot has 2DOF articulated joints between each two segments, 1DOF joints controlling cameras, and actuated wheels. This gives a total of 3 control variables per segment, and 21 in total.

We assumed position feedback from joints and vision feedback from two cameras mounted on both ends of robot. With these two cameras robot will have advantages similar to Kohga robot providing operator with view from the nose camera and perspective view from behind and above the robot when tail of Wheeeler is lifted in scorpion-like manner (see Fig. 2).

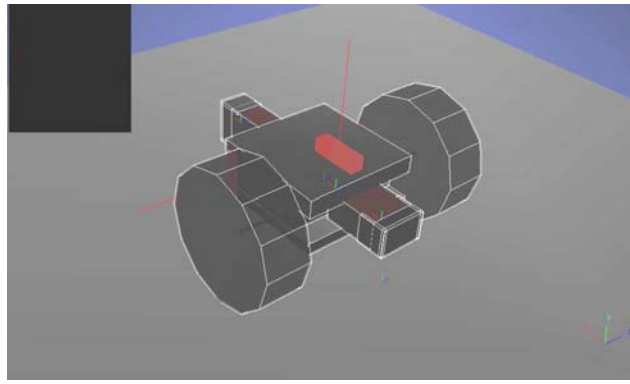


Fig. 1. Segment of Wheeeler

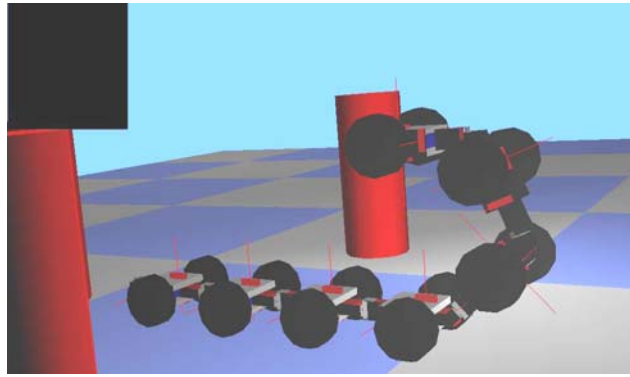


Fig. 2. Scorpion-like pose of Wheeeler

We assume that robot is able to raise two front segments, and therefore climb obstacles with height at least 1.5 times higher than the robot itself, (see Fig. 3). Inter-segment joints working in vertical direction have a range of movement close to ± 90, while in horizontal direction it is a little over ± 45.

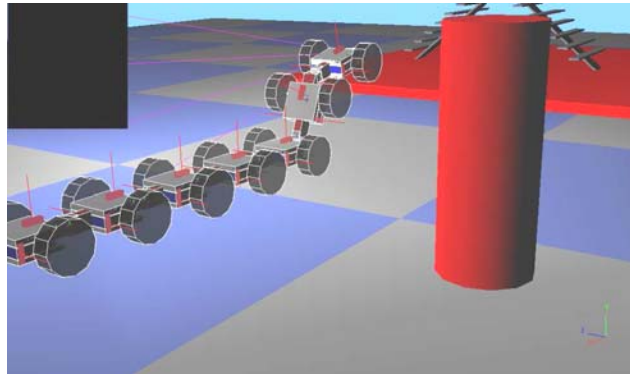


Fig. 3. Wheeeler climbing an obstacle in virtual environment

These ranges combined with short segments and zigzag posture of robot (e.g. as shown in Fig. 4) can compensate for lack of all side tracks (known from Moira or OmniTread). When rotation of upper wheels is opposite to the lower wheels robot is able to enter pipes, shafts, or low ceiling environments.

Each segment is equipped with four distance sensors, 3 axes accelerometer, a gyroscope, and a potentiometer to measure position of suspension mechanism. This gives a total of 9 sensors per segment, 63 sensors in total to be processed.

In current stage of the project, all distance sensors, potentiometers, accelerometers and cameras are implemented. These sensors are available in simulation environment. All information read from them is processed by robot controller and streamed to the client-operator. Gyroscopes are currently being implemented, since in the Webots software they are not supported. However, the functionality of the simulator can be extended through physics plug-in written by user.

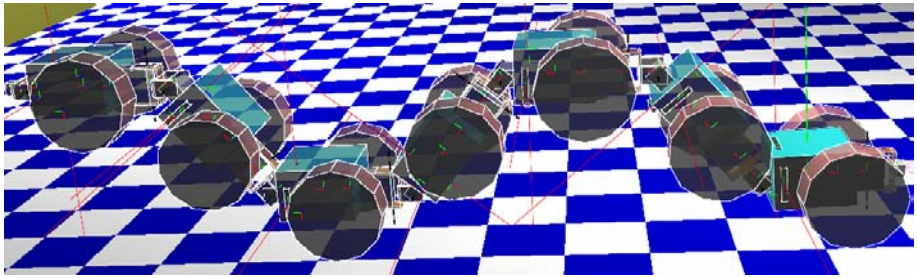


Fig. 4. Wheeeler in zigzag configuration

2.3 Mechanical Details

For precise mechanical design we have been using both AutoCAD and ProEngineer. The 3D CAD models helped us to visualize structure of the transmission and prepare suspension design as shown in Fig. 5.

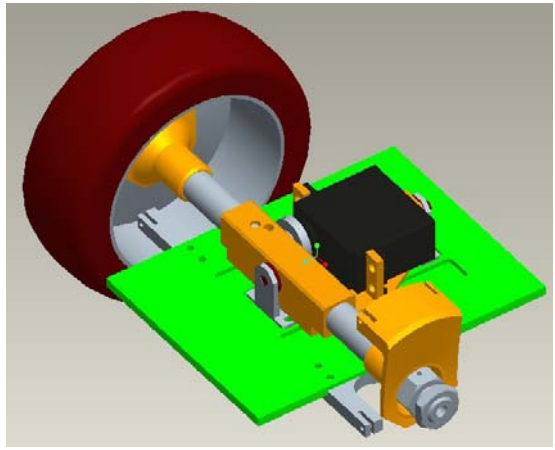


Fig. 5. 3D model of driving system of Wheeeler with passive suspension (springs not shown)

Earlier designs of robots of this kind, very often did not take into account a fact, that the useful force generated by the robot (it's segment), which propels robot and/or generates reaction on hindrance is dependent on the torque/force between considered segments and the ground. This reaction is dependent not only on the gravity component of the mass of considered segment but is dependent on reaction forces between this segment and the neighbor segments of snake-like robot structure. Typical design of articulated mobile robot consists of the number of segments connected by passive and/or active joints, driving systems containing motor with wheels for generating driving forces, and what is essential, driving system is rigidly connected to the body of the segment.

In our design, as shown in Fig. 5 and Fig. 6, the driving module with gear and wheels is connected to the body of segment by the rotational joint. Driving module

consists of motor with gear 200:1 and additionally conical gear 1:1 for changing the direction of rotational motion. This module is mounted in the base of the segment of the robot rotationally in such a way, that it can rotate ±10° over longitudinal axis of the robot. This angle is measured with rotary potentiometer. Axis of wheels is supported by two ball-bearings in the funnel, which at ends is connected with the body of segment using springs.

This solution assures enlargement of the angle between the surface of junction with the ground of one segment in the relation the neighbor segment (or two segments) as can be seen in Fig. 7. Beside of this, driving wheels with motor are connected to the body of segment with springs, that additionally improves the reaction forces between the wheels and the ground. Therefore, friction forces between wheels and the ground are greater, what improves performances of the robot.

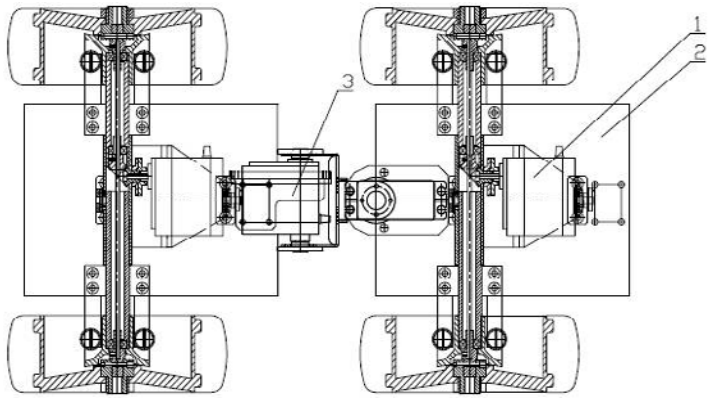


Fig. 6. Basic components of the robot: 1. – driving motor (module contains bevel gears), 2. – base of the segment, 3. – bending motor of the joint

Robot's segment includes driving part and two actuators for changing (controlling) angles of orientation in the relation to two neighbor segments. Obtained solution is characterized by very good integration of all mechanical and/or electrical/electronic components. Current design ensures space for all control/sensory system and a complete equipment for typical mission as was described in previous section.

We are very pleased with the behavior of Wheeeler's passive suspension, which helps to travel in an uneven terrain, as shown in Fig. 7, preserving continuous contact with ground for all wheels of the robot. Even for obstacles as high as half of wheel's diameter springs allow each segment to conform to the ground conditions and provide good grip for all tires.

In Fig. 8 we can see the behavior of springs depending on the position of wheel with respect to the floating platforms. If wheel is lifted up by an obstacle springs extend as shown on the left part of Fig.7, while springs on the wheel which is lower are compressed (right part).

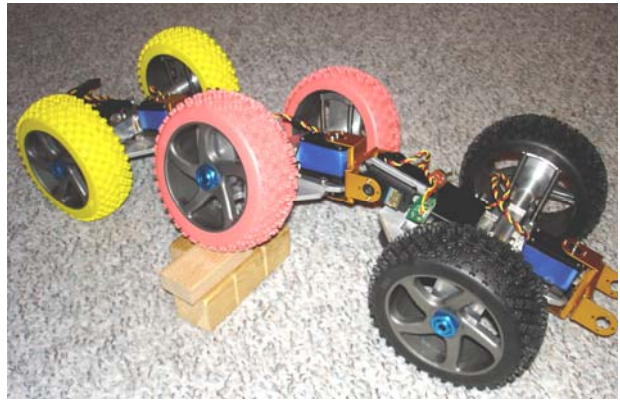


Fig. 7. Prototype of Wheeeler on an uneven terrain – 3 segments (out of planned 7 segments) connected with 2 DOF joints

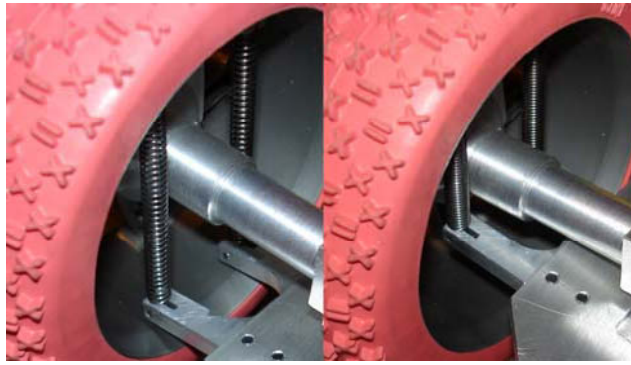


Fig. 8. Behavior of springs in passive suspension during riding over obstacle

3 Control

At first, basic teleoperation with only a visual feedback was introduced. A simple, IP network, socket based client-server application was built, featuring ability to send elementary commands to the robot. Communication was unidirectional, allowing client (operator) send one of the following instructions:

- new angular velocity of the axle of specified segment,
- new position of the horizontal or vertical joint of a segment,
- stop all segments.

This form of control would be very inconvenient in a real application; therefore a simple propagation algorithm for angular position of joints was introduced.

3.1 Follow the Leader Approach

To clearly explain this method, let us draw segments as presented in Fig. 9. To simplify, we assume that maximum angle of turn does not exceed the range of a single, horizontal joint of a segment. In an instant of time t_0 (left part of Fig. 9) robot started to turn and the angle of the turn is α_0. In a next simulation step (right part of Fig. 9), robot moved by Δx. In this case we can express the angle of turn as a sum of α and β. Length of a segment is represented by l, and it is a distance between two following segment joints working in the same direction. This lets us calculate joint variables in the next step of simulation according to equation (1).

$$\alpha = \arcsin\left(\frac{l-\Delta x}{l}\cdot\sin(\alpha_0)\right) \tag{1}$$

The equation (1) assumes that the linear velocity of all segments which have not started to turn is known and common. We have to remember that the velocity does not have to have the same value as the linear velocity of segments which turned already. Different situation can be seen in case of wheels on the turn curve. Their angular velocity is not very significant since in this robot, because of its structure, a slip on the turn curve cannot be avoided, therefore this variable will not be analyzed here. If we would represent the displacement of segments after the turn by Δy, according to the law of cosines (2),

$$\Delta y^2 = (l-\Delta x)^2 + l^2 - 2\cdot(l-\Delta x)\cdot l\cdot\cos(\beta) \tag{2}$$

we can see that displacement Δy according to (3)

$$\Delta y^2 = \Delta x^2 + (2\cdot l^2 - 2\cdot l\cdot\Delta x)\cdot(1-\cos(\beta)) \tag{3}$$

can be equal to Δx only if $cos(\beta) = 1$ or when $\Delta x = l$.

Both situations may not occur due to logical or physical limits. This led us to conclusion, that when the velocity of a robot would be set globally, it could only mean a value for the single, specified segment. If a robot is changing horizontal direction of move, linear velocities of other segments have to be calculated accordingly.

Of course this robot operates in three-dimensional space; therefore the algorithm for propagation of angles may be applied to the vertical joints between segments, with respect to the physical limits derived from segments geometry.

In the above it was assumed, that the robot moves in one, specified direction, first segment is defined by the direction of motion. This robot has to be operated in a user friendly way, therefore for the control purposes a mapping of segment addresses algorithm analyzing current linear velocity is required. The angular velocity of an axle, and therefore wheels is not the only parameter required to find the linear velocity of the robot. The other parameter known through measurement is the gravity direction measured by accelerometers. For the convenience of the operator the control application has to adapt to present conditions, letting one control the first segment, however segment numbering depends on real direction of motion. Currently – in the model – angular velocity of wheels and the pose of the first segment define the direction of robot used for remapping of axes, however in the real case more variables will have to

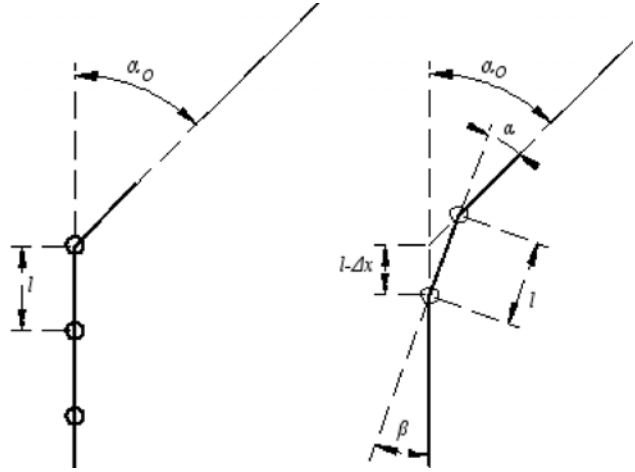


Fig. 9. Kinematics of the horizontal turn

be analyzed. A single sensor can always break, as well as some other distortions may appear, therefore in a real application a sensor fusion mechanism needs to be employed. Fusion of sensor may appear to be even more serviceable in case of attempts to estimate, for example, slippage of the wheels, or other parameters of motion, which cannot be measured in a straight forward way.

The functionality discussed above introduced new control commands to be sent to the robot:

- set/reset auto mode,
- cameras control commands – since in automatic mode we do not control other joints individually.

3.2 Sensor Fusion

To improve the behavior of propagation algorithm in a rough terrain – such as debris, where wheels may bounce on the surface, a basic sensor fusion algorithm was proposed, as schematically shown in Fig. 10. This algorithm assumes that if some of the following segments form a straight line, measurements obtained from these segments should be comparative, therefore they are grouped and the sensors' readings are averaged in a presented way. In the situation when groups are degraded to one segment and acceleration readings are above threshold we assume larger belief to encoder readings (70%). Algorithm shown in this figure is simplified; iterations of calculations for segments or segment groups were omitted.

3.3 Communication Layer

With the further development, operator–robot communication framework was changed. CORBA communication was introduced. The choice was made because of its portability and flexibility and detailed explanation can be found in [13]. With robot development and sensory suite extension the larger amount of data had to be transferred over network, including:

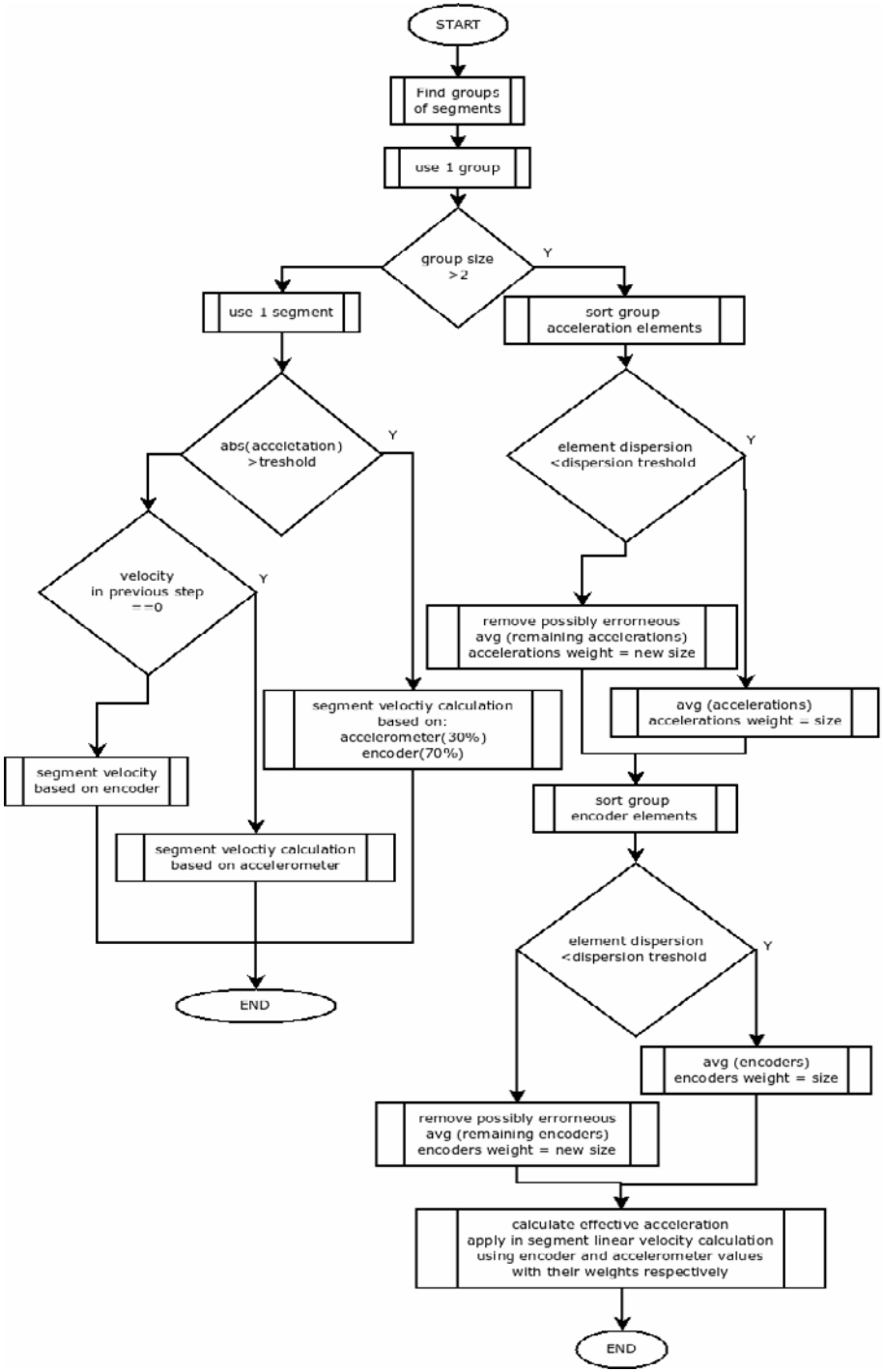


Fig. 10. Basic sensor fusion algorithm

• control commands to robot,
• sensor data from robot,
• video streaming (future plans).

The selected mechanism allows for easy extension of communication features, decreasing probability of programming errors to occur.

In our robot, communication can be divided into 2 sections: control data (shown in Fig. 11) and sensor data (see Fig. 12).

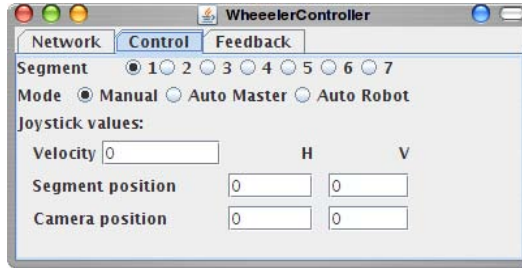


Fig. 11. Control information tab

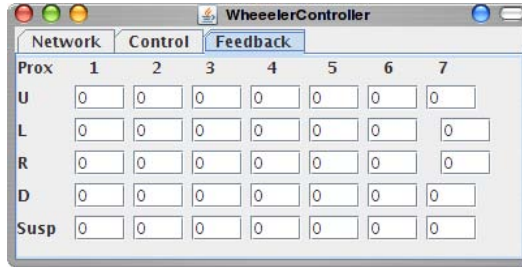


Fig. 12. Sensor information tab

Let us present the current stage of the network interface. The WheeelerControl interface, presented in Fig. 13 is a set of remote methods included in the robot-side servant to be executed by the operator part of control application (details can be found in [14]).

`WheeelerSensors` interface is a servant implemented on the operator side of application. These methods are supposed to be executed by the robot to deliver sensor data to the operator. This interface will be extended with video streaming features as soon as they are implemented in the real robot. Additional set of methods in the `WheeelerControl` interface to access sensors can be considered redundant, however, its real purpose is debugging of application in case of instability in the development phase of the project. In the interfaces mentioned, `set` and `get` methods can be seen, names of which are self explanatory also providing the direction of data flow. Methods starting with prefix `send` are supposed to provide the robot with the references to the remotely available objects on the client side.

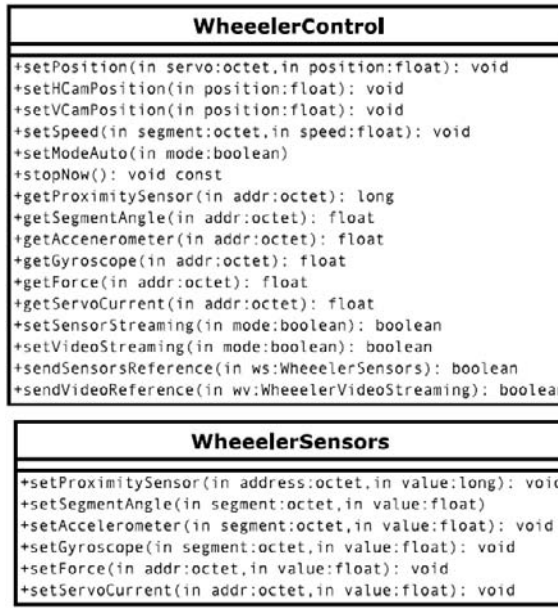


Fig. 13. Declaration of CORBA network interfaces

3.4 Controllers

In order to control all the sensors envisioned in Wheeeler and in order to simplify mechanical fit into the segment we have designed and built specialized controller based on the AT90CAN128 (Atmel), as shown in Fig. 14. We have chosen this processor for the fast AVR structure and relatively high processing power, as for 8-bit controllers. Additionally, in-system programming from Atmel offers reprogramming of each processor of the system directly through CAN bus. This will simplify development procedure of the robot's lowest level software. Local controllers are augmented with all necessary peripherials: 3-axis accelerometers LIS3LV02DL (STMicroelectronics), single axis gyroscope ADIS16100 (Analog Devices), quadrature counters LS7366R (LSI/CSI) and IR distance sensors GP2D120 (Sharp). Functionality of controller can be further extended through serial communication interfaces: CAN, SPI, I2C and RS232. CAN bus is used as a main communication means for data acquisition and control.

Local controllers are daisy-chained along robot's body and connected to the main controller realized on PC104 computer. This main controller gathers data from robot and forwards to operator's station via wireless link. It will also be used to transfer video signal. In the opposite direction, control orders come from operator; they are being analyzed in main controller and distributed to local ones. We are planning that main controller will be also responsible for basic autonomous behavior of Wheeeler.

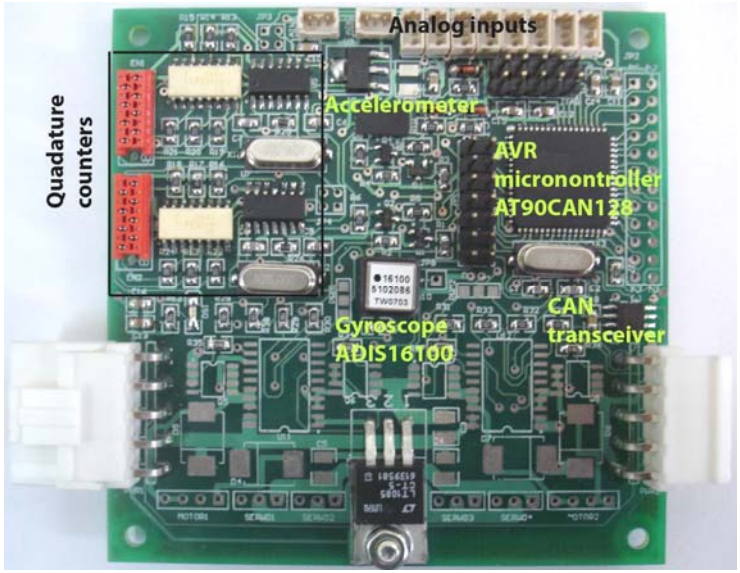


Fig. 14. Local controller mounted on each segment of Wheeeler

4 Experiments

First experiments have been made in the simulation environment. To verify the correctness and efficiency of a suggested control algorithm for this robot a test group of children aged 8-12 was asked to try to fulfill a set of simple tasks (following predefined paths). In the basic one, there was a need to drive through a narrow, straight corridor and traverse debris. In the extended task - climbing up and down the stairs and traversing a trench was required.

After a brief introduction to robot control and some preliminary practice, each member of a test group was asked to make several attempts of fulfilling given tasks with various controller features enabled or disabled. After each attempt the experiment participant was asked about opinion on robot behavior and difficulties in its operation. During all attempts robot's path, all variables concerning active joints and forces vectors were recorded. Also, after the use of the most basic version of the controller, group members were asked to form a team to drive the robot manually controlling each joint variable and each axle velocity.

In the experiment it was proven, that this robot would require a lot of practice to be team-controlled. After one hour training the group was unable to fulfill any of the tasks due to joint variables and velocity synchronization problems. With the most basic controller which included joint variable propagation algorithm it was always possible, after 5 minutes of training, to complete the most basic task, however, propagation accuracy was lost on the debris. It was proven that even with the most basic algorithm it is possible to climb up and down the stairs or traverse a trench; however, the operator must have the ability to enable or disable vertical propagation of joint variables. It has to be stated, that for teleoperated stairs climbing it is required to use a scorpion-like pose, preferably with front camera able to be moved vertically.

After building the proof-of-concept version of Wheeeler, consisting of 3 segments, we have made some preliminary tests to verify robot's behavior, power consumption and performance. Results are presented in the following tables. Table 1 compares speed of the robot on the flat terrain (carpet) and supply current measured for three levels of supply voltage. Table 2 shows the current consumption during driving on the inclined steel flat surface (measurement for two inclinations and three voltage levels).

Table 1. Performance of 3 segment Wheeeler on the flat terrain

Supply voltage [V]	Speed of robot [cm/s]	Starting current [A]	Nominal current [A]
5	32.2	1.7	0.7
6	40.0	1.7	0.7
7	46.0	2.3	0.9

Table 2. Power consumption of 3 segment Wheeeler on inclined surface

Inclination [deg]	Supply voltage [V]	Starting current [A] Going up (down)	Nominal current [A] Going up (down)
16.2	5	2.6 (1.5)	1.4 (0.3)
	6	2.7 (1.6)	1.5 (0.3)
	7	2.8 (1.6)	1.7 (0.5)
21.3	5	2.6	2.0 (0.2)
	6	2.7	2.0 (0.3)
	7	2.8	2.0 (0.4)

5 Conclusions

Hypermobile robots, which are a subcategory of articulated, snake-like robots, are a very interesting field of research. There is a need for robots which are capable of operating in a rough terrain - natural or urban. Typical tasks to be considered are:

- climbing or avoiding obstacles,
- going through trenches,
- moving in narrow spaces such as tunnels, ventilation shafts, ruins inspection.

The common features of hypermobile robots are actuated wheels or tracks, multiple joints on the robot's body allowing it fit into narrow spaces, and relatively large length compared to its width (or field of its cross-section).

The most significant difficulty in designing a hypermobile robot is the need of synchronization of every actuated element during the robot movements. This issue is the main objective of presented project of Wheeeler.

We expect that currently implemented control algorithms let the above requirements to be fulfilled. Nevertheless, for the convenience of the operator and improved safety and reliability of robot operation several improvements should be implemented:

- robot pose should be verified – whether it has not overturned. In such case the robot should be able to restore its default pose automatically, if only surrounding objects do not make it impossible,

- image recognition could be implemented to increase robot autonomy in case of loss of communication with the operator or just to improve teleoperation convenience,

Also a very important feature to be implemented in the near future is video streaming, through CORBA. It is supposed not only to provide a visual feedback, but also supply additional information about the environment through further video processing.

References

1. Baker, J., Borenstein, J.: The Joysnake A Haptic Operator Console for High-Degree-of-Freedom Robots. In: 2006 International Joint Topical Meeting: "Sharing Solutions for Emergencies and Hazardous Environments", Salt Lake City, USA, February 12-15 (2006)
2. Granosik, G., Hansen, M., Borenstein, J.: The OmniTread Serpentine Robot for Industrial Inspection and Surveillance. Industrial Robots International Journal, Special Issue on Mobile Robots IR32-2, 139–148 (2005)
3. Granosik, G., Borenstein, J., Hansen, M.G.: Serpentine Robots for Industrial Inspection and Surveillance. In: Huat, L.K. (ed.) Industrial Robotics. Programming, Simulation and Applications, pp. 633–662. the plV pro literatur Verlag, Germany (2007)
4. Granosik, G.: Hypermobile robots. In: Aschemann, H. (ed.) New Approaches in Automation and Robotics, pp. 315–332. I-Tech Education and Publishing, Vienna (2008)
5. Hirose, S., Morishima, A.: Design and Control of a Mobile Robot With an Articulated Body. The International Journal of Robotics Research 9(2), 99–113 (1990)
6. Hirose, S., Morishima, A., Tukagosi, S., Tsumaki, T., Monobe, H.: Design of Practical Snake Vehicle: Articulated Body Mobile Robot KR-II. In: Fifth Int. Conference on Advanced Robotics, 'Robots in Unstructured Environments', vol. 1, pp. 833–838 (1991)
7. INSPECTOR SYSTEMS Rainer Hitzel GmbH, `http://www.inspector-systems.com`
8. Kamegawa, T., Yamasaki, T., Igarashi, H., Matsuno, F.: Development of the Snake-like Rescue Robot KOHGA. In: Proc. IEEE Int. Conference on Robotics and Automation, New Orleans, LA, April 2004, pp. 5081–5086 (2004)
9. Klaassen, B., Paap, K.L.: GMD-SNAKE2. A Snake-Like Robot Driven by Wheels and a Method for Motion Control. In: Proc. Of IEEE International Conference on Robotics and Automation, Detroit, MI, May 10-15, pp. 3014–3019 (1999)
10. Long, G., Anderson, J., Borenstein, J.: The OmniPede: A New Approach to Obstacle Traversion. In: Proc. IEEE Int. Conf. on Robotics and Automation, USA, pp. 714–719 (2002)
11. Michel, O.: WebotsTM: Professional Mobile Robot Simulation. International Journal of Adavnced Robotic Systems 1(1), 39–42 (2004)
12. Osuka, K., Kitajima, H.: Development of Mobile Inspection Robot for Rescue Activities: MOIRA. In: Proc. IEEE/RSJ Int. Conference on Intelligent Robots and Systems, Las Vegas, Nevada, pp. 3373–3377 (2003)
13. Pytasz, M., Granosik, G.: Applying CORBA technology for the teleoperation of Wheeeler. In: Kozłowski, K. (ed.) Robot Motion and Control 2007 in the "Lecture Notes in Control and Information Sciences", pp. 311–318. Springer, London (2007)
14. Pytasz, M., Granosik, G.: Object oriented network control of Wheeeler – the hyper mobile inspection robot. In: CD Proc. of IEEE International Workshop on Safety, Security, and Rescue Robotics, Rome, Italy, September 27-29 (2007) ISBN: 978-1-4244-1569-4

15. Schempf, H., Mutschler, E., Goltsberg, V., Skoptsov, G., Gavaert, A., Vradis, G.: Explorer: Untethered Real-time Gas Main Assessment Robot System. In: Proc. of Int. Workshop on Advances in Service Robotics, ASER 2003, Bardolino, Italy (2003)
16. Scholl, K.U., Kepplin, V., Berns, K., Dillmann, R.: Controlling a multi-joint robot for autonomous sewer inspection. In: Proc. IEEE Int. Conference on Robotics and Automation, ICRA 2000, vol. 2, pp. 1701–1706 (2000)
17. Streich, H., Adria, O.: Software approach for the autonomous inspection robot MAKRO. In: Proc. IEEE Int. Conference on Robotics and Automation, New Orleans, LA, USA, pp. 3411–3416 (2004)
18. Takayama, T., Hirose, S.: Development of Souryu-I connected crawler vehicle for inspection of narrow and winding space. In: 26th Annual Conference of the IEEE Industrial Electronics Society, IECON 2000, vol. 1, pp. 143–148 (2000)
19. Zhang, H., Wang, W., Deng, Z., Zong, G., Zhang, J.: A Novel Reconfigurable Robot for Urban Search and Rescue. International Journal of Advanced Robotic Systems 3(4), 359–366 (2006)

The qfix Robot Kits

Stefan Enderle

University of Applied Sciences Isny, Germany
qfix robotics GmbH, Senden, Germany
enderle@qfix.de

Abstract. Currently, there is a strong movement in comprehensive schools towards teaching natural science in a more integrated way. For example, the areas mechanics, electronics and software can be studied individually, but they can also be combined and a mechatronical system can be investigated or even be built. For such classes, we developed a robust, modular, reusable and cost-effective kit for building autonomous mobile robots. These kits consist of aluminium elements, modular controller boards, different sensors and actuators, as well as the free GNU C++ environment and a graphical programming environment.

1 Introduction

Robot building projects are a good means to bring the interesting field of robotics to schools, high-schools, and universities. Studying robotics the students learn a lot about mechanics, electronics, and software engineering. Additionally, they can be highly motivated and learn to work in a team.

Performing a lot of robot building labs with pupils and students, we found that there is a gap between the relatively cheap toy-like kits, like LEGO Mindstorms [1] or Fischertechnik Robotics and the quite expensive mechatronics kits from FESTO or even professional off-the-shelf robots. The toy kits offer a good opportunity to start building robots, but they mostly support the control of only 2 or 3 motors and the same number of sensors. Off-the-shelf robots (see e.g. [2,3,4]) are completely built up, so typically only the programming of the robot can be studied.

Alternatively, there exist a number of controllers, like the 6.270 board or the Handy-Board [5] which come without mechanical parts and so must be used in combination with other toy kits, like RC-controlled cars, or custom-built robots. However, these boards, can control only small motors and are not very expandable.

After building RoboCup robots from scratch [6,7,8,9] and supporting schools developing their own RoboCupJunior robot [10], the authors gained a lot of experience about reasonable mechanical concepts and controller architectures for a usable robot development kit. Thus, we decided to develop the robot kit familiy *qfix* and to provide it to schools and universities.

2 Crash-Bobby – A Differential Drive Robot Kit

We started in 2004 to develop a professional but affordable mobile robot kit. The first one became the so called “Crash-Bobby” kit, a mobile robot platform with two independently driven wheels and a caster wheel. As sensors, the well-known Sharp distance

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 84–95, 2009.

sensors GP2D120 with a range up to 30cm are used. Motors, sensors, the controller board and a battery package are mounted on a 10cm x 10cm base plate (see Fig. 1). The controller board "BobbyBoard", which is the first board developed, drives the two motors as well as three infrared distance sensors in order to implement e.g. a simple collision avoidance behaviour.

Fig. 1. Crash-Bobby, a differential drive robot with two driven wheels and three IR distance sensors

In order to make this robot kit interesting for schools, where it is mainly used today, it was clear that the price for the kit could not exceed the price for a LEGO mindstorms kit (which in 2004 was about EUR 250,-). Thus, our goal was to create a robot kit consisting of aluminium parts for the price of plastics kit.

Today, an improved version of the kit contains a controller board with USB and additionally uses bumpers for detecting collisions, a line sensor for moving along a line and an LCD for displaying messages or status information.

This new Crash-Bobby robot kit is already used in hundreds of schools for teaching the basics of mechanics, electronics and programming. This is mostly done either in the regular physics or informatics classes, or in special work groups with the aim to join e.g. a Eurobot or RoboCup contest.

3 The *qfix* Modules

The main concept behind *qfix* (see [11]) is modularity in the following dimensions:

- **Mechanics:** The mechanical parts are aluminium parts including rods, plates and holders for different sensors and actuators. These parts are the building blocks for constructing mechanical and mechatronic systems, like cars, walking robots, etc. Most parts contain threads and can easily be screwed together, so very robust models can be build.
- **Electromechanics and Electronics:** There already exist many compatible electromechanical and electronical parts including a variety of sensors, actuators, and controller boards. With these components it is possible to make the mechanical

models *move* (by DC motors, servo-motors, stepper motors), *sense* (by tactile, infrared and ultrasonic sensors), and *think* (by powerful controller boards which can be programmed on the PC).

– **Software:** In the software area, modularity is no big deal. The *qfix* software comes with the powerful free GNU C++ toolchain (WinAVR for windows, respective libraries or RPMs for linux). Additionally, it contains an easy-to-use C++ class library for accessing all *qfix* electronics components.

  Since beginners, say, of an age from 12, have problems going directly into C or C++ programming, we developed a graphical programming environment called GRAPE in order to simplify the programming of self-built robots. This software directly produces C++ code from the graphical description and thus supports the beginner in learning object-oriented programming.

3.1 Mechanics

The basic building blocks of the *qfix* system are anodized aluminium rods with holes of diameter 6mm along all four sides and two M6 threads on the front and back side. The currently existing rods have a length from 20mm to 100mm and include rods with a 45^o angle (see Fig. 2).

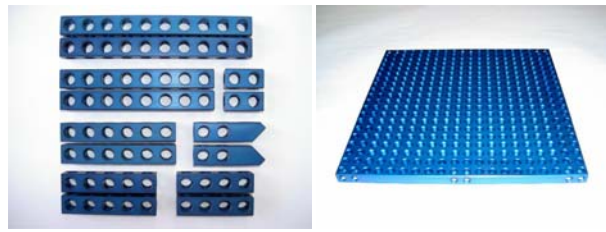


Fig. 2. Left: Basic rods from 20mm to 100mm. Right: Plate with 200x200mm and 400 threads.

Other basic elements are a variety of plates with holes and threads. These plates can be bolted to the rods using a screw and a nut or only a screw exploiting the rods' frontal threads. Like the rods, the plates are given in different lengths and widths, currently up to 200mm x 200mm (see Fig. 2 for an exemplary plate).

All mechanical parts use holes and threads according to DIN/ISO standards and have a grid of 10mm.

In Figure 3, some additional mechanical elements can be seen: wheels, casterwheels, gears, and axles. They are usually used to implement dynamic models which can be driven by different motors as shown in the next section.

3.2 Electromechanics/Electronics

Motors: In order to make a model move, motors are needed. Typical robotics applications often use different kinds of motors for different tasks: DC motors, servo motors,

Fig. 3. Left: Wheels, axles and gears. Right: Omni wheel with its axle.

Fig. 4. Exemplary motor bearing for a DC motor

and stepper motors. *qfix* supports these different categories by providing the respective motor bearings (see Fig. 4) and electronics components for driving motor and wheel encoders.

Sensors: When building robots, it is also necessary to make them able to gather information about their environment. This can be done by mounting simple switches signalling bumps into obstacles, or by adding distance measuring devices like infrared or sonar sensors. Like for the motors, we also developed bearings for numerous sensors (see Fig. 5) in order to guarantee an easy but fixed mount of the sensor to the robot.

Controller boards: Obviously, the motors and sensors must be driven by an electronics component. For *qfix*, we developed a new, modular controller board architecture

Fig. 5. Left: Bearing Sharp IR distance sensor. Right: Bearing for a CNY70 line sensor.

which is both powerful and easy-to-handle. The smallest controller we use is the Atmel ATmega32. This controller was used on the first board developed in 2004, the "BobbyBoard". The board has the same characteristics as the new board from 2008, the "MiniBoard", but could only be programmed from the PC by a parallel (printer port) cable. The newer "MiniBoard" (see Figure 6) uses the same controller and supports the following I/Os as well as program and data transfer via USB:

- 2 DC motor controllers (battery voltage, 1A)
- 4 digital inputs (0/5V)
- 4 analog inputs (0-5V)
- 8 digital outputs (battery voltage, 100mA)
- 4 LEDs
- 4 buttons
- I^2C-bus for extensions
- USB for program download and data transfer to the PC

Fig. 6. "MiniBoard": controller board with Atmel ATmega32 controller and USB port

Further developed main boards are the "CAN128Board" which shows the same I/O capabilities but uses an Atmel AT90CAN128 controller with CAN interface, more memory and more speed. And, the "SoccerBoard" with 8 analog and 8 digital inputs, 8 digital outputs, 6 motor drivers, and optional CAN and USB interface (see Figure 7). This board is specifically designed for the requirements of RoboCupJunior, where often omnidrive platforms with three driven wheels plus a kicker and a "dribbler" are used combined with multiple sensor systems.

All controller boards can basically be programmed ("flashed") from the PC via a serial, parallel or USB link. However, for the newer boards, the USB connection is preferred and supported by a USB bootloader mechanism.

Extension Boards: The main idea behind the *qfix* boards is their flexible modular architecture: The main controller board runs the main program and communicates with expansion boards for setting actuator values and getting sensor data. The expansion boards themselves are responsible for controlling the attached devices, so the main processor does not have to perform expensive tasks, like feedback motor control, etc.

Fig. 7. “SoccerBoard”: controller board with Atmel ATmega128 controller

The controller boards contain an I^2C-bus and optionally a CAN bus which both allow to chain dozens of boards of the same or different kinds to a large controller network. So, it is possible to either control more I/Os or even to implement distributed applications with decentralized control (see e.g. [12]).

The following extension boards based on I^2C-bus are currently available:

- **Servo board 1:** The servo-board uses a Atmel mega8 for controling 4 servo motors independently.
- **Servo board 2:** This servo-board is designed for humanoid robots and can control 24 servo motors independently. It contains a mega128 controller and a Xilinx FPGA for fast I/O control.
- **Stepper-board:** The stepper-board can control 4 stepper motors independently. Both, full and half step mode are supported.
- **DC-power board:** The DC-power-board is capable of controlling two DC motors with 4A each. It also contains two encoder input lines for each motor.
- **LCD display board:** An LCD display with 4 lines of 20 characters each (see Figure 8).
- **Relais boards:** There are two relais boards: one to be connected to the digital output of the controller board and one to be connected via the I^2C-bus.

Further expansion boards, e.g. for Polaroid sonar sensors [13] and a camera board are currently under development.

3.3 Software

With *qfix* we provide the free GNU C++ toolchain including generic tools for downloading programs to the controller boards. Additionally, we provide a C++ class

Fig. 8. “LCD display board”: expansion board with LCD display

library supporting all *qfix* boards. On Windows, the generic tools mainly consist of the WinAVR GCC environment for Atmel controllers which includes the extensible editor *programmers notepad* and powerful download tools, like *avrdude*. All tools also run on Linux/Unix and Mac, so cross-platform development is fully supported.

The easy-to-use *qfix* C++ class library hides the low-level hardware interface from the programmer and supports the complete *qfix* extension board family. The main idea is to provide a specific C++ class for each *qfix* module. Therefore, the library provides the classes `MiniBoard, SoccerBoard, LCD, SlaveBoard, StepperBoard, ServoBoard, RelaisBoard,` etc. For example, when building an application with the BobbyBoard and the LCD you use the respective classes, like the following:

```
#include "qfixBobbyBoard.h"      // include BobbyBoard library
#include "qfixLCD.h"             // include LCD library

int main()
{
  BobbyBoard board;              // construct object "board"
  LCD        lcd;                // construct object "lcd"

  board.ledOn(0);                // turn on LED 0
  board.waitForButton(0);        // wait until button 0 is pressed
  board.motor(0,255);            // turn on motor 0 to full speed
  lcd.print("Engines running");  // print a text on the LCD
}
```

As can be seen from the comments of the code, two instances of two classes are constructed: `board` and `lcd`. Their methods are called in order to let the main board turn on a LED and a motor, wait for a button press, and output text on the LCD.

Both classes hide a lot of functionality in their constructors. When constructing the object `board` for instance, the constructor initializes all I/O pins and starts an interrupt routine for motor PWM control. When constructing `lcd`, the constructor opens an I^2C-bus channel and starts communicating with the physically connected LCD display. This mechanism works perfectly as long as expansion boards of different types are used only.

When using multiple expansion boards of the same type, the extended construction syntax can be used in order to connect each object to the correct physical board. This is show in the next listing. Imagine you have a controller board and three identical LCD display boards:

```
#include "qfixBobbyBoard.h"      // include BobbyBoard library
#include "qfixLCD.h"             // include LCD library

int main()
{
  BobbyBoard board;              // construct object "board"
  LCD        lcd0(0);            // construct object "lcd0"
  LCD        lcd1(1);            // construct object "lcd1"
  LCD        lcd2(2);            // construct object "lcd2"

  board.waitForButton(0);        // wait until button 0 is pressed
  lcd0.print("Hallo");           // print a text on LCD 0
```

```
  lcd1.print("World!");          // print a text on LCD 1
  lcd2.print("Engines running"); // print a text on LCD 2
}
```

In this example, each one of the three `lcdx` objects is connected to the physical LCD board with the respective ID. This ID can be hardcoded to the LCD by flashing the LCD board, or it can be dynamically changed by calling the method `changeID(newID)` of class `LCD`.

For those who want to connect multiple controller boards but do not want to go into detail with programming the I^2C-bus, we provide a class `SlaveBoard` which can be used as a "remote control" for connected BobbyBoard main boards:

```
#include "qfixBobbyBoard.h"      // include BobbyBoard library
#include "qfixSlaveBoard.h"      // include SlaveBoard library

int main()
{
  BobbyBoard master;             // construct a master board object
  SlaveBoard slave0(0);          // construct a slave board object
  SlaveBoard slave1(1);          // construct a slave board object

  master.motor(0,100);           // turn on motor on master board
  slave0.motor(0,100);           // turn on motor on slave board 0
  slave0.waitForButton(0);       // wait for button on slave board 0
  slave1.ledOn(0);               // turn on LED on slave board 1
}
```

4 Graphical Programming Environment GRAPE

In addition to the C++ environment, we developed a new software system called *GRAPE* (which stands for GRAphical Programming Environment). With GRAPE it is possible to program the *qfix* controller boards in an object oriented way without having experience in C++.

The GRAPE application consists of three tabbed windows which are used sequentially: In the first tab, the desired classes (e.g. BobbyBoard and LCD) are loaded. Each class can then be instantiated by one or more objects. The object names can be freely chosen. The second tab holds the main window for graphical programming. Here, symbolic blocks are arranged intuitively in order to get a flow chart with the desired program flow (see Figure 9).

For each symbolic icon, a properties dialog can be opened to define the semantics of the icon in a semi-graphical way: For *commands*, the user can select an object from the list of instantiated objects, then chose a method from the object's possible methods, and then select the desired parameters from the list of possible parameters for the chosen method. This selection defines all parts of a typical object-oriented method call: `<object>.<method>(<parameters>)`.

After filling all graphical blocks with their respective meaning, the flow chart can be saved as a XML description file. This makes it possible to perform, e.g. in an individual tool, the translation to any object-oriented (or even classically procedural) programming

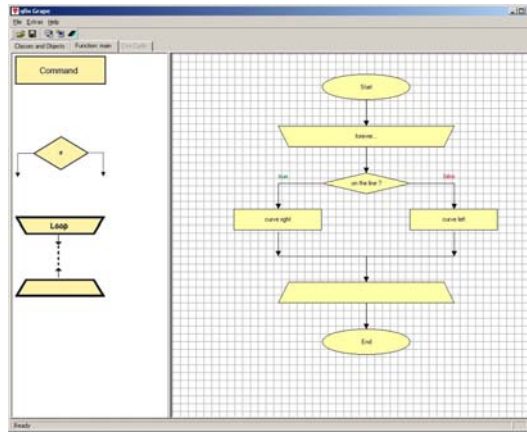


Fig. 9. Graphical program in GRAPE

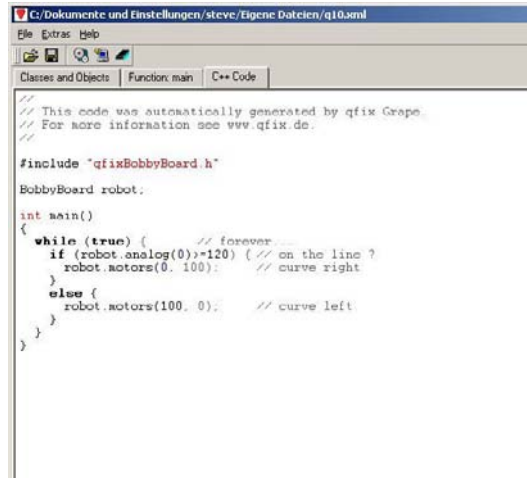


Fig. 10. Automatically generated C++ code in GRAPE

language. In GRAPE, this translation is already integrated and the flow chart (or internally, the XML representation) is automatically translated to C++ code (see Figure 10).

With this approach, the basic concepts of a procedural programming language can easily be learned: commands, sequences of commands, if-clauses, while loops. And, it can be studied how these concepts are translated to C++ or another programming language. In addition to that, the users learn to use given class libraries.

5 Experiments

In order to demonstrate the feasibility of the *qfix* parts and controller boards, we developed some typical robot applications.

5.1 Offroad Robot

Figure 11 shows an ”offroad” robot which was initially built in order to test the power of the motor controllers of the BobbyBoard.

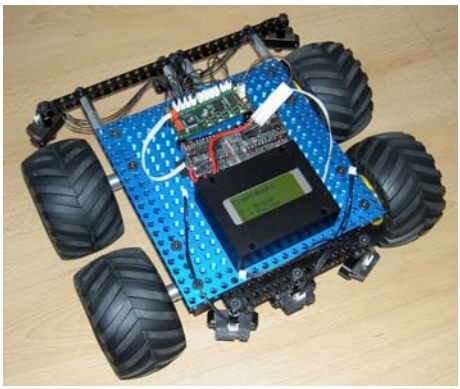


Fig. 11. Offroad robot with differential drive

So, for this robot, the same mainboard as in the small differential drive robot is used and drives four stronger motors, where the left and the right ones are connected in parallel. The complete platform is much bigger (main plate of 20x20cm) than the Crash-Bobby platform and includes a boxed version of the LCD display. You can also see the infrared distance sensors to the front and to the back with a range of up to 80cm (Sharp DP2D12).

5.2 Soccer Robot

Another application, a special soccer robot, was built in order to demonstrate the flexibility of both mechanics and electronics components. As main platform we used a round plate of about 21cm diameter with three omnidirectional wheels (see Figure 12).

Fig. 12. Omnidrive platform with three omnidirectional wheels

In order to control the three motors, two controller boards were connected via the I^2C-bus and communicate with each other to establish a reliable movement coordination. Additionally, the resulting soccer robot uses a kicker device and a so called "dribbler" to hold the ball near the robot. As sensors, infrared light sensors are used for detecting a RoboCupJunior ball. For obstacle avoidance, infrared or ultrasonic distance sensors can be attached. The complete soccer robot including a trendy skin or "tricot" is shown in Figure 13.

Fig. 13. Soccer robots

6 Conclusion

We presented *qfix*, a construction kit for developing autonomous mobile robots and other mechatronics applications. *qfix* was mainly developed for educational and edutainment purposes. The kit consists of solid mechanical and electro-mechanical parts, powerful modular controller boards with several extension boards, and a complete C++ class library for easy support of all functionality.

Since the kits are often used in the RoboCupJunior area, where the users are only 12 or even less years old and have no programming experience, we developed the graphical programming environment GRAPE. This tool supports object oriented programming on a graphical level but directly generates C++ code which can be studied and edited.

The complete *qfix* robot kit family proves to be an appropriate tool for robot development. It is already used in educational classes and labs in schools and at universities. Additionally, the open architecture encourages the robotics community to help improving the kits.

Acknowledgements

This work is sponsored by qfix robotics, Germany (`www.qfix-robotics.com`). We also thank Bostjan Bedenik who mainly implemented the Grape software.

References

1. Baum, D., Gasperi, M., Hempel, R., Villa, L.: Extreme Mindstorms – An Advanced Guide to LEGO Mindstorms. Apress (2000)
2. ActivMedia: Pioneer 1 Operation Manual. RWI, Jaffrey, NH (1996)
3. K-Team: Khepera – user manual (1999)
4. Nomadic, http://www.robots.com
5. HandyBoard (1998), http://lcs.www.media.mit.edu/groups/el/Projects/handy-board/index.html
6. Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawa, E., Matsubara, H.: RoboCup – a challenge problem for AI. AI magazine, 73–85 (Spring 1998)
7. (RoboCup), http://www.robocup.org
8. Enderle, S.: The Sparrow-99 robot. Technical report, University of Ulm, Internal report (1999)
9. Utz, H., Sablatnög, S., Enderle, S., Kraetzschmar, G.K., Palm, G.: Miro – Middleware for mobile robot applications. IEEE Transactions on Robotics and Automation, Special issue of on Object Oriented Distributed Control Architectures (2002)
10. RoboCupJunior, http://www.robocupjunior.org/de
11. Enderle, S.: The robotics and mechatronics kit qfix. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006: Robot Soccer World Cup X. LNCS, vol. 4434, pp. 134–145. Springer, Heidelberg (2007)
12. Kaiser, J.: Real-time communication on the CAN-bus for distributed applications with decentralized control. In: 4th IFAC International Symposium on Intelligent Components and Instruments for Control Applications, Buenos Aires, Argentina (2000)
13. POLAROID: Ultrasonic Ranging System. Polaroid Corporation, 784 Memorial Drive, Cambridge, MA 02139 (1991)

Pilot Study of Person Robot Interaction in a Public Transit Space

Mikael Svenstrup[1], Thomas Bak[1], Ouri Maler[1], Hans Jørgen Andersen[2], and Ole B. Jensen[3]

[1] Department of Electronic Systems, Automation & Control, Aalborg University, 9220 Aalborg, Denmark
{ms,tba}@es.aau.dk
[2] Department for Media Technology, Aalborg University, 9220 Aalborg, Denmark
hja@cvmt.aau.dk
[3] Department for Architecture and Design, Aalborg University, 9000 Aalborg, Denmark
obje@aod.aau.dk

Abstract. This paper describes a study of the effect of a human interactive robot placed in an urban transit space. The underlying hypothesis is that it is possible to create interesting new living spaces and induce value in terms of experiences, information or economics, by putting socially interactive mobile agents into public urban transit area. To investigate the hypothesis, an experiment was carried out at a bus terminal serving both as a transit space and a shopping mall, where an autonomous robot were to detect and follow random people. The people that were followed were asked to fill out a questionnaire for quantitative analysis of the experiment. In addition video documentation of the experiment was used in the evaluation. The results showed that people were generally positive towards having mobile robots in this type of environment where shopping is combined with transit. However, it also showed harder than expected to start interaction with commuters due to their determination and speed towards their goal. Further it was demonstrated that it was possible to track and follow people, who were not beforehand informed on the experiment. The evaluation indicated, that the distance to initiate interaction was shorter than initially expected, but complies with the distance for normal human to human interaction.

Keywords: Human-Robot Interaction, Transit Space, Pilot Study.

1 Introduction

When robots move from the laboratory or factory, and out into the everyday human environments, they will have to interact with humans, and accordingly human-robot interaction is a novel and growing research field [1,2,3]. For robots

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 96–106, 2009.

to be able to behave appropriately in human environments, they will have to possess skills to interact with, and learn from people as well as behave in a social way that will be acceptable, as perceived by humans. Experiments with robot motion and people's perception of robots have been investigated by several authors. For example, the level of comfort for the test subjects are investigated in [4,5,6] through wizard-of-oz experiments.

Interaction between a robot and a human not only rely on the ability to input, output, and process information, but to move socially around, it is also necessary to know something about the spatial relationship between the two actors. In [7] human-human proxemics, i.e.the spatial distances between persons in interaction, were studied, and in [8] experiments were made, that support the division into these zones. [9] concerns models describing social engagement based on spatial relationships, and this is also investigated in [10], where a robot system is implemented to learn, perceive and understand the spatial dimension of an indoor human made environment. This calls not only for skills for interacting with humans, but also for navigating in populated public areas. In [11] an approach for coordinating motion of a robot in a crowded human environment with moving obstacles is described, and an experiment with a robot at a train station is done in [12]. In [13], a navigation strategy enabling a robot to join a group of persons engaged in a conversation is developed. The above research concerns how to implement a robot in public spaces, but where are the actual potential of mobile agents in public spaces?

The global transportation activity and congestion has increased the time we spend in transit [14], and thus *transit spaces* opens new opportunities for human interactive robots. The spaces of transit are seen as sites of interaction where people meet and thus (potentially) engage [15]. In Denmark for example there is a clear correlation between GNP and Mobility during the period 1994-2004 [16]. Mobility thus makes up an increasing proportion of contemporary everyday life ([17]). Much mobility is done in transit and in designated *transit spaces* such as airports terminals and shopping centers.

Focus in this paper is on the willingness of people to interact with a robot, and how people reacts when robots suddenly is a part of their daily environments.

The general hypothesis is that it is possible to add value, both in terms of experience, information and economics, to urban transit spaces by inducing socially intelligent robots. This is supported by the assumption that, by putting interactive social robots into a transit space, interesting interaction will arise.

To be able to verify this hypothesis and create robots suitable for these public spaces, it is necessary to conduct experiments investigating how robots are perceived by humans in the public space. It is also necessary to find out how the robot should be designed and what behavior algorithms should be used to get into interaction with humans. The effect of placing a robot in a human environment is analyzed through an experiment where a robot is placed in a transit space and shopping center, and tries to start interaction by following people it detects.

2 Methods

The experiment documented in this paper is designed to give some indication to the validity of the hypothesis as outlined in the introduction. To do this two experimental elements have been explored: 1) demonstration of the ability of a robot to detect, track and follow people in a natural way, 2) investigation of peoples perception of the robot, the space and the interaction. So both the technical aspects of placing mobile robots in complex open-ended environments are investigated, and so is the understanding of the nature and potential of robotic agents in public urban transit spaces.

2.1 Experimental Setup

Location. The experiment was performed in the combined central bus station and shopping mall the *Kennedy Arkaden* in Aalborg, Denmark, on 13th of December 2007 from 9am to 1pm. The space is characterized by being used both by commuters and shoppers having patterns of motion ranging from very determined to wandering behavior. Hence, the space may be categorized a combined public and urban transit space giving a very dynamic and diverse experimental environment. In the setup there was no primarily information about the experiment for the persons involved.

Robotic Platform. The basis for the experiment was a robotic platform, Robotino, from FESTO. To facilitate interaction the robot was dressed as Santa Claus (the experiment was carried out just before Christmas) and equipped with a loudspeaker playing the well known Christmas jingle "Jingle bells" when a person was detected and the robot tried to initiate interaction. In addition the robot was equipped with a head with 126 red diodes which enables it to to express emotions like happy, sad, surprised or confused or even showing a question mark. Expressing emotions, is an important feature for robot-human interaction as indicated in [18,19].

Person Detection. To enable detection of persons in the environment, a simple method relying on a laser range finder was employed. An algorithm for detecting legs of persons and converting these to persons described in [20], was implemented on the platform. This algorithm provides a list of persons within the range of the laser, and the algorithm was modified to keep track of the individual persons in the list, so that a specific person can be chosen as the target for interaction.

Control of the Robot. The robot behavior is inspired by the spatial relation between humans (proxemics) as described in [7]. Hall divides the zone around a person into to four categories, 1) the public zone $> 3.6m$, 2) the social zone $> 1.2m$, the personal zone $> 0.45m$, and the intimate zone $< 0.45m$. Social spaces between robots and humans were studied in [8] supporting the use of

Hall's proxemics distances and the human robot interaction is therefore designed to be able to experiment with different distances.

For control the robot programming framework Player [21] was installed on the Robotino platform and the following behavior scheme was developed (see Fig. 1):

1. Roam randomly around until a person is detected.
2. Start smiling and play a jingle.
3. Follow that person keeping a specific distance, until the person is lost, or a certain time interval has elapsed.
4. Change facial expression and start roaming again.

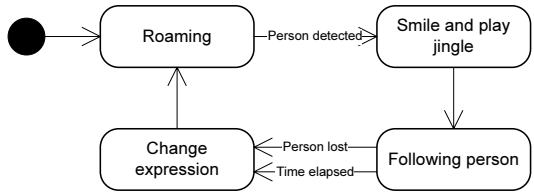


Fig. 1. The state diagram, which was implemented on the robot for the experiment

To keep the desired distance to the person being followed, two decoupled PID controllers, one for velocity and one for rotation, was used. The desired distance was initially set to 1,4 meters to make the robot stay in the social zone of the persons. Later in the experiment, the distance was decreased to 1 meter to investigate if it has any effect on the initiation of interaction.

2.2 Evaluation Methods

The outcome of the experiment was evaluated by, questionnaires, video recordings, and in situ observations. The questionnaire was done by interviews of the persons that were in interaction with the robot. All interviews were done immediately after the interaction.

The observations and the video analysis was used to gain knowledge about the reactions of people and positive and negative aspects of the procedure of the experiment.

Questionnaire. The interviewed persons were asked about age, sex, occupation, business at the location, and the frequency of their visits to this specific transit space. The interviewed persons were then asked to rate the following questions on a five point Likert scale from good to bad. How would you describe the arcade as a public space in terms of

- What is your impression of this transit space?
- Is it a good place for social networking?

These questions were followed by questions about the experience with the robot:

- How did you experience the robot when you first noticed it?
- How did you feel when it followed you?
- How do you think a robot like this fits into a place like this?
- How do you think robots in 20 years will be a part of transit spaces? (assistants - as entertainment - surveillance - there will be no robots)
- How many robots do you think will be present in future transit spaces? (few - as today - they will be everywhere)

All answers were recorded for future analysis.

3 Results and Observations

During the experiment 48 different persons were interviewed, and most of the them answered all the questions. Only two persons answered too few questions to be used in the evaluation.

3.1 Person Detection and Robot Controller

A typical scenario from the experiment is seen in Fig. 2, where a mother and a child interact with the robot. The output from the laser range finder from that same scene is shown in Fig. 1.

Fig. 2. A mother and a child interacting with the robot, a typical situation from the experiment

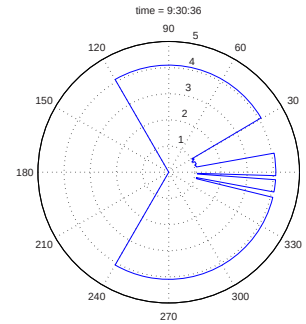


Fig. 3. Typical readings from the laser rangefinder. The laser image corresponds to the scenario in Fig. 2. The radial scale on the polar plot is in meters.

The experiment showed that the algorithm was able to detect the persons in the area. However, due to persons generally moving faster than expected, and only 4 meters of laser vision, the control loop for caching up with persons to be followed, sometimes reacted too slow, and people were lost on that behalf. Because of this, some parameters of the controller were tuned during the experiment to make the robot move faster. This increased the ability to follow

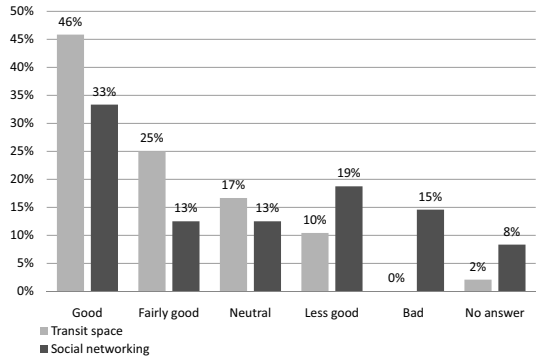


Fig. 4. Answers from questions on peoples overall experience of the space as a transit space and as a place for social networking

people, but the robot still did not move fast enough for following all people. The controller could be tuned to make the robot move even faster, but this is also a compromise between fulfilling the desire to make the robot move in a slow, smooth and comfortable way.

3.2 Answers to Questionnaire

Of the 48 respondents, 47% were commuters, while 29% were shoppers, reflecting the combined transit and shopping space. The average age of the respondents was 45 years and 65% were women. A summary of the answers to a selection of the questionnaire are displayed in the three histograms in Fig. 4-6.

88% of the questioned people think of the transit/shopping space the "Kennedy Arkaden" as Neutral to Good place. So people in general experience the space as a comfortable not hostile environment - a good basis for interacting with the robot (see Fig. 4). 58% rate the space as Neutral to Good as a space for social networking.

Regarding questions about how people experienced the robot the majority (92%) answered that they were positive towards (Neutral to Good) when they first noticed it and it started to approach them. Further, in general people (67% Neutral to Good) did not feel uncomfortable when the robot followed them, though a few persons answered that did not like the robot following them (see Fig. 5).

Of the people questioned 90% found that a robot would fit Neural to Good into a transit space like the "Kennedy Arkaden", see Fig. 6.

When asked if they expect robots and what role they would have in transit space in 20 years, 23% expect robots to have a role in entertainment, 40% see robots as assistants, helpers or guides when used in transit spaces and 6% expect them to have a role in surveillance. 50% expect robots to be everywhere in 20 years, 29% only expect a few and 8% expect a situation as today.

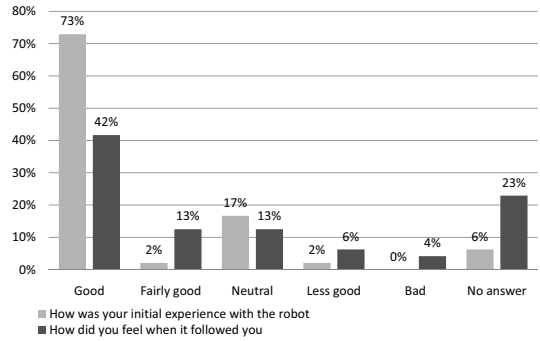


Fig. 5. Answers from questions on peoples experience with the robot, initially and after the robot started following them

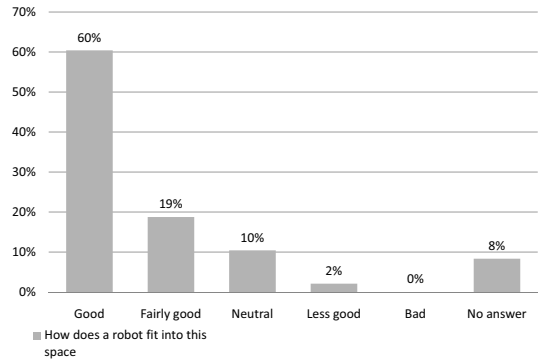


Fig. 6. Answers from question on peoples opinion regarding the use of robots in transit spaces

3.3 Observations

During the experiment and by video analysis the following were noted:

- When individual persons were near the robot, they either did not notice it, give it a curious look and then go, or, in a few cases they played with it a bit. However, when larger crowds formed around it, they seemed to get bolder; there was much more playing with the robot then, essentially turning it into an attraction of the arcade. This actually contradicts the results found in [22], where an experiment shows that if a person is close to the robot, other people tend not to get close to the robot also.
- People in transit like e.g. commuters (who knew exactly where to go) tend to completely ignore the robot even when it followed them. It was more people who had time to stop, people with children, or people who just liked the entertainment, who reciprocated the interaction.
- When changing the desired distance from 1,4 meters to 1 meter, i.e. moving into the personal zone, people tended to start interaction much more often.

– A lot of the time the robot was roaming around in one end of the transit space having nothing to do. But at the same time, there might be people in the other end of the environment, who the robot could not see and who were potential subjects for interaction.

4 Discussion

4.1 People Detection

The detection of persons seemed to work very well, but the persons are modelled with no dynamics, which means that persons should be close to where they were before to be certain to detect that it is the same person in the next iteration of the algorithm. When e.g. the robot is turning, the person positions as seen from the laser changes rapidly, and this can sometimes be a problem for the laser. It is therefore suggested to improve the person detection algorithm with information about the movement of the persons and the robot itself, or a range finder with a longer field of vision.

Instead of trying to make the robot better at following people, this could also be interpreted in an other way; the robot should not try to start an interaction when a person does not show the slightest interest in the robot. If for example you each day passed through the transit space to get to the office or home, then you would not be interested in, or even be annoyed when, robots try to interact with you all the time. Therefore research should focus on the persons who have interest in interaction, either when they want information, help, experience or other things. In this context it would be of great benefit if the robot were able to learn to see, what the intentions of each person are. This could for example be done using Case Base Reasoning, which is used in [23] and [24].

Sometimes during the experiment, some people were standing and looking at other persons who were interacting, or several persons tried to get in contact with the robot at the same time. At these times it was the robot who needed assistance. Therefore it could be interesting to investigate the possibility of multiple cooperating robots, who helps each other in completing the tasks in transit spaces. This could enable robot A to call for help, or informing robot B that at some place, there is a person who needs assistance, but robot A is already occupied. Sharing information would also enable the robots to get a better overview over the whole situation, and e.g. make an adaptive map over the environment. Such a system could also be supported by other sensors, like surveillance cameras or sensors detecting people moving into or out of the area. This could also eliminate or reduce the cases, where the robot roams around in one end, and therefore does not observe interesting persons in other ends of the transit space.

4.2 Quantitative Evaluation

Generally the people questioned were positive towards the "Kennedy Arkaden" as a place for transit and shopping (see Fig. 4). Of the respondents most were commuters. The observation was that commuters were more difficult for the

robot to follow, and the overall number of commuters in the space is hence higher than indicated in the numbers. The high number of commuters may also influence the attitude towards the "Kennedy Arkaden" as a place for social networking. The people positive towards the space as a place for social networking was lower than the people positive towards the space in general. The response to the two questions as outlined in Fig. 4 indicate a positive attitude towards the transit space, which is the basis for the experiment. Introducing a robot in an environment where the general attitude is hostile, may be much more difficult.

The majority of the respondents where were positive towards having a robot entering the transit space (see Fig. 5). This supports the hypothesis that it is possible to add value to urban transit spaces by putting interacting robots into them. Furthermore, most people feels comfortable towards having the robot in the environment, even when the robot were following them, though it was slightly less in this case. As the interaction interface was very simple (sound and motion), it may not have been clear, what was the purpose of the interaction.

People started to interact more when the interaction distance was set to be closer, probably due to the robot getting into the personal zone, where it can not just be ignored. This result complies with logic, since when we are moving normally around in public spaces, we pass a lot of people within the social zone. But unless it is very crowded, we try go stay out of the personal zone of people we do not interact with. In this context, it is also important for the robot to stay out of the intimate zone, which according to [25] is uncomfortable for test persons passing a robot.

Also supporting the hypothesis is the fact that 90% (Neutral to Good) found that robots could be used in transit spaces. The questionnaire also gave some idea of the potential use of robotics as seen by the respondents, where the majority (40%) sees robots as assistants. 50% expect robots to be everywhere in 20 years, which given the relative high average age (45 years) of the respondents is interesting and provides a very good basis for the extension of robotics into peoples everyday lives.

5 Conclusion

This paper has described an experiment with an autonomous robot, driving around and following random people in an urban transit space. The results show that people are generally very positive towards the idea of having more robots assisting or entertaining persons in public spaces. About interaction it was found that initiating interaction, is much easier, if the robot comes into the personal zone of the persons. It was also demonstrated that the algorithm for detecting, tracking and following persons worked, if the persons did not move too fast (below approximately 1,5 m/s), though it was possible to tune the controller for better behaviour during the experiment.

The purpose of the experiment was to get some indication of the hypothesis that it is possible to add value to urban transit spaces by putting social robotic agents into them. The experiment supports the hypothesis in terms of most persons being positive and interested in the robots existence in the public space.

However, there is still a long way to go before we see sociable robots naturally integrated into everyday human environments.

In this experiment essential knowledge have also been gained about making experiments in an open-ended environment, where you are not able to know all factors beforehand as opposed to laboratory experiments.

An interesting prospect for future work is to actually try to detect if people is interested in interaction from their movement pattern, and only try to start interaction if that is the case. Additional it can be concluded that it would be of great benefit if several robots were able to cooperate about solving the tasks for mobile robots in urban transit spaces.

The robot described in this paper is the relatively "dumb" first generation mobile agent in a transit space. Taking small steps towards the final goal of a human interactive robot in a public space, the next generation could be overlayered with one-way information technologies (e.g. traffic information). Third generation robots could be added interactive technologies (e.g. games and interactive communication) and thus become the "digital passenger" of the future.

References

1. Dautenhahn, K.: Methodology & themes of human-robot interaction: A growing research field. International Journal of Advanced Robotic Systems 4(1), 103–108 (2007)
2. Thrun, S., Bennewitz, M., Burgard, W., Cremers, A., Dellaert, F., Fox, D., Hahnel, D., Rosenberg, C., Roy, N., Schulte, J., Schulz, D.: Minerva: a second-generation museum tour-guide robot. In: Proceedings of IEEE International Conference on Robotics and Automation, May 10-15, vol. 3, pp. 1999–2005 (1999)
3. Kanda, T.: Field trial approach for communication robots. In: Proc. 16th IEEE International Symposium on Robot and Human interactive Communication RO-MAN 2007, pp. 665–666 (2007)
4. Syrdal, D.S., Dautenhahn, K., Woods, S., Walters, M.L., Koay, K.L.: Doing the right thing wrong personality and tolerance to uncomfortable robot approaches. In: The 15th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN 2006), Hatfield, UK (September 2006)
5. Sisbot, E., Alami, R., Simeon, T., Dautenhahn, K., Walters, M., Woods, S.: Navigation in the presence of humans. In: 5th IEEE-RAS International Conference on Humanoid Robots, December 5, pp. 181–188 (2005)
6. Koay, K.L., Walters, M., Dautenhahn, K.: Methodological issues using a comfort level device in human-robot interactions. In: IEEE International Workshop on Robot and Human Interactive Communication, 2005. ROMAN 2005, August 13-15, pp. 359–364 (2005)
7. Hall, E.T.: A system for the notation of proxemic behavior. American anthropologist 65(5), 1003–1026 (1963)
8. Walters, M.L., Dautenhahn, K., te Boekhorst, R., Koay, K.L., Kaouri, C., Woods, S.N., Lee, D., Werry, I.: The influence of subjects personality traits on personal spatial zones in a human-robot interaction experiment. In: Proc. IEEE Ro-man, Hashville, August 2005, pp. 347–352 (2005)
9. Michalowski, M., Sabanovic, S., Simmons, R.: A spatial model of engagement for a social robot. In: The 9th International Workshop on Advanced Motion Control, AMC 2006, Istanbul (March 2006)

10. Mozos, O.M., Jensfelt, P., Zender, H., Kruijff, G.J.M., Burgard, W.: An integrated system for conceptual spatial representations of indoor environments for mobile robots. In: Proceedings of the IROS 2007 Workshop: From Sensors to Human Spatial Concepts (FS2HSC), San Diego, CA, USA, November 2007, pp. 25–32 (2007)
11. Prassler, E., Bank, D., Kluge, B.: Motion coordination between a human and a mobile robot. In: IEEE/RSJ International Conference on Intelligent Robots and System, 2002, September 30- October 5, vol. 2, pp. 1228–1233 (2002)
12. Hayashi, K., Sakamoto, D., Kanda, T., Shiomi, M., Koizumi, S., Ishiguro, H., Ogasawara, T., Hagita, N.: Humanoid robots as a passive-social medium: A field experiment at a train station. In: Proceedings of the 2007 ACM/IEEE Conference on Human-Robot Interaction, Arlington, VA, United States, pp. 137–144 (2007); Train station; Communication robot; Human robot interaction
13. Althaus, P., Ishiguro, H., Kanda, T., Miyashita, T., Christensen, H.: Navigation for human-robot interaction tasks. In: Proceedings of IEEE International Conference on Robotics and Automation, 2004. ICRA 2004, April 26-May 1, vol. 2, pp. 1894–1900 (2004)
14. Urry, J.: Mobilities. Blackwell, Malden (2007)
15. Jensen, O.B.: Facework, flow and the city – simmel, goffman and mobility in the contemporary city. In: Mobilities. vol. 2, pp. 143–165 (2006)
16. Infrastrukturkommissionen: Infrastrukturkommissionens betaenkning (January 2008)
17. Cresswell, T.: On The Move. Mobility in the Modern Western World. Routledge (2006)
18. Nair, R., Tambe, M., Marsella, S.: The role of emotions in multiagent teamwork. In: Fellous, J.M., Arbib, M. (eds.) Who needs emotions: the brain meets the robot. Oxford University Press, Oxford (2005)
19. Breazeal, C.: Designing Sociable Robots. MIT Press, Cambridge (2002)
20. Xavier, J., Pacheco, M., Castro, D., Ruano, A., Nunes, U.: Fast line, arc/circle and leg detection from laser scan data in a player driver. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005. ICRA 2005, April 18-22, pp. 3930–3935 (2005)
21. Collett, T., MacDonald, B.A., Gerkey, B.P.: Player 2.0: Toward a practical robot programming framework. In: Sammut, C. (ed.) Proceedings of the Australasian Conference on Robotics and Automation (ACRA 2005), Sydney, Australia (December 2005)
22. Nabe, S., Kanda, T., Hiraki, K., Ishiguro, H., Kogure, K., Hagita, N.: Analysis of human behavior to a communication robot in an open field. In: HRI 2006: Proceeding of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction, pp. 234–241. ACM Press, New York (2006)
23. Likhachev, M., Arkin, R.: Spatio-temporal case-based reasoning for behavioral selection. In: Proc. IEEE International Conference on Robotics and Automation, ICRA, vol. 2, pp. 1627–1634 (2001)
24. Kracht, S., Nielsen, C.: Robots in everyday human environments. Master science thesis, Department of Electronic Systems, Automation & Control, Aalborg University (June 2007)
25. Pacchierotti, E., Pacchierotti, E., Christensen, H., Jensfelt, P.: Evaluation of passing distance for social robots. In: Christensen, H. (ed.) Proc. 15th IEEE International Symposium on Robot and Human Interactive Communication ROMAN 2006, pp. 315–320 (2006)

A Mobile Robot for EUROBOT Mars Challenge

Tomáš Krajník, Jan Chudoba, and Ondřej Fišer

The Gerstner Laboratory for Intelligent Decision Making and Control
Department of Cybernetics, Faculty of Electrical Engineering
Czech Technical University in Prague
{tkrajnik,chudoba}@labe.felk.cvut.cz, fisero1@fel.cvut.cz

Abstract. The aim of this paper is to present an intelligent autonomous robot for competition EUROBOT 08. In this "Mission to Mars", two robots attempt to gather, sort and dump objects scattered on a planar rectangular play-field. This paper descripts robot hardware, i.e. electromechanics of drive, chassis and extraction mechanism, and software, i.e. localization, collision avoidance, motion control and planning algorithms. The experience gained by participating on both national and international round is evaluated.

1 Introduction

1.1 EUROBOT

The EUROBOT[1] association holds amateur robotics open contests, organized either in student projects, in independent clubs, or in educational projects. It started in 1998 in France as the national robotics cup. Nowadays, participants are not required to be Europeans, the only restriction is age: only one team member may be over 30 years old. This team member is allowed to advise and lead, but should not participate directly during implementation.

In a typical EUROBOT match, two autonomous mobile robots compete on a planar field with rectangular shape. Robots are limited in size - the maximum height is 0.35 m and their convex hull circumference must not exceed 1.2 m. After start, the robot can deploy its devices and extend its perimeter up to 1.4 m. Match duration is 90 seconds.

Competition rules change every year. This prevents from the situation in other leagues (e.g. FIRA[2]), where new teams have disadvantage compared to veteran participants and are selbom able to reach a good rank. The challenge of this year is called "Mission to Mars".

In this challenge, two opposing robots pick up floorball balls representing biological samples or frozen regolith and put them in three separate containers. Each successfully placed ball adds a certain score depending on its color, robot team color and container type. Special score bonus is added, if balls placed in a standard container form a certain pattern, representing biological samples encircled by ice. Detailed and precise rules description can be found at [3].

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 107–118, 2009.

1.2 Project Goal

The main objective of participation in this competition is educational. The competition proved to be an excellent opportunity for a group of students to design and manufacture a functional device able to solve the assigned task. This is a big difference compared to common educational project, when students solve only particular tasks and the "right" solution is often known in advance. Moreover, students have to solve problems related to different areas of engineering: mechanics, electronics, computer science and artificial intelligence. The experience gained by working in a team under a time pressure is also invaluable.

1.3 Paper Structure

The paper is organized as follows: It begins with introduction. Next division is concerned with an overview of robot concept and desired activity. This will be succeeded by sections describing robot motion, extraction, localization, collision avoidance and planning subsystems. In the next section, we will evaluate the progress of the robot on both national and international round, compare its behaviour to other concepts. Subsequent chapter will conclude about robot ability to fulfill the desired task. Conclusion will be followed by acknowledgments and references.

2 Robot Overview

Our robot hardware consists of an alluminium chassis, two-wheel drive, ball extractor, power subsystem, two control units (drive and extractor), onboard PC, optical sensors, mechanical bumpers and laser rangefinder. We decided to use Ångström Linux operating system to be installed on the onboard computer. The software can be divided into localization, planning, motion control, collision avoidance and waste handling modules. Onboard PC software is programmed in C/C++, algorithms of the extractor and motion control boards have been realized in C.

Skeleton of the robot is formed by interlocked X-shaped alluminium beams. These proved to be firm and stable enough to support robot devices and protect them reliably during collisions. The remaining parts of the chassis is made of alluminium or cuprexit plates, which support sensors, electronics and extraction mechanism.

Drive subsystem has its own control unit capable of positioning both wheels independently. Motion control is based on a set of predefined standard movements, (e.g. from the start position to the vertical dispenser), reactive movement routines (e.g. moving along the standard container, docking) and on a position regulator. Three bumpers (two in the front and one in the back) serve as primary collision detection sensors. Aside from signals from these bumpers, motion control board monitors start, emergency stop and other buttons.

Localization module is based on odometry and URG-4LX laser rangefinder [4] located at the front of the robot. Two cooled containers at field corners serve as

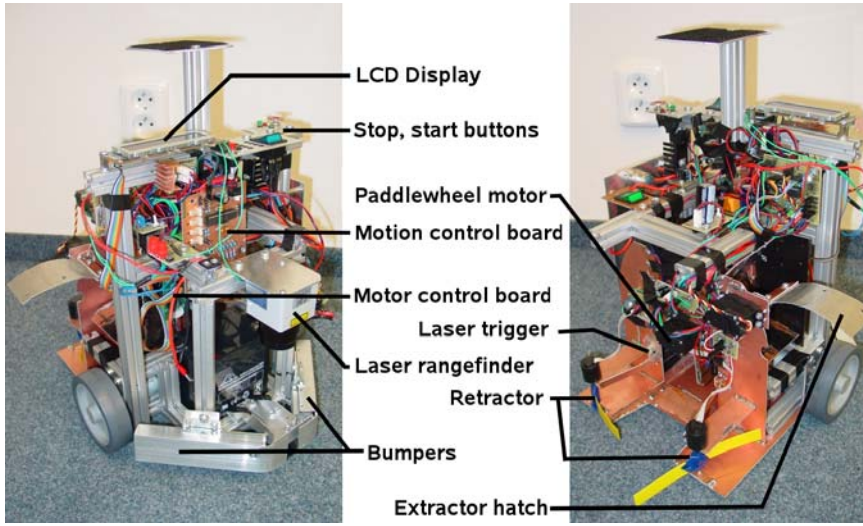


Fig. 1. Robot overview

easily detectable landmarks. Opponent robot position is calculated as well and taken into account in motion planning.

Extraction mechanism is a five vane paddlewheel with its own controller board, ball acquisition mechanism and hatch.

Planning subsystem implements a predefined Petri net [5], where places represent functions of individual software modules and transitions describe return values of these functions. In our approach, a place in a Petri net represents some functionality of robot subsystems (e.g. extractor: eject sample) mentioned in previous sections.

Collision avoidance uses A^*[10] algorithm operating on a grid map in combination with reactive avoidance.

At the beginning of the match, the robot attempts to pick up two colored balls from the vertical dispenser. During the extraction, its laser rangefinder provides data on opponent movement and reports it to the planning module. Planning module then decides, whether to pick up three white balls from ground plane or from the second vertical dispenser. After picking up three white balls, the robot manipulator is full and the robot heads towards the standard container while avoiding the opponent. During this movement, the robot calculates a sequence of manipulator actions in order to maximize the score while minimizing dumping time. Preparatory actions, i.e. preparing the first dumped sample at the hatch, are taken during docking. During the dumping procedure, laser rangefinder data are processed in order to detect adversary position. Opponent movement is analyzed to estimate the next course of action, i.e. picking from ground or extraction from dispensers.

During forward movement, laser rangefinder serves as a collision avoidance sensor. However it cannot provide opponent position during ball pickup, because

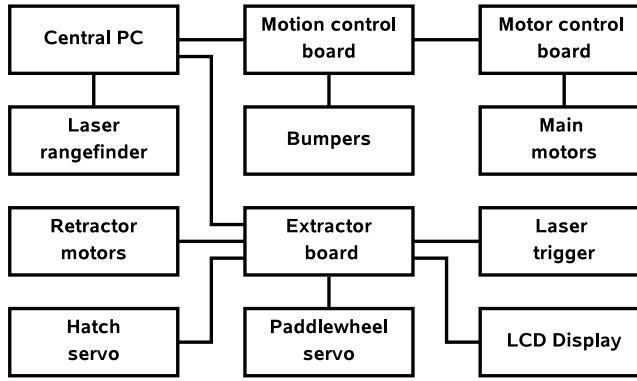


Fig. 2. System scheme

it is placed on the opposite side of the robot than the extraction mechanism. During ball pickup from the ground, the robot will have to rely on its bumpers to detect collisions. If the opponent is detected on collision course (or a collision is detected), our robot will wait if the opponent is going to clear the way. In case the path is not clear after a small moment, planning module will choose an alternative target. If no alternative target would exist, collision avoidance module tries to devise an alternative path to current destination.

3 Motion

The motion subsystem is in charge of moving the robot around the field while preventing damage done to our or opponent robot. It consists of two wheels with DC motors, motor control board, motion control board and mechanical bumpers. We have decided to use EGM30 12V/4.5W motors equipped with encoders and a 30:1 reduction gearboxes. The MD23 dual motor driver is able to control either speed or position of both motors independently. It can also report the current state of motor IRC counters. Acceleration and speed are monitored and limited by this controller to prevent wheel slipping and subsequent odometric imprecision. This motor controller is connected to the motion control board via an I^2C bus. The motion control board handles signals from bumpers and issues commands to the motor controller.

The motion control board is based on an Atmega microcontroller. It receives orders from onboard PC via an RS232 interface and translates them to commands for motor controller. In addition, it processes signals from three bumpers during the match and buttons used during robot starting procedure. These buttons are "system test", "change color", "change strategy" and "start". The "start" button is connected paralelly with a start cable connector.

Motion control unit stores latest information from bumpers and when requested, it transmits these data to the onboard PC by RS232 interface. Signal "bouncing" during the switch transitions is filtered by first order RC filters.

Collision detection, realized by three mechanical bumpers prevents damage by a robot attempting to move in blocked direction. In such situations, rotating wheels erode surface of the field - robot, which damages field gets disqualified. Detected collision also indicates, that odometric information is no longer reliable source of position information. Each frontal bumper is composed of two microswitches covered with a metal plate. Rear bumper is realized by a microswitch and serves as a indicator, that the robot has successfully docked to a dispenser.

Three basic operating modes are realized by the motion subsystem. The first one is a set of predefined sequences of commands for the drive controller. These are simple, short moves, during which we do not expect the opponent robot to interfere. A typical example is the first move of the match, i.e. moving from start area to first vertical dispenser. Second operating mode consists of a set of reactive behaviours. These translate sensory (bumper) inputs to simple commands. The third mode realizes "follow the carrot"[6] regulator. In this mode, onboard PC sends robot-relative coordinates of a target position and the robot moves towards it.

4 Extractor

Extractor is a part of robot which is responsible for sample pickup, transport and dump. It occupies almost half of the robot body. It is composed of a five vane paddlewheel, hatch, retractor, two laser triggers, control board and LCD display.

The most important part is the paddlewheel propelled by a DC motor with a IRC sensor providing 90 impulses per revolution. Paddle profile is optimized for efficient and reliable sample storage and dump. The wheel has five slots, each can store one ball. An infra red sensor provides contact-less detection, that the paddlewheel slots are in right position. The retractor is made of two floppy rotating lamellas. Its task is to push balls inside the paddlewheel lowest slot. The first laser trigger, placed behind the retractor, consists of a 5mW laser diode aimed at a photosensitive transistor. This light gate indicates that the sample was pushed inside by the retractor. Second laser trigger is placed inside the paddlewheel and indicates, that a ball was successfully extracted. One slot of the paddlewheel is fitted with a controllable hatch for sample ejection.

Extractor control board is based on an ATmega microcontroller. This microcontroller generates the PWM signal for the paddlewheel, retractor and hatch motors, controls laser trigger and handles LCD display. The controller is interconnected with the onboard PC via an RS232 interface. Apart from printing its own status on the LCD display, it acts as a bridge between the display and onboard PC. Microcontroller is using internal interrupts and timers for better communication and event handling.

The extractor is designed as a fairly independent function module capable of its own actions. Its main task is sample extraction from the field or vertical dispensers. During the sample pickup, retractor lamellas rotate ant push balls in front of the robot inside the paddlewheel slot. Once the ball acquisition is

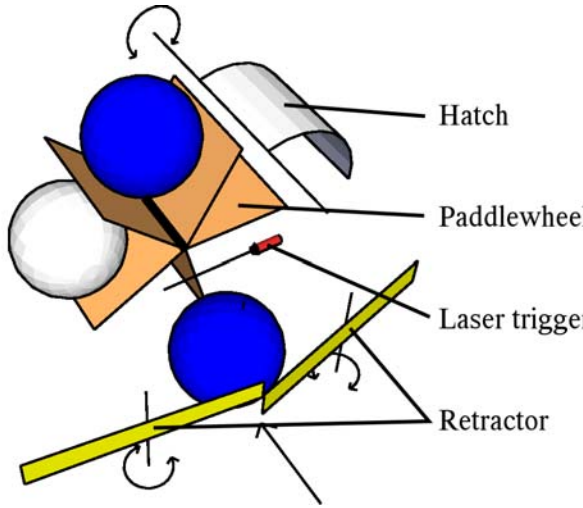


Fig. 3. Extractor scheme

detected by the first laser trigger, the paddlewheel rotates to prepare a free slot for next picked sample. The second laser trigger then checks, if the acquired ball is in correct position. Once the sample extraction is terminated by planning module or the extractor is full, retractor motion is stopped.

To initiate sample dump, the computer announces assumed paddlewheel status, i.e. which slots are occupied by which samples. After that, an optimal ball ejection sequence (e.g. so that the colored and white balls alternate) is planned. The sequence is then realized by rotating the paddlewheel and opening/closing the hatch. The last action status, i.e. success or reason of failure, is reported to onboard PC and printed on LCD display.

5 Localization

The task of the localization system is to estimate position of our and opponent robot on the playfield. To achieve this, we fuse data from odometry, bumpers and laser rangefinder.

Pulses of wheel IRC sensors are first counted by motor controller. These values are regularly read by the motion control board, which translates them to cartesian coordinates and heading. Onboard PC can obtain these data from motion control on request.

Odometry is insufficient, because when robot collides or slips, odometry data lose relevance. Furthermore, knowledge of the opponent robot position have proved to be useful - and this cannot be measured by odometry.

Because of that we use an independent localization sensor, which is based on an URG-04LX laser rangefinder mounted in front part of robot. Its scanning plane is parallel with field plane and is located at 0.16 m above the field level.

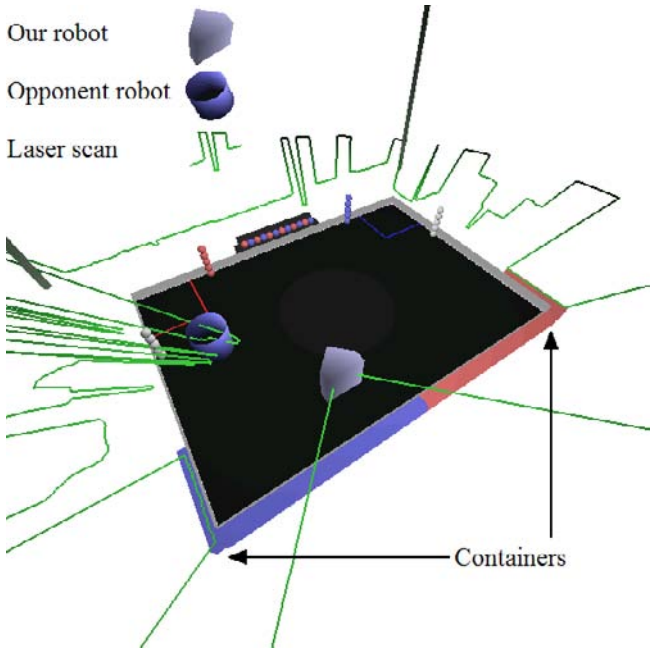


Fig. 4. Scanner data interpretation

Field of view is approximately 250 degrees, because it is slightly limited by the robot itself, sensor range is limited to 4 m, its resolution is up to 1 mm. One complete scan takes approximately 0.1 s and provides the robot with 720 distance measurements.

Localization algorithm works as follows: Laser rangefinder data are requested and motion control board is asked for odometric data. Lines of 0.6 m length, which correspond to cooled containers, are detected in the obtained scan. Cartesian positions of centers of detected lines are then computed.

From known and measured container positions, the pose of our robot is estimated. Most of the time, localization results are ambiguous, a result closest to odometry readings is chosen. This result is reported to motion control board to correct its position information. After computing our robot position, rangefinder measurements are transformed to the field absolute coordinates. Measurement laying inside of the field are segmented and centroid of the largest segment is calculated. This centroid represents opponent robot position.

6 Collision Avoidance

The purpose of collision avoidance module is to adapt robot movement to opponent robot position. In the first and second mode of motion (see section 3) collision avoidance module causes the robot to slow down or pause its movement.

During the "follow the carrot" mode, collision avoidance first checks, whether the desired trajectory intersects opponent position. If trajectory target actually lies inside of opponent position, collision avoidance reports failure and planning module has to devise a solution. If so, an occupancy grid model of the field with is created (cell size is 0.1 m). After that, A^*[10] algorithm is utilized to search the shortest collision free path. The resulting sequence of cells is first converted to a serie of points representing the planned trajectory. The trajectory is then smoothed out and redundant points are erased. The point closest to the robot then serves as an input for "follow the carrot" controller.

7 Planning

Planning is a process of creating a sequence of actions, which we have to apply to a system in order to transfer it from its start state to the goal states. Planning can be realized as search through the space of all possible plans, while looking for the optimal plan according to given optimality criterion. In our case, plan is a series of robot actions with optimality criterion given by game scoring rules. These are based on a number of collected and correctly placed balls during a ninety-second time interval. We do not utilize automatic methods for finding optimal plan, instead of this we try to propose suitable strategy leading to the maximum score. Our team has developed a fixed plan of actions, which should deal with every likely situation (e.g. crash with other robot, incorrectly identified sample). We decided to utilize Petri net [5] model, because it gives clear and effective representation of proposed plan. The Petri net is a bipartite graph consisting of places, transitions and oriented arcs. Every place in a Petri net has a positive integer number of tokens. The projection from a set of places to an integer set is called "marking" and represents state of modeled discreet system. A transition has input and output places. For input place, there exists an arc running from it to the transition and vice versa. If there are enough tokens in every input place of a transition, it can be "fired". During firing, tokens from input places are removed and appear in output places. The number of removed and added tokens is determined by arc weight. The major advantage of Petri nets is the ability to model parallelism and synchronization.

In our approach, a place in a Petri net represents some functionality of the robot subsystems (e.g. extractor: eject sample) mentioned in previous sections. These functions can be thought as simple robot behaviours [8]. If a token is present in a place, corresponding behaviour is activated. There can be more than one behaviour active at the same time, but within one subsystem, only one behaviour is in effect. Possible outcomes of particular behaviour are represented by output transitions (e.g. opponent far, opponent at dispenser). Because of Petri net formalism, we can precisely examine many important properties (deadlock, reversibility, liveness) of our plan. We also measure time needed for each action and use timed Petri nets extension to estimate plan duration. This can give not only quantitative evaluation of our robots ability to win, but also shows functions, which have major impact to plan length. During robot tuning in later stages of development, we can optimize these behaviours to obtain greater speed.

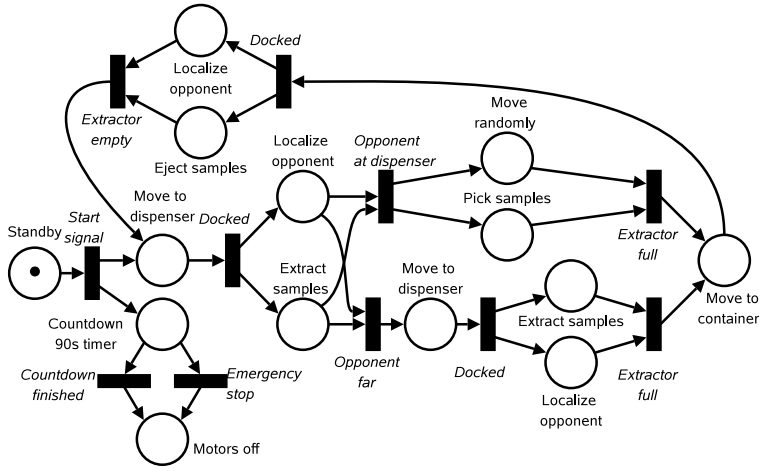


Fig. 5. Intended plan represented by a Petri Net

8 Testing and Competing

The robot was operational approximately seven weeks before the Czech national round. Thanks to other teams, which have build a testing field, we have been able to perform extensive testing. During the test trials, the overall concept has proven to be feasible. However, we have encountered several problems and had to deal with them by changing robot algorithms and hardware. These improvements were completed before the Czech national round and thus we were quite well prepared.

8.1 Testing

Wheel slipping was causing much trouble and we had to lower robot movement acceleration. The slippage problem was supressed, but not eliminated. Laser-based localization had significant problems if there were other planar objects around the field. These objects were often mistaken for containers and the robot lost track of its position. Since the rotation velocity of the URG04LX laser is quite low, laser data obtained during robot movement were deformed. Because of that, the robot had to stop before performing a localization procedure.

When dumping balls to the standard container, the robot position had to be 1-2 cm apart from the field edge. If the robot was closer or farther, balls bounced back to the field. Since the precision of the localization system is lower than 1 cm, the robot had to use its bumpers to detect the container. Balls ejected in the standard container collided with each other and sometimes switched their position. Therefore, we had to slow down the extractor in order to ensure the intended sequence of the balls in the standard container.

The most troublesome operation was extracting balls from vertical dispensers. When the robot did not dock precisely, acquisition of the samples took too much

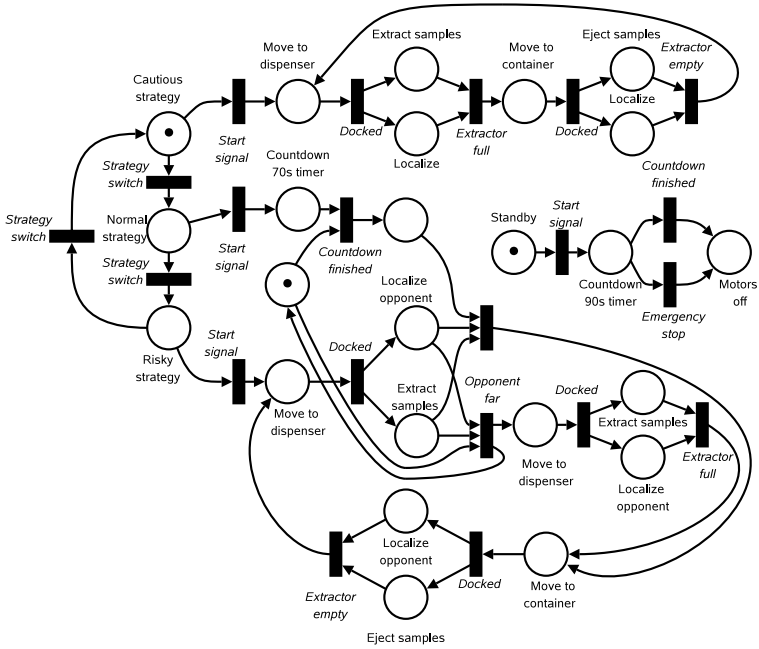


Fig. 6. Plan strategies represented by a Petri Net

time or the extractor failed completely. In order to improve precision, the laser scanner was used to detect dispenser position prior to docking. The rear bumper had to be adjusted to increase docking precision and reliability.

All these modifications improved reliability at the cost of speed. During trials, the duration and success rate of elementary actions were measured. We decided to acquire samples only from dispensers, since picking them up from the field has proven to be ineffective. For docking and movement actions, there was a tradeoff between speed and success rate. Whenever possible, the planning module was allowed to select current action speed and therefore its success rate. Now, the planning module implements three strategies, which are selected by a switch during the startup procedure:

– "Cautious" strategy maximizes the chance to score. All actions are performed slowly, the robot does not attempt to build a sample sequence. The robot extracts five colored balls and dumps them in the standard container. Then, the robot returns to the white sample dispenser, extracts all samples and dumps them as well. This strategy works best against weak teams, which are not likely to score more than 10 points. Maximum achievable score is 15 points. Success rate was approximately 80%.
– "Risky" performs all actions as quickly as possible. The robot attempts to create a full sequence of ten samples. If successful, the score reaches 27 points. Success rate was rather low - less than 20%, but worth of a try against strong teams.

– "Normal" maximizes average score. Action speed selection is based on remaining match time. The robot attempts to form the first sample sequence and then decides, whether to build another sequence or pick up and dump only colored samples. Maximum achievable score is 27 points. However, in 90% of the trials, the remaining time did not allow to form a second sequence. Thus, the most frequent score was 19 points. The approximate success rate is 50%.

8.2 Competition

During the Czech national round, we have tried "Normal" and "Cautious" strategies. The robot scored, but sometimes got lost when moving from the standard container to the dispenser. Later on we noticed, that the field has deteriorated (probably due to high humidity at storage location), which might be the cause of odometric localization error. Most robots suffered from the same problem even more, because their localization was based on odometry only. The robot was able to avoid moving opponents and switch to alternative plan is the opponent obstructed its path. We have maintained a fair score during qualifying rounds and reached the final. In the finals, we won by a fluke and ended up as Czech national champion.

At the international competition at Heidelberg, the robot kept stucking after it docked to the first dispenser. Out of five matches, the robot got stuck three times and did not score at all. However, we won one match with a score of 19 points.

9 Conclusion

Our robot has won the Czech national round and competed successfully at the international round at Heidelberg. However, extensive testing made it too much adjusted to the testing field. Therefore, the robot scored less than expected during competitions. It had problems to extract balls from different dispensers and its odometric localization was not working quite well due to wheel slippage.

Although the aim of this project was to create a robot capable of performing a task given by match rules of EUROBOT 08 competition, its main goal is educational. It should teach students to work as a team, evolve their knowledge in areas which are encompassed in robotics and apply theoretical principles they have learned previously. The robot itself is functional, and efforts, enthusiasm and increasing labor effectivity of team members indicate that educational goal has been achieved as well.

For the EUROBOT 2009, we would like to make modifications to the robot drive and odometric system in order to achieve greater speeds and more reliable odometric localization.

Acknowledgments

Construction of this robot would not be possible without support of Dr. Libor Přeučil, head of IMR, FEE CTU in Prague. We thank EUROBOT organizers,

volunteers and team members for the oportunity to participate in the contest. The presented work has been supported by the Ministry of Education of the Czech Republic under program "National research program II" by the project 2C06005.

References

1. http://www.eurobot.org
2. http://www.fira.org
3. http://www.eurobot.org/eng/rules.php
4. http://www.hokuyo-aut.jp/
5. Petri, C.A.: Kommunikation mit Automaten. Ph. D. Thesis. University of Bonn (1962)
6. Barton, M.J.: Controller Development and Implementation for Path Planning and Following in an Autonomous Urban Vehicle. Undergraduate thesis, University of Sydney (November 2001)
7. Duda, R.O., Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures. Comm. ACM 15(1), 11–15 (1972)
8. Arkin, R.C.: Behavior Based Robotics. MIT Press, Cambridge (1998)
9. Kulich, M.: Lokalizace a tvorba modelu prostředí v inteligentní robotice (in czech). Ph.D. thesis. CTU-FEE, Prague, Dept. of cybernetics (2004)
10. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach, pp. 97–104 (2003) ISBN 0-13-790395-2

The Fly Algorithm for Robot Navigation

Rodrigo Montúfar-Chaveznava[1], Mónica Pérez-Meza[2], and Ivette Caldelas[3]

[1] Engineering Division, ITESM Santa Fe, Av. Carlos Lazo 100, Del. A. Obregón, 01389 México D.F., Mexico
rmontufar@itesm.mx
[2] Universidad de la Sierra Sur, Miahuatlán de Porfirio Díaz, Oaxaca, Mexico
[3] Instituto de Investigaciones Biomédicas, Universidad Nacional Autónoma de México, México, D.F., Mexico

Abstract. The fly algorithm is a strategy employed for 3D reconstruction of scenes, which employs the genetic algorithms and stereovision principles to determine clusters of points corresponding to different objects present in scene. The obtained reconstruction is partial, but enough to recognize obstacles in the robot space work. This 3D reconstruction strategy also allows to know the dimensions of the detected objects. Many parameters are involved in the fly algorithm, and then it is difficult to assign the optimal values for the best performance. In this work we test different parameters values, analyze the results and present some improvements to the algorithm considering the fly algorithm can be employed in robot navigation.

Keywords: The fly algorithm, Evolutive Strategies, Vision stereo, Robotics.

1 Introduction

In particular, in this work we are looking for a strategy to reconstruct the world where an autonomous mobile robot operates using a stereovision system. The reconstruction of the entire world is practically unmanageable, but, a partial reconstruction is possible and it provides the basic idea of the world structure. In this sense, the use of both, stereovision theory and evolutive algorithms, allow us to reconstruct partially the world. Note that some parameters are involved in the performance of the reconstruction, principally those related to the evolutive algorithm. In this work we analyze these parameters to determine the optima, considering that external factors such as illumination can modify the performance of the reconstruction strategy.

2 Reconstruction 3D

Usually, the 3D reconstruction algorithms have been limited to the schemes defined by Marr [1], which are based on the image exploitation and extraction of all information necessary for a scene interpretation. These schemes have three stages: segmentation, reconstruction and recognition. At first stage, the low level primitives are extracted from images. Next, these primitives are merged to produce a 2D reconstruction. Finally, the object identification is performed by comparison.

Many reconstruction algorithms have emerged from these approximations, and some of them employ stereovision techniques to find correspondences between images and triangulation processes to obtain a 3D structure.

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 119–127, 2009.

Stereoscopy is a very popular technique to carry out 3D reconstruction, based on the process performed by the biological vision. It requires at least a pair of images, captured from different positions in a scene, to obtain the structure of the space or infer the 3D position of spatial points. Stereoscopy is an active area of computer vision research [2, 3] based on the works achieved in the field of photography at the beginning of the last century. Vision stereo problem is easily solved by triangulation if the camera calibration parameters are known. The determination of camera calibration parameters is also another problem in 3D reconstruction that has been exhaustively studied and several solutions have been proposed [4, 5, 6].

3 Evolutionary Algorithms

Evolutionary algorithms model natural processes, such as selection, recombination, mutation, migration, locality and neighborhood. These algorithms work on populations of individuals or solutions, performing a parallel search of the best ones.

3.1 The Fly Algorithm

The use of evolutive algorithms for exploration in a parameters space in image analysis problems has been exhaustively explored in last years [7]. There are some works [8, 9] that extract 2D primitives using evolutive algorithms. The fly algorithm [10] is an approximation that searches in the parameters space for the best 3D model consistent with the stereo images. The scene model is defined as a cluster of 3D points or flies (Fig.1a) and the algorithm explores the 3D space looking for the most representative cluster in scene.

An optimization method is the Parisian evolution [11], which considers the global solution is given by a population and each individual is part of it. A special case of the Parisian evolution is the Fly algorithm where individuals (flies) are defined as 3D points with coordinates (x, y, z). The objective of the algorithm is to guide an important part of the population to suitable areas of the search space, corresponding to the surfaces of visible objects in scene.

The fly algorithm can be considered an image processing technique based on the evolution of a population of flies projected on a pair of stereo images. The evolution is regulated by a function of adjustment designed to make the flies converge on the objects located in the scene. This adjustment function is called the *fitness function*.

The algorithm projects the flies on the stereo images, producing a new pair of coordinates: (x_R, y_R) and (x_L, y_L), corresponding to the position of the flies in the right and left images respectively. Initially, the population of flies is created randomly in the space formed by the intersection of the view of both images. Flies then evolve following the rules of evolutionary algorithms. We identify three components in the algorithm:

The fitness function. Compares the fly projections on the left and right images. If the fly is posed on an object, the projections will have similar pixel neighborhoods on both images, producing a high fitness value. If the fly is "flying", the projections will have different pixel neighborhoods, and the fitness value will be low. The idea is illustrated in Fig. 1b. There, fly_1, which is settled on an object, has a better fitness value than fly_2. The fitness function employed is:

$$F = \frac{\left|\nabla(M_L)\right|.\left|\nabla(M_R)\right|}{\left(\sum_{colors}\sum_{(i,j)\in N}(L(x_L+i,y_L+j)-R(x_R+i,y_R+j))^2\right)} \quad (1)$$

(x_L, y_L) and (x_R, y_R) are the coordinates of the projected fly on the left and right images; $L(x_L + i, y_L + j)$ and $R(x_R + i, y_R + j)$ are the pixel color values; N is the neighborhood population introduced to obtain a more discriminating comparison of the fly projections; $\left|\nabla(M_L)\right|$ and $\left|\nabla(M_R)\right|$ are the norms of the gradients of Sobel on the projections of the fly. That is intended to penalize flies when they are posed on uniform regions.

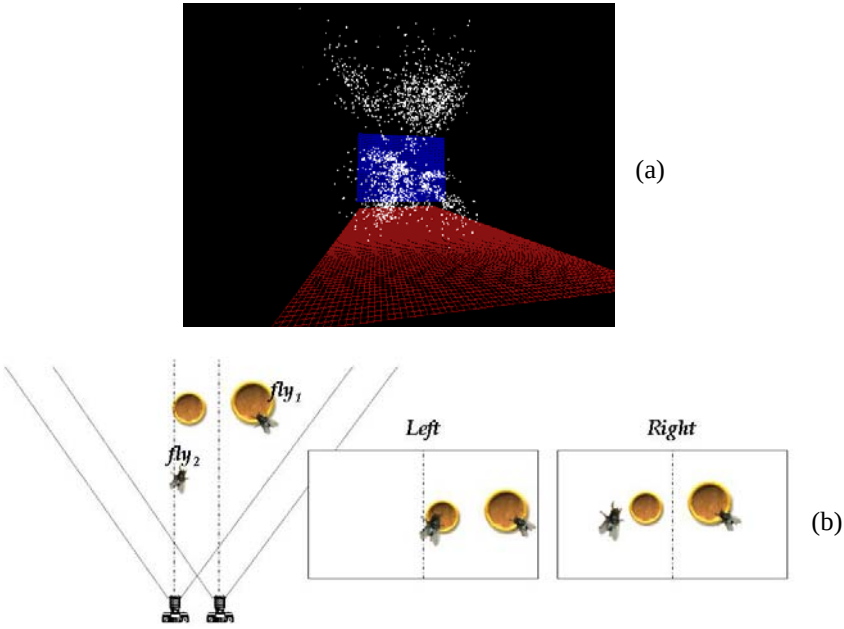


Fig. 1. (a) The space of the fly algorithm. The scene model is represented as cloud of 3D points. (b) The projection of flies: The pixel neighborhoods for fly_1 are very in stereo images; the pixel neighborhoods for fly_2 are different. This observation is employed in the fitness function.

Selection. Classifies the flies according the fitness values, preserving the best individuals. It uses sharing rules that splits the population, forcing a part of it to explore other areas on the search space by replacement. The flies with low fitness values are replaced and new flies are generated using genetic operators.

Genetic operators. We use the following operators to generate new flies:

– Barycentric cross-over. From two parents F_1 and F_2, the heir F is crated such as $\overrightarrow{OF} = \overrightarrow{\lambda OF_1} + (1-\lambda)\overrightarrow{OF_2}$, where λ is a random value between [0, 1].

– Gaussian mutation. The mutation operation adds Gaussian noise to each component of the 3D coordinate in a discriminated fly to produce a new one..
– Immigration. This operation extends the exploration area in search space, creating new individuals randomly; to ensure a constant exploration of the whole space of search.

4 Results

The performance of the fly algorithm depends on the dynamics of the world and the values provided to the involved parameters such as stop criterion, search area definition (image division), percentage of crossing, mutation and immigration, etc.

Table 1. Results obtained using different genetic operators values: Sets A, B and C

Population size	**Stop criterion**	**Set A**			
		No. iterations		*Time (ms)*	
		2 reg.	*4 reg.*	*2 reg.*	*4 reg.*
500	90 %	37	26	1096	787
	95 %	37	30	1096	906
1500	90 %	26	21	1674	1368
	95 %	26	22	1693	1449
3000	90 %	25	20	2896	2294
	95 %	26	22	2996	2490
5000	90 %	27	23	4743	4180
	95 %	27	24	4783	4409
Population size	**Stop criterion**	**Set B**			
		No. iterations		*Time (ms)*	
		2 reg.	*4 reg.*	*2 reg.*	*4 reg.*
500	90 %	27	25	813	768
	95 %	28	28	818	834
1500	90 %	22	19	1427	1259
	95 %	23	22	1455	1412
3000	90 %	22	20	2584	2309
	95 %	23	22	2674	2566
5000	90 %	26	24	4796	4262
	95 %	27	25	4868	4578
Population size	**Stop criterion**	**Set C**			
		No. iterations		*Time (ms)*	
		2 reg.	*4 reg.*	*2 reg.*	*4 reg.*
500	90 %	21	21	653	665
	95 %	23	23	709	696
1500	90 %	19	18	1231	1193
	95 %	20	19	1293	1249
3000	90 %	21	19	2424	2124
	95 %	21	20	2437	2336
5000	90 %	21	21	3793	3818
	95 %	22	22	4048	3827

4.1 Pair of Stereo Images

The pair of stereo images are divided in two and four regions. Initially, the population is equally divided and posed in each region. During execution, the standard deviation of each region is computed and employed to determine the region characteristics.

We test three sets of genetic operator percentages:

Set A: Selection 40%, Crossing 10%, Mutation 40%, Immigration 10%.
Set B: Selection 50%, Crossing 20%, Mutation 20%, Immigration 10%.
Set C: Selection 40%, Crossing 20%, Mutation 20%, Immigration 20%.

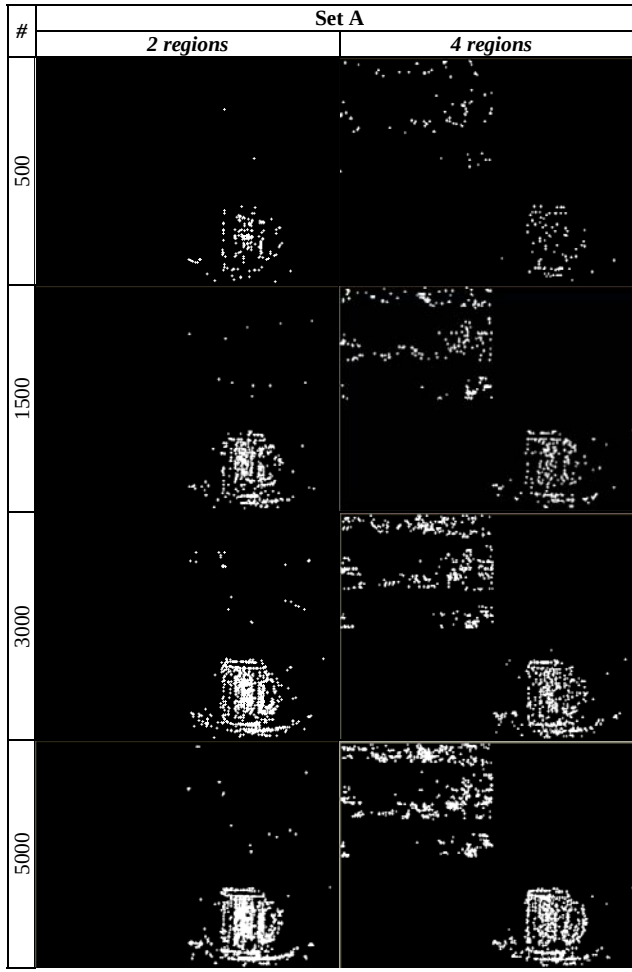


Fig. 2. Results obtained using the genetic operators set A, image division for 2 and 4 regions and population size of 500, 1500, 3000 and 5000

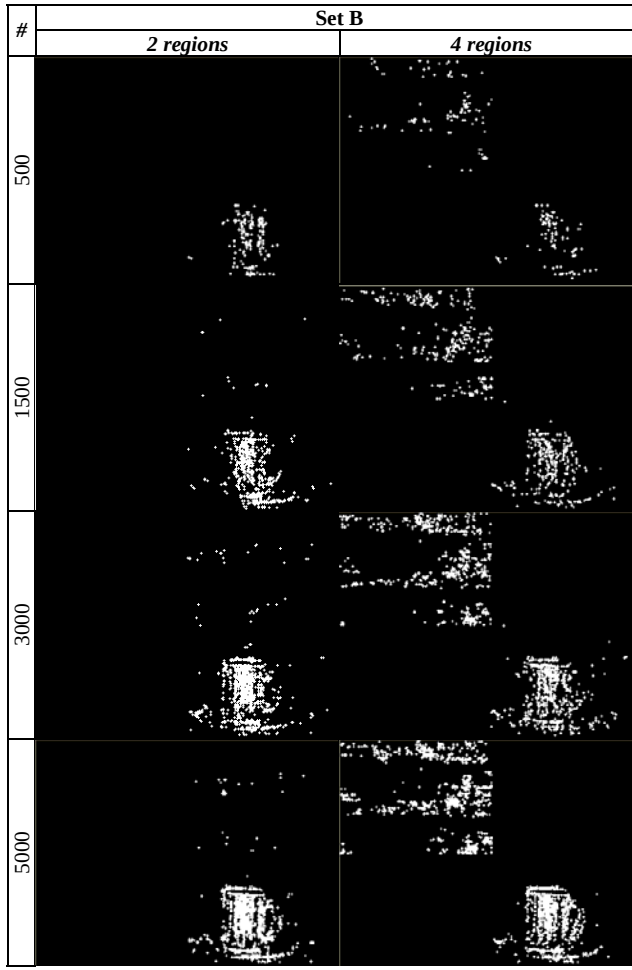


Fig. 3. Results obtained using the genetic operators set B, image division for 2 and 4 regions and population size of 500, 1500, 3000 and 5000

A threshold value was defined as a stop criterion considering the values observed in the fitness function.. We try different population sizes. The results corresponding to Sets A, B and C are presented in Table 1. We compare the number of iterations and processing time to reach the stop criterion. Results are showed in Figs. 2, 3 and 4.

4.2 Image Sequence: Robot Navigation

The system was tested in an educational robot, three paths were defined and the navigation was controlled to capture the sequence of stereo images necessary for the reconstruction. The robot has monocular vision (one camera system), and then it is necessary to provide an adequate control to get accurate camera positions. The genetic

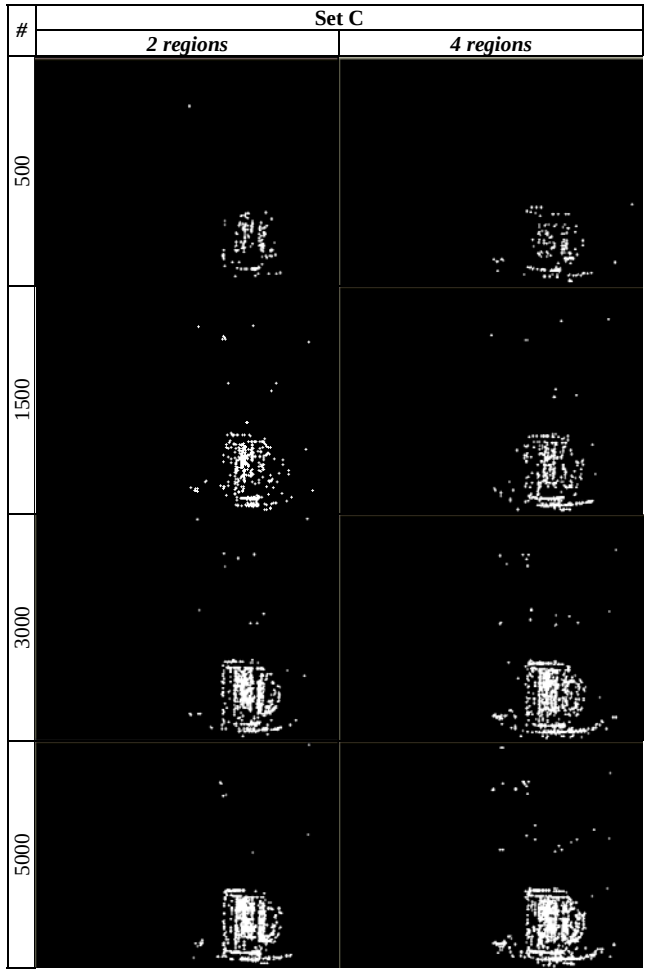


Fig. 4. Results obtained using the genetic operators set C, image division for 2 and 4 regions and population size of 500, 1500, 3000 and 5000

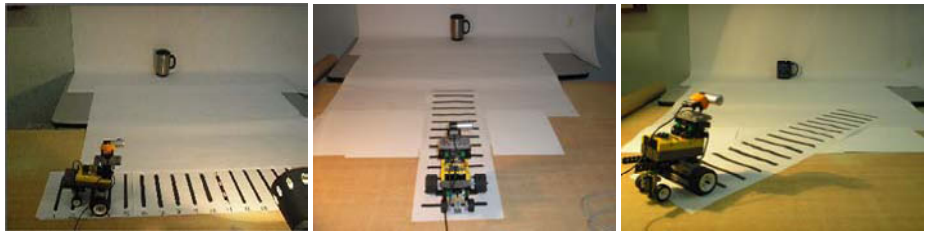


Fig. 5. Navigation paths. At each mark the robot stops and captures an image. Every pair of captured images is employed in the fly algorithm and the reconstruction carried out.

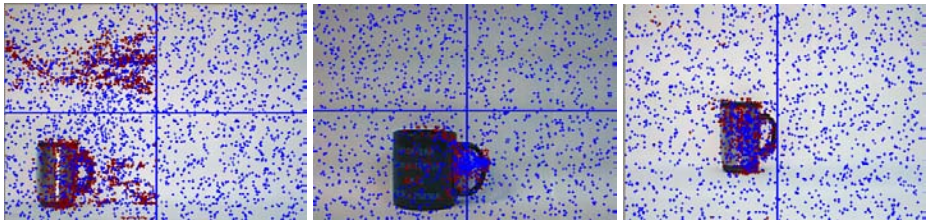


Fig. 6. Results obtained at the end of each path. The flies are posed on the object.

Fig. 7. Visual reconstructed planes *XY*, *XZ* and *YZ* corresponding to path (a). The object detected is well defined. Corresponding 3D reconstruction is showed in Fig. 1.

operators employed correspond to set A in Table 1 using 5000 flies. Fig. 5 shows the navigation paths followed by the robot. Fig. 6 shows the visual results for every path. Fig. 7 shows the different reconstruction planes obtained in the first path.

5 Conclusions

We have presented the results of different tests carried out varying the values of the genetic operators, the population size and the image division. We can say that a population of 3000 flies, the use of the set C and an image division in four regions present the best results.

A reasonable well-defined 3D reconstruction can be performed during robot navigation in harmony with the robot control. The use of additional image operators allows obtaining well defined clusters of flies and a better recognition of objects. The computed world metrics are very exact when the camera calibration is well performed and the robot navigation system provides the correct coordinates. We conclude the fly algorithm for 3D reconstruction in robot navigation is an excellent option, producing almost real-time results even with a monocular vision system.

References

1. Marr, D., Poggio, T.: A computational theory of human stereo vision. Proc. of the Royal Soc. of London. Series B, Biological Sciences 204(1156), 301–328 (1979)
2. Gutierrez, S., Marroquin, J.L.: Disparity estimation and reconstruction in stereo vision. Technical communication No I-03-07/7-04-2003. CC/CIMAT, Mexico (2003)

3. Quam, L., Hannah, M.J.: Stanford automated photogrammetry research. Technical Report AIM-254, Stanford AI Lab (1974)
4. Zhang, Z.: A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(11), 1330–1334 (2000)
5. Abdel-Aziz, Y.I., Karara, H.M.: Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. In: Proc. of the Symp. on Close-Range Photogrammetry, Falls Church, VA, pp. 1–18 (1971)
6. Tsai, R.Y.: A versatile camera calibration technique for 3D machine vision. IEEE Journal of Robotics and Automation RA-3(4), 323–344 (1987)
7. Alander, J.T.: Indexed bibliography of genetic algorithms in optics and image processing. Rep. 94-1-OPTICS, Univ. of Vaasa, Dep. of Inf. Tech. and Prod. Ec. (1995)
8. Lutton, E., Martinez, P.: A genetic algorithm for the detection of d geometric primitives in images. In: Proceedings of the International Conference on Pattern Recognition, ICPR 1994, Los Alamitos, CA, pp. 526–528 (1994)
9. Roth, G., Levine, M.D.: Geometric primitive extraction using a genetic algorithm. IEEE Trans. on Pattern Analysis and Machine Intelligence 16(9), 901–905 (1994)
10. Louchet, J.: Using an individual evolution strategy for stereovision. Genetic Programming and Evolvable Machines 2(2), 101–109 (2001)
11. Collet, P., Lutton, E., Raynal, F., Schoenauer, M.: Individual GP: an alternative viewpoint for the resolution of complex problems. In: Proc. of the Genetic and Evolutionary Computation Conference GECCO 1999, San Francisco, CA, vol. 2, pp. 978–981 (1999)

Scarabaeus: A Walking Robot Applicable to Sample Return Missions

Sebastian Bartsch and Steffen Planthaber

DFKI Robotics Lab Bremen, Germany

Abstract. Recently there was a growing interest in the applicability of walking robots for sample return missions especially in the context of space missions. Samples found in hazardous terrain are of particular scientific interest, especially walking robots have a high degree of mobility in such environments.

In this paper we present the six-legged robot Scarabaeus, which is prepared to demonstrate such a mission using its custom-made claw. We present the robot itself, the method of sample detection as well as the use of piezo-electric elements attached to the claw for the detection of a successful grasp.

1 Introduction

On the basis of their high degree of mobility especially in rough and steep terrain, there is a rising interest in using walking and climbing robots in space missions.

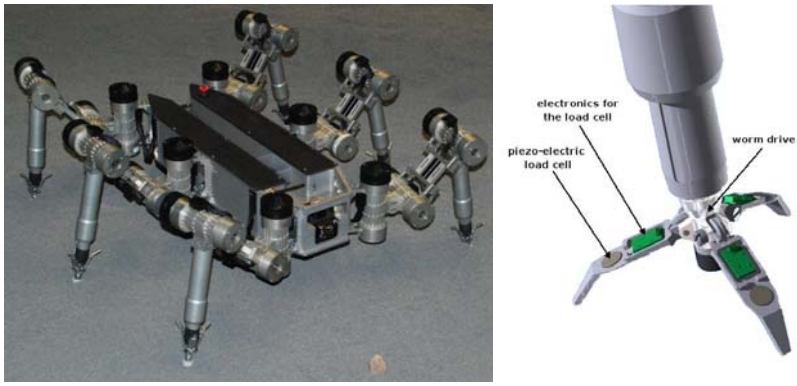
Sample return missions like STARDUST[1] or HAYABUSA[2] are also of scientific interest. Instead of sending humans on such missions, robots (1) can provide a better cost efficiency (2). A closer look at the advantages and disadvantages of legged robots (3) for planetary missions as compared to tracked and wheeled systems (see also (4)) shows that the former kind of robots could concentrate on retrieving a sample from an area which is difficult to access but near the robot's point of departure, where it may have been transported to by a wheeled rover.

When using walking machines for this task, a grabbing or collecting device is crucial. Walking machines offer a high degree of freedom in their legs. So it is self-evident to use a leg as manipulator. To manage space restrictions for sensors to detect contacts at the claw tips, we are using piezo-electric elements.

In order to study the feasibility of walking machines for this task, we constructed and programmed a six-legged robot called Scarabaeus (Fig.1(a)) as successor to the four-legged ARAMIES robot (5) and the eight-legged SCORION (6). All those robots are contolled by the same biologically inspired, behavior-based control concept(7; 8).

[1] http://stardust.jpl.nasa.gov/home/index.html
[2] http://www.isas.jaxa.jp/e/enterp/missions/hayabusa/index.shtml

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 128–133, 2009.



(a) The fully integrated Scarabaeus (b) Detailed description of the claw

Fig. 1.

2 Mechanics

Each of the six legs has four actuated degrees of freedom. The first joint is directly connected to the torso and moves the leg along the body. The second one is used to lift and lower the leg. The third joint is used for the lateral movement of the lower leg. The last actuator is responsible for opening and closing the grabber which is attached to the end of the lateral segment (Fig.1(b)). A spring, which is integrated into the lower leg, is used to absorb shocks while walking and to counteract tensions between the legs.

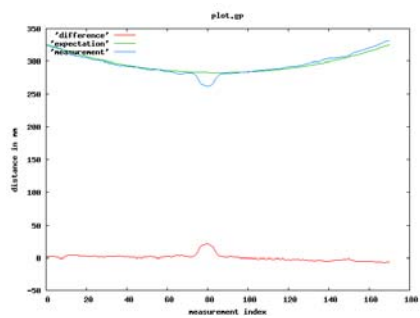
The grabber consists of three claws which are actuated by a worm drive. These elements were developed to perform two functions: It is intended to avoid sinking into dusty surfaces by spreading the claws to enlarge the contact area and it provides the capability to use the legs, which innately have a large operating space, as manipulators.

3 Electronics

For the high-level computations an embedded PC with a 300MHz Geode Processor is used. The embedded PC is connected to a Motorola MPC565 microcontroller via serial connection (RS232) and to the user interface via ethernet (LAN or WLAN). The MPC565 is responsible for time-critical low-level locomotion behaviors like forward and backward walking, controlling the posture, and executing reflexes.

The MPC565 is controlling an FPGA which is responsible for the communication with the five Motor controller boards via Dual-Port-RAM. Each Motorboard is able to control six motors and to read in their sensory information. Thus the MPC565 writes the desired joint angles derived from the trajectory curves to

(a) Scarabaeus Head



(b) Stone Detection Plot

Fig. 2.

the FPGA, and also reads the sensory information when it accesses the Dual-Port-RAM. The sensory information provided by the joints consists of position, temperature, and its required current in milliampere. A linear potentiometer is included to the spring-damped distal segment to measure its compression which indicates the bearing pressure and is used to sense ground contact. To detect whether a claw has contact with an object or not, each claw finger is equipped with a piezo-electric load cell which provides information about the gradient of the force applied to the material.

To select and track the object to be collected, the head (Fig.2(a)) consists of a visual module containing the Hokuyo URG-04LX Scanning Laser Rangefinder and a standard USB camera. It is actuated via a standard servo in order to rotate it around the pitch axis.

4 Control Software

The distinction of the low-level locomotion control and "highlevel" behaviors between two processors using a dezentralized behavior-based control approach is highly beneficial. The locomotion running on the MPC565 is independent of the processor load of the embedded PC which is responsible for i.e. obstacle detection, navigation and planning. Therefore the stability of locomotion is assured even in times of very high cpu load on the embedded PC system.

In our former projects, Bézier curves were used to specify the joint angles in a chronological sequence to generate rhythmic motion patterns like walking forward, backward, left, and right. Thus the Bézier curves are similar to the output of Central Pattern Generators (9). For the Scarabaeus we added an additional layer to the low level framework (10), which solves the inverse kinematic for each leg. Now the Bézier curves describe the endpoint of the feet in Cartesian coordinates, which still results in a rythmic movement pattern. By using these

coordinates it was made easier to create new movement patterns because the developer could directly describe at which x-, y- and z-coordinate the feet should be at a certain point of time in the sequence. The ability to overlay different motion patterns such as forward and lateral walking to produce a diagonal walking persists.

Furthermore, it is possible to use this layer for the stone-collecting behavior to move the manipulator to the Cartesian object coordinates sent by the object detection.

5 Visual Object Position Detection

In order to detect a suitable stone for grabbing, an expectation value for the measurement of the laserscanner is computed (green line without amplitude in Fig. 2(b)[3]). A fitting difference (red line with positive amplitude) between the actual measurement (blue line with negative amplitude) and the expectation indicates a suitable stone to be collected as a sample.

Only when a suitable object is found, Cartesian coordinates are computed and sent to a collecting behavior within the low level framework running on the MPC565, which then takes care of grabbing the stone using the claws and placing it into a storage container on its back.

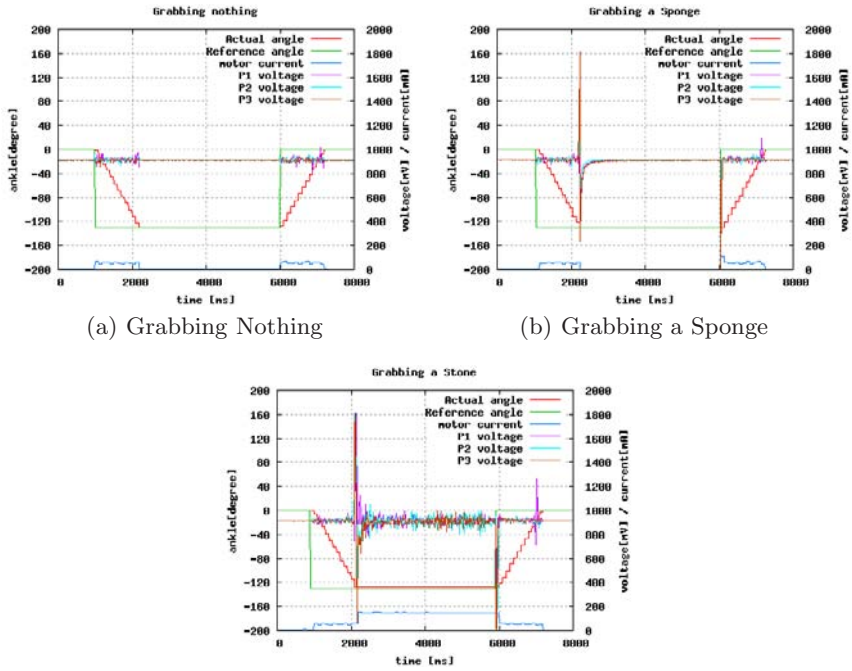
As opposed to other object recognition methods for detecting grabbing objects (i.e. (11)), our method has very low computational costs.

6 Tactile Object Contact Detection

To collect a sample it is necessary to detect whether an object was grabbed or not. The usage of piezo-electric elements for manipulation is an approach to provide tactile information without the need to use very specialized sensors as decribed in (12).

To determine whether the sensory data provided by the grabber are useful to detect if an object was grabbed or not, several experiments were performed. In the first one, the grabber was closed without grabbing anything in order to get information about how strong the deflection of the piezo-electric load cells is as a result of the vibration of the claw motions. It showed that there were negligible voltage amplitudes of the piezo elements (Fig.3). The other experiments were conducted to analyze whether it is possible to detect the grabbing of hard and soft materials and whether it is possible to distinguish between them. One of them was using a sponge, the piezo-electric elements showed a high amplitude when the object was contacted and released. In the other experiment using a stone, the power consumption of the motor increased because it was not able to reach the desired position due to the fact that the stone was too hard.

[3] The plot was taken when a one cm high object was in the scanline with an angle of about 45^o.



(a) Grabbing Nothing

(b) Grabbing a Sponge

(c) Grabbing a Stone

Fig. 3. Test results of grabbing detection. P1, P2, and P3 are the sensor values of the piezo electric load cells of the three claws.

7 Conclusion and Outlook

In our work we proved the usability of walking robots for sample return missions. We showed that it is possible to use the komplex kinematic of a leg which was extended with an additional grabbing device to take up a sample and place it into a container on the back of the robot. Thus there is no need to add a complete manipulating device to the system to perform such a task. In addition we showed that the task of stone collecting can be accomplished by methods with very low computational costs.

The piezo-electric load cells which are integrated in the claws can be used to detect whether an object was grabbed or not. The data on the power consumption of the motor can be used to get an idea about the stiffness of the material.

The use of piezo-electric load cells is a satisfactory way when the available space for sensors is constrained. Their use to control the motor current to allow tactile manipulation of objects is only one possible application. Also ground classification should be possible, if their amplitude is evaluated while the claw is spread and sinking into the subsoil.

Bibliography

[1] Schenker, P.S., Huntsberger, T.L., Pirjanian, P., Baumgartner, E.T., Tunstel, E.: Planetary rover developments supporting mars exploration, sample return and future human-robotic colonization. Autonomous Robots 14(2-3), 103–126 (2003), http://www.springerlink.com/content/g55m107335658367/fulltext.pdf

[2] Coates, A.J.: Limited by cost: The case against humans in the scientific exploration of space. Earth, Moon, and Planets 87(3), 213–219 (1999), http://www.springerlink.com/content/v624w0uv3p815705/fulltext.pdf

[3] Silva, M.F., Machado, J.T.: A historical perspective of legged robots. Journal of Vibration and Control 13, 1447–1486 (2007), http://jvc.sagepub.com/cgi/reprint/13/9-10/1447

[4] Patel, N., Scott, G.P., Ellery, A.: Application of bekker theory for planetary exploration through wheeled, tracked and legged vehicle locomotion. In: Proc. of Space 2004 Conference, pp. 1–9 (2004), http://aecoms.com/Documents/Space2004%20Paper%20(Final%20Rev'd).pdf

[5] Hilljegerdes, J., Spenneberg, D., Kirchner, F.: The construction of the four–legged prototype robot aramies. In: Proceedings of CLAWAR 2005, London, UK (September 2005)

[6] Spenneberg, D., Kirchner, F.: Scorpion: A biomimetic walking robot. In: Robotik 2002, VDI, Ed. VDI, 2002, vol. 1679, pp. 677–682 (2002)

[7] Spenneberg, D., Strack, A., Zschenker, H., Kirchner, F.: Reactive leg control for four–legged walking based on cpg and reflex models. In: Proceedings of the 9th International Conference on Climbing and Walking Robots (CLAWAR 2006) (2006)

[8] Spenneberg, D., Strack, A., Zschenker, H., Kirchner, F., Hilljegerdes, J., Bosse, S.: Control of a bio–inspired four–legged robot for exploration of uneven terrain. In: Proc. of ASTRA 2006 Workshop, ESA–ESTEC, Noordwijk, NL (2006)

[9] Hooper, S.L.: Central pattern generators. In: Embryonic ELS (2001), http://crab-lab.zool.ohiou.edu/hooper/cpg.pdf

[10] Spenneberg, D., Albrecht, M., Backhaus, T.: M.o.n.s.t.e.r.: A new behavior–based microkernel for mobile robots. In: Proceedings of the ECMR 2005 (2005)

[11] Recatalá, G., Chinellato, E., del Pobil, Á.P., Mezouar, Y., Martinet, P.: Biologically-inspired 3d grasp synthesis based on visual exploration. Autonomous Robots (2008)

[12] Melchiorri, C.: Tactile Sensing for Robotic Manipulation. Lecture Notes in Control and Information Sciences, vol. 270, pp. 75–102. Springer, Heidelberg (2001), http://www.springerlink.com/content/hqfu29l9d4k0g7b1/fulltext.pdf

Cybertech Robotic Competition

Iñaki Navarro and Ramón Galán

Universidad Politécnica de Madrid,
José Gutiérrez Abascal 2, E-28006 Madrid, Spain
{inaki.navarro,ramon.galan}@upm.es

Abstract. *Cybertech* is a robotic competition organized yearly by the Universidad Politécnica de Madrid (UPM) in which undergraduate students design and build mobile robots that can compete in different events. Students from UPM can follow a related course in which they learn how to build a robot. Marks obtained in the course depend partially on the results obtained in the competition. The characteristic event of the competition is the *bullfighting* in which each team must build a bullfighter robot that shows its skills against a bull robot provided by the organization.

1 Introduction

Robotic competitions in which students design, build and test a robot to compete against other machines demonstrating its abilities, are widely spread around the world. They provide many benefits to undergraduate students in terms of both learning and increasing motivation towards engineering, as it is shown in [1,2,3,4,5,6]. Students have to deal with real world problems, building a device that must work fulfilling the specifications given in the rules. They must face the different stages of engineering (requirements analysis, design, verification and redesign) while constraining to budget and time limitations. In addition they have to integrate interdisciplinary skills from mechanics, electronics and computer science, and learn how to work in teams.

There are dozens of different competitions; some of them oriented towards pushing new research frontiers, like *Urban Search and Rescue* competitions [7] and robot soccer competitions [8], while others like *Firefighting Robot* [9], *Robocup Junior* [4], *Micromouse Contest* [10], *Eurobot* [11] and *Hispabot* [12] are more focused on teaching and education. Both types of competitions attract potential students to engineering degrees.

Cybertech is a competition organized yearly by the Universidad Politécnica de Madrid (UPM) open to universities all around the world, where undergraduate students work in teams to design and build a robot that participates in different events. The aim is to attract the interest of the students while they discover the problems arisen from real applications and learn how to work in teams. Students participating in the competition can follow a course where they are taught some of the basics to develop their robots. They are graded partially taking into account the results obtained in the competition.

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 134–144, 2009.

The competition is described in Sec. 2, while the related course is explained in Sec. 3. An international course and competition held in 2008 is described in Sec. 4. Results and conclusions can be found in Sec. 5.

2 The Competition

The competition consists of several independent events, that have evolved over the different editions. *Maze* event, *Line-following*, event, *Solar Cars* event, *Simulated Robots* event and *Bullfighting* event took place during the last 2007 edition.

2.1 Maze Event

Maze event, as well as *Line-following* event, are held in every edition of the competition since they allow novice students to learn the basics of mobile robotics and mechatronics, allowing them to move to more difficult events when they are more experienced. *Maze* event is similar to the *Micromouse Contest* [10], where robots have to get out of a maze in a minimum time. The maze has an entrance where the robot starts and an exit to reach. Each robot has to travel through the maze in the minimum time. The maze is changed in each one of the different rounds. A picture of a robot inside the maze can be seen in Fig. 1.

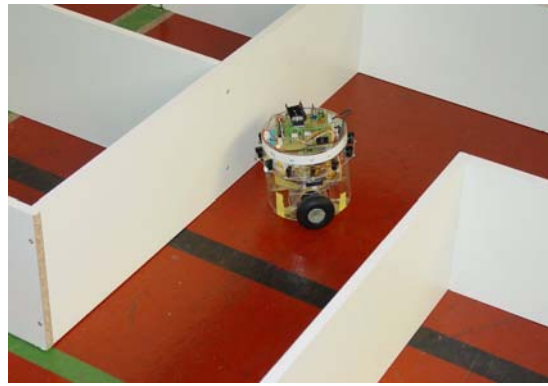


Fig. 1. The maze event

2.2 Line Following Event

In the *Line-following* event robots must follow a black line over a white background. The difficulty is increased with the different rounds by introducing bifurcations in the path and obstacles that must be avoided by losing the line during few centimeters, to get back to it later. In addition, in some rounds two or more robots can follow one line at the same time, being the second robot forced to overtake the first in order to win. Thanks to the increasing difficulty of the event,

Fig. 2. The line following event

experienced and beginner participants can compete at the same time. Placing more difficult tasks in the last rounds makes best robots get better rankings. Overtakings and obstacles also make the event appealing for the public since there are more interaction between the robots. In Fig. 2 a picture of one of the line-following races with an obstacle in the field can be seen.

2.3 Solar Cars Event

Solar Cars event expects students to be concerned about new sources of energy at the same time they learn about electronics. They have to build an autonomous device that should be able to move inside a circuit being propelled just by solar energy, without possibility of accumulating it. The solar panel is provided by the organization and the students have to build the mechanics and the electronics to convert the energy. The solar car circuit can be seen in Fig. 3(a), while a solar robots is shown in Fig 3(b).

(a) The solar car circuit.

(b) A solar car.

Fig. 3. The solar cars event

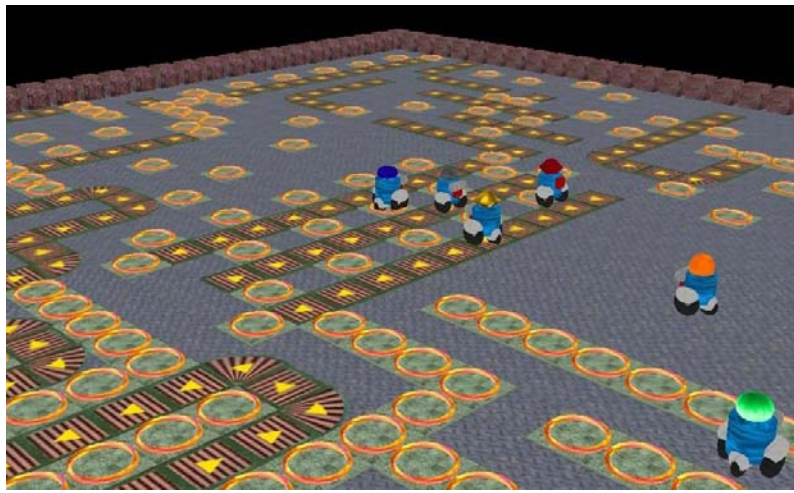


Fig. 4. The simulated robots event

2.4 Simulated Robots Event

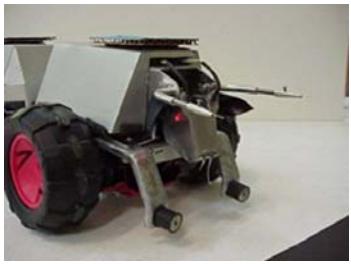
In the *Simulated Robots* event participants have to develop a computer program to control a virtual robot that moves in a simulated maze. Robots must interact between them avoiding to fall into traps; their goal is to find the way out of the maze. This event is not a multidisciplinary one in the sense that students just need to program, but on the other hand, it allows to develop more complicate algorithms. A screenshot of the simulated environment and few robots competing is shown in Fig. 4.

2.5 Bullfighting Event

The main event that makes the competition unique is the *Bullfighting*. Each team has to build a bullfighter robot that fights in the arena against a bull robot provided by the organization. The bullfighter robot must show its bullfighting skills surviving to the bull robot attacks.

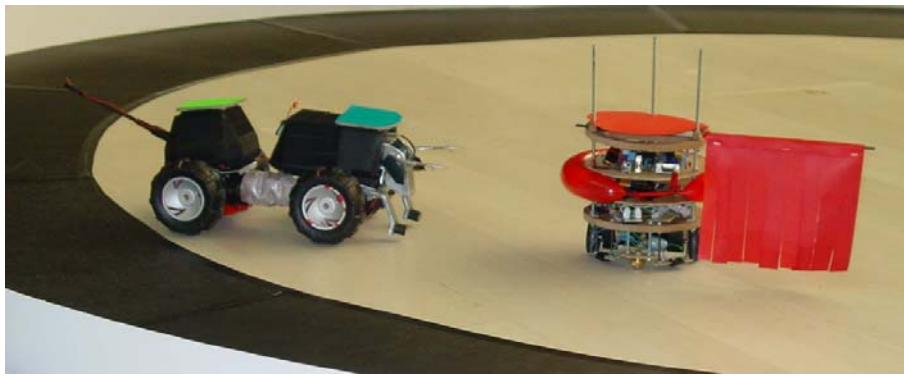
Each bullfighter robot must wear a red balloon attached around its body at a certain known height. The bull robot has sharp horns in the front part used to prick the bullfighter balloon. The most basic behavior of the bull robot is to pursue and attack red things in motion. The bullfighter robot is considered to be dead when the balloon is pricked, so its main mission is to keep it safe. It must be autonomous, all sensors and processing must be inside the robot, except the information about the positions of the robots, provided by the organization and received by a radio link. Each robot can carry a red cape outside its body in order to cheat the bull robot and make it follow the cape, avoiding to be injured.

The arena where the robots fight consists of a circle of 4 m diameter surrounded by a wall of 20 cm height. The surface is divided in an inner white



(a) The bull robot.

(b) The bullfighting arena.



(c) The bull against a bullfighter.

Fig. 5. The bullfighting event

circle of 3 m diameter and a black external ring of 50 cm wide. The bullfighter robot should carry out its movements on the white circle, remaining unevaluated any action carried out over the black zone. In Fig. 5, a picture of the arena with the robots, a picture of the bull robot and a picture of a bullfight are shown.

An overhead camera is placed over the arena looking at it and connected to a computer that tracks the positions of both robots. This information is used by the computer to control the actions of the bull robot, taking into account the positions of both robots and the elapsed time. The longer time the bullfighter robot survives on the arena, the more aggressive the bull robot becomes. The computer sends the motor commands to the bull robot using a radio link. In addition, the information about the positions of both robots is sent to the bullfighter robot that can make use of it. More technical details about the implementation of this system and about the behavior of the bull robot are explained in [13].

Bullfighter robots face the bull robot one by one, getting a score that depends on the time they stay alive on the arena, this is the time they keep the red balloon safe, and on the difficulty of the movements around the bull robot they make. This last part of the score is subjective and so, given by a panel composed of three professors. Their decision depends on how many times the bullfighting robot passes in front of the bull robot, and how close it does, considering also

the reactions of the audience as it is done in a real bullfight. A better score is obtained by doing a good use of the cape, making the bull robot to pass under it. No score will be obtained if the bullfighter robot escapes all the time from the bull robot and does not approach it.

Each one of the robots shows its abilities in four different rounds of maximum 3 minutes. The winner of the competition is the one that obtains more points adding the scores of the best 3 rounds. In each round the bullfighter robot gets 1 point for every 10 seconds that it remains on the arena without being injured, and 10 extra points if it is able to stay without being injured the whole 3 minutes. The panel gives between 0 and 20 additional points per round to the bullfighter according to the bullfighting abilities shown.

2.6 Bullfighting Event Rules

The rules of the *Bullfighting* event are presented in the following paragraphs.

Article 1. Aim: The aim of the contest is to show the skill of an autonomous robot behaving as a bullfighter robot. Each team must design, build and program the robot.

Article 2. Teams: Teams must be formed by three or four students.

Article 3. Rules: These rules are fundamental and must be obeyed.

Article 4. Change in Rules: Any changes of these rules can be made by the Organization. Any modification will be communicated to the teams.

Article 5. The Judges: The judges will be in charge of taking any decision during the competition related to disqualifications, winners, etc.

Article 6. Robot Size: The maximum width and length of the robot are 30 by 30 cm. In addition its height can not exceed 40 cm.

Article 7. Robot Weight: The maximum allowed weight of the robot is 4 Kg.

Article 8. Autonomy: The robot must be fully autonomous. Sensors, actuators, energy and processing must be incorporated inside the robot. Just information provided by the overhead camera of the arena will be provided from outside.

Article 9. The Red Balloon: The robot will carry a red balloon around its body, which will be pricked by the horns of the bull robot. The bullfighter robot must have a specific area, where the red balloon will be placed, with the following characteristics:

- It will cover the whole perimeter of the bullfighter robot.
- It will be a strip of 8cm placed at a height 12 cm.
- It will not be allowed to carry any protection that will make more difficult to prick the balloon.

Article 10. The Red Cape: Every robot might carry a red *"capote"* (cape) whose aim is to make the bull go against it and not against the robot.

Article 11. The Horns: When behaving as a bull, the robot will carry horns to prick the opponent's red balloon.

Article 12. The Arena: The arena will consist of a circular area of 3 m of diameter. This area will be formed by an inner white circle of 2 m diameter, with a black ring 50 cm wide around it.

Article 13. Wall: There will be a white wall 25 cm high around the arena that will help the robots to locate themselves and avoid getting out of the arena.

Article 14. Illumination: The arena will be illuminated with artificial light, which will be as homogeneous as possible.

Article 15. Aim: The aim of the bullfighter robot is to stay as much time as possible on the arena without being injured by the bull robot. The bullfighter robot will be considered *dead* when the red balloon attached to it is pricked.

In addition, the bullfighter robot should show its abilities and skills on the arena, using its cape and making the bull robot turn around it while not being injured.

The aim of the bull robot is to find the bullfighting robot and prick its balloon.

Article 16. Rounds: Each match will consist of four different rounds of maximum 3 minutes in which two teams fight against each other. The winner of the competition will be the one that will get more points adding the scores of the best 3 rounds.

If two teams draw, the fourth round will be taken into account. If there is still a draw a fifth round will be done.

Article 17. Scores: Both the time that the bull fighter robot is able to stay on the arena without its balloon being pricked and its bullfighting skills, will be taken into account to calculate the score.

The bullfighter robot will get points as follows:

- The bullfighter robot will get 1 point for every 10 seconds that it remains on the arena without being injured (the red balloon is not pricked).
- Additionally, if the robot is able to stay without being injured the whole 3 minutes, it will get 10 additional points.
- According to the ability and skills bullfighting, the judge will give the bullfighter between 0 and 20 additional points per round.

3 The Related Course

An undergraduate course related to the competition has been offered to UPM students since 2005. The main objective of the course is that students learn how

to design and build an autonomous robot following a set of specifications. In fact students have to present a working robot at least at one of the events of the competition.

The lessons of the course consist of five thematic workshops where students learn the basics of mechatronics and mobile robotics. The first workshop is dedicated to soldering the main control board, and understanding its different hardware subsystems. During the second one they learn how the microcontroller works and how to program it. In the third workshop, the basics of motor control, and mobile robot kinematics are explained. The different type of sensors and their processing are taught during the fourth workshop. The architectures for the control of mobile robots are explained in the last session. Two pictures of the lessons can be seen in Fig. 6.

These lessons are taught by professors but also by undergraduate and Ph.D. students that have experience in mechatronics and *Cybertech* competition. The previous experience of these students helps a lot while teaching their university mates, since they know those small but important problems concerning a competition like this one.

The control board used is common to every student, and is provided by the UPM without any cost for the students. It is similar to the electronics described in [14]. In addition students have a small budget, also provided by the UPM to buy the different sensors, and actuators that they might need. This material is shared by each group made up by 4 students that compete together.

Each year the course is complemented with three speeches about a common topic, given by relevant authorities. The topic was field robotics in 2005, artificial intelligence in 2006 and climate change and solar energy in 2007. The reason for 2007 topic was the introduction of the *Solar Cars* event.

The final marks are calculated taking into account the following issues: the quality of the project developed, explained on a short paper; an oral examination

(a) A lesson.

(b) Working with the control board.

Fig. 6. The related course

on the previous day to the competition, where students explain how the robot works and show it; and the final ranking in the competition.

4 The International Course

The main novelty for the 2008 edition was the inclusion of a parallel international course and competition on mobile robotics. The course was self-contained and students with basic knowledge on computer science, electronics and/or mechanics built and programmed a mobile robot using a commercial kit. The course finished with a small competition between the different teams, consisting of a simplified version of the *Bullfighting* event.

This course was organized in collaboration with BEST, a non-profit organization aiming to internationalize students of technology via complementary education abroad. Forty students from twenty different European countries visited Madrid for one complete week, to follow the course and participate in the competition while learning about the different European cultures.

The forty students were split into ten groups of four students. Each group had different nationalities and background skills. First day, the students were given basic notions and ideas about mobile robotics and mechatronics. In addition, they built the robots using the commercial kits and performed some software tests using basic examples. Second day, the teams prepared their strategies for the competition and designed the modifications in the mechanical structure of the robots. Third and forth days were spent implementing and testing the algorithms for the *Bullfighting* event. Fifth day the competition took place. An image of a bullfight and teh robots used in the international course can be seen in Fig. 7.

The results of the competition were highly satisfactory. The robots of the ten teams worked properly, being able to show their bullfighting skills and scape from the bull robot. It was quite impressive since students only had one week to build and program the robot.

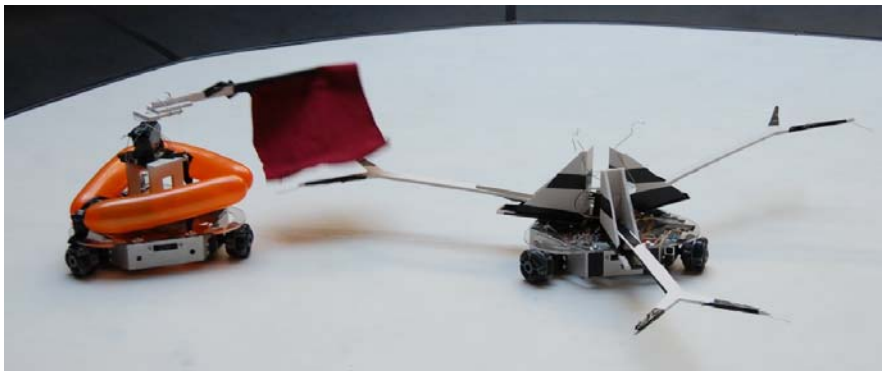


Fig. 7. A bullfight during the competition of the international course

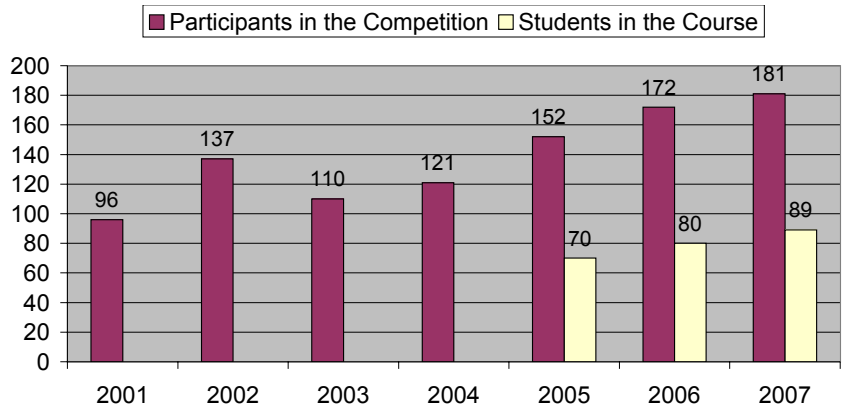


Fig. 8. Participation in the competition and in the course

5 Results and Conclusions

Cybertech competition has been a success since the beginning. It has been held yearly since 2001 with an increasing participation, starting with 96 students in the first year and having 181 in the last edition, that took place in 2007. *Cybertech* course started in 2005, when 70 students followed it, increasing up to 89 during the last 2007 edition. In Fig 8, a bar chart summarizing the participation over the editions is presented.

The quality of the robots presented has increased over the years, and it has been necessary to modify the rules of some of the events to make them more difficult and attractive. This is the case of the *Line-following* event, where obstacles and overtaking were not present in first editions. However, given the structure of the event, inexperienced students can also participate.

The inclusion of the course was one of the reasons of the improvements in the robots, and a big motivation for the students to get involved on robotics and on the competition. In addition, many students that take part in the competition write their master thesis at the Automation Department, where the competition is organized.

Every year high-school students are invited to watch the competition. Thus, students get more interested in technology and get to know the UPM, becoming potential students.

Both Cybertech Competition and Cybertech course have been a succes yearly: from the point of view of the robots presented and the organization. The competition has been always carried out during the days given and following the expected schedule.

Acknowledgements. *Cybertech* has been funded by the Universidad Politécnica de Madrid, the *Escuela Técnica Superior de Ingenieros Industriales* (ETSII) and the *Sociedad de Amigos de la Escuela*, a non-profit society that cooperates with the ETSII. I. Navarro is sponsored by Madrid Region with a Ph.D. grant (FPI-2005).

References

1. Murphy, R.: Competing for a robotics education. Robotics & Automation Magazine 8(2), 44–55 (2001)
2. Chung, C.A., Anneberg, L.: Robotics contests and computer science and engineering education. In: 33rd Annual Frontiers in Education, vol. 2 (2003)
3. Ahlgren, D.J., Verner, I.: An international view of robotics as an educational medium. In: International Conference on Engineering Education, Manchester, U.K (August 2002)
4. Asada, M., D'Andrea, R., Birk, A., Kitano, H., Veloso, M.: Robotics in edutainment. In: IEEE International Conference on Robotics and Automation, vol. 1, pp. 795–800 (2000)
5. Verner, I.M., Ahlgren, D.J.: Robot contest as a laboratory for experiential engineering education. ACM Journal on Educational Resources in Computing 4(2), 1–15 (2004)
6. Pack, D., Avanzato, R., Ahlgren, D., Verner, I.: Fire-fighting mobile robotics and interdisciplinary design-comparative perspectives. IEEE Transactions on Education 47(3), 369–376 (2004)
7. Osuka, K., Murphy, R., Schultz, A.: Usar competitions for physically situated robots. Robotics & Automation Magazine 9(3), 26–33 (2002)
8. Braunl, T.: Research relevance of mobile robot competitions. Robotics & Automation Magazine 6(4), 32–37 (1999)
9. Verner, I.M., Ahlgren, D.J.: Fire-fighting robot contest: Interdisciplinary design curricula in college and high school. Journal of Engineering Education 91, 355–360 (2002)
10. Chen, N.: An updated micromouse competition. In: Proceedings of 26th Annual Conference Frontiers in Education Conference, November 1996, vol. 3, pp. 1057–1059 (1996)
11. Nicosia, V., Spampinato, C., Santoro, C.: Software agents for autonomous robots: the eurobot 2006 experience. In: Proceedings of the 7th WOA 2006 Workshop, From Objects to Agents, Catania, Italy (September 2006)
12. Pastor, J.: Alcabot-hispabot autonomous robot competition in spain. In: Eurobot Workshop on Educational Robotics, Catania, Italy (2006)
13. Lafoz, I., Mora, A., Rodriguez-Losada, D., Hernando, M., Barrientos, A.: Behavior control architecture for a life-like creature: "the robotaurus". In: 11th IEEE International Workshop on Robot and Human Interactive Communication, pp. 542–547 (2002)
14. Gutierrez, A., Navarro, I., Amor, D., Castro, J., Donate, J.: Pi2: A new initiative in educational mobile robotics. In: Eurobot Workshop on Educational Robotics (2006)

Robotics Exhibits for Science Centres. Some Prototypes

Orazio Miglino[1,2], Michela Ponticorvo[1], Angelo Rega[1], and Beniamino Di Martino[3]

[1] Department of Relational Sciences, University of Naples "Federico II", Italy
[2] Institute of Cognition Science and Technologies, National Research Council, Rome, Italy
[3] Department of Information Engineering, Second University of Naples, Aversa (CE), Italy

Abstract. Robots are attractive for public as they represent a clear and ambitious scientific challenge: an artificial being creation similar to natural ones by man. The popular enthusiasm grows up in parallel with huge scientific research progress in this field. On this success wave many Science Centers, Science Museums and Science Festivals propose spaces and moments to spread theoretic, methodological and technical issues of this discipline.

In this paper we describe some of our exhibits prototypes that were designed and carried out to communicate main approaches of modern robotics (telerobotics, cognitive robotics, autonomous and evolutionary robotics and collective robotics). These exhibits have been conceived as laboratories where a visitor can experiment and put the hands on various kinds of robot. They have been accomplished by putting together cheap and easily available materials. This way it has been possible to present these exhibits in many important scientific divulgation events throughout Europe with a relatively modest expense.

1 Introduction

Between all the myths created by men, the automaton is one of the most ancient: since ever men have wondered of a parallel world inhabited by synthetic agents. At the same time this myth is very modern because it is a mirror of the times, as the possibility of its realization reflects the technologic level of an epoch: nowadays, more than ever, there are the right conditions to realize it. To strengthen this claim it is sufficient to remember that it is possible to find robots in factories, in research labs, between the games of children: Puma, a robotic arm, is employed in advanced research; various robots, similar to insects, show an efficient walking behaviour; the SONY robot dog Aibo, can play interacting with the environment. Technology offers the bases which is possible to work on, but gaining the challenge requires efforts coming from a varied heap of disciplines.

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 145–155, 2009.

In fact, the way toward the realization of this goal is still long: the described examples are just a pale imitation of what a self-operating machine should be! At the end automata should be able to support themselves energetically, reproduce, repair themselves, interact with other forms of life and being self-conscious.

Building such a machine requires knowledge from Chemistry and Science of Materials: the matter of which this creatures will be made of must be deeply different from the materials that are used now; from Biology: the reproductive and adaptive capabilities shown by natural organisms can be reproduced only with a complete understanding of biological principles; from Psychology: to have an intelligent machine, it's necessary to comprise what Intelligence is; from Computer Science and Engineering: these disciplines will be fundamental in designing this creatures and translating the indications of other sciences in the final artefact. It is clear, then, that the challenge of building a robotic creature is intrinsically multidisciplinary and needs a huge contribution of scientists and researchers, keeping in mind that the central issue in this ambitious enterprise is that the artificial agent will have to show an adaptive behaviour, a feature that is shared by every form of life that has been able to survive, namely the capacity to find solutions to specific problems (for example finding food, escaping predators, reproducing) autonomously through a dynamic interaction with the environment.

Side by side to this enterprise, it is then necessary a divulgation about the actual state of robotics and the nature of the challenge of creating a synthetic agent. An interesting and, by the way, funny way to do this, is preparing exhibitions through which visitors can entertain while learning about robotics. In the present paper we propose and describe a tour in a City of Robots, a pathways along the steps that have been already made in the effort to create an adaptive artificial agent.

It is necessary to remember that, talking about the characteristics that an artificial organism should have, we have proposed that it should be an intelligent machine. But giving a scientific definition of intelligence is itself an hard challenge. In psychological literature there are as many definitions of intelligence as the number of researchers that have addressed the question. Between these, some definitions, even if not very current, have the quality to be wide enough to include aspects of intelligence that are shared by a great part of living beings and are therefore useful for the definition of an intelligent machine. For Woodrow intelligence (1921)is the capacity to acquire capacity; Dearborn (1921)underlines the capacity to learn or to profit by experience; Pintner (1921) defines intelligence as the ability to adapt oneself adequately to relatively new situations in life and Colvin (1921)draws the attention on the ability to learn or having learnt to adjust oneself to the environment.

According to these suggestions, the notion of intelligence we aim at building along the steps that will be described later, is centred on autonomy, adaptive behaviour and on the fruitful interaction with environment as fundamental features in intelligence.

2 Towards Autonomous and Adaptive Robots

The exhibit "The City of Robots" is articulated into two sections: the first one is descriptive and follows the history of robotics, illustrating the main achieved results and delineating the future steps to do, while the second one is made up of various didactic exhibits that allow to experiment directly "what's up" in robotics. In particular, the attention will be focused on how it is possible to determine the behaviour of a machine as the visitors will be conduced to interact with robots that are more and more autonomous. Participating in it, visitors follow indeed a twofold pathway which we can be describe in two slogans: Using Animats in education and Educating about Animats. In fact, covering the stages that we will describe in detail in the following sections, visitors can learn different things at different level:

1. the biological, sociological, psychological concepts (evolutionary theory, adaptive behaviour, collective dynamics, complex systems) together with technical aspects (for example programming, robot control) that have been used to model robots' behaviour and how it's possible to move to more and more adaptive behaviour, modelling more faithfully what happens in real life.
2. different problems related to robotics (what is a sensor, an actuator, a control system, which are the problems that can be encountered moving form simulation to a real, changing and noisy environment) and how, on a higher level, the behaviour of a machine can be determined: the interaction with machines that are more and more autonomous, displaying increasing adaptive behaviours invites question on, for example, how they react to environment, what is the role of this latter in different approaches to Animats, why different ways of building an Animat produce various behaviours, in which aspects these latter are different.

The different exhibits, that are described in detail in the following sections, are built with various materials. Some of them are materials that already existed and could be purchased in common shops, but have been adapted to the specific purpose of the exhibition, while others have been realized just for it.

2.1 Extending Our Sensory-Motor Apparatus through an Artificial Body (Telerobotics)

The first step of the tour we propose is characterized by a very high degree of technological sophistication but it's quite simple on a theoretical level: visitors will be introduced to Telerobotics (Niemeyer and Slotine, 1990), the area of Robotics that is concerned with the control of robots from a distance. The robot expands perceptual and motor capabilities of man, becoming hands, legs and eyes of the humans where they cannot go. The machine, in fact, can be sent in distant or hostile environment, in the sea depth or on a planet surface, while it is controlled from a distance by a human user. This kind of robot has no intelligence or autonomy, it is just a prolongation of humans, one of the other

Fig. 1. RoboBug: the small mobile robot that can be teleguided

sensors-effectors he/she can use. It was possible to see and appreciate some examples of Telerobotics during last decade explorations of Mars: Pathfinder Lander or Sojourner Rover were guided by NASA's experts on Earth and could collect precious information on the Red Planet. Likewise, in the exhibit we propose, the visitor, with the help of a guide, can control RoboBug (Fig.1), a small mobile robot that explores an environment that is not directly accessible. The user can use a console to control the robot, while another user that controls a mechanic adjustable arm helps him/her in reaching fixed goals. The available information is provided by 3 monitors that show scenes in real time from the environment RoboBug must explore. It is constituted by a rectangular arena with hills, valleys, harshness of land. In particular the images are retaken by 3 camera posed respectively on the robot itself, on the arm and on the ceiling of the room. This arrangement of the cameras supplies different points of view: allocentric and egocentric.

The camera on the ceiling gives us a perspective that is similar to the one we have when playing a remote-control car: we see it from the outside, immersed in the environment where it moves. This view is commonly called bird eye's view (allocentric in a more formal language) and it is quite easy solving an exploration task exploiting this view, because the spatial characteristics of the environment can be seen entirely and the vehicle can be controlled with relatively few effort. On the contrary, guiding a vehicle not directly seen that sends images from its sensory apparatus (on-board camera, infrared sensors), poses in a completely different perspective: subjective or egocentric. This is what happens with the cameras on RoboBug and on the arm.

The difficulty of solving the task becomes then to put together these different frames of reference of the world to have a reliable representation of it that permits to reach the target. Moreover it proposes the problems that can be encountered commonly with Telerobotics, as the burden of analyzing data in real time or the delays in receiving signals. Robobug and its setting has been developed by the Institute of Cognition Science and Technologies of the National Research Council of Rome.

2.2 Putting Our Intelligence in an Artificial Mind (Programming a Robot/Cognitive Robotics)

The first step to provide a robot of a certain degree of autonomy is to program its behaviour, just like normally traditional personal computers are programmed. The programmer defines in detail a particular behaviour and writes the code to implement it using traditional programming languages. In this case the intelligence of the programmer is transferred in the robot that is released in the environment and can act according to the instructions it has received by the programmer. These instructions are usually expression of a "human-like" way of facing a problem, referable to high-level cognitive abilities.

The discipline that is concerned with the design of robots that function in changing, incompletely known and unpredictable environment by using high-level cognitive abilities is Cognitive Robotics. The robots produced in this frame can have knowledge, beliefs, preferences, goals, motivational attitudes, can collect information from the environment, plan and finally execute plans, can reason about goals, perception and actions. To do this the programmer has to describe, in the language of programming, the properties of the robot, its ability, its representation of the environment to endow the robot of a high-level controller.

In this exhibit the visitors will have the possibility to program Lego robots, built with the kit Lego Mindstorms ®, Robotic Invention System. It is composed by Lego Bricks, gears, pulleys, pneumatics, motors, wheels, sensors (light, temperature, rotation) and by the RCX (Robotic Command Explores), a programmable Brick that contains a microprocessor that allows for up to three input ports and controls three output ports. The RCX is also equipped with an infrared port that can be used to download programs from a computer via an infrared tower (attachable to the computer's serial port). The visitor, with the help of a guide, can build the robot, create a programs for his/her robot using Lego's own visual programming environment on a desktop computer, download it on the robot and check if the behaviours he/she has programmed are suitable for a certain goal.

This approach works with relatively simple behaviours, in fact, as soon as the behaviours to implement become more complex, it is impossible to foresee in advance every environmental condition the robot can encounter.

2.3 Training/Breeding an Artificial Organism (Adaptive Robotics)

Living beings are not programmed, they adapt to the environment they live in or, under some circumstances, they can be bred or trained by other living beings. This establishment has given the start to a wide field of research: Adaptive Robotics. The robots are endowed with the capacity of learning from their experience and their learning processes are often supervised or canalized by a human being. The exhibit allows the visitor to train a population of robots, awarding the individuals that prove to be the best in solving a spatial orientation task. In particular users are introduced to a setting consisting of some mobile robots, an arena and an associated software, in a word, they will interact with Breedbot.

Breedbot is an integrated hardware/software system that is suitable for users with no technical or computer experience using which it is possible to breed a

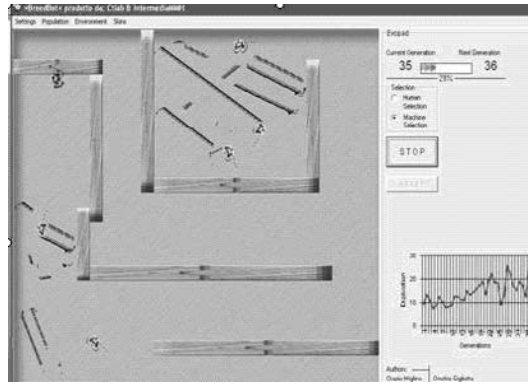


Fig. 2. A snapshot representing the Graphical Interface of BreedBot software. On the left side there are the 9 robots while they explore a rectangular arena with walls. On the right side there is a display that indicates the current generation, STOP and RESET buttons and a graph, updated after each generation, that summarizes the performance in the exploration task along generations.

small population of robots in a software environment that simulates a process of artificial evolution, based on Darwinian selection. The visitors are aided by two guides that introduce to the use of the simulator and to the physical robot. At the beginning of each simulation, the computer screen shows a first generation of robots in action. Figure 2 shows BreedBot's graphical interface. The left hand side of the screen displays the behaviour of the nine simulated robots in the arena, which is surrounded by walls. The right hand side provides information about the state of the system (the number of the current generation and a graph showing changes in the performance of the population) along with a number of commands allowing the user to stop the system and to choose between human and artificial selection. The pull-down menu in the top left corner contains system utilities (to change the geometry of the environment, save configurations etc.). After a certain time, some of the robots are selected to produce offspring. Users can let the system select the "best robots" or make the decision themselves, choosing with the radio button between human and machine selection. If the system makes the decision, it rates the robots by their ability to explore the environment, and selects those with the highest scores. Human users, on the other hand, choose the robots according to the criterion they prefer.

The exhibit merges then two key techniques used in Artificial Life: User Guided Evolutionary Design (UGED) and Evolutionary Robotics (ER). UGED (Bentley, 1999) allows users to "evolve" software and hardware objects, using a computer simulation system. The simulation system proposes a set of variants of the object users are trying to evolve, they choose a subset of these variants and the simulation system produces a new set of variants. The selection/production process goes on until users have obtained the desired objects. This technique is used when visitors operate the artificial selection by themselves.

Fig. 3. The transfer of the control system (an Artificial Neural Network) from the simulator BreedBot to the real Lego MindStorms robot, through the infrared port

On the other side, when the artificial selection is automatic, Breedbot utilizes techniques belonging to Evolutionary Robotics (Nolfi and Floreano, 2000), a field of research that evolves robots with a design process directly inspired by biological evolution. In accordance with what happens in Breedbot, small populations of robots evolve behavioral adaptations to a user-specified environment. Once the selection procedure is over, the system creates clones of the selected robots. During this process it introduces random mutations into their control systems. The robots created in this way constitute a new generation. The control systems (the rules that determine the behaviour of the robot) are different for every member of population and this determines different behaviours. This selection/cloning/mutation cycle can be iterated until the 'breeder' finds a particularly capable robot. At this point the "brain" of the simulated robot can be uploaded to a physical robot, as shown in Figure 3, built using motors, infrared sensors, bricks and an on-board computer from the Lego Mindstorms© kit (Figure 4), and observe its behavior in the real world.

Breedbot is designed to be easy to use for breeders of small robots and to be highly interactive: visitors can use the system's graphical interface, conduct their artificial evolution and if they want, they can select the individuals which will be allowed to reproduce. They can stop the program at any time, choose what they consider to be an interesting robot and use the infrared port to upload its control system to the physical robot. In fact, putting the robot near the infrared tower and pressing "implant" key the simulated robot will be transferred to the brain of the real robot. Moreover Breedbot allows visitors to design the spatial characteristics of the arena and once the robot brains have been downloaded to the physical robots, visitors can interact with the physical hardware.

Breedbot has been produced by the Institute of Cognition Science and Technologies of the National Research Council of Rome to try to merge two main families of robotics products: kit for the robot construction and pre-assembled robots. The first category contains those products with sensors, motors, materials

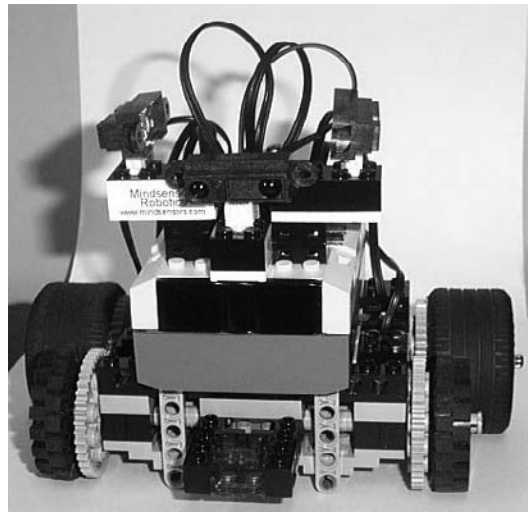


Fig. 4. The physical BreedBot built with Lego MindStorms

that users can use to build the robot body together with the software to determine the control system. This application is useful because the user is active in the designing and building process and, for this reason, is often used as an educational tool. On the other side, the second one proposes complete and ready robots that are able to interact with a human being or an other robot. The robots are conceived as artificial organisms that can capture the interest of the user through the relation and the involvement in the human/(artificial) animal interaction. Breedbot allows to build one's own robot, to make it evolve, "taking care" of the artificial organism, joining the activation of the cognitive processes such as planning and decision with emotional aspects like the anxiety of the wait, the joy for good result, etc. Moreover the user interacts both with the software (the simulation environment) and the hardware (the mobile robots). This approach is undoubtedly fascinating, but it encounters a big limit in the problem that it still doesn't exist a solid scientific theory about learning/adapting processes so the current robots are able to learn on the basis of quite simple processes. Anyway, the robots that are obtained following this approach have a certain degree of autonomy and can potentially adapt to unknown or unforeseen environmental situations.

2.4 Looking after Artificial Organisms to Produce Collective Intelligence (Swarm Robotics)

Living beings live in community and the behaviour of an individual influences other individuals. Often the sum of several simple individual behaviour produces a complex collective behaviour. This is, for example, the case of ants: every single ant has just simple abilities, but, thanks to the interaction with other simple

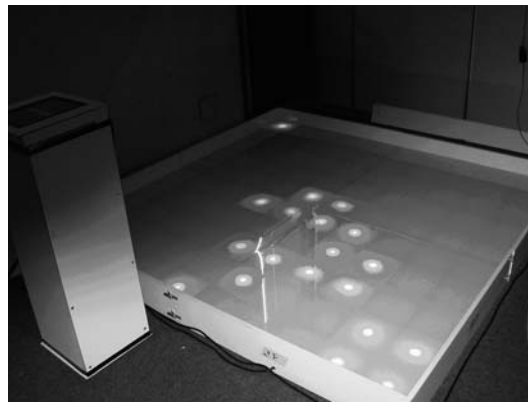


Fig. 5. ARS, a setting for artificial ethology and the console to control it

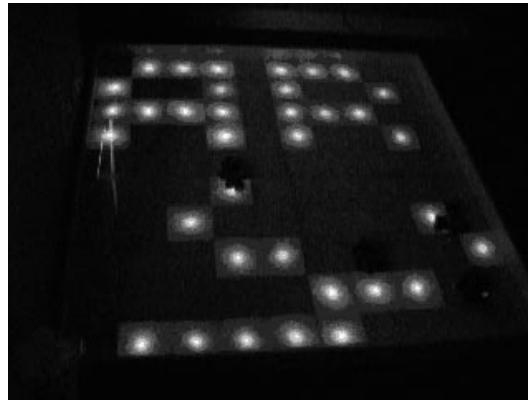


Fig. 6. The setting ARS with lights

agents, can display a complex and emergent behaviour, as cooperating for transporting objects. At this step of the tour the behaviour of robots is determined by taking inspiration from this natural phenomenon. The exhibit proposes, in fact, an experience of collective robotics in which the visitor influence the behaviour of two swarm of robots (Dorigo et al., 2004) manipulating the environment. In more detail visitors will interact with ARS, Autonomous Robotics Setting, a setting for artificial ethology, shown in Fig.5, that has been implemented by Institute of Cognition Science and Technologies of the National Research Council of Rome.

ARS is a platform of 2*2m, divided in 10*10 cells and surrounded by a wall 30 cm high. Every cell contains a lamp that can be turned on and off by an external console (Figure 6). This configuration permits to create quite easily mazes, illuminating the opportune cells, of different complexity. It is enough large for small teams of mobile robots. The visitors, after an introduction by

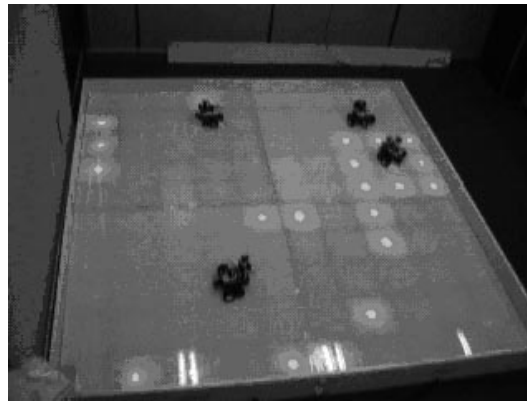


Fig. 7. Two families of robots acting in ARS

a guide, can interact with a team of 4 robots, built with Lego Mindstorms Kit (Figure 7). These robots have 2 infrared sensors (pointed toward the floor to check luminosity of cells) and 3 sensors of distance. Moreover every robot can receive an infrared signal emitted by its "companion". Two robots halt on bright cells and move on the dark ones (group A), while the others (group B) have the opposite behaviour (move on bright cells and halt on the dark ones). When a robot is reached by a signal emitted by its companion produces a sound and changes direction to avoid bumps. The visitor can influence the behaviour of these two families, manipulating the console of the arena and creating different paths. Varying the characteristics of the environment artificial organisms face more and more complex tasks. The prototype is in fact born with the goal to permit experiments of autonomous robotics with increasing behavioural complexity. The human intervention creates at each manipulation different scenarios in which the 2 teams of robots move in dynamical interaction. This decentralized control leads to a continuously changing situation that produces emergent behaviours.

3 Conclusions

The proposed tour illustrates how artificial organisms can become more and more similar to their natural counterpart, showing at each step, an increasing adaptive behaviour. This is achieved thanks to the sequence of exhibits, all provided with physical agent, a dimension that cannot de disregarded in robotics research. Different degrees of human intervention reflect in different definitions and implementations of the control system and consequently in the different kind of link that relates the robot, the human and the environment the robot lives in that, all together, determine the behaviour. The exhibit has been shown in occasion of "Futuro Remoto" at Città della Scienza in Naples for various year. In those circumstances thousands of people, both children and adults, have interacted with the exhibition, showing a deep interest and active participation. In

fact the next steps in this research will be focused, on one hand, on the attempt to measure quantitatively the reactions of the users, to understand what these products make emerge, in the point of view of robot-humans interaction. On the other side we would like to expand these exhibits with the introduction of new kind of robots, possibly more flexible and robust, as the new e-puck produced by the Ecole Polytechnique Fédérale de Lausanne (EPFL)in Switzerland.

Following these future directions will represent, for the ones who will work to realize it, as well as for the ones who will just take part in the exhibit, a further step to realize the ancient dream of living machines.

Acknowledgement

We would like to thank Franco Rubinacci, Iginio Sisto Lancia and Massimiliano Caretti for their precious work in the experiences the present paper refers to, that has allowed many people to get in touch with the City of Robots.

References

1. Bentley, P.: Evolutionary Design by Computers. Morgan Kaufmann, San Francisco (1999)
2. Colvin, S.S.: Intelligence and its measurement. Journal of Educational Psychology 4, 136–139 (1921)
3. Dearborn, W.F.: Intelligence and its measurement. Journal of Educational Psychology 12, 210–212 (1921)
4. Dorigo, M., Birattari, M., Blum, C., Gambardella, L.M., Mondada, F., Stützle, T.: Ant Colony Optimization and Swarm Intelligence. In: Dorigo, M., Birattari, M., Blum, C., Gambardella, L.M., Mondada, F., Stützle, T. (eds.) ANTS 2004. LNCS, vol. 3172. Springer, Heidelberg (2004)
5. Niemeyer, G., Slotine, J.J.: Stable adaptive teleoperation. In: Proc. American Control Conference, vol. 2, pp. 1186–1191 (1990)
6. Nolfi, S., Floreano, D.: Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines. MIT Press/Bradford, Cambridge (2000)
7. Pintner, R.: Intelligence and its measurement. Journal of Educational Psychology 16, 303–317 (1921)
8. Woodrow, H.: Intelligence and its measurement: A symposium (XI). Journal of Educational Psychology 19, 463–470 (1921)

Quantitative Cognitics and Agility Requirements in the Design of Cooperating Robots

Jean-Daniel Dessimoz and Pierre-François Gauthey

HESSO-Western Switzerland University of Applied Sciences,
HEIG-VD, School of Management and Engineering,
CH-1400 Yverdon-les-Bains, Switzerland
{Jean-Daniel.Dessimoz,Pierre-Francois.Gauthey}@heig-vd.ch

Abstract. The paper makes two main theoretical contributions and also presents a cooperating robot for domestic use, as a practical study case, where the concepts agile control can be illustrated. The first contribution relates to quantitative cognitics, an approach which helps in analyzing complex situations and in designing advanced systems involving cognition, such as cooperating robots. Cognitive systems usually feature a hierarchy of subsystems. For humans, as for autonomous, cooperating robots, the functions of perception, decision, or in a more refined fashion yet, vision or trajectory planning, are good examples of (sub-) processes and tasks, which can be addressed per se, in specific subsystems. In particular, vocal communication is discussed below, as instance of a very stochastic process. For such cases, the probability of error is to be taken into account for the relevant assessment of knowledge quantities. The second contribution relates to a critical feature: relative agility of control elements. For complex cases, where many resources are necessary (e.g. groups of robots), a multi-agent structure may be useful, but then interactions occur and this involves loops, where information flows, and consequently stability becomes increasingly of concern. Several elements of solutions are proposed.

Keywords: Cognitics, cooperating robot, agility, multi-agent architecture.

1 Introduction

Automated processes allow for humans, today, to control systems in rather complex and demanding tasks and applications. We shall not refer here to energy nor material components, usually also necessary, but will focus instead on information-related flows and processes.

Nowadays, many data flows are commonly processed without direct human action, i.e. in an automated fashion, involving cognitive operations in complex and demanding applications. Novel formal definitions and units for cognitive properties need be defined (re. e.g. [1,2]). Such flows are far beyond the complexity and abstraction levels of early « messages », which used to consist in a few bits only, directly related to physical signals, and were typical of telephone communications around which the "information theory" has been invented, in the middle of last century [e.g.3]. This

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 156–167, 2009.

new need, which relates to progresses to be achieved in the cognitive world (knowledge management, abstractions, learning, and more broadly speaking, cognitics – this word has been coined to describe the science and techniques of automated cognition) is widely recognized and very acute today, appearing in one way or another in many fields ranging from technical domains such as in manufacturing workshops, to computer-based context and even further to social sciences and humanities areas [e.g. 4-6]. Assistance at home is one field of particular interest for us.

In order to progress in such areas, several contributions are presented in this paper. In general, a quantitative approach is recommended (quantitative cognitics) and in particular the assessment of knowledge for cognitive systems that sometimes make errors is shown below. Another point involves considerations about the agility of controllers, which is shown below to be a critical factor to take into account for the design and stability of multi-agent systems; the latter may consist in very different forms: e. g. a group of robots, a set of internal components of a single robot, or a mixed group of humans and robots.

The importance to go quantitative has been too long underestimated in AI and in control. For ages cognitive properties were not only not measured, but even not properly defined. Consider as an analogy the mechanical task for a person of jumping over a wall. Most of the issue critically depends on a quantitative criterion: the height considered for the wall; 30cm high, the wall is easily walked over but 3m high it becomes a definite obstacle. A decade in variation is enough to make the task either easily done or impossible to achieve. Similarly, as shown below, a decade of variation in relative agility for a feedback controller turns it from very easily feasible (on-off control) to impossible to realize (the system to control must be reengineered or other, complementary controllers must be designed)! Playing chess or tennis, translating from Russian to Chinese, following a person or guiding another one by a robot: in all these cognitive cases success critically depends on specific metric threshold values retained on cognitive entities (in particular, task complexity and required expertise).

The case of designing autonomous, cooperating robots, as for domestic applications in Robocup-at-Home league [7, 8], provides a good example where the above considerations can be performed and appropriate solutions practiced.

The paper is organized as follows. Section 2 presents knowledge quantity estimation when delivered information is not totally error-free , as is typically the case in vocal communication. Section 3 discusses the fact that cognitive processes can be distributed at different levels of granularity, usually including, at a high level, perception, decision and action functions; it also shows where is typically the largest load. Section 4 reminds the reader of the crucial importance of relative agility for control units in any single loop (action and feedback paths), and shows that a multi-agent system architecture may lead to many loops, which may each set specific, different requirements; this provides the basis for Section 5 where it is shown that time and delays must be tracked, or at least represented in a model when tracking is not feasible in order to keep distributed, complex, multi-agent systems stable and effective; alternately, architecture may sometimes need be adapted. Section 6 provides a practical case, a cooperating robot for domestic use, where above theory can be illustrated, practiced and tested.

2 Knowledge Estimation in Presence of Errors

In the MCS model [e.g. 2], the concept of knowledge is usually presented for the basic case, namely for the case of systems that deliver correct information; possibly limited to a small domain, but nevertheless always correct.

Let us remind the reader of the MCS equation for assessing knowledge, K, which is the following:

$$K = \log_2\left(n_o \cdot 2^{n_i} + 1\right) \text{ [lin]} \tag{1}$$

where n_i is the quantity of information entering the system, and n_o is the quantity of information delivered by the system.

Equ. 1, includes, in its core, a quantity M, defined below, which can be viewed as the complexity of the process, or, in principle, as the size of a (virtual) memory containing all the possible messages delivered, for all possible input configuration (this memory is virtual, in the sense that in nearly all cases, it would be totally impossible to realize such a memory; yet this is an interesting equivalent model, to be considered as a reference for quantitative assessment).

$$M = n_o \cdot 2^{n_i} \text{ [bit]} \tag{2}$$

An extension is very useful for assessing cognitive properties in the case where a cognitive system delivers information flows that are not totally error-free[1]. In such a case the system does not perfectly know a given domain D_m.

A particular output message, d_{osj}, does not necessarily correspond to the correct corresponding one, d_{oj}. Equation 1 is still applicable, but the part of out-flowing information that does not correspond to D_m, which could be called "noise" or "error", should be removed from the equation. The quantity of correct information delivered by the system, n_{osc}, must then be estimated in each case, and injected into Equ. 1:

$$M_s = n_{osc} \cdot 2^{n_i} \text{ [bit]} \tag{3}$$

The quantity n_{osc} is defined in the following way:

$$n_{osc} = \sum_{j=1}^{n} p\left(d_{os}j\right) \cdot p\left(d_{os}j = d_{o}j\right) \cdot \log_2((p\left(d_{os}j\right))^{-1}) \text{ [bit]} \tag{4}$$

where $p(d_{osj})$ is the probability of occurrence of message d_{osj} flowing out of the system, and d_{oj} is the corresponding correct result, i.e. the result that belongs to the knowledge domain under consideration when a specific message, d_{ij}, enters the system. The term $p(d_{osj} = d_{oj})$ is the probability of the jth output message of the system to be correct. The basic idea here is that the information quantity delivered by each output message should be weighted by its probability of being correct. If system answers actually are all correct, the second term on the right side of Equ. 4 has a null effect

[1] An early version of this extension can be traced to [11], even though other symbols were used there.

(factor equal to 1) and consequently the three quantities no, nos, and nosc will be the same. On the other extreme, if output messages are not related to the knowledge domain, or, to put it briefly, answers are wrong, n_{osc} will be zero, leading to zero [lin] of knowledge, even if n_{os} is much larger than n_o.

3 Cognitive Quantities in Perception, Decision and Action Processes

In principle, the MCS model can describe as well human processes as those processes running on man-made systems. Nevertheless, « cognitic » rather than « cognitive » could be used, when relating specifically to man-made systems rather than to human beings. The quantitative assessment of cognitic and cognitive properties show that in real world systems, the performance levels of cognitive or cognitic systems may not always lie where expected.

Observing humans, the overall cognitive system could be validly considered as a single black box, characterized in particular by overall input and output information flows. At the other extreme, it might also been useful in many circumstances to analyze things at a much finer granularity level and this is also possible within MCS model. We shall consider here the traditional fragmentation of overall process into three main parts : perception, decision, and action.

In general, perception processes imply much larger cognitic/cognitive quantities (in particular complexity and knowledge) than action, and, even more so, than decision. In A.I. however, attention tends to be focussed on decision processes, while for real-world systems, perception and action processes cannot be ignored.

Considering the elementary task of starting or stopping a car at a crossroad, as a function of red or green states of a traffic light, the following quantities may be estimated.

$$K_{decision} = \log_2\left(1 \cdot 2^1 + 1\right) \approx 1 \quad \text{[lin]} \tag{5}$$

$$K_{perception} = \log_2\left(1 \cdot 2^{30000} + 1\right) \approx 30'000 \quad \text{[lin]} \tag{6}$$

assuming a good enough traffic view, compatible in quality with what a 100 row x 100 column, color camera can acquire

$$K_{action} \approx \log_2\left(200 \cdot 2^1 + 1\right) \approx 10 \quad \text{[lin]} \tag{7}$$

assuming a 1 meter long trajectory to be travelled, with 1 cm accuracy, in 3-D space, for a leg actuating the gas or the brake pedals.

4 Behavioral Stability of Groups

It has been shown in [1] that the agility of closed-loop controllers, relative to the dynamic behaviour of systems to be controlled, is critical for success (re. fig.1).

Taking two examples in robotics can make this point very clear. As a first example consider the current limiting loop in a motor coil: measuring, taking a decision, and switching power circuits can be done much faster than current may vary. Therefore on/off control is feasible (green area). On the contrary, a control loop including Earth decisions for a rover on Mars will require up to 40 minutes of time delays, and therefore no process involving seconds or even minutes on planet Mars can be validly managed with closed loops in this context (red area).

A single robot, as in Robocup-at-Home applications, consists, even alone, in many components, which feature various time characteristics and specifications. Supervision processes may be run by regular PC's or laptops, with typical control times of the order of 0.1-10s. Joint coordination may be ensured only with additional, faster control resources, such as synchronized servocontrollers. And low-level power circuits at joint level must be capable of control time-span much shorter than 1 ms.

In the case of multiple systems, such as a group of cooperating robots, interactions will usually occur, thus creating a potentially large number of individual, elementary control (decision) loops.

For a successful overall system behaviour, it is critical that in all control (decision) loops, the relative agility be good enough. Taking the relative agility as an indicator provides a sound basis for task allocation and decision priorities in group negotiations.

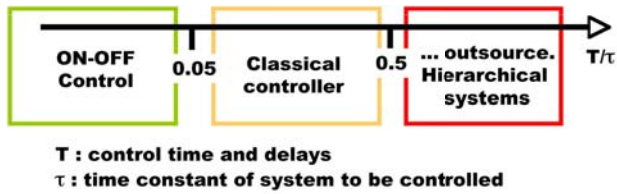


Fig. 1. Control may be easy to be achieved, or quite impossible, depending on the relative agility of controllers (T/τ)

For a successful overall system behaviour, it is critical that in all control (decision) loops, the relative agility be good enough. Taking the relative agility as an indicator provides a sound basis for task allocation and decision priorities in group negotiations.

5 Necessity of Time Modelling and Time Reference in Complex, Distributed, Multi-agent Systems

Interacting systems exchange information and thereby may yield numerous control loops. The danger in those cases is that when systems grow, involving more agents and processes the global, collective behaviour becomes unstable. As shown in previous paragraph, an interesting indicator of the reliability of decision making in any considered loop is provided by the agility of the control (decision-taking) element, including communication delays, relatively to some dynamic properties of controlled system elements (characteristic time constant).

In complex, distributed multi-agent systems (consider for example a group of robots as in a soccer team, or the water-circuit management resource of a warship, as

presented by Rockwell Automation), the number of interaction loops may be very large, with very different agility features. The situation is even more difficult to manage if reconfigurations may dynamically occur. In such cases it seems difficult during design phase to forecast all possible configurations and to predefine all the appropriate decision paths and units.

In a soccer game for example instability will suddenly occur if a robot can move fast, but is controlled through an information path (loop) relatively slow, because of including communication through several team members.

Approaches worth to be considered include the following ones: 1. to characterize extensively signals (samples) in terms of phase (add accompanying time-stamps). 2. to dynamically identify the time responses of elements to be controlled (for all relevant loops). 3. to reduce as much as possible the occurrence of loops by organizing subsystems in as decoupled a way as possible (functional and topological autonomy). When feasible, approach 3 is drastic in avoiding loops and thereby the risk of instability. Approaches 1 and 2 allow applying in real-time the agility criterion mentioned in the previous paragraph for allocating instantaneous decision rights.

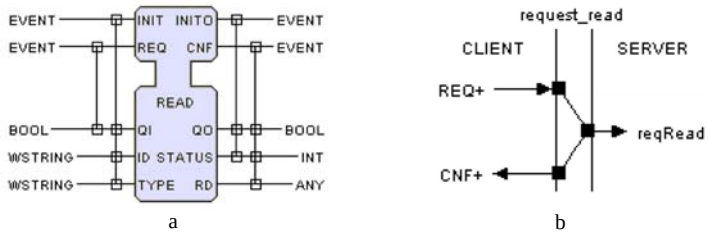


Fig. 2. Example of functional block in IEC61'499 framework (a), and example of associated elementary signal sequence (b) [9]

For example, in novel proposals for so-called intelligent control, such as in particular in O3NEIDA context [e.g. 6, 10], a good basis is already provided by the formalism of functional blocks (FB). As shown in fig. 2, time is already explicitly taken into account in terms of sequence and causality. An additional modelling step making delays explicit seems to be practically feasible for FB's and would provide the basis for robust, distributed behaviour in complex systems in this context.

Other interesting examples are numerous in the domain of cooperating robots.

6 Case Study – Quantitative Assessment of Cognitive Performance Levels for a Mobile, Cooperating Robot, in Domestic Environment

This section relates to mobile, cooperating robots. It presents and illustrates the general idea that perception tasks are usually much more demanding, in cognitive performance levels, than action and, even more so, than decision tasks. It also helps demonstrating how MCS metrics can be practiced and prove useful.

Fig. 3. RH3-Y, our mobile, cooperating robot for demonstrations in domestic applications. Blue trays, with individual covers, are there for user purpose, in « at home » applications. The lower level contains electrical and electronic devices, servo controller and PLC ;. the supervision computer can be lying on top of the trays, for development phases, but is normally smaller and also confined in the lower part of the robot; or is replaced by a fixed, regular computer, operating remotely via Ethernet and TCP/IP connection.

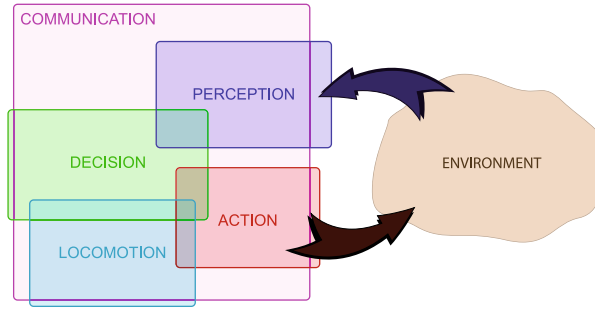


Fig. 4. Overview of main cognitive functions of a mobile, cooperating robot

Fig. 3 presents RH3-Y, the third version of our mobile, cooperating robot for demonstrations in domestic applications, which has followed previous designs for Eurobot. The first version, RH1-Y, has taken part in competitions [7] and has for example proven capable, in principle, to follow a human, which is a basic ability for many potential home applications (carrying goods, accompanying persons in order to be ready for services, being trained for preferred paths, etc.).

As overviewed in Fig. 4, and shown in a more detailed form in Fig. 5, the main cognitive processes can be more or less distributed in specialized functional units: perception (in particular word recognition, visual object recognition, obstacle localisation), decision, action, etc.

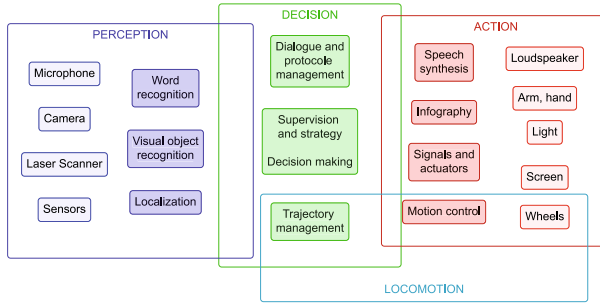


Fig. 5. Refined view of main cognitive functions and resources of RH2-Y (This robot is RH1-Y augmented by the addition of basic arm and « hand », among other improvements)

Let us consider some specific tasks to be handled by RH1-Y. In the context of Robocup-at-home, very precise tasks have been defined in 2006, and they are updated every year. There has been in particular a « navigation » task, which required in principle that the system visit 3 locations, according to user's choice, among about 10 specific predefined possible locations

In order to have a convenient human-robot interaction, vocal dialogue is particularly well suited

As stated so far, and globally, the quantity of information as input of the system is about 10 bit, considering that we have 3 words, each one being equally probable among 10 possibilities. Here the output quantity is the same, the cognitive process being purely to transfer on the output what is fed as input.

$$n_i = n_o = \log_2(10) * 3 \approx 10 \text{ [bit]} \tag{8}$$

These input and output quantities provide the essential substance in order to qualify the necessary amount of knowledge required for the task as stated.

$$K_{global} = \log_2\left(10 \cdot 2^{10} + 1\right) \approx 13 \text{ [lin]} \tag{9}$$

This quantity is small.

In practice, the global view is, schematically speaking, also the view at the level of the decision unit. However the user does not feed the decision unit with a nicely encoded, 10 bit signal. The user « just speaks », in English, with the restricted vocabulary mentioned (3 times one word among 10 possible topological names).

Consequently, the robot needs a perception stage. As shown on Fig. 5, the sound path starts with a microphone, connected to the perception unit. At the input interface of the latter, the amount of information received is about 50'000 bit per word. At least 150'000 bit for all three of them (assuming a 0.5 s duration per word, 10 kHz of sampling frequency, and 1% accuracy; classical information theory).

When everything works perfectly, the three words (out of ten possible) are abstracted from the sound-wave, i.e. recognized, which means that about 10 bit of correct information is indeed delivered by the perception unit to the decision unit. (In case the recognition is not totally error-free, the equation of paragraph 2, above, should be taken)

Thus the knowledge quantity required for the perception stage is the following:

$$K_{WordPerception} = \log_2\left(10 \cdot 2^{150'000} + 1\right) \approx 150'000 \text{ [lin]} \tag{10}$$

On the other hand, once decided, a destination location has still to be reached. Assuming, first, a 10 cm accuracy, second, an average coordinated motion along a 5 m path, to be done three times, and third, a motion in the plane (3 degrees of freedom), the action function has to concretely deliver (« synthesize ») about 2'500 [bit] of (correct) information in order to define the trajectory:

$$n_o = 3 \cdot \frac{5}{0.1} \cdot 3 \cdot \log_2\left(\frac{5}{0.1}\right) \approx 2'500 \text{ [bit]} \tag{11}$$

Thus the knowledge quantity required for the action (or locomotion) stage is the following:

$$K_{TrajectoryPlanning} = \log_2\left(2'500 \cdot 2^{10} + 1\right) \approx 30 \text{ [lin]} \tag{12}$$

Looking back at Equ. 9, we can see that the estimation of knowledge required is not realistic there. In the case we review, the input information is vocal, and on the output side a full trajectory is to be defined (and travelled). The perception stage and the action stage should not be overlooked.. Consequently the following expression appropriately describes the amount of knowledge required for the task to be successfully performed:

$$K_{global2} = \log_2\left(2'500 \cdot 2^{150'000} + 1\right) \approx 150'000 \text{ [lin]} \tag{13}$$

The design of cognitic systems featuring such a large amount of knowledge is usually not obvious. Fig. 4 gives an overview of the control resources and architecture adopted for the design of RH1-Y.

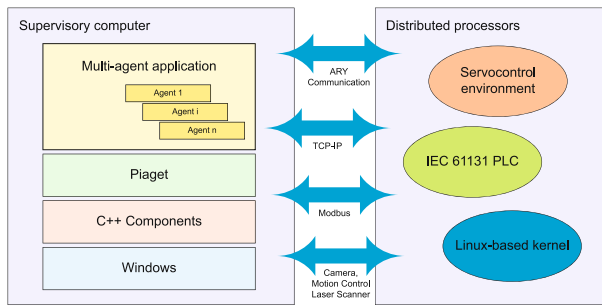


Fig. 6. Overview of the control resources and architecture of RH1-Y

It can be observed in the figure that a variety of embedded agents are used, in order to match very different time-scales: supervisory computer for large-scope, relatively less agile control loops; and, for more agility in reflex loops a PLC (re. IEC 61'131 programming standard), complemented with yet more agile servo controllers and

specialized processors in smart sensors (in particular color camera, laser scanner). These various resources are interconnected but remain to a large extent autonomous. This is well in line with the measures proposed in paragraph 5: adapting the agility of each controller to the specific requirements of the task they control; and keeping to a minimum level the amount of interaction between agents.

Fig. 7. Our robot RH3-Y follows our team member Illkyun Jeon at speed of up to 1m/s through the "Home" at Robocup competition 2008 in Suzhou, China, successfully crossing next another team

7 Conclusion

The paper has presented several contributions to the field of cognitics and multi-agent systems.

The first main contribution relates to a quantitative approach for cognitive systems, and in particular to the importance of abstraction and concretization processes, Cognitive systems can be viewed as single black-boxes, processing information, However they can also be considered as a structure where more detailed processes are present (or conversely as larger systems, such as a group of robots for instance). For humans or autonomous, cooperating robots, perception, decision, or, in a more refined fashion yet, vision or trajectory planning are good examples of (sub-) processes that can be addressed independently, as specific functions. The MCS metric system is generally helpful in order to guide understanding and developments of cognitive systems, Applying this metrics shows that typically the largest cognitive load is put on perception, then much less is required for action, and even less, for decision functions; while by intuition people usually tend to refer mostly to the latter.

The second main contribution shows how time and time-related properties are crucial indicators for the design and operation of complex, distributed, and in particular

multi-agent systems. For complex cases, many resources are necessary, a multi-agent structure may be useful, but then interactions occur and this implies loops where information flows, and consequently stability becomes increasingly a concern. The paper shows that a critical property in every loop is the relative agility of control elements, and therefore time and delays must be tracked, or at least represented in a model when tracking is not feasible, in order that decision power be allocated dynamically to the appropriate elements, thus keeping distributed, complex, multi-agent systems stable and effective. Another element of solution is to reduce interactions and loops by isolating components, giving each of them as much autonomy as possible.

The third and final main contribution of the paper relates to a cooperating robot for domestic use, as a practical study case, where above concepts have been illustrated, practiced and tested. In particular, vocal communication has been shown to be very demanding in terms of cognitive performance and, with current implementation, does not always behave totally without errors. For such cases, the probability of error is to be taken into account for the relevant assessment of knowledge quantities.

The contributions of many students in past Eurobot projects [12], as well as numerous contributions from colleagues, and from the technical departments of HESSO.HEIG-VD, are gratefully acknowledged here.

Cognition classically involves information processing and modeling. Recent works with careful attention to quantitative estimation have made it clear however that these prerequisites keep cognition quite disconnected from reality [13]. No matter what element of reality is considered, in common practice only an infinitesimal portion of its nature is perceived and explicitly described.

References

1. Dessimoz, J.-D., et al.: Ontology for Cognitics, Closed-Loop Agility Constraint, and Case Study – a Mobile Robot with Industrial-Grade Components. In: Proc. 5th IEEE Intern. Conf. on Industrial Informatics INDIN 2006, Singapore (August 2006)
2. Dessimoz, J.-D.: Beyond information era: cognition and cognitics for managing complexity; the case of “enterprise”, from a holistic perspective. In: Proc. ICIMS-NOE (Intelligent Control and Integrated Manufacturing Systems- Network of Excellence), ASI - 2000 (Advanced Summer Institute) and Annual Conf., CNRS-ENSERB-Univ. de Bordeaux, France, September 18-20, pp. 164–170 (2000)
3. Shannon, C.E.: coll. papers, Sloane, N.J.A., Wyner, A.D. (eds.). IEEE Press, Piscataway (1993)
4. Performance Evaluation as a Tool for Quantitative Assessment of Complexity of Interactive Systems. LNCS. Springer, Heidelberg (2005) ISSN 0302-9743
5. : Towards a Knowledge Society - Is Knowledge a public good? Dynamics of Knowledge production and distribution. In: ESSHRA (Extending the Social Sciences and Humanities Research Agenda) International Conference, Bern, Switzerland, June 12-13 (2007)
6. Vyatkin, V., Christensen, J., Lastra, J.L.: An Open, Object-Oriented Knowledge Economy for Intelligent Distributed Automation. IEEE Trans. on Industrial Informatics 1, 4–17 (2005)
7. RobocupAtHome League (2007), `http://www.robocupathome.org`

8. Dessimoz, J.-D., Gauthey, P.-F.: RH2-Y – Toward A Cooperating Robot for Home Applications. In: Robocup-at-Home League, Proceedings Robocup 2007 Symposium and World Competition, Georgia Tech, Atlanta, USA, June 30-July 10 (2007); Or: Dessimoz, J.-D., Gauthey, P.-F.: RH3-Y – Toward A Cooperating Robot for Home Applications. In: Robocup-at-Home League, Proceedings Robocup 2008 Symposium and World Competition, Suzhou, China, July 14-20 (2008) (accepted for publication)
9. IEC Project 61131-3 - Programmable Controller Languages, Annex H Interoperability with IEC 61499 devices, IEC (2007), http://www.holobloc.com
10. Vyatkin, V.: IEC 61499 Function Blocks for Embedded and Distributed Control Systems Design, 271 p., O3NEIDA - Instrumentation Society of America (2006) ISBN - 10: 0-979-1330-1-7
11. Dessimoz, J.-D.: Towards a Theory of Knowledge. In: Proc. 3rd Ann. Conf. of the Swiss Group for Artificial Intelligence and Cognitive Sciences (SGAICO 1991), Inst. for Informatics and Applied Mathematics, Univ. Bern, Switzerland, March 1992, vol. IAM-91-004, pp. 60–71 (1992)
12. Dessimoz, J.-D.: Ten Years of Experience with Eurobot; Achievements, Lessons Learned and General Comments. In: Proc. Eurobot Conference 2008, Heidelberg, Germany, May 22-24 (2008) ISBN: 978-80-7378-042-5
13. Dessimoz, J.-D.: Contributions to Standards and Common Platforms in Robotics; Prerequisites for Quantitative Cognitics. In: International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR) 2008. First International Workshop on Standards and Common Platform for Robotics, Venice, Italy, October 3-7 (2008)

Assembly of Fuel Cells and Stacks with Robots

Peter Konold, Avdo Muminovic, and Manfred Wehrheim

Abstract. PEM-fuel cell stacks (PEM = **P**olymer-**E**lectrolyte-**M**embrane) are still produced manually to a large extent. Therefore identical components have to be positioned precisely. Besides higher total costs, logistical and weight problems is also a result in a higher statistical sensitivity for failures due to assembly errors and the total of assembly tolerances. The aim of the project is to optimize the economic assembly of fuel cells in a significant way. Therefore the development of an automatic assembly process to join a membrane-electrode-arrangement with bipolar plates to a single cell stack in a leak-proof way is of essential importance. The single cells which have been produced and checked by a robot can be stored on work piece holders. These single cells are joined process-sure to bigger stacks also by an assembly robot. The benefits of this effective assembly by robots will be explained here.

1 Introduction

The production of PEM-fuel cell stacks (PEM = **P**olymer-**E**lectrolyte-**M**embrane) is still carried out manually to a large extent. Therefore identical components have to be positioned precisely. Besides higher total costs, logistical and weight problems, the large number of components also results in a higher statistical sensitivity for failures due to assembly or material errors and the total of assembly tolerances. The aim of the project is to optimize the economic assembly of fuel cells in a significant way. Therefore the development of an automatic assembly process to join a membrane-electrode-arrangement with bipolar plates to a single cell stack in a leak-proof way is of essential importance. The single cells which have been so produced and checked can be stored or held in readiness on work piece holders (part one of the research project).

In another automated assembly-step these single cells are joined process-sure to bigger stacks. Here the scaling-up to many-cell-stacks will be examined for logistic and assembly benefits (part two).

2 Automatic Joining of the Fuel Cell

Within the production of fuel cells at the ZSW (Centre for hydrogen and solar technology in Ulm) a robot assembling cell has already been built up. This cell is restricted to the mounting or combining of single components of the fuel cell.

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 168–179, 2009.

In this project of the University of Applied Sciences in Ulm in close collaboration with the ZSW the function of this robot-system will be expanded and then transferred into fully automated operation. The leak-proof function of the fuel cell has been archived by using a metal frame gaskets or the application of gaskets directly onto the main components. In this case the weight and cost saving technique of direct application of a glue gasket is used. For this a dispenser-system for applying the glue-gasket has been integrated into the robot cell. To control the glue gaskets a vision processing system with camera and PC has been installed. For part hardening, final control of the parts and following assembly to finished stacks a transfer system (TS 2plus from Rexroth) with carriers and identification systems (ID40) which transfers the fuel cells in and out of the robot cell has been introduced.

These measures required large changes and extensive works on the gripper, the integrated SPC (stored program control), the sensors and the valve-technology.

The system has the following components:

- Robot system (6 axis jointed-arm robot, with a lifting capacity of 12 kg and an operation distance of 1385 mm) with a SPC to control the periphery
- Batcher with a dispenser system containing
    - Dosing unit for gaskets, squirt dosing unit: a fully integrated system able to dispense products of a liquid to pasty consistency from 10ml plastic injection devices
    - Gasket-glue in 10ml plastic injection devices
    - Profiles and angles for fixed installation
    - Injection-holder with simple positioning system
- NeuroCheck Software for industrial vision processing with 16mm lens camera and PC, Fire wire connection and Digital I/O-card
- Transfer system TS 2plus with double belting and carriers, identification-system (ID 40), Supply and transport system to lift and position the parts.

In three cycles glue is applied to the bipolar plate1, a membrane and the bipolar plate2, controlled via the picture processing system and placed on the correct carrier. A single fuel cell is manufactured.

Plates or membranes with damaged glue gasket are removed from the assembly-process immediately and the process is repeated with a new plate or membrane.

Thanks to the space saving placement and the optimal exploitation of the carrier, three fuel cells can be placed on each carrier and transported out of the assembly-cell to harden.

If the leak control system is included in the assembly cell, the fuel cells will be kept in the assembly cell for hardening. Then they will be checked by the leak control system and then transported out of the assembly cell on carriers as assembled single fuel cells. This is where the identification system (ID40) which is installed on the carriers will come into operation. This system allows documentation of a single cell, the assembly time, the batch number, the leak check results before the assembling of the stack as well as statistics for a general analysis like number of manufactured cells,

number of good parts, place and time of production. The data can be transferred contact free at any step in the system with a RW (Read Write) head and sent to data processing.

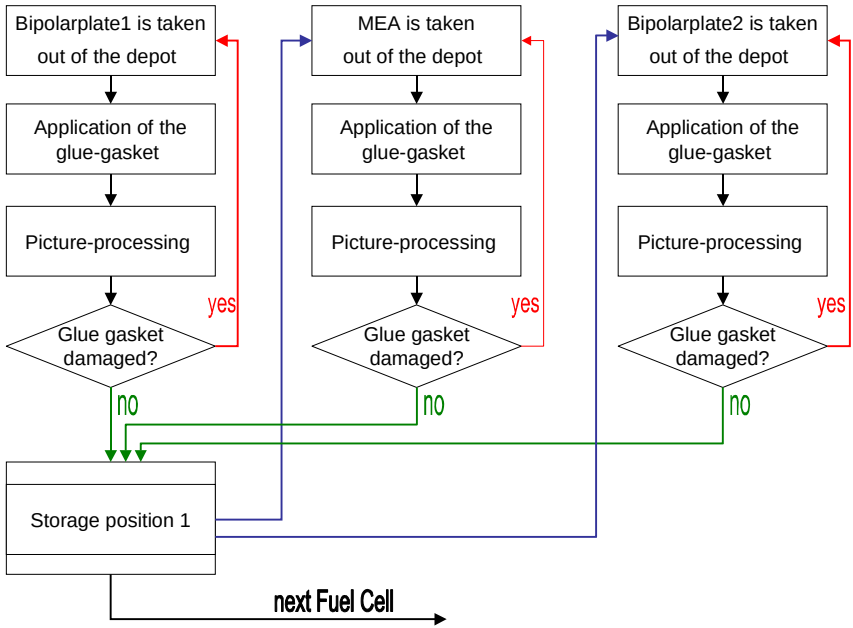


Fig. 1. New Assembly Structure

Glue application bipolar plates/MEA

The glue application, which is the application of a squirted gasket, is a sensitive process. Before the process can begin, the viscosity of the glue has to be checked with the help of the dosing unit. Also the injection needle must be free of glue.

The glue applications for the BP1 (figure 2) and the MEA are identical (large glue gasket), because of its different shape the glue gasket for the BP2 is smaller. The height of the glue gasket must not exceed 0.7mm and the gasket must be applied evenly. In case of an unevenly applied gasket problems may occur regarding the join and leak proof seal.

The glue-gasket is checked by a visual processing testing program created with NeuroCheck. The hardware used here consists of a FireWire camera with over 1 mega-pixel, a lens with a light intensity of 1.4, a PC and an I/O-card.

During the check there must be no change in light conditions which could affect the recording quality (e.g. extraneous light coming from the sun). The dim out of the checking area prevents reflections from the robot, gripper and other surrounding parts.

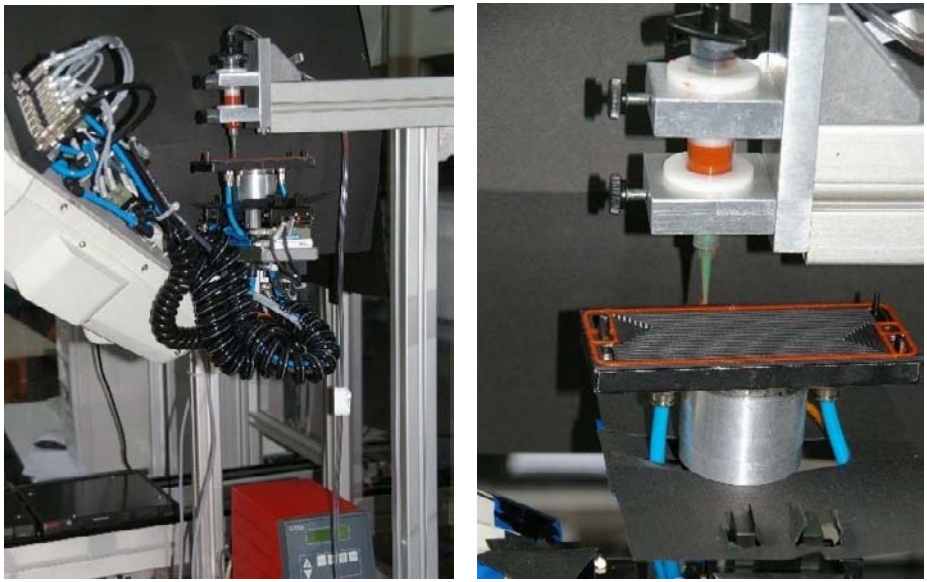


Fig. 2. Glue-Application with robot

Checking the glue-gasket

Through optimized adjustment and illumination the unbroken continuity of the gasket is checked. The program execution of the single checking-steps and the binary image used by the computer while processing the data is shown in figure 3.

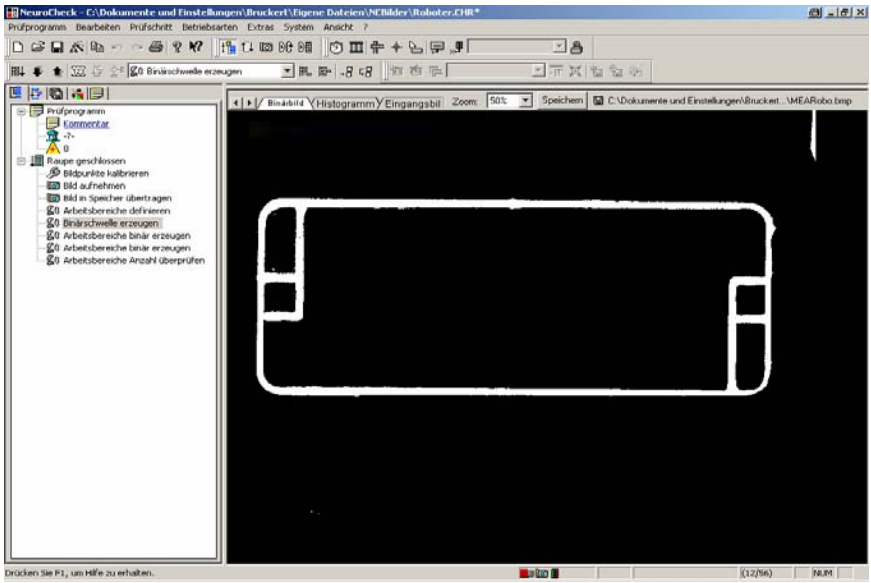


Fig. 3. Checking the glue gasket

The glue gasket on a MEA is also visible in figure 3. The adjustment of the optical exposure of a gasket on the MEA is much more difficult than on a bipolar plate, because the reflection of the foil has to be considered. Conclusion

Within this first part of the research project the automatic assembly of a single fuel cell has been developed and realized. To check the glue application the picture processing system NeuroCheck was used. The linking to other assembly and process stations is managed by a transfer system with carriers.

This research work "automated assembly of single fuel cells for the use in fuel cell stacks" is supported by the project "Innovative Projekte" of the Universities of Applied Sciences in Baden-Württemberg as well as in cooperation with the ZSW /4/.

With this robot cell (figure 4) it is now possible to automatically assemble single fuel cells beginning with lot size 1. The following tests will show how many cells can be produced, at what quantity a different assembly cell setup is required and what momentary advantages there are as well as the saving of an extra gasket due to the robot assembly.

Fig. 4. Carrier in the assembly station with 3 storage places

3 Construction of PEM Fuel Cell Stacks

In another process the hardened single cells are checked for leak proofness, and sorted into performance and quality categories.

The constructed fuel cell stack is a functioning power supply. Depending on the required Voltage it is necessary to assemble the number of needed single cells and the other elements for the operational discharge. In figure 5 these parts are shown before and after assembly.

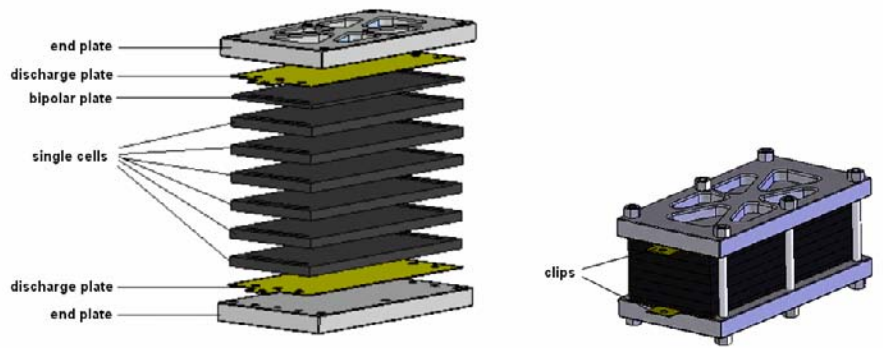


Fig. 5. Elements of a fuel cell stack – not assembled/assembled

Because parts such as the end plate, the flux voltage plate or the thread rods are needed only in small quantities for each fuel cell stack, these parts are considered as enclosed components by the material flow and are provided on staging carrier.

Automated assembly of stacks

The compact final assembly is done from staging carriers by a robot. The layout of this robot-station in figure 6 shows the transfer-circle with the material flow and the transport way of assembled fuel cell stacks. The screwing station attached to the

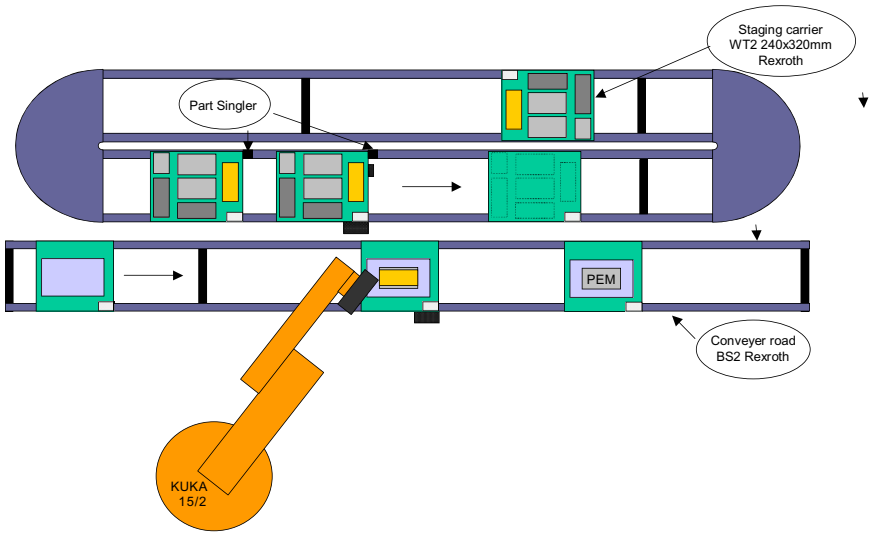


Fig. 6. Layout of stack-assembly with robot

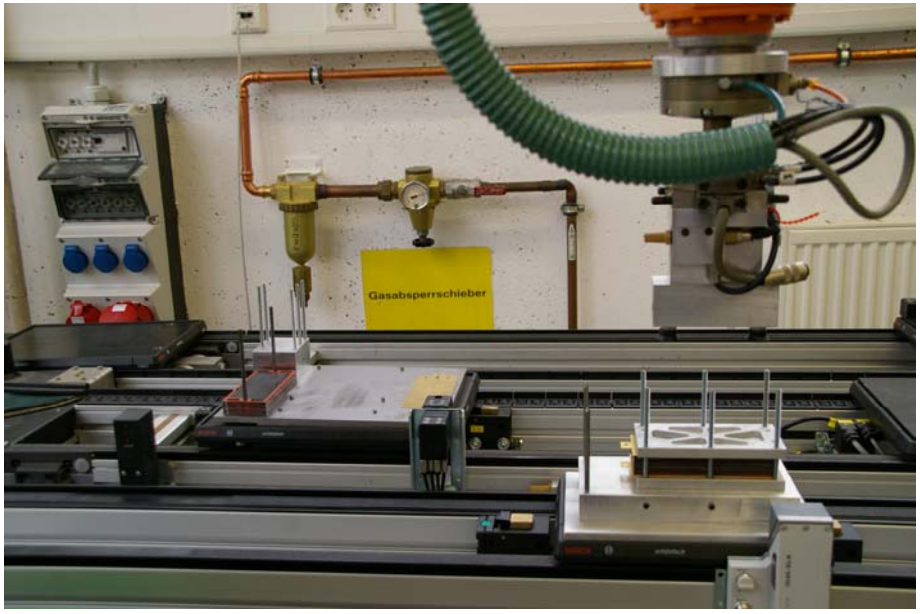


Fig. 7. Robot with gripper-change-system an vacuum-gripper

transport way is not taken into consideration here. For a universal application (e.g. further assembly methods) a robot with a 15kg lifting capacity is used during the experimental stage.

When the part supply is carried out this way, the assembly can also be managed by a cheaper SCARA-robot, because only xyz movements and rotations are necessary. The plates and flat parts are picked up by a vacuum gripper and the pins or rods are joined by a parallel gripper, therefore a gripper change system is recommended.

Transfer system

While designing the system a special focus was laid on a high flexibility and economic effectiveness. In our design the different sized staging carriers are transported via two double-belt transport systems. Each system is equipped with a lifting and positioning system for exact fixation of the staging carriers for part removal or final assembly. It also is equipped with a part singler. The electrical identification-system (ID40) attached to the staging carriers allows part tracing and the production of different voltage stacks.

Programming

The Kuka robot was coded at the Hochschule Ulm. A basis program allowed the assembly of different voltage fuel-cell stacks. Currently the system is set up for low voltage fuel-cell stacks (3V-12V). The assembly time of a 3V fuel cell stack is approximately 1.5 min including the gripper changes.

Fig. 8. Programming (teaching) of a Kuka robot

4 Economic Effects on Production

Analyses and tests of the automated assembly of fuel cell stacks show that fuel cells and fuel-cell stacks can be manufactured by robots. Here the production of a smaller lot size with a more complex robot cell or a higher quantity with more robot cells is the goal. If costs can be cut on the part itself (extra metal frame gasket), the production becomes even more economical. The costs with and without the extra metal frame gasket are shown in figure 9.

Depending on the size of the cell the costs of the metal frame gasket can be up to 25% of the other parts of a fuel cell. The costs for the extra supply carrier integrated in the production process and the extra production time also have to be added.

The momentary costs should not be the only thing taken into consideration. In a cost-benefit analysis the non-financial criteria such as the early selection of rejects due to defects shown in figure 10. It shows the considerable advantages of the glue-gasket robot assembly.

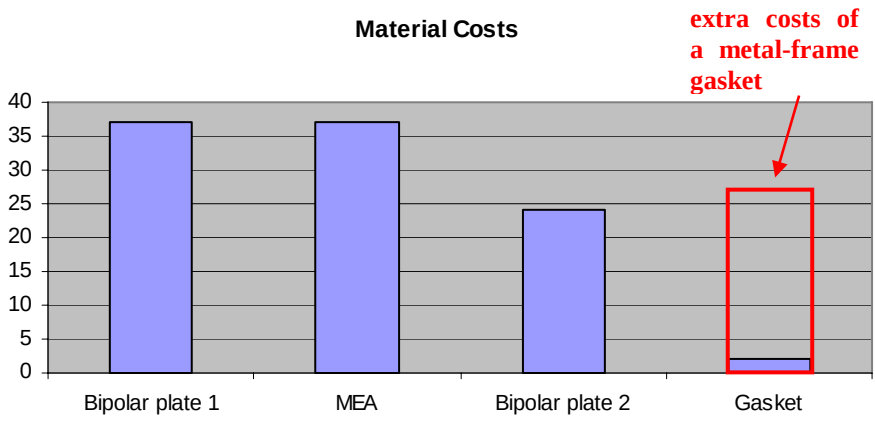


Fig. 9. Cost influence of the metal frame gasket on a fuell cell

		Conventional		Glue application	
Factor	**Weighting**	**Fulfilment**	**Value of benefit**	**Fulfilment**	**Value of benefit**
Higher durability and reliabiliy	30	4	120	8	240
Improvement in process-sureness	23	4	92	8	184
Relif of labor conditions	14	2	28	4	56
Early selection of defective goods	23	3	69	8	184
Design	10	3	30	7	70
Points total	**100**		**339**		**734**

Fig. 10. Non-financial criteria evaluation

In a cost benefit comparison of different experts of both production types it becomes even clearer. Unfortunately up until now we have not been able to document this with exact numbers. We have had to make do with projections for higher piece numbers.

If our calculations were transformed into a layout for a production system, a linear or parallel production line could be realized.

Because of the big difference in the way single cells and cell stacks are assembled a special eye has to be kept on the capacity utilisation. At the present stage of knowledge an assembly of nine single cells to a cell stack (figure 13) is the most efficient production type. The above shown parallel production relates to a production of a 12V PEM fuel cell with a base scale of 50x120 mm². Every modification in size, voltage etc. changes the layout, the number of production steps and robots in the production process. A yet-to-be-done optimisation of the robot movements and positioning of the staging carriers could result in a much higher productivity.

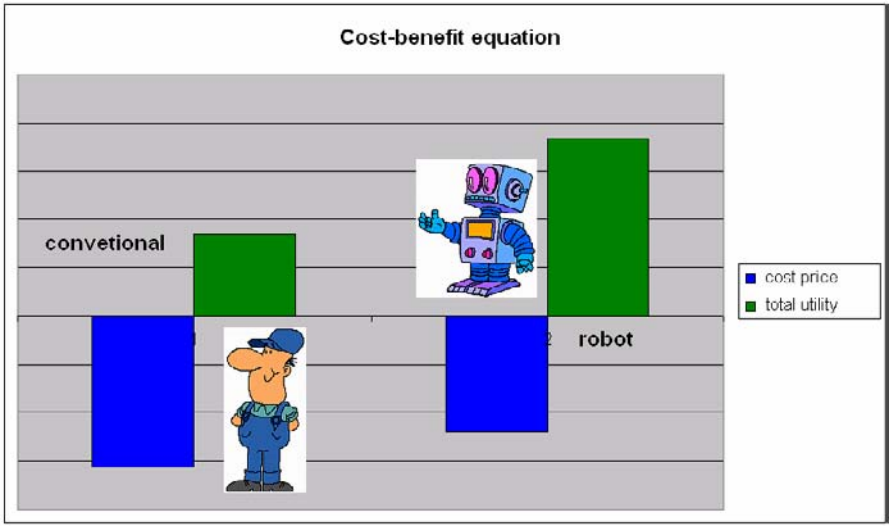
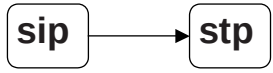


Fig. 11. Cost benefit comparison of conventional and robotic assembly

Line production



	single-cell production	cell-stack production
Available assembly time	1.870 h	1.870 h
Time per piece	75 sec	95 sec
Max. number of pieces	89.760 pcs	70.863 stacks
Number per stack		12
Max. Stack number		7.480 stacks
Utilisation	**100.00 %**	**10.56 %**

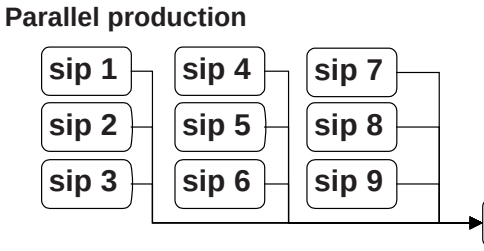
Legend

sip single-cell production

stp stack production

Fig. 12. System layout for a linear production

It is our goal to optimize the basic operation sequence and to make the production even more process sure to gain a high product quality which leads to high capacity cell stacks.



	single-cell production	cell-stack production
Machines	9 machines	1 machine
Available Asseblmy time p	1.870 h	1.870 h
Overall assembly time	16.830 h	1.870 h
Time per piece	75 sec	95 sec
Max. piece number	807.840 single-cells	70.863 stacks
Single-cells per stack		12 single-cells
Max. stack number		67.320 stacks
Utilisation	**100.00 %**	**95.00 %**

Legend

sip single-cell production

stp stack production

Fig. 13. System layout for a parallel production

5 Future Perspective

Analyses and tests of the automated assembly of fuel cell stacks show that fuel cells and fuel cell stacks can be manufactured by robots. With these robot cells the production of fuel cells with a very small lot size is possible. The goal is that big piece numbers of single cells will be produced by many robot cells and the cell stacks with one high performance robot. Further research will be conducted to support this.

Through the automation realized here and cost saving on the part itself the production will become even more economical.

The advancement of this technology which is environmentally friendly, smaller and weighs less than the conventional lead battery could be driven forward in terms of the price performance ratio through the automated production.

The procedure shown here indicates that if only one production step is optimised the costs can be reduced by up to 25 %.If further optimisation is done e.g. during the production of the MEA or the bipolar plates it can be estimated that in a few years the production costs could be so low that a mass production of energy through this technology is within reach.

If a good hydrogen distribution grid, a cheap and low in co2 emission production of hydrogen through for example wind energy or electrolysis could be established, a competitive system for multiple applications could be reached.

List of Literature

(1) Böhmer, E., Ehrhardt, D., Oberschelp, W.
Elemente der angewandten Elektronik, 15. Auflage, Vieweg Verlag, 2007
(2) Knaupp, Katrin
Programmierung einer Anlage zur automatisierten Montage von Brennstoffzellen durch Industrieroboter - Optimierung und Erweiterung der Steuerung und Bedienung
Hochschule Ulm SS 2007
(3) Bruckert, Annabel
Programmierung einer Montageanlage und Einsatz der Bildverarbeitung zur Herstellung von Brennstoffzellen
Hochschule Ulm SS 2007
(4) Zentrum für Sonnenenergie und Wasserstoff-Forschung Ulm, www.zsw-bw.de
(5) Krausa, Michael
Batterie oder Brennstoffzellen?
Fraunhofer-Institut für chemische Technologie
(6) Schmidt-Walter, Heinz
Bericht: Brennstoffzellen als Batterieersatz, 2005
(7) Kappeler, Gabriele
Automatisierte Montage einer Brennstoffzelle
Hochschule Ulm SS 2007
(8) Forschungszentrum Karlsruhe
Die eierlegende Wollmilchsau?, Presseinformation Forschungszentrum Karlsruhe, www.fzk.de
(9) Scholta, Joachim, Zedda, Mario
Portabel Anwendungen mit PEM-Brennstoffzellen,
Forschungsverbund Sonnenenergie „Themen 1999/2000“
(10) Hzwei, Das Magazin für Wasserstoff und Brennstoffzellen, www.linde-gas.de
(11) M. Schmidt, Volkmar
Mobile Energieumwandlung: Batterien und Brennstoffzellen
VDI-Berichte Nr. 2007 s.S. 23-42 2007
(12) Bericht von dem 4. Deutschen Wasserstoff Congress , Essen
Wasserstoff kommt langsam in Fahrt
VDI-Nachrichten 29.02.2008

Prof. Dipl.-Ing. Peter Konold teaches in the faculty of Manufacturing Engineering & Management at the University of Applied Sciences. Special Qualifications are Robotic, Assembly, Vision-Systems and Speechrecognition.

Dipl.-Ing.(FH) Avdo Muminovic is Production-Manager at the Institute of Manufacturing Technology and Material Testing (IFW) and employee on the research project.

Prof. Dr.-Ing. Manfred Wehrheim is Vice President for research and transfer at the University of Applied Sciences Ulm and teaches in the faculty of Manufacturing Engineering & Management the fields Production, CAM and Tool Machines.

The authors are working at the IFW with facilities and laboratories for research and study.

Grape – Graphical Robot Programming for Beginners

Stefan Enderle

University of Applied Sciences Isny, Germany
qfix robotics GmbH, Senden, Germany
enderle@qfix.de

Abstract. Using robot kits for education in schools and universities, we found that there is a lack in tools for teaching the structures of an object oriented programming language. Thus, we decided to develop a graphical programming environment for the beginner, using procedural concepts together with given objects.

The described tool is able to produce C++ code from the graphical user input. So, the mapping between a flow chart and the syntax of the programming language is directly visualized. Additionally, the environment can easily be extended by the user to use additional C++ classes or to create code for different controllers or PC processors.

1 Introduction

The author works in the field of robot kits, which are modular kits (see [1]) consisting of mechanical and electronical components which can be used to build an autonomous mobile robot. After physically building up the robot platform, the created vehicle must be programmed by some programming environment, mostly running on the PC. These robot kits are used for example in school tournaments, like RoboCupJunior [2] where young students from an age of 10 years start to work with and to program robots.

Advanced students have no problem using a real programming language like C/C++ or JAVA. This is because the structural programming entities, like commands, loops, if-then-else-clauses, etc. which are the same for all procedural programming languages, are well-known. Thus, in this case learning a programming language means lerning the *syntax* of a programming language. However, for younger students who do not know the structures of programming, it is important to introduce these entities in a graphical way (like flow-chart elements), and, to make them understand the mapping from the graphical design to the program source code.

Companies developing toy robot kits, like LEGO [3] or Fischertechnik, also provide graphical programming environments for their kits. However, these environments are all dataflow oriented, i.e. the boxes used in these systems are *processes* according to a "input-process-output- model", or simpler *functions* with input and output. The ins and outs of such functional boxes can be combined to larger systems and e.g. the sensor data flows along the described channels being modified by the boxes until it flows into a last box, e.g. a motor, where they performs some action.

This approach is also used in sophisticated programming systems like labView or Simulink where even embedded realtime systems can be programmed graphically.

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 180–192, 2009.

However, due to their complexity these systems cannot be used for absolute beginners. And, we think that the procedural approach where a program is firstly considered as a sequence of statements is much easier to learn than the data flow or functional approach.

This led to the development of *Grape*, an easy-to-use graphical programming environment with which a flow chart (representing the program flow) can be built and the meaning of the individual elements of the flow chart are defined. The complete steps of creating a *Grape* program are explained in the following.

The main design issues for *Grape* are the following:

- **Extensibility:** The set of classes that is given for the user should be easily extended by additional classes. Thus, a simple class description language must have been created.
- **Generic program representation:** There should be a generic XML representation of the graphical program to be able to write additional tools, like translators to different programming languages.
- **Automatic code generation:** The tools should be able to generate C++ (or any other OO language) souce code in order to directly see the translation from the graphical program identities to the respective lines of code.
- **Platform independent:** The tool should run on at least Windows and Linux. So, we chose to use the qt toolkit ([4]) for implementation.

Most of these design issues are discussed in more detail in the following.

2 Grape

The easiest way to introduce *Grape* is to look at a small standard project and how it is implemented using *Grape*. First, have a look at a typical “hello robot-world” program in C++:

```
#include "qfixSoccerBoard.h"

SoccerBoard robot;

int main()
{
  robot.motor(0,255);
}
```

This will let one motor of the robot turn with full speed. In the short example you can see a typical sequence of four basic actions:

- Including a (class) library
- Declaring an object
- Defining the main function
- Using the object by calling its methods

The next sections show how these basic actions are performed in *Grape* leading to a graphical program. After that we biefly look at the process of automatically transforming the graphical program to C++ code.

2.1 Classes and Objects

Grape comes with a list of predefined classes for robot programming. Figure 1 shows the tab “Classes and Objects” where classes can be included and objects of the included classes can be declared.

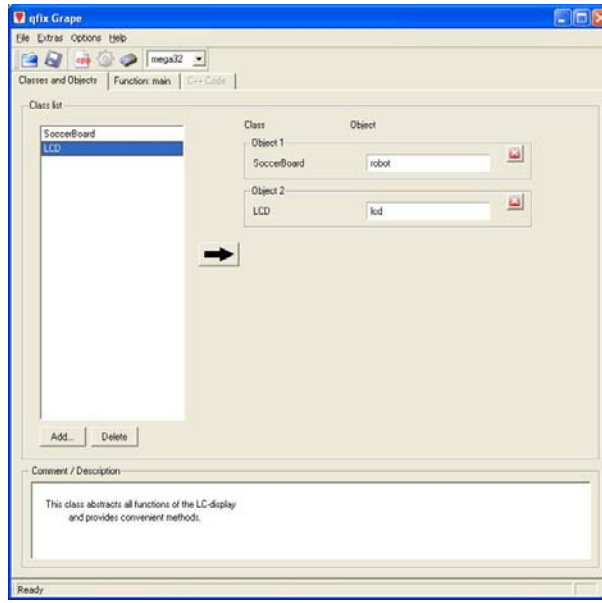


Fig. 1. Two included classes and objects thereof

In the Figure 1 the two classes (`SoccerBoard` and `LCD`) are included and an object of each class (called `robot` and `lcd`) are declared.

This list of predefined classes can be extended by own classes using a simple XML syntax which is described in Section 3.1.

2.2 Graphical Programming

The second tab “Function: main” consists of an almost empty grid holding only the stub of the main program represented by a `Start` and a `End` node (see Figure 2).

Here, to program graphically means to arrange symbolic blocks to yield the desired flow chart. The available building blocks are represented on the left side of the window and the user has to drag-and-drop them over an already existing arrow in the program. Figure 3 shows a first program containing an infinite loop.

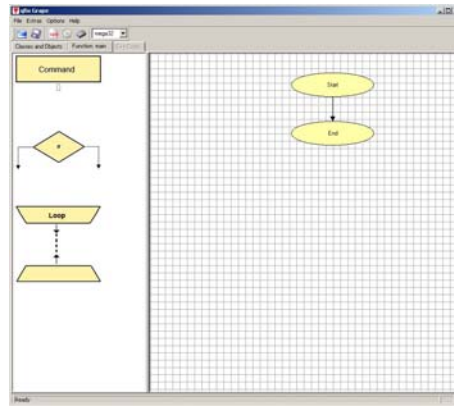


Fig. 2. Main window with program stub

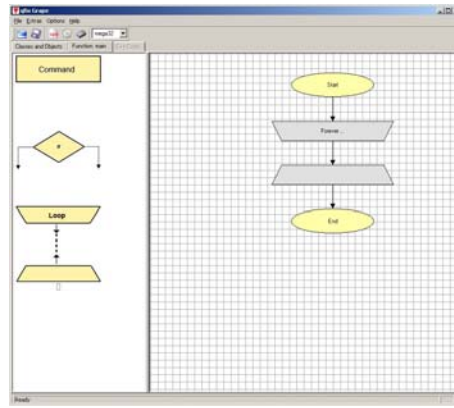


Fig. 3. Infinite loop as first element

The symbolic building blocks currently include the following programming concepts:

- commands (or statements)
- if-statements with else-clause
- loops

Figure 4 shows a graphical program including all three concepts.
Further programming concepts that are not implemented yet are

- own functions or procedures
- variables

Note that the concept of *dragging over an arrow* does not allow the user to produce a logically incorrect program! So, for example, it is not allowed to have an if-statement with a *true*-path ending at another place as the *else*-path. This would result in a "goto"-functionality (as it is done in other graphical programming systems in which boxes and arrows can be placed arbitrarily).

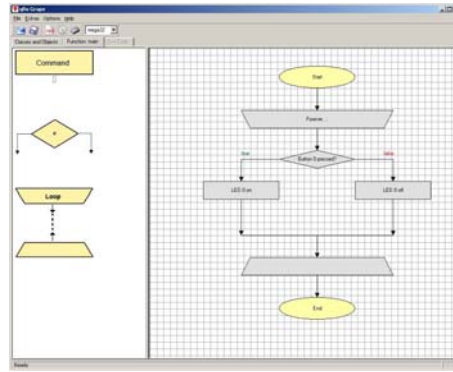


Fig. 4. Graphical program including commands, if-statement and loop

2.3 Dialog-Based Implementation

In the last figures, we see that the selected programming elements are grey. This means that the logical structure is existing, but their meaning (semantics) is not defined, yet. The user can decide if he first finishes the graphical program and then implements the individual boxes, or if he performs the two steps in parallel. However, before being able to generate source code, all boxes have to be implemented with the desired semantics.

The indicator for an element being defined or not is its color. A grey box indicates the pure element without meaning, a yellow box indicates an implemented command, loop, etc.

In order to define the semantics of a specific element it can be double-clicked which opens the respective dialog for the implementation. Figure 5 shows the dialog belonging to a command box.

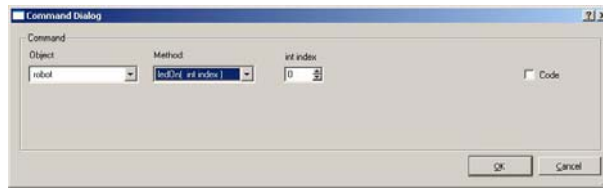


Fig. 5. Command dialog

You can see that one of the declared objects can be selected and one of the selected object's methods can be chosen. According to the method's signature the respective arguments can then be specified. With this approach it is not possible to produce any syntactical error.

A respective dialog is used for loops and if-then-else clauses.

3 Internal Representations

3.1 Class Representation

The classes that can be used in the *Grape* environment are regular C++ classes which are represented by an XML file describing the following properties of a class:

- the class name
- the C++ header file with the class definition
- the class' methods including their parameters (name, type, and value range)
- descriptions of the class and the methods

See the following XML code for a representation of the class `BobbyBoard` which represents a robot controller board (the `DOCTYPE` header is left out for convenience):

```
<qfixClass>
 <header>
   <includefile name="qfixBobbyBoard.h" />
 </header>
 <class name="BobbyBoard">
  <description>
    This class abstracts all functions of the
    BobbyBoard and provides convenient methods
    to access all input and output channels.
  </description>
  <method type="void" name="motor" >
   <params>
    <param type="int" name="index"
     min="0" max="1" />
    <param type="int" name="speed"
     min="-255" max="255" />
   </params>
   <description>
     Sets the motor with the given index to
     the given speed.
   </description>
  </method>
  <method type="int" name="analog" >
   <params>
    <param type="int" name="index"
     min="0" max="3"/>
   </params>
   <description>
     Returns the value of the given analog
     input port.
   </description>
  </method>
 </class>
</qfixClass>
```

A class description file is read when the user switches to the first tab "Classes and Objects" and presses the "Add.." button in order to add a new class.

3.2 Flow-Chart Representation

In Figure 4, we saw a graphical program consisting of commands, an infinite loop and an if-then-else clause. Now, we describe the internal representation of this flow-chart, firstly regarding only the structure of the program without code.

Displayed graphically, the XML structure looks like displayed in Figure 6 (note, that we left out several nodes for better overview).

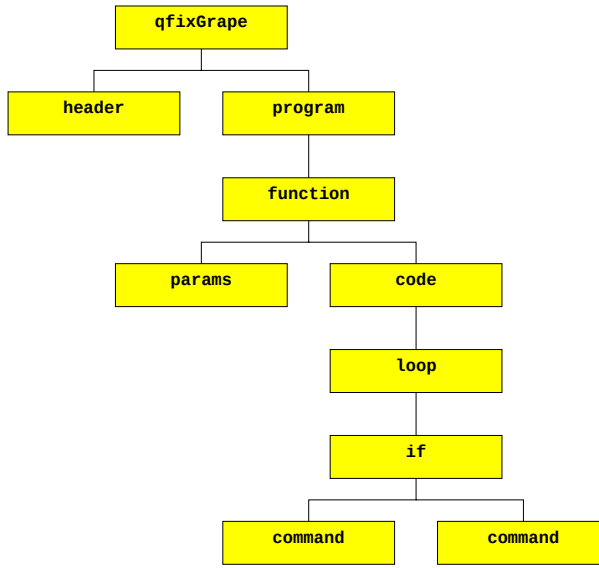


Fig. 6. Rough XML structure of the program in Figure 4

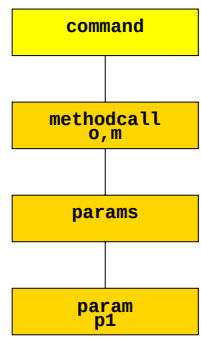


Fig. 7. XML structure of a command with implementation

3.3 Code Representation

When the user uses the dialog-based implementation to fill a graphical element with its semantics, the internal XML representation is updated in the way that the respective graphical element receives a child node describing the implementation of the concept.

For example, when one of the `command` nodes above is filled with the implementation *call object* `o`, *method* `m` *with parameters* `p=`{`p1`}, the command node expands to the one in Figure 7. Thus, one can say that the semantics information is *embedded* into the graphical representation.

4 Code Generation

The XML representation discussed above can be automaticall translated into C++ (or any other object oriented programming language). This is done by a relatively simple mapping schema from XML to C++ which is explained in this section.

As an example, mapping the graphical program from Figure 4, which ist internally represented by the XML code in Section 3.1, produces the following C++ source code:

```
#include "qfixSoccerBoard.h"
#include "qfixLCD.h"

SoccerBoard robot;
LCD lcd;

int main()
{
  while (true) {                    // forever ...
    if (robot.button(0)==true) {    // Button 0 pressed ?
      robot.ledOn(0);               // LED 0 on
    }
    else {
      robot.ledOff(0);              // LED 0 off
    }
  }
}
```

This mapping is performed by applying the following schema:

- First, the `<header>` node is visited:
  - nodes `<includefile>` expand to
    `#include "<name>"`
  - nodes `<global>` expand to
    `<type> <name>;`
- Then, the `<program>` node is visited:
  - nodes `<function>` expand to
    `<type> <name> (<params>)`
    {
      `<code>`
    }

- nodes `<loop>` of type `forever`[1] expand to
```
while(true) {
  <code>
}
```
- nodes `<if>` expand to
```
if(<left-term><operand><right-term>)
{
  <true>
};
else {
  <false>
}
```
- nodes `<command>` of type `methodcall`[2] expand to
```
<object>.<method>(<params>);
```

The user, obviously, is not bored with these mapping details. When having finished his graphical and dialog-based development – i.e. all graphical elements turned to yellow – he simply presses the function key F4 and receives the respective C++ source code in the third tab "C++ code" (see Figure 8).

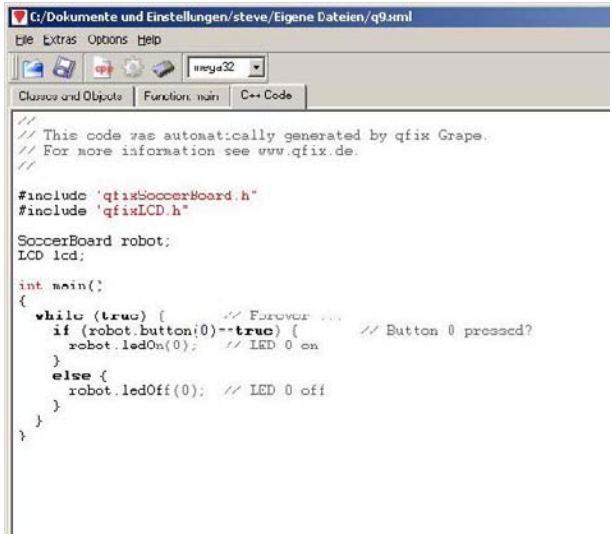


Fig. 8. Respective code in GRAPE

5 Tools

Working with mobile robot kits, what you always have to do despite coding is to compile your program and download it to the robot controller. Thus, it is desirable to have

[1] Other loop types including conditions are possible.
[2] Other command types are possible.

the compile and download tools integrated into the programming environment. However, since there are often multiple controllers that must be supported at the same time, the environment must be kept flexible in order to integrate multiple tool sets according to the required task.

In *Grape*, we use a schema-based configuration concept to import additional tools and being able to activate them by a shortcut key. A tool is represented by a command-line (e.g. a compiler call or a shell script) with additional arguments e.g. for the program name or directory.

in the XML tools configuration file you can add your own tools to the desired schemas and configure a hot-key which shall be used to invoke the respective tool. See the following code for an exemplary configuration file with three schemas and two tools for compilation and download in each schema:

```
<qfixGrapeConfig>
 <schemes>
   <scheme name="mega32">
     <tool name="compile" key="F5"
       command="c:\WinAVR\compile-mega32.bat"
       params="%f" />
     <tool name="download" key="F6"
       command="c:\WinAVR\download-mega32.bat"
       params="%f" />
   </scheme>
   <scheme name="mega128">
     <tool name="compile" key="F5"
       command="c:\WinAVR\compile-mega128.bat"
       params="%f" />
     <tool name="download" key="F6"
       command="c:\WinAVR\download-mega128.bat"
       params="%f" />
   </scheme>
   <scheme name="can128">
     <tool name="compile" key="F5"
       command="c:\WinAVR\compile-can128.bat"
       params="%f" />
     <tool name="download" key="F6"
       command="c:\WinAVR\download-can128.bat"
       params="%f" />
   </scheme>
 </schemes>
</qfixGrapeConfig>
```

In this example, you can see three schemas “mega32”, “mega128” and “can128” which correspond to three different microcontroller settings. For each schema, the tools “compile” and “download” are defined which are bound to the hotkeys F5 and F6. Pressing one of these keys, the respective commmand line is executed with the current filename (see the “%f”) as argument.

Note that this solution is not only flexible enough for compiling and downloading programs for a family of similar robots with similar controllers. It is also possible to

call completely different tools for other robots with different controllers or even to call an independent program, like LaTeX or a compiler for a PC GUI applications.

6 Experiments

In order to demonstrate the feasibility of the *Grape* system, a development task was given to five groups of students. They should developed a graphical program with *Grape* and use the additional tools for compiling and downloading to the robot. The task was to have a small robot equipped with an analog light sensor to follow a black line. At the beginning, the program should wait for a start button. While moving along the line, it should check if a stop button was pressed and in that case stop the motors and wait for the start button again. Figure 9 displays the used robot.

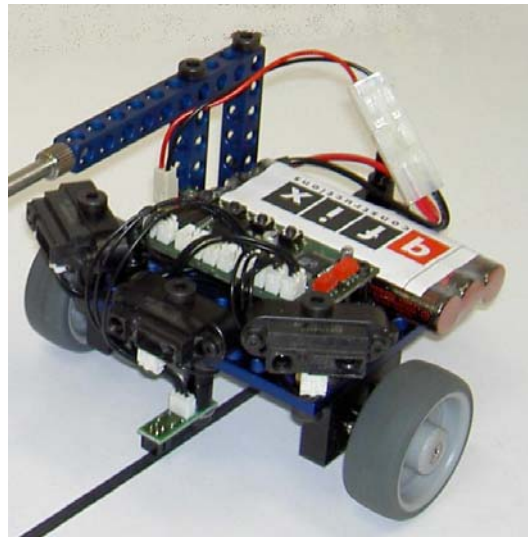


Fig. 9. Differential drive robot with two driven wheels, three IR distance sensors and a analog light sensor for ground line detection

The solution program that all students could easily solve is shown in Figure 10. After waiting for the start button it runs into the main loop which checks if the robot is on the line. If so, a slight left curve is driven, if not, a slight right curve is driven. After that selection, the stop button is checked. If it is not pressed, nothing happens. If it is pressed, the motors are stopped and the program waits until the start button is pressed.

From top to buttom, the program entities are filled with the following semantics:

```
– robot.waitForButton(0)
– forever
– if (robot.analog(0) <= 120)
– robot.motors(-100,255) / robot.motors(-255,100)
– if (robot.button(1) == true)
– robot.motorsOff()
– robot.waitForButton(0)
```

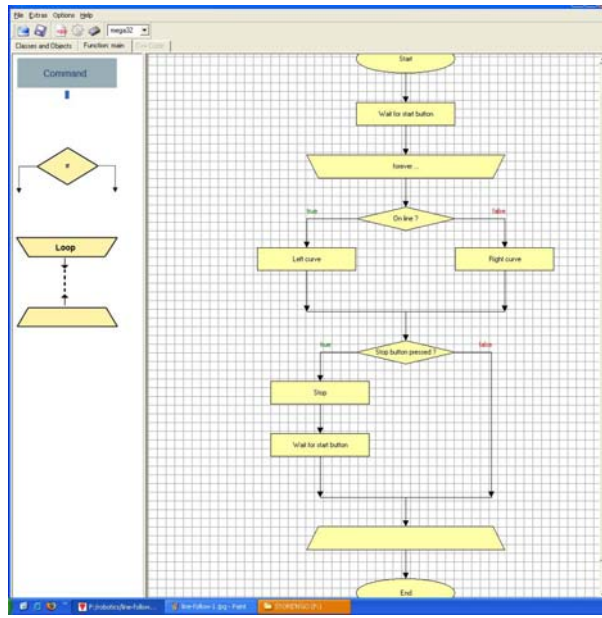


Fig. 10. Solution program for line following

Generating C++ source code for this program generates to following code which can be compiled and downloaded to the robot controller.

```
#include "qfixBobbyBoard.h"

BobbyBoard robot;

int main()
{
  robot.waitForButton(0); // Wait for start button
  while (true) {          // forever ...
    if (robot.analog(0)<=120) {
                          // On line ?
      robot.motors(-100,255);  // Left curve
    }
    else {
      robot.motors(-255,100);  // Right curve
    }
    if (robot.button(1)==true) { // Stop button pressed ?
      robot.motorsOff();         // Stop
      robot.waitForButton(0);    // Wait for start button
    }
  }
}
```

Playing around with the threshold value (here: 120) the robot is indeed able to follow a black line by wobbling along the black/white edge of the line. The experiment showed that the students could easily use the Grape tool for developing such behaviour.

7 Conclusion

We presented *Grape*, a graphical programming environment for object oriented programming. The main goal of *Grape* is the students to understand the *concepts* of programming languages including commands or statements, loops, if-clauses, etc. With *Grape*, the student can play around with these concepts in a graphical way.

To go further into programming, the student has to define the *semantics* of each graphical entity with respect to his program in mind. From this, *Grape* can generate C++ code which can also be studied in order to learn the mapping between the flowchart and the source code. This is enforced since the user can immediately see how a change on the graphical side affects the generated code.

Thus, *Grape* is a powerful tool for educational purposes and has a good chance for being used in other areas despite programming small mobile robots, too.

8 Open Work

Note that currently, only a subset of object-oriented programming is supported by *Grape*, namely the usage of existing predefined classes. It is not supported to define own classes within *Grape*, yet. However, for a novice of programming, this is sufficient.

The next programming concepts to be included into *Grape* are subroutines/functions and variables. Subroutines and functions can be easily integrated since the function concept is already defined (see the main function in the examples above). The representation of variables is still open.

Especially for robotics applications, the concept of multiple processes/threads is important, since the robots can be programmed much easier using a behaviour-based approach including multi-threading.

Acknowledgements

This work is sponsored by qfix robotics, Germany (`www.qfix-robotics.com`). We also thank Bostjan Bedenik who mainly implemented the Grape software.

References

1. Enderle, S.: The robotics and mechatronics kit qfix. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006: Robot Soccer World Cup X. LNCS, vol. 4434, pp. 134–145. Springer, Heidelberg (2007)
2. RoboCupJunior, `http://www.robocupjunior.org/de`
3. Baum, D., Gasperi, M., Hempel, R., Villa, L.: Extreme Mindstorms – An Advanced Guide to LEGO Mindstorms. Apress (2000)
4. Trolltech: Qt c++ library, `http://www.trolltech.com`

Robotic Exploration of Planetary Surfaces – Rover Technologies Developed for Space Exploration

S. Klinkner[1,2], C. Lee[1], H.-P. Röser[2], G. Klingelhöfer[3], B. Bernhardt[1], I. Fleischer[3], D. Rodionov[3,4], and M. Blumers[3]

[1] von Hoerner & Sulger GmbH, Schlossplatz 8, 68723 Schwetzingen, Germany
klinkner@vh-s.de
[2] Universität Stuttgart, Institut für Raumfahrtsysteme, Stuttgart, Germany
[3] Johannes Gutenberg Universität Mainz, Institut für Anorganische und Analytische Chemie, Mainz, Germany
[4] Space Research Institute IKI, Moscow, Russia

Abstract. Mobility is a key feature for any science mission and for space exploration in general. Missions with mobile systems provide a much wider spectrum of outcomes by employing a higher number of samples within an increased area of exploration. The additional degree of freedom of a rover in comparison to a lander or even a robotic arm allows the mission to be flexibly adapted to the landing site as it is encountered.

Nevertheless, rover vehicles developed for the exploration of planetary surfaces are extreme complex systems, which have to be specialised for the environmental conditions they are dedicated for. With the variation of the environmental conditions on missions to different target planets, the requirements are varying for the landing system, the rover as well as the payload.

Since 1989 the company von Hoerner & Sulger is doing research in the field of robotic systems and planetary exploration. Given that, the company is in the mean time well situated in the development and manufacture of rovers and established a good cooperation with academic institutes. The company gained the experience to develop the matching rover chassis for a variety of mission scenarios:

The Nanokhod rover is a small mobile scientific platform, designed to transport a package of scientific instruments and to carry out in-situ measurements of rocks and small craters in the vicinity of the landing point. The Microrover has a volume of 160×65×250 mm, it weighs 3,2 kg including a payload mass of 1 kg and has a peak power need of max. 5 W. The Nanokhod is a tethered system that uses the Lander for power supply and as a data relay to Earth. The Nanokhod has recently been designed to withstand the demanding requirements of a flight model on a mission to Mercury. Based on this design, an engineering-level hardware model was built which is suitable for environmental testing, preparing the rover design for a variety of possible future missions.

The Solero rover is an innovative Minirover concept, designed for regional exploration of a planetary surface. The vehicle has a passive chassis

A. Gottscheber et al. (Eds.): EUROBOT 2008, CCIS 33, pp. 193–206, 2009.

concept with exceptional climbing abilities, which provides the ability to adjust to all kinds of terrains and thus minimises control needs.

The company vH&S is leading already the second ExoMars rover chassis breadboard design and manufacturing activity, which is part of the rover development for the first European Mars rover. The second breadboard is designed and built by vH&S GmbH in collaboration with DLR and two Swiss collaborators. The ExoMars rover for the ESA Cornerstone Mission Aurora, will be a mobile Laboratory having an Exo-biology Payload (Pasteur), including a geochemical package, and carrying a drill that is reaching probes up to a depth of two meter.

This paper describes the gained experiences and most important aspects of a rover design for the purpose of planetary exploration. In addition it presents the newest designs and the manufactured models in relation to their missions ...

1 Introduction

von Hoerner & Sulger GmbH (vH&S) is a small scale enterprise (SME) that is working since 1971 successfully in the field of space exploration. This covers scientific space instruments, rocketborne systems, cameras and sensors for space applications as well as robotic systems for planetary exploration. The areas of operation of vH&S include concept finding, feasibility studies, development, fabrication, and qualification of space systems for applications in extraterrestrial missions.

Since its foundation the company has produced more than 10 flight-qualified scientific space instruments. Famous examples from the past are the first-ever mass spectrometers for cometary dust, PIA and PUMA (1 & 2), which were built between 1981-1984 for ESA's Giotto mission and for the Russian Vega 1 & 2 missions. These three experiments were the first and only ones that met comet Halley in 1986, giving insight into its chemical composition. After these brilliant results, vH&S became the prime company to provide mass spectrometers for both NASA and ESA missions.

This is only one example how vH&S has earned profound experience in the space mission activities and a worldwide high profile reputation in the space business, both in prime and subcontractor roles.

Based on this experience in the sector of space missions the company has expanded its portfolio in the late 1980's with the development of systems for in-situ planetary exploration. Since vH&S is active in this field it was involved in the development of various rover chassis concept matching for distinct mission scenarios. These chassis concepts cover the rover categories from a highly-integrated Mircorover to a big rover with a mass of up to 220 kg.

The mission scenarios vary from very hot to very cold environmental conditions, for targets with and without atmosphere and for planetary surfaces which are rough and rocky or smooth and covered with fine dust particles called regolith – namely environments on the planets Mars and Mercury, or our Moon. The most prominent example that made vH&S a leading European name for

robotics is the *Nanokhod* Microrover, with a mass of only 3,2 kg including 1 kg of scientific payload mass. On the other end of the rover size scale vH&S is involved in ExoMars activity that is building the rover for the European Mars mission being part of the ESA Aurora programme.

2 Mobility for Space Exploration Missions

Current trends in space exploration aim for mobile systems[1]. In order to fulfil the whole range of scientific interests and to have a well-funded scientific examination of a planetary surface more than one sample has to be considered. This requires a mobile system to reach the different areas of interest.

A mobile system means that either several samples of different materials have to be collected and transported to the instrument or the instrument itself has to be moved to the different sample sites. A wide spectrum of results gives a wider view of the explored target; either way though requires the ability of movement. The application of a rover which transports instruments and carries out in-situ analysis has the advantage that the range of the rover system is farther.

The mobility aspect of the rover makes it possible to explore with a single space system a whole region of up to some tens of km around a landing point depending on the implemented system. This regional exploration provides a deeper knowledge and allows a more general characterisation of certain areas of a planetary surface. The mobility of such a system gives the scientific explorers the freedom to reach and examine specific points of interest. This offers also the opportunity to carry out a systematic exploration of the region. Depending on the system used and on the environmental conditions encountered this can even compensate in some cases the uncertainty of the landing ellipse.

For the mobility in space infrastructure the following development can be observed. For nearby planetary bodies which have been explored before, the implemented mobile systems are developed with an increasing reach ($m > km$). While for bodies which require a long transfer and which are examined for the first time, the focus is rather on a system with very low mass. Mission success in this case is to gain in-situ analysis in the near surroundings of the landing point.

A variety of rover systems have been developed in the previous years in order to accomplish all needs for the exploration of target surfaces. These systems differ in their range, complexity, their demands on control, power needs etc. Thus a rover system can be adjusted to the expected mission scenario.

3 Rover Technology for Science Missions

There are several performance drivers for rovers used for space applications. One of the main drivers is the perception capability. This means either their ability to recognise and understand the environment they move through, or the identification of the targets they take samples of. The rover has to percept the near surroundings, by seeing it, processing the seen information and than analyse

what it means for the further rover activity. This routine has to be done for each segment of movement as well as for the taking or measuring of samples.

A further driver closely related to this is the mobility capabilities. This is necessary in order to reach scientific targets, when moving through all kinds of terrain the rover might encounter. But the mobility aspects become even more important for the processing and handling of samples to prepare them for analysis. A rover has to enable a close contact between an instrument and the sample and it may grasp and collect samples; in some cases it might even have to dig, grind or drill, to prepare the sample for the scientific analysis with certain instruments.

Especially for long distance mission targets with long signal delays, a further driver for space rovers is their operational capabilities. For rover missions, there is a high demand for autonomy, in order to cope with hard real-time situations, as well as to not put any additional constraint on the mission duration because of signal delays. Furthermore the need for a high level ground interaction is also always a cost factor for the mission. The demand for autonomy includes also autonomous decision making in an unknown and unstructured environment. The rover system has to be able to cope with unexpected situations and has to provide solutions for any contingency. Autonomous operation implemented in space rovers also simplifies the payload (P/L) operations as well as the scientific target selection.

A rover system equipped with a high number of sensors for scientific analysis can utilise these capabilities also for the planning of single operations. Like that the rover can use instruments facilities for mission planning aspects. This use of synergy effects enhances the system efficiency, by reducing any additional instrumentation and thus system overhead by a careful integration of the instrumentation into the system.

Autonomy makes the rover adaptable to a variety of situations and missions and provides the necessary flexibility which the systems needs in a completely unknown environment. An optimised space exploration approach of a rover system development can be achieved, by developing the rover together with the scientific payload, in order to plan for synergies.

4 Microrover Nanokhod for the Mercury Surface Exploration

Since 1992 the company von Hoerner & Sulger GmbH has the leading role in the development history of the Nanokhod rover. The rover is based on an originally Russian design, but has since undergone a significant development to now provide a practical flight implementation. Part of the devlopment were for example the accommodation of scientific payload instruments and the building of prototypes to test the mobility performance, like step and slope climbing abilities, and sealing concepts to prevent the rover inside from regolith.

The latest development is now a Nanokhod design which is able to cope with the challenging requirements for a flight model on the Mercury surface. A hardware model of this design is realised for environmental tests such as vibration,

shock and thermal vacuum to the extreme requirements of a Mercury mission profile. The project to develop a mature rover for the Mercury environment is based on the ESA Cornerstone mission BepiColombo.

Initially, the BepiColombo mission was still including a Surface element, and the Nanokhod rover was selected to be part of it as a mobile scientific platform - the Mercury Robotic Payload (MRP). Unfortunately the Surface element, was cancelled from the BepiColombo mission and with it the flight opportunity of the rover. Nevertheless, it was decided to continue with the development to prepare the rover for future missions by solving detailed technical problems of a real implementation and thus gaining a better understanding. Also because the technology developed for the challenging nature of the Mercury environment is considered to be applicable with moderate modifications on a variety of other planetary bodies, with and without atmosphere, like Mars or Moon.

Small systems are required due to a limitation of financial resources and energetically more demanding missions. As a small and mobile system the Nanokhod thus fulfils two trends - mobility and low mass - for future missions, with additionally a very good payload mass to system mass ratio. And now with the design meeting the demanding flight model requirements for the conditions of a Mercury mission the rover is very well prepared for a variety of future mission scenarios.

4.1 Mercury Mission Profile

After some initial studies of the rover mission under day- as well as night side conditions of Mercury, the landing site was decided to be on the night side of the planet, where the absence of the sun's radiations reduces the thermal range and makes the mission feasible. Despite this the environment remains severe, coupling a high vacuum with surface temperatures estimated to −180°C. However, the night side landing site causes the disadvantage that all energy must be supplied chemically.

The MRP Nanokhod shall be able to move across the Mercury surface which is expected to consist of fine regolith similar to the Moon surface with a speed of 5 metres/hour and to negotiate steps of 10 cm height and trenches of 10cm width. The mission foresees one in-situ analysis per Earth day with each of its three instruments: Microscopic camera (MIROCAM), APXS and Mössbauer Spectrometers. The rover shall operate near-autonomously due to a single communication period once a day.

4.2 Design Drivers Resulting from the Mercury Mission Scenario

This mission profile induces as main design drivers mass and volume for the whole landing system. With an interplanetary journey between Earth and Mercury that requires a high amount of fuel to reach and enter the Mercury orbit, and with a landing scenario which has to rely only on chemical propulsion, it becomes obvious that every kilogram to land on the planet is very cost intensive. This is why the mass and the volume of the Lander and thus the rover have to be

reduced to a minimum in order to make the mission feasible. Opposing this, the rover has to be designed to be robust enough to cope with the vibration and shock environment of such a landing scenario. In order to reduce the amount of chemical propulsion to a minimum the landing shock is expected to be 200 g for a duration of up to 20 ms.

Also related to the mass limitation issue is the energy consumption of the rover. The power is provided by the Lander batteries and with a given mission duration, the consumed energy relates directly to the battery capacity and thus to the battery mass. This makes the energy consumption to be the next significant design driver of the rover system.

During the mission, the rover has levels of different energy consumption, depending on the rover activity. Based on an analysis of the mission and the duration of the different activities the energy consumption of the rover was optimised, in order to decrease the overall power needs as far as possible. In the case of the Nanokhod for the Mercury mission the energy consumption has to be as low as possible for the measurements phases as well as for the non-active periods of the rover.

A low power consumption of the system is also realised by implementing a passive thermal control and mechanisms which do not require power to maintain their state. Passive thermal control in this instance means that the rover does not attempt to maintain its temperature to a set level but the rover is allowed to heat up and cool as defined by the environment, the mechanical design and selected materials and finishes. As there may be extended periods when the rover is powered off, all components have to function from the surface temperature upwards.

4.3 Overview of the Nanokhod Design

A brief overview of design is given in the following paragraph. For a more detailed description of the rover design including its instruments please refer to [2]. The main components which are generic to all Nanokhod rovers are:

- Two locomotion units (LU) enclosed by walls and the driven caterpillar tracks which provide the method of locomotion
- The tether unit (TU) which rigidly attaches both locomotion units and contains the spools from which the tether wire is deployed
- The payload cabin (PLC) containing the scientific payload instruments
- Arms connecting the PLC to each of the LU giving the PLC two degrees of freedom allowing the instruments to place next to sample sites and for the PLC to act as an extra limb for negotiating obstacles.
- Four internal drive units used to drive the caterpillar tracks and position the arms relative to the LU and the PLC.

The main components are identified on the current design in Figure 1. The overall structure has been upgraded to withstand the rigours of vibration and shock with the inclusion of four rigid yokes in the LUs. Analyses have been performed on all components to ensure that they are compatible to the mechanical

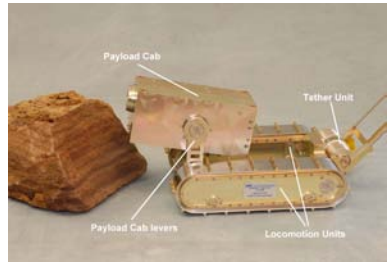


Fig. 1. Main components of the Nanokhod rover

and thermal environment. A completely new drive system based on a similar concept for all four drives within the rover has been implemented, which was developed in a close cooperation with the Harmonic Drive AG. Due to the high vacuum environment it is not possible to use standard DC motors for extended durations and so the Faulhaber AM1020 stepper motor was selected as the motor for the drives. Dicronite dry lubricant was used on the Harmonic Drive and the crown gear, the application of which had been tested at Harmonic Drive AG. The motor and the planetary gearhead were supplied pre-lubricated with MoS_2 by the manufacturer.

Electronically the system is has been partitioned into a number of nodes each of which perform a distinct function. Power for the drive system nodes is supplied by a 28 V line which is also controlled by the tether interface node. When the power is removed from this line all drive unit nodes are powered off, minimising the power consumption during instrument operations.

4.4 Conclusions for the Microrover

In table 1 the main parameters of the highly integrated Nanokhod are listed. Although a mission to the Mercury surface is currently unlikely, the Moon has now become very popular in consideration for proposed visions, but also other

Table 1. The main dimensions of the Nanokhod rover

Nanokhod properties	
Mobility:	
Overcome obstacles up to:	0,1 m
Locomotion speed:	2,7 m/h
Electrical Power:	
During movement:	5.7 W peak
Other modes:	1,3-3,4 W
Rover Dimensions	
Mass including P/L and module on Lander:	$< 3{,}2$ kg
Size:	$< 250 \times 160 \times 65$ mm

targets stay possible. The night side environmental conditions of the Moon are similar to Mercury and would allow easy adaptation of the current concept. For a dayside landing the new Nanokhod model is still very applicable although new attention would need to be given to the thermal design, having a thermal model of the rover already on hand. For missions with greater demand for mobility performance such as obstacle negotiation, rover range and greater payload capacity, the MRP Nanokhod rover provides an ideal design baseline.

The technical challenges to miniaturise a practical flight implementation to 3,2 kg have been overcome and with a relaxation of volume and mass constraints the design allows a safe and quick realisation of the new mission scenarios. Possibilities of modification include replacing the tether system with internal power provision and RF communication as well as upgrading the navigational capabilities.

The MRP Nanokhod rover is a huge advance towards a practical flight model of a highly integrated miniaturised mobile payload for planetary exploration despite limited resources that were available. Solutions for open issues have been implemented allowing the rover to be subjected to the currently ongoing thermal vacuum testing and further environmental tests.

5 Minirover Concept Solero for Regional Exploration

The mobility requirements for a chassis of the minirover type call for a typical travel distances of several kilometres. The Solero combines this with an average locomotion speed of up to 100 m/h (EXOMARS). It is evident that navigation and control of such a vehicle needs another approach as compared to local mobility systems, which remain in the vicinity of a stationary lander. Autonomous navigation is required to enable at least the typical operational increment between two consecutive ground control interventions, which is one sol for the Mars mission baseline of this design. Besides the principal autonomy requirements, the inherent stability of the locomotion concept is another feature driving the control system. If a locomotion system can cope with a large variety of terrains without active control, the whole control system overhead can dramatically decrease.

The Solero concept follows this approach; a detailed description on the activity is given in [3]. It uses an innovative locomotion concept originating from EPFL (Ecole Polytechnique Fdrale de Lausanne), which is suited to provide a high degree of design-inherent stability with respect to locomotion in rough terrain.

5.1 Solero Requirements

The mission scenario for the Solero rover is regional exploration for a geochemical mission. The payload for the Solero rover was chosen with reference to the *Nanokhod* microrover. It consists of an Alpha Particle X-Ray Spectrometer (APXS), a Mössbauer Spectrometer (MIMOS) and a microscopic camera (MIROCAM). Accordingly, the top-level requirement for the rover system is to transport and operate these instruments for in-situ geochemical exploration.

The minimum on-surface travel distance for the Solero locomotion is specified to 10 km, with an effective locomotion speed of 220 m per day. In addition the rover system must be able to withstand Mars environmental conditions (e.g. outside ambient temperatures between −100°C and +30°C, Mars atmosphere, dust).

The Solero has a max. mass of 10 kg, an autonomous power supply using solar power collection, and – similar to the Nanokhod – has no active thermal control.

5.2 Solero Flight Concept for Mars

When defining a complete rover system flight concept, a variety of system elements have to be considered, defined and adjusted in an iterative and heuristic process. The design drivers for the Solero system design are:

- *Rover configuration, operations and control:* the rover must be able to carry out operations autonomously for a duration of at least one day. This implies the capabilities to autonomously solve the problems of rover localisation, path planning, and trajectory execution.
- *Power provision, storage and control:* the system has to work with a minimum electrical storage. As a consequence the diurnal power profile is driving the operational capabilities of the rover, i. e. driving, payload operation, telecommunication sessions, hibernation.

Further subsystems need of course to be assessed for the complete system concept, they are however not the strongest drivers.

5.3 The Solero Chassis Concept

Using a rhombus configuration, the rover has one wheel mounted on a fork in the front, one wheel in the rear and two bogies on each side. The parallel architecture of the bogies and the spring suspended fork provides a high ground clearance while keeping all 6 motorised wheels in ground contact. This ensures excellent climbing capabilities.

The front fork has two roles: its spring suspension guarantees optimal ground contact of all wheels at any time and its particular parallel mechanism produce a passive elevation of the front wheel if an obstacle is encountered. The front wheel has an instantaneous centre of rotation situated under the wheel axis that is helpful to get on an obstacle.

The bogies provide the lateral stability. To ensure similarly good ground clearance and climbing capabilities, the virtual centre of rotation of the bogie is set to the height of the wheel using a parallel configuration.

The steering of the rover is realised by synchronising the steering of the front and rear wheel and the speed difference of the bogie wheels. This allows for precise manoeuvres and even turning on the spot with minimum slip.

The payloads as well as rover subsystems can be accommodated in the central body of the rover. The flight concept of the Solero rover is equipped with a solar panel, scientific payload and two navigation instruments: an omni-directional camera and a stereovision camera for 3D obstacle recognition.

Fig. 2. The Breadboard of the Solero Minirover

5.4 Conclusions for the Solero Minirover

The development of the Solero chassis was carried out within a very low cost technical demonstration activity (TDA), leading to a first system conceptual design and a development model (breadboard) to demonstrate aspects of locomotion, payload accommodation, and power provision. Although all system areas have been addressed in a first instance, some issues need thorough design work in order to establish a detailed design suitable for a flight rover. The technical parameters of a Solero flight model are given in Table 2.

Table 2. The Solero flight concept is designed with following technical attributes

Solero Properties	Target Value
Total Mass:	10 kg
Overall Dimension:	ca. 880×600×400 mm
Payload Mass:	1 kg
Solar Power:	16 W typical daily peak power
Mean Locomotion Power:	8 W
Navigation:	Autonomous Navigation up to 1 km distance by 3D obstacle recognition and negotiation
Telemetry:	Close to the Beagle 2 design: direct telemetry link to orbiter total data size ca. 2 Mbit per day
Locomotion Speed:	20 cm/s

This concerns in particular the control system. The current Solero model has only a very simple control functions implemented. It is therefore proposed to bring all subsystem areas to a detailed design level in a first development step called *Detailed System Design*. In a second step, the design, development and manufacturing of the EQM (Model) and FM (Flight Model) rovers could be implemented efficiently and with relatively low risk.

The positive results of the Solero activity recommend the Minirover concept as a promising solution for a future rover mission as well on Mars as on other

planetary surfaces. It also recommends itself as a good base platform for various kinds of payloads which require an excellent mobility.

6 ExoMars Breadboard for European Mars Mission

Bridget is the first full size ExoMars breadboard rover which was commissioned by Astrium UK Ltd. and built by a consortium of companies and institutes led by von Hoerner & Sulger GmbH, [4]. In the mean time the second ExoMars chassis breadboard was designed and built under the lead of vH&S together with institutes and the Swiss company Oerlikon Space.

The primarily intention of a full scale rover is to investigate the capabilities of the suspension and traction system for use on a future ExoMars rover. It is also used to study additional rover system components such as the navigation system or a drill. The scale of the rover chassis allows valuable experience to be gained not only in the performance of the systems but also from practical aspects such as accommodation of components and AIV aspects related to the handling of a full scale vehicle.

The breadboard chassis design reflects the shape and form of the proposed ExoMars rover at the time. However, pragmatic decisions in the use of materials and off-the-shelf hardware had to be made in order to keep to a reasonable cost for the rover whilst maintaining flexibility for its future use.

6.1 ExoMars Breadboard Design

The main design requirements of the rover chassis are based on the output from Astrium UK's Exomars/Pasteur Phase A mission study during which von Hoerner & Sulger GmbH had led the chassis study team.

The original phase A study proposed a 6 wheel rover with a RCL-C type configuration (based on the ESROL study) as the mission baseline, as it was a good compromise between complexity (and mass) and its performance in terms of stability and body movements during obstacle negotiation.

However, further comparison with miniature hardware rover models conducted for Astrium UK by ETHZ identified an undesirable characteristic which under certain conditions the outside wheels would effectively lift the centre wheel off the ground. A more complex control algorithm could be used to provide a solution to this problem but this is highly undesirable due to extra risk and resources it would entail. For this reasons plus the fact that RCL-E offers a reasonable mass advantage in a flight design, it was decided to proceed with a RCL-E type chassis configuration for the first breadboard design. This consists of a main body, two parallel bogies in the front and one lateral bogie in the rear

For the first breadboard the focus was placed onto an ideal passive suspension for obstacle negotiation in which there is only a minimum of longitudinal displacement of the wheels positions when the rover negotiates an obstacle. Longitudinal displacement of the six wheels relative to the centre of mass will cause the loads on each wheel to vary from the ideal situation where all wheels are

(a) Breadboard *Bridget* - 2006

(b) Breadboard Phase B1 - 2008

Fig. 3. Development of two full-size ExoMars rover chassis Breadboards

equally loaded. This however, ignores the effect of the rover body inclination as it changes from horizontal. The longitudinal displacement was thus minimised for the Bridget Breadboard by optimising the geometry of the parallel bogies for the step range the rover had to cover, see figure 3(a).

The drawback of such an ideal passive suspension unit is the additional mass, caused by the parallel beams. With the main focus on minimising mass for the second breadboard, the optimisation in the longitudinal displacement was given up. The new chassis was thus a simplified Concept E rover using simple bogies instead of parallel bogies. Only for the rear bogie the breadboard foresees the possibility to study the effects of both the parallel bogie and the simple bogie by applying or removing a locking device, see figure 3(b).

During the Phase A Study of the rover chassis it had been highlighted that the use of a flexible wheel would be beneficial to vehicle performance and efficiency in several ways:

- For a properly designed flexible wheel, the larger (and longer) ground contact footprint will lead to less slip and higher thrust as compared to a similarly sized rigid wheel, resulting in better drawbar performance (and thus improved slope climbing capability)
- Overall motion resistance of a properly designed flexible wheel is lower than that of a similarly sized rigid wheel, resulting in smaller losses or, equivalently, a better mileage (energy to be spent per distance driven).

This is why both breadboards have been equipped with flexible wheels.

For both breadboards, the main body accomplishes the chassis concept by providing the linkage in-between the bogies. In both cases the body was made of profile frame in order to easily integrate additional systems and payloads.

6.2 Conclusions for the ExoMars Breadboards

Table 3 compares the main parameters of the two ExoMars breadboards, which have been designed so far by vH&S. Since delivery Bridget has been used by Astrium (UK) for testing both in Tenerife and the UK, after the first preliminary tests at vH&S. The second breadboard is currently extensively tested at the

Table 3. Main attributes of the ExoMars Breadboards

Properties	Breadboard Bridget	Breadboard Phase B1
Overcome obstacles up to:	0,3 m	0,25 m
Locomotion speed:	100 m/h	108 m/h
Power:	60 W peak, Battery	48 W Battery
Breadboard Mass:	115 kg,	80 kg
Payload Capacity:	185 kg	40 kg
Size:	1600×815×1000 mm	1650×1000×1200 mm

company Oerlikon Space, where it was delivered to after completion of the assembly in Schwetzingen by vH&S. Valuable experience has already been gained during the project which will be put to good use for the next phases of the ExoMars project. Further field trials and performance test undertaken will build further on this experience. Both rovers have appeared extensively in press and television, being the first visible steps of the European rover mission to Mars.

7 Conclusion

The development and design of rovers is a multi-disciplinary task due to the complexity of a space system. Such a system has one common goal, which is to transport and operate scientific instruments on a planetary surface in order to collect data and provide knowledge of our solar system. Nevertheless, the space system rover consists of several different components, like drive units, the control unit, the chassis etc., which can be defined as subsystems. These subsystems have again their own properties and functions and in-between them there are a high number of interactions and exchanges. This interaction between the different elements requires an overall design, which allows the integration of all elements and subsystems and thus maximising the possible synergies. In addition, the

Fig. 4. The magnitude variety of vH&S rovers

definition of such a subsystem as well as the input and output values can change depending on the applied point of view.

Applying to a space system additional requirements imposed by the mission scenario, the encountered environment or the influences of the rover size onto the chassis, it becomes obvious that the development of a matching rover is not only the application of size factor.

Quite the contrary, each rover design has to be considered as a "network-type" problem. For all missions a new set of requirements has to be developed and designed to. This means the design follows a heuristic, non-sequential methodology to meet the demanding requirements any space mission for planetary exploration imposes.

Nevertheless vH&S has proven a large experience in this field with the rover designs on hand. The company has solutions available for issues concerning small-scale systems, minirover scale as well as large rover systems, see figure 4.

References

1. Klinkner, S., Laufer, R., Graf, T., Nagy, M., Roeser, H.-P.: Lunar Exploration using Small Satellite with Micro Rover Technology. In: 10th ISU Annual International Symposium, Strasbourg, France (2005)
2. Schiele, A., Romstedt, J., Lee, C., Henkel, H., Klinkner, S., Rieder, R., Gellert, R., Klingelhöffer, G., Bernhardt, B., Michaelis, H.: The new NANOKHOD: Engineering model for extreme cold environments. In: 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space, Munich, September 5-8 (2005)
3. Michaud, S., Schneider, A., Bertrand, R., Lamon, P., Siegwart, R., Van Winnendael, M., Schiele, A.: SOLERO: Solar Powered Exploration Rover. In: ASTRA 2002, Nordwijk (November 2002)
4. Lee, C.G.-Y., Dalcolmo, J., Klinkner, S., Richter, L., Terrien, G., Krebs, A., Siegwart, R., Waugh, L., Draper, C.: Design and manufacture of a full size Breadboard ExoMars Rover chassis 2006, Nordwijk (November 2006)

Author Index