Pablo Gonzalez de Santos
Elena Garcia
Joaquin Estremera

Quadrupedal Locomotion

An Introduction to the Control of Four-legged Robots

Springer

Quadrupedal Locomotion

Pablo Gonzalez de Santos, Elena Garcia
and Joaquin Estremera

Quadrupedal Locomotion

An Introduction to the Control of Four-legged Robots

With 135 Figures

Springer

Pablo Gonzalez de Santos, PhD
Elena Garcia, Dr. Eng
Joaquin Estremera, PhD

Instituto de Automatica Industrial
Ctra. Campo Real, Km. 0,2
28500 Arganda del Rey
Madrid
Spain

British Library Cataloguing in Publication Data
Gonzalez de Santos, P. (Pablo Gonzalez de Santos)
Quadrupedal locomotion : an introduction to the control of four-legged robots
1.Mobile robots 2.Robots - Control systems 3.Robots - Motion
I.Gonzalez de Santos, Pablo II.Garcia, Elena III.Estremera, Joaquin
629.8'932
ISBN-13: 9781846283062
ISBN-10: 184628306X

Library of Congress Control Number: 2006923486

ISBN-10: 1-84628-306-X e-ISBN 1-84628-307-8 Printed on acid-free paper
ISBN-13: 978-1-84628-306-2

Printed in Germany

9 8 7 6 5 4 3 2 1

Springer Science+Business Media
springer.com

To Pili, Jose Pablo and Javier; they form my personal stable tripod.
Pablo Gonzalez de Santos

To Luis Angel and Irene; you give direction to the walk of my life.
Elena Garcia

To Beatriz, to whom I will run back soon.
Joaquin Estremera

Preface

Legged robots have proven to be a promising locomotion system, capable of performing tasks that conventional vehicles cannot perform. Even more exciting is the fact that this is a rapidly developing field of study for researchers from a variety of disciplines. Over the past three decades, legged locomotion technology has been developed all over the world, resulting in the invention of many important new machines and methods. However, only a few books have been published on the subject of multi-legged robots. The main objective of this book was to explore some of the major issues that the authors have been analyzing over the past ten years. A second objective was to write a book that only encompasses quadruped locomotion, the first specialized book on this topic. The book is divided into two parts: Walking Measurements and Algorithms, and Control Techniques. The first part is devoted exclusively to the theoretical aspects of quadrupeds. The first chapter is an introduction to the historic development of multi-legged robots, highlighting their advantages and disadvantages, main features, and potential and actual applications, as well as discussing basic concepts and the trade-off between quadrupeds and hexapods. Finally, new and traditional stability measurements and gait generation algorithms for quadrupeds are explained. The second part of the book deals with general design and control algorithms (kinematics and dynamics) and techniques aimed at improving the main features of robots, such as speed and ground detection, interfaces, *etc.* These techniques are used for legged robots in general, but this book applies them specifically to quadruped robots. The material presented in the book is the result of a true group effort involving many different individuals. We are especially grateful to the members of the Industrial Automation Institute (CSIC) who provided many valuable contributions to the machining and maintenance of the SILO4 walking robot. We would also like to thank our colleagues at the Department of Automatic Control, who provided direct assistance with the experimental work. We are deeply indebted to the department head, Dr. M. Armada, for his unconditional support. We would like to express our gratitude for the contributions of Dr. M.A. Jimenez. She could have been one of the authors of this book,

but instead she decided to follow her husband on another exciting adventure in The Netherlands. The support of Dr. J.A. Galvez, who created the main mechanical design of the SILO4 walking robot, is also greatly appreciated. Finally, we would like to acknowledge the financial support of the Spanish Minister of Education and Science. Most of the results contained in the book were funded by grants from this institution (ROB1990-1044-C02-01, TAP94-0783, TAP1999-1080-C04-01, DPI2001-1595 and DPI2004-05824). The second author also gratefully acknowledges funding from the European Social Fund for her CSIC-I3P contract.

Industrial Automation Institute - CSIC
Arganda del Rey, Madrid,
May 15, 2005

Pablo Gonzalez de Santos
Elena Garcia
Joaquin Estremera

Contents

Part I

Walking Measurements and Algorithms

1

Walking Robots

1.1 Introduction

Nature is made up of wonderful creatures and human beings have always been curious, interested or excited about their behavior and have tried to understand, enjoy or imitate them. Emulating live creature performances is an attractive idea but extremely difficult to accomplish. Generally, we have to settle for building some simple apparatus that can imitate only minute aspects of what we ordinarily sense from our surroundings: creatures that can see, smell, manipulate, and ... walk.

This book is devoted to the imitation of walking by the development of machines with legs, also widely known as robots[1]; in other words, mechanical systems that move themselves by using devices that resemble legs. Based on the number of legs the robot has, there are bipeds like humans or birds, quadrupeds like mammals and reptiles, hexapods like insects, and octopods like spiders. Robots with one (Raibert's hopper (1986)), three (OSU Triped (Berns, 2005)), five (Hitachi hybrid robot (Todd, 1985)), eight (ReCUS (Ishino *et al.*, 1983)) or more legs (Nonaped, (Zykov *et al.*, 2004)) are unusual, but not impossible. This book is focused in particular on quadrupeds walking under static stability, *i.e.* machines with four legs that exhibit some special features and use specific control algorithms that we will highlight throughout the following chapters. Nevertheless, mention will inevitably be made of other multi-legged robots (monopods and bipeds will not be considered in this book) and related features when required.

The first documented walking mechanism appeared in about 1870 and was based on a four-bar mechanism invented by the Russian mathematician P. L. Chebyshev as an attempt to imitate natural walking (Artobolevsky,

[1] This chapter does not discuss the issue of whether a walking machine directly controlled by an operator is a robot or not. In any event, the sequence of leg motions is performed automatically; therefore the terms machine, vehicle, and robot will be used interchangeably.

1964). Some extraordinary machinery was later developed for leisurely use, and around 1893 the first patents for legged systems were registered with the US Patent Office.

Some decades later, in around 1940, researchers began to consider the real possibilities of using legged robots for practical applications. As usual, military applications came first. The UK and the US military sponsored important projects to study the applications of legged mechanisms as war machines. Many activities were later envisaged as prospective applications for legged robots based on their theoretical advantages.

The challenge of creating something that walks was fascinating but very complex at that time, and researchers did not succeed very often. Nevertheless, some interesting examples were designed and built during this period. Once again, the key was computing technology. When researchers were able to use powerful and compact computers in the industry, the number of robot developments increased and they were more successful. By the mid-1970s the first computer-controlled legged robot was tested at the Ohio State University (OSU). After that, American and Japanese universities and research centers began carrying out a great deal of activity in this field, including the development of vehicles considered to be mythical walking robots. Research in Europe was delayed for a few years. The first walking robot was documented around 1972 at the University of Rome in Italy (Mocci *et al.*, 1972). However, the hexapod developed at the Moscow Physio-Technical Institute in 1977 is sometimes reported as the first walking robot in Europe (Gurfinkel *et al.*, 1981).

Walking robots exhibit many hypothetical advantages over their more traditional counterparts, and the scientific community has developed a large number of computer-controlled walking robots in order to prove this assertion (Berns, 2005). Most of these machines, however, still exist as simple laboratory prototypes. Only a handful of walking robots have been equipped with the appropriate features: ASV (Song and Waldron, 1989), Dante II (Bares and Wettergreen, 1999), and Timberjack (Plustech-Oy, 2005), *etc.*, although they still far outperform the current possibilities for wheeled and tracked robots. The poor condition of the state-of-the-art of walking-robot technology is mainly due to the fact that the development of walking robots is much more complex than what was originally expected, not only in terms of mechanisms, but also in terms of electronic systems, sensing and control algorithms.

This book presents some of the basic concepts related to the design of statically-stable control algorithms for quadruped robots based on the author's prior experience: RIMHO (Jimenez *et al.*, 1993) (see Fig. 1.8); ROWER (Gonzalez de Santos *et al.*, 2000) (see Fig. 1.9) and SILO4[2] (Gonzalez de Santos *et al.*, 2003) (see Fig. 1.11); and it is devoted to the control of statically stable quadrupeds from an engineering point of view. That means that dynam-

[2] RIMHO, ROWER and SILO4 have been developed at the Industrial Automation Institute (IAI) of the Spanish National Research Council (CSIC).

ically stable algorithms and biologically inspired robots are out of the scope of this book. In this first chapter, we will introduce legged robots in general, and emphasize certain aspects such as their historical development, the advantages of walking locomotion over traditional wheeled locomotion, potential and real walking robot applications and a comparative study of quadrupeds and hexapods, the most common type of robot that has been built so far.

1.2 Historical Perspective

Walking-locomotion technology started with the development of very simple walking mechanisms that resembled toys which were able to move only under favorable conditions, including flat and level ground. Later scientists observed and recorded the walking modes of certain species in order to try to understand the functioning of motion in nature. Afterwards, gaits were formulated and studied based on mathematical models in an attempt to improve the features of walking mechanisms. Thus, stability measurements and gait-generation algorithms were created based on ideal cases, and improvements were eventually made leading to the present condition of such technology.

In considering the history of walking machines, it is normal to include the milestones in making progress on the understanding of how humans and animals walk. However in this section, we will focus on those milestones related exclusively to mechanisms and algorithms, and we will avoid mentioning other significant landmarks regarding photographic reports on motion (Muybridge, 1957), and comparative studies on zoology and biology (Wilson, 1966; Alexander, 1977; McMahon, 1984), *etc.*

This section covers the most important historical landmarks in the development of walking robots. It is based on the work done by Orin (1976), Todd (1985), Raibert (1986), Messuri (1985), and Song and Waldron (1989). We recommend the books and PhD theses written by these authors for readers who are interested in the historical development of walking robots.

1.2.1 Walking Mechanisms

The first walking mechanism, as mentioned above, was built around 1870 by Chebyshev. It was based on a system he had devised about two decades earlier. It consisted of a device based on four bars, like the one illustrated in Fig. 1.1, placed in such a way that when Link 1 rotates around axis A1, perpendicular to the sheet, the foot (and also point P1) follows a quasi straight-line trajectory, T1, at certain times during the cycle, and it moves off the ground during the rest of the cycle, T2. The shape of the trajectory and the quality of the straight line, T1, depend on the link lengths. The trajectory in Fig. 1.1(a) has been obtained for $\overline{A_1A_2} = 0.15\,\text{m}$, $\overline{A_2A_4} = 0.41\,\text{m}$, $\overline{A_3A_4} = 0.4\,\text{m}$, $\overline{A_3P_1} = 0.9\,\text{m}$ and distance $\overline{A_1A_2} = 0.3\,\text{m}$. Using this simple device it is

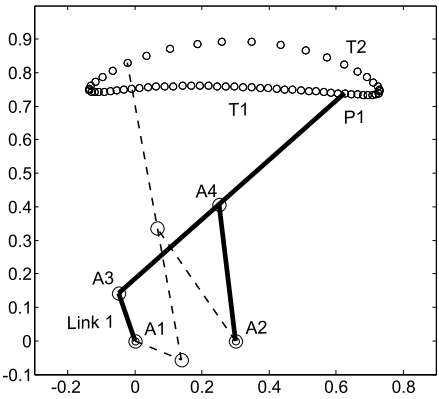
possible to alternate the support (stance) and transfer (swing) phases[3]. The walking machine devised by Chebyshev, and sketched in Fig. 1.1(b), arranges the legs in two pairs (contralateral, non adjacent legs). Thus, it is possible to perform a trot gait by alternating stance and swing phases of each pair of legs (diagonal gait). However, such a device can only walk (dynamically) on perfectly flat, levelled terrain because it does not have any kind of terrain adaptation mechanism or independent leg motion, which determines foothold selection. Nevertheless, the Chebyshev mechanism was incorporated as part of two popular machines during the final quarter of the last century: MELWALK (Kaneko *et al.*, 1985) and Dante (Wettergreen *et al.*, 1993).

Before the development of Chebyshev's mechanism, some astonishing moving devices that are not really walking systems were developed, essentially for entertainment purposes: Vaucanson's mechanical duck (1738); the writer, the musician and the draughtsman built by the Swiss watchmaker Jacquet-Droz about 1774, *etc.* (Logsdon, 1984). These are remarkable examples of the human interest in emulating live behavior. Another example which is considered a milestone in the history of the development of walking machines is the Mechanical Horse, which was registered by L. A. Rygg with the US Patent Office in 1893 (see Fig. 1.2). A rider provided the machine with power by using pedals, which caused the horse legs to move using links and cranks. That is traditionally considered to be the first patent of a legged system, though it is unclear whether or not it was actually built.

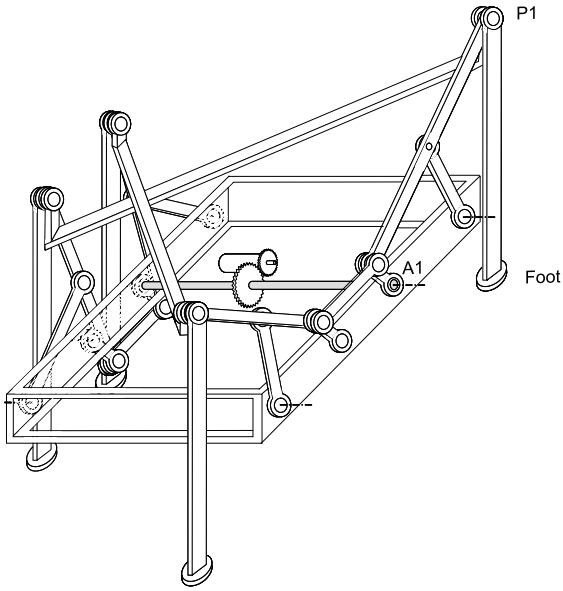
Many years later, in the 1940s, researchers and engineers developed new potential applications for walking robots. Military and space researchers were attracted by the possibilities provided by legged locomotion and some interesting applications were proposed, first in the UK and then in the US, as machines of war and planetary exploration.

The first serious undertaking to build a legged vehicle with independently controlled legs and terrain adaptability was made in the UK in 1940. A. C. Hutchinson believed that legs could be more efficient than wheels or tracks for very heavy vehicles in the range of 1000 tons. Hutchinson worked together with F. S. Smith in the development of legged systems capable of decoupling horizontal and vertical motions so that only two hydraulic actuators would be needed to move the machine. Finally, they were able to build a scaled down four-legged mechanism which was 60 cm tall, with eight joints which were moved by an operator using wires. Apparently, the machine was tested for use in armored vehicles but at the time the UK was immersed in World War II and the British War Department was not interested in such developments. As a result, further development of the machines was halted (Todd, 1985). As far as the authors were able to ascertain, this was the first quadruped machine built with terrain adaptability properties.

[3] The terms support phase and stance phase will be use interchangeably as well as the terms transfer phase and swing phase.



(a)



(b)

Fig. 1.1. (**a**) Mechanism of Chebyshev: configuration for the stance trajectory (*solid line*) and configuration for the swing trajectory (*dashed line*); (**b**) Sketch of the whole machine obtained from the planar drawing in (Artobolevsky, 1964)

During the next 20 years some important theoretical research was carried out in the US based on grants from NASA and the US Army. The discoveries made by the Polish engineer M. G. Bekker in the US Army Tank-Automotive Center were especially noteworthy. These discoveries allowed for the develop-

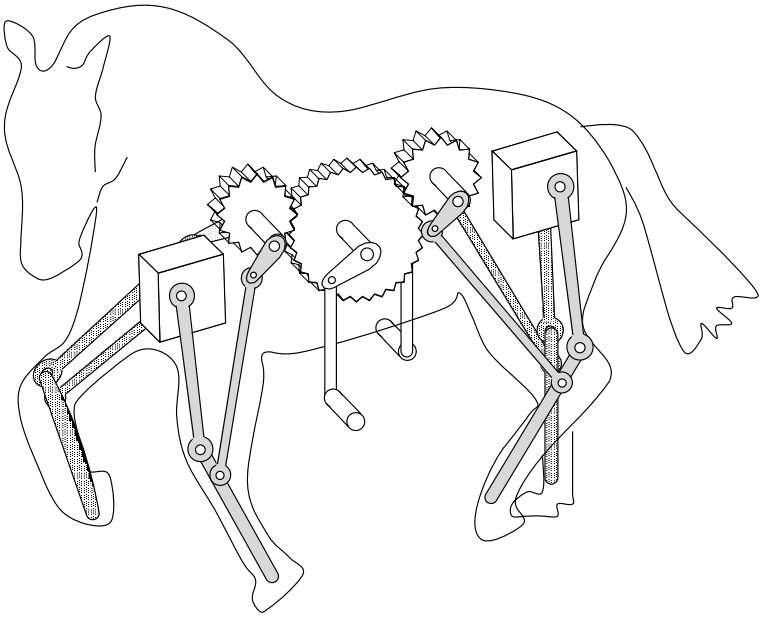


Fig. 1.2. The mechanical horse patented by Rygg in 1893. Drawing inspired by U.S. Patent No. 491,927 dated Feb. 14, 1893.

ment of the GE Walking Truck, which was built in close collaboration with General Electric Corporation. Work on this quadruped, with a 3 m long body, weighing 1400 kg, was began by R.S. Mosher in 1962. It used a 90 HP gas engine, so it was an autonomous machine from an energy-related point of view; however, control of legs was performed by an operator riding the robot. The operator commanded all 12 joint of the machine by using handles and pedals controlled by his/her hands and feet. Todd (1985) mentioned in his book that the burden of coordinating such a large number of handles and pedals made the operator unable to control the system for longer than 5 min. Raibert (1986), in contrast, claimed that the director of the project was able to drive the vehicle surprisingly well after about 20 h of training. In any case, this project was tremendously importance and it motivated R. McGhee of the University of Southern California to expand on existing locomotion techniques. Fortunately, McGhee had seen a demonstration of the GE Walking Truck in the mid-1960s and he understood that the main problem with the machine was the operator's inability to coordinate leg motions even for a short period of time. By chance, he was collaborating at the time with R. Tomovic of the University of Belgrade on a theory of finite-state control. He realized that an automatic cycling system capable of substituting the operator was the perfect solution.

Around the same time, NASA and the US Army once again sponsored projects to explore the possibilities of creating legged robots for military transport, planetary exploration, and for disabled assist applications. One important result of this research was the construction of the Iron Mule Train by the Space General Corporation. The Iron Mule Train was an eight-legged robot considered to be another milestone in the history of the development of walking machines (Morrison, 1968). More than 20 years later, Todd (1991) made a robot based on the Iron Mule Train machine with some slight modifications that served to make the components easier to manufacture. He concluded that the new robot was not a technical advancement, but on balance it was still a worthwhile invention for illustrating the advantages and limitations of such a robotic concept.

From 1966 to 1969 the Bucyrus-Eire Company worked on the construction of the Big Muskie, the largest legged machine ever seen. It was a 13500 ton dragline designed for working in open-air mines. The machine was based on four feet that were hydraulically driven. The feet rotated around a fixed shaft that moved forward, while the body of the machine stayed on the ground. When all four feet were touching the ground, the machine lifted up and moved forward to a new placing position. The feet could be seen as a one-radius wheel and the machine was capable of reaching speeds of 270 m/h. The Big Muskie was considered proof that Hutchinson's ideas worked, and surprisingly it was operative until 1991 (Big-Muskie, 2005).

In 1966, McGhee continued his research and, along with A. A. Frank, built a medium-sized (50 kg) quadruped called the Phony Pony (see Fig. 1.3). Each leg was based on a two-degrees-of-freedom system with rotary joints actuated by electric motors. The feet were based on an inverted T-shape structure that provided stability in the frontal plane. Each joint had a number of sensors for detecting whether the joint was locked, in forward motion or in backward motion. With these three different states, and using electronic logic based on flip-flops, they created a state machine with six synchronized states. The robot performed the quadruped crawl and the diagonal trot depending on the selected state diagram.

The Phony Pony was a milestone of paramount significance because it inspired McGhee, then at the Ohio State University (OSU), to build new machines that also became important milestones in the history of walking robots: the OSU hexapod and the Adaptive Suspension Vehicle (ASV) (see Fig. 1.4).

The OSU hexapod, built in 1977, was the first computer-controlled walking robot. Its legs were based on an insect leg type with three rotary joints driven by electrical motors. This robot became the experimental testbed for a large number of scientific results related to gait generation, robot control, and force distribution algorithms. In 1986, McGhee along with Waldron, who was still at OSU, built and tested the ASV hexapod (see Fig. 1.4), possibly the largest and most extraordinary terrain-adapted walking machine ever built (Waldron and McGhee, 1986; Song and Waldron, 1989).

In 1980, Professor Hirose at the Tokyo Institute of Technology (TIT), in Japan, began developing a large family of quadrupeds. The first one was the Pre-ambulate Vehicle (PV-II) (see Fig. 1.5), which is recognized as an important milestone in the development of legged robots in general, and quadrupeds in particular, although before it there was a precursor named KUMO. The PV-II weighed 10 kg and was about 1 m high. Its leg was based on a three-degree-of-fredom (DOF) pantograph mechanism, which was patented as the PANTOMEC. Since then, this device has been widely used in the construction of walking robots (see Figs. 1.8 and 1.13(a)). A few years later, Hirose began development of the TITAN series and he has been developing the TITAN-IX since 2001 (Kato and Hirose, 2001).

In 1983 Odetics Incorporated launched the ODEX I, a hexapod robot with legs based on a pantograph mechanism and placed in a circular configuration (Russell, 1983). This robot did not provide any important scientific contributions, but it is always included in the table of milestones as the first commercialised walking robot. The company built an advanced version to use for inspection of nuclear power plants (Byrd and DeVries, 1990). Afterwards the company apparently stopped its development of legged robots.

All of the robots that have been mentioned so far are basically statically stable systems. The first quadruped capable of walking – running indeed- with complete dynamic stability was developed by M. Raibert (1986) at the Massachusetts Institute of Technology (MIT) (see Fig. 1.6). Most of the researchers involved in developing dynamically stable robots started out working on statically stable multi-legged robots and later moved on to dynamic systems. Raibert did the opposite; he thought that by solving the one-leg problem he could apply his research to machines with two, four, or any number of legs. In the end, he succeeded.

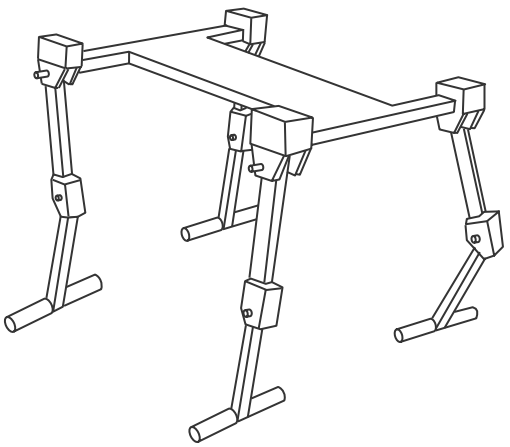


Fig. 1.3. Sketch of the Phony Pony

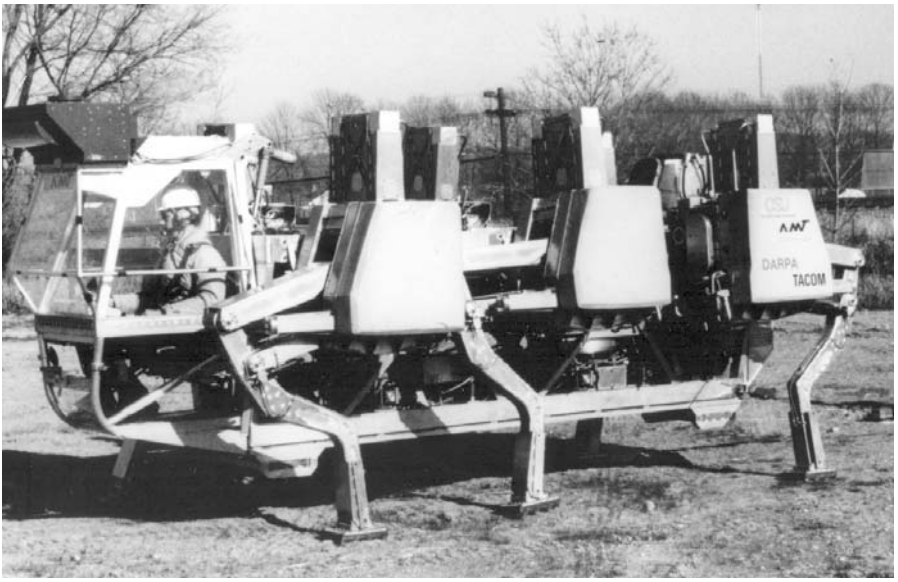


Fig. 1.4. The Adaptive Suspension Vehicle (ASV) (photograph courtesy of Professor Waldron)

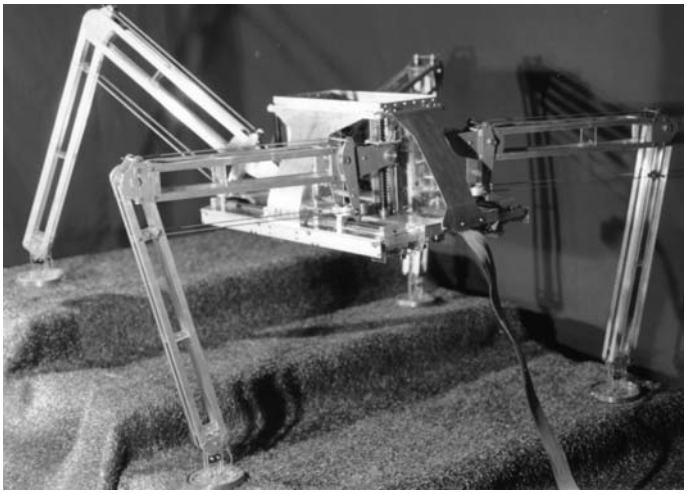


Fig. 1.5. PV-II walking robot (photograph courtesy of Professor Hirose)

Dynamic walking is essential for achieving high rates of speed for legged robots. This is one of the important features for obtaining mobility; however, for most tasks carried out by a legged robot, such as handling things, collecting measurements, *etc.*, the robot must be stationary or walking at very low speed; therefore static-stable gaits are required.

From the early 1990s to today, the number of walking robots being built around the world has increased dramatically. More than 200 different robots have already been catalogued (Berns, 2005). About 20% of them are quadruped robots. This proves that there have been a lot of developments in creating this kind of walking robot which exhibits the special features that are described later.

Table 1.1 summarises the history of the development of multi-legged robots, highlighting the milestones in the history of quadruped walking robots. Bear in mind that since 1990 the true milestones in the development of walking vehicles have been devoted to biped locomotion.

1.2.2 Gait Design

After creating a walking mechanism, it is necessary to perform a leg and body motion sequence to make the mechanism walk. This sequence is known as the "gait." The gait has been researched for many years; however, for a long time the results only functioned on surfaces with very favorable conditions: flat and level terrain.

The first significant progress that was made relating to gaits for quadrupeds was carried out by M. Hildebrand (1965). He described and compared the quadruped gaits of particular species based on physical features and performance. Hildebrand discovered 164 different gaits for quadrupeds, and he introduced the concept of gait formulae mainly based on observation and intuition; thus, his work was criticized as lacking in scientific foundation. Later, McGhee expanded on Hildebrand's ideas, including such factors as stride length, duty factor and leg phase. He defined these parameters using a consistent mathematical formula (see Chap. 3 for parameter definition).

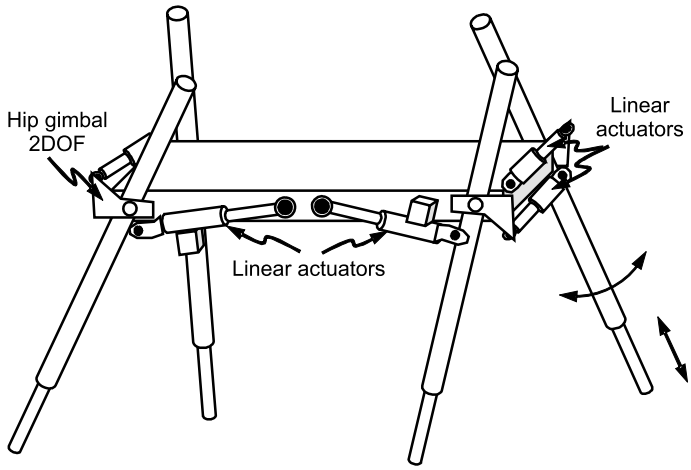


Fig. 1.6. Sketch of Raibert's dynamically stable quadruped

One of the main milestones in gait generation was the result of research by McGhee and Bessonov, who each studied and formulated continuous gaits separately. McGhee and Frank (1968) studied gaits for quadrupeds and concluded that a symmetric, periodic, regular type of gait known as the wave gait provided the optimum amount of stability for quadrupeds. A few years later, Bessonov and Umnov (1973) demonstrated, through numerical experimentation, that the stability of a wave gait is also the optimum option for hexapods. In 1982, Orin studied gaits for hexapods taking into consideration a crab angle, which is the angle that forms the body longitudinal axis with the direction of motion (Orin, 1982). Then, Kumar and Waldron (1988) derived a modified wave gait to generate continuous crab gaits for hexapods. Later, Zhang and Song (1990) formulated the wave-crab gait for quadrupeds, which allowed these robots to move in any direction. All of these gaits performed properly on flat and level terrain, but they had to be adapted to be able to walk on irregular terrain.

Several authors studied this problem from different approaches. The main approach was to use sensors for ground detection and to modify some parameters of the known algorithms, while maintaining some other parameters. Kumar and Waldron (1989) proposed a modified wave gait that automati-

Table 1.1. Milestones in the development of multi-legged walking robots (milestones for quadrupeds are indicated in bold)

Year	Researchers	Milestone
1870	**Chebyshev**	**Design of a four-link mechanism to alternate stance and transfer phases**
1893	**Rygg**	**Mechanical horse patent**
1940	**Hutchinson**	**Four-legged prototype**
1961	Morrison	Design and testing of the Iron Mule Train
1968	**Frank and McGhee**	**Design and testing of the Phony Pony**
1968	**Mosher**	**Testing of the GE Walking Truck**
1969	**Bucyrus-Erie Co.**	**Big Muskie, a 13,500 Ton walking dragline, is built and operated until 1991**
1972	University of Rome	First walking robot in Europe
1977	McGhee	Testing of the first computer controlled walking robot (OSU hexapod)
1980	**Hirose and Umetani**	**Development of the PV-II at the Tokyo Institute of Technology**
1983	Odetics Inc.	ODEX I, the first walking robot to be commercially available
1987	Waldron and McGhee	First demonstrations of the ASV, the best walking robot built in modern times
1989	**Raibert**	**First quadruped to perform trotting, pacing, and bounding gaits under dynamic control**

cally adapted to the selected footholds, for a wider range of speeds. They also described strategies for computing the desired speed based on modifying the gait parameters. Basically, the gait is modified by changing its duty factor, leg phase, leg stroke or cycle time. Some other authors proposed a gait-control strategy for a blind walking robot that automatically adapted the wave-crab gait to uneven terrain in real time by adapting the gait parameters while the vehicle was performing the walking algorithm (Jimenez and Gonzalez de Santos, 1997).

There is also a type of gait known as the discontinuous gait. At first, it might seem inferior to the continuous gait, but it actually has interesting capabilities on irregular terrain. Discontinuous gaits are formulated in much the same way as wave gaits, *i.e.* they are regular, symmetric and periodic, and they are generated using a fixed pattern. However, they differ from continuous gaits in their use of body motion. Continuous gaits perform the leg motion sequence while the body is moving at a constant speed. Discontinuous gaits, however, propel the body with all of the legs on the ground, and the leg transfer motion is performed while the body is stationary and supported by all of the other legs. This gait offer more stability and greater speed than wave gaits for certain duty factors, and it intrinsically provides greater terrain adaptability, as shown in Chap. 3. Finally, they are easier to implement than continuous gaits. Such ease of implementation is a mandatory feature of real machines.

One problem with gait algorithms is dealing with the navigation of real machines on a given path. Both continuous and discontinuous gaits can be used to follow a straight trajectory forming a crab angle with the body reference frame, but combining these straight segments to match a predefined trajectory is a troublesome task, especially when using continuous gaits. Periodic gaits are formulated as straight-line crab gaits which require fixed starting foot positions in the body reference frame in order to carry out the movement. These positions depend on the crab angle, *i.e.* the angle between the longitudinal axis of the body and the direction of the movement. Thus, the foot positions must be changed, with extra leg motions, at the beginning of the new trajectory segment in order to adapt the movement of the body to a new trajectory, for example. Note that this is a difficult task for quadrupeds, but it is not a problem for hexapods performing a wave-crab gait with a duty factor of half the period (alternate tripods). In the 1990s several authors studied methods to connect the different gaits for quadrupeds, and a number of algorithms were proposed as a means to find extra walking periods to connect two crab gaits, while certain features were maintained as a constant, such as the speed and stability margins (Lee and Song, 1990; Jimenez and Gonzalez de Santos, 1997).

Another important point to consider is that using fixed support patterns to propel the robot hampers motion over zones with forbidden cells. These shortcomings were the basis of the study on non-periodic gaits, also known as free gaits, characterized by the selection of leg sequence and adequate footholds

in real time. McGhee and Iswandhi (1979) designed a free gait for hexapods while they were working on a previous non-periodic algorithm developed by Kugushev and Jaroshevskij (1975). This method functioned by selecting a sequence of support points to enable a walking robot to negotiate terrain containing certain zones that were unsuitable for support, so the terrain had to be *a priori* divided into permitted and forbidden cells. The algorithm did not consider the irregularities of the terrain, and was only tested in simulations. Furthermore, the algorithm was intended to maximize the number of legs in the transfer phase; thus, it was adequate for hexapods, but it did not prove to be very useful for quadrupeds that were unable to have more than one leg in the air. A few years later, Hirose (1984) proposed a free gait that was specially designed for four-legged robots. It had positive results when it was tested in computer simulations. The terrain was again divided into permitted and forbidden cells, but it still needed terrain adaptation features. Apart from the lack of ground adaptation, free gaits have beneficial features for path-tracking purposes, since they can change direction at any time.

The basic developments in gaits for multi-legged robots are summarized in Table 1.2. Bear in mind again that since 1990 the true milestones on the development of gaits for walking robots have been devoted to biped locomotion. Algorithms for generating continuous, discontinuous and free gaits are included in Chaps. 3 and 4.

1.2.3 Stability Measurements

The main function of a walking robot is to provide stable motion; that is, to stay in motion for a period of time without falling. Stability in walking robots is a binary concept: a walking robot is either stable or unstable; there is no in-between. However, it is necessary to quantify the degree of stability,

Table 1.2. Milestones in the development of gaits for multi-legged robots

Year	Researchers	Milestone
1965	Hildebrand	Classification of animal gaits
1968	McGhee	Mathematical gait formula
1968	McGhee and Frank	Optimum wave gaits for quadrupeds
1973	Bessonov and Umnov	Optimum wave gaits for hexapods
1975	Kugushev and Jaroshevskij	Initial formulation of free gaits
1979	McGhee and Iswandhi	Completion of the formulation of free gaits
1989	Kumar and Waldron	Study of adaptable gaits
1990	Zhang and Song	Formulation of crab-wave gait

if possible, for an idea of how close or far the position of a robot is from being unstable.

The concept of "stability" was first defined by McGhee and Frank around 1968. They stated that a walking robot is statically stable if the horizontal projection of its center of gravity (COG) lies inside of the support polygon, formed by joining all of the feet in support. As stability measurements, they calculated the shortest distance from the projection of the COG to the boundaries of the support polygon and defined it as the Stability Margin (S_{SM}). Thus, a short stability margin means that the robot is close to instability. Zhang and Song (1989) formulated the crab-wave gait and defined the stability margin of this gait. To simplify the final formula, they defined the Longitudinal Stability Margin (S_{LSM}) as the shortest distance from the COG to the boundaries of the support polygon in the direction of the robot's longitudinal axis.

The stability measurement based on the projection of the COG does not factor in the height of the COG. Instinctively it seems that the higher the COG, the less stable the configuration. The COG position is especially important when the robot is standing on a slope. The S_{LSM} is not contingent on the terrain inclination; however, the robot is prone to tip over downhill rather than uphill. In order to solve this problem, Messuri proposed a new measurement called the Energy Stability Margin (S_{ESM}) (Messuri, 1985; Messuri and Klein, 1985). This measurement is defined as the minimum potential energy that is required to knock over the robot around the edges of the support polygon. This energy depends on robot weight. In 1998, Hirose and colleagues normalized this measurement with respect to robot weight; they thus defined the Normalized Energy Stability Margin (S_{NESM}) (Hirose *et al.*, 1998).

The S_{ESM} and S_{NESM} factor in the external impact energy required to knock over the robot; however they still use static measurements since only potential energy is calculated, not including the robot's inertia before impact. When a robot moves at medium speed, dynamic effects could occur, thus altering the amount of energy required for tumbling. This might also occur when the robot is carrying a manipulator, for instance. Dynamic measures should therefore be taken, at least under these circumstances.

The first attempt to define a dynamic measurement was made by Orin (1976). He defined the Dynamic Stability Margin (S_{DSM}) as the shortest distance from the Center of Pressure (COP) to the boundaries of the support polygon. The COP is defined as the projection of the COG along the resultant of all of the forces acting on the COG. Therefore, under static conditions the COP coincides with the COG and the S_{DSM} is thus limited in the same way; that is, it does not factor in body height.

Throughout the last decade, some researchers tried to define stability margins capable of sensing dynamic effects. Although they were based on various criteria, at the end, they resulted in very similar definitions. The issue is still being researched, it is summarized in Table 1.3 and will be expanded on in Chap. 2.

1.3 The Advantages of Walking Locomotion

Apart from the interest in creating machines to imitate natural motion, some researchers envisaged the potential advantages of legged systems over traditional vehicles, based on wheels or tracks, for use in industry or services. Some of these advantages are discussed below.

1.3.1 Mobility

Legged robots exhibit better mobility than wheeled robots because they are intrinsically omni-directional systems. That is, a legged robot can change direction independently of the direction of the main body axis, just by changing its footholds. On the other hand, a conventional wheeled robot would have to do some manoeuvring to be able to change direction.

Likewise, a legged robot can move and orientate its body while maintaining the footholds just by changing its leg extension. This feature provides the robot's body with six additional degrees of freedom (DOF). Figure 1.7 illustrates these features that require legs based on 3-DOF mechanisms. It is worth noting that a wheeled robot that has wheels with traction and directional motors can substantially improve its directionality, but with the added cost of making the system more complex. There are also robots that use special wheels, such as the Ilonator wheel, which provide omnidirectionality, but only on flat surfaces (Ilon, 1975).

1.3.2 Overcoming Obstacles

A legged robot can overcome obstacles that are at a lower level than the maximum ground clearance, just by stepping on them. On the other hand, a wheeled robot can only overcome obstacles with heights of up to half of the wheel radius (McKerrow, 1991). The tracks consist of a virtual wheel with a radius of half the track length; so a tracked vehicle can surmount higher obstacles than a wheeled one, but using large body motions (see Fig. 1.7).

Table 1.3. Milestones in the development of stability measurements

Year	Researchers	Milestone
1968	McGhee and Frank	Definition of static stability and the Static Stability Margin
1976	Orin	Dynamic Stability Margin
1985	Messuri	Energy Stability Margin
1989	Zhang and Song	Longitudinal Stability Margin
1998	Hirose and colleagues	Normalized Energy Stability Margin

1.3.3 Active Suspension

A legged robot provides intrinsically active suspension by adapting the leg lengths to terrain irregularities. In this manner, a legged robot can cover highly irregular terrain with the body levelled. Thus, legged systems provides riders with smooth and comfortable motion. In contrast, the body of a wheeled robot is always parallel to the terrain and adopts similar tilts to the ground (see Fig. 1.7).

1.3.4 Energy Efficiency

Hutchinson suggested in 1940 that the efficiency of very heavy legged vehicles might be better than that of wheeled vehicles (see previous sections). Later, Bekker proved through experiments that Hutchinson was correct in asserting that legged systems under very irregular terrain conditions are more efficient than wheeled or tracked systems. Table 1.4 shows the data obtained by Bekker (1960) in the comparative study on vehicles and animals.

1.3.5 Natural Terrain

Wheeled vehicles require very expensive, continuous paved surfaces to move efficiently. In principle, legged systems do not require prepared terrain, like wheeled vehicles do, and they can move on sandy, muddy, stiff, and soft terrains with similar efficiency. Another advantage of legged systems is that they do not need continuous terrain to move.

1.3.6 Slippage and Jamming

Wheels tend to sink in soft terrain, which makes it difficult for wheeled vehicles to move. However, if one leg is placed vertically on the ground it only compacts soft ground in the same direction. Leg lifting is performed vertically, without interfering the ground. When the body is propelled, feet rotate around their joints; therefore, legs do not interact with the ground and do not cause any jamming problems. The same is true for vehicle slippage when propelling forward/backward (see Fig. 1.7).

Table 1.4. Bekker's study of vehicles and animals

	Average speed on highly irregular terrain (km/h)	Power required to move over a plastic strip that is 25 cm thick (HP/ton)
Tracked vehicles	8 – 16	10
Wheeled vehicles	5 – 8	15
Animals	> 50	7

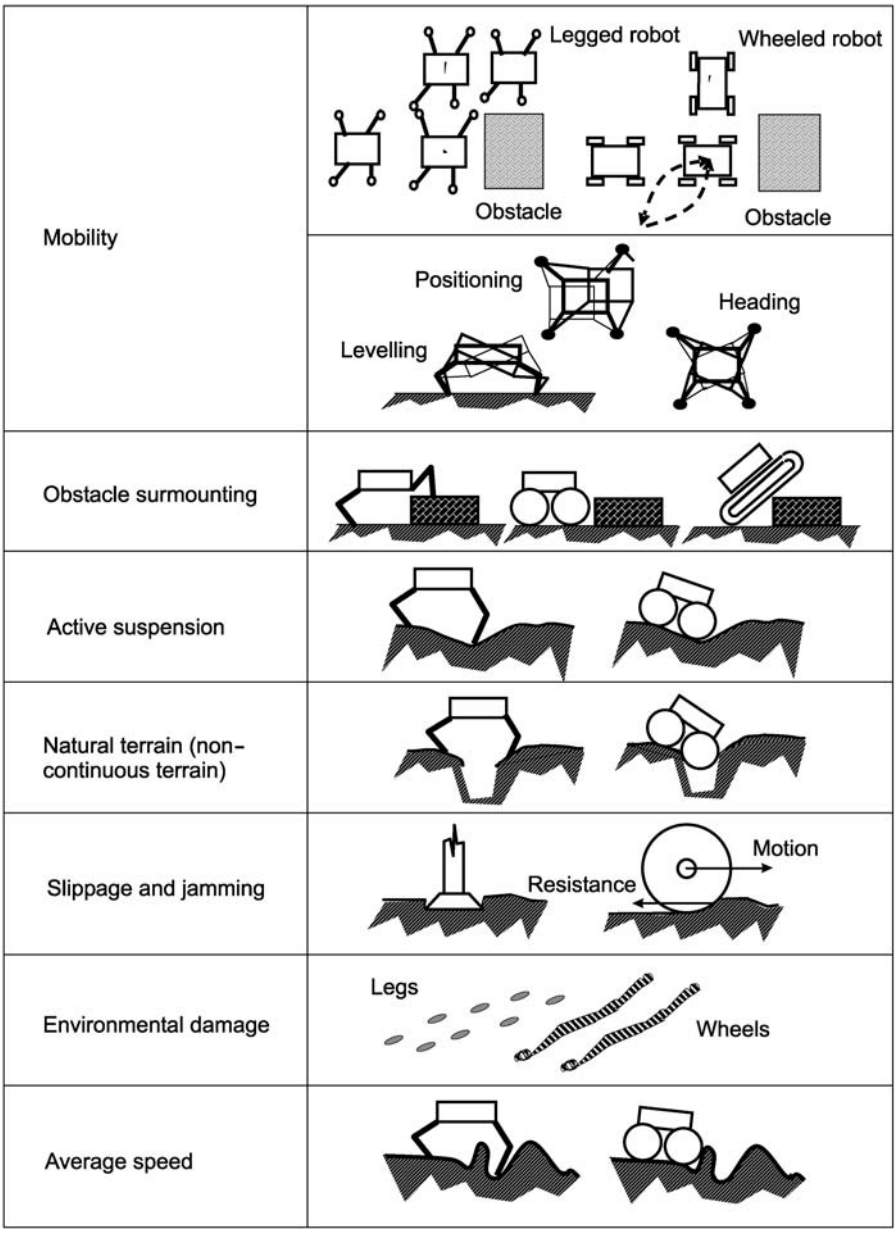


Fig. 1.7. Advantages of legged *vs* wheeled robots

1.3.7 Environmental Damage

Legged vehicles require discrete contact points with the ground, while wheeled or tracked vehicles use a couple of continuous paths along the ground. Therefore, legged robots touch the ground less than traditional vehicles do, thereby causing less environmental damage (see Fig. 1.7).

1.3.8 Average Speed

Traditional vehicles can move at high speeds on prepared surfaces. However, when the terrain is more uneven, the vehicle speed decreases rapidly. Legged systems (mammals, for instance) are able to adapt quite well to terrain irregularities, and they are able to maintain similar average speed over very different kinds of terrain. Bekker's study proved this phenomenon (see Fig. 1.7).

1.4 Disadvantages of Walking Locomotion

Of course, legged machines are not the general solution for locomotion; they also have problems and disadvantages that have kept them from being used for industry and services. The first problem is their complexity. Legged vehicles are more complex than wheeled vehicles, not only in terms of the machine itself, but also in terms of electronics and control. Another problem with legged robots is their speed. Statically stable legged robots are intrinsically very slow machines. Dynamically stable robots are still in the very early stages of development, and they do not appear to move as fast as robots with wheels. The total cost is another major factor. These problems are further expanded below.

1.4.1 The Machine

A wheel is an extraordinarily simple mechanism that consists of a disk with a rotary joint. A leg consists of several links and joints (rotary or prismatic). However, this system is obviously more complex than that of a simple wheel. A wheel requires only one actuator to propel it, and another actuator to steer it. The most simple statically-stable wheeled vehicle is the tricycle, consisting of a wheel for traction and steering (two actuators), and two more passive wheels. There are other wheeled configurations, such as the differential system, or the latest Self-Balancing Two-Wheeled Vehicle (Tirmant *et al.*, 2002), that also requires only a few actuators. The most complex system has four wheels with independent traction and a steering actuator – eight in total. On the other hand, a leg needs at least three DOFs, which means that it needs three actuators to provide traction and direction. At least 4 legs are needed for a statically stable robot; therefore, the number of actuators for a legged robot must be at least 12. Therefore the electromechanical system is more complex and expensive for a legged robot than for a wheeled one.

1.4.2 Electronic System

Every actuator has an associated power driver. Thus, legged robots require more electronic systems than wheeled robots. Another problem is that the robot joints must be controlled; therefore, the control system requires sensors for that purpose. Once again, legged robots need more sensors than traditional vehicles, and there are other problems as well.

Wheels are always in contact with the ground, while legs alternate between the stance and swing phases; this means that they need sensors to determine when the foot touches the ground. Including a touch sensor, or some type of similar sensor, for each leg increases the total number of sensors, as well as the number of electronic cards required for processing the sensor information.

Algorithms for controlling legged robots are more complex than the algorithms used to move wheeled robots (see Sect. 1.4.3). However they do not cause much of a computational burden; there are no special computing requirements for legged robots, and the computers for both kinds of robots are quite similar.

1.4.3 Control Algorithms

A wheel driver or a steering driver for a wheeled robot only requires a signal from the controller. Normally this voltage is proportional to the speed required or to the steering angle required, respectively. However, a quadruped robot must simultaneously coordinate the motion of all twelve of its joints, as well as its foot sensors, in order to provide stable motion. The control algorithms for legged robots are undoubtedly more complex than wheeled-robot algorithms.

1.4.4 Achievable Speed

We have already mentioned that a legged robot can achieve higher speeds than a wheeled robot on highly irregular terrain. On prepared surfaces such as roads, streets and factory floors, the speed of wheeled vehicles is definitely greater. For instance, fast animals such as horses or cheetahs can reach speeds of up to 60–80 km/h, while wheeled vehicles can reach speeds up to 350 km/h. Legged robots cannot compete with these animals and vehicles, and it is not believed that they will be able to in the future.

1.4.5 Cost

The total cost of a system is proportional to its complexity in terms of the machine, electronics, sensors, *etc.* Thus, a legged robot is much more expensive than a wheeled robot. Table 1.5 presents some figures on the complexity of a quadruped and the simplest wheeled robot, for purposes of comparison and cost estimation.

In summary, legged robots are not expected to substitute completely traditional vehicles on prepared surfaces. The only appropriate uses for legged robots are those where they present clear advantages over traditional vehicles: natural terrain, highly irregular terrain, special geometric structures, stairs, *etc.* The next section describes some of the potential and real applications for walking robots based on their theoretical advantages.

1.5 Potential and Real Uses for Walking Robots

Potential uses for walking robots are based on their advantages over wheeled or tracked vehicles for each specific task. Thus, there are some advantages to using legged robots in traditional vehicle applications, such as military missions, inspection of complex or dangerous scenarios, terrestrial, underwater and space exploration, forestry and agricultural tasks, construction activity, and civil projects, for example. Nevertheless, legged robots can also be used as the perfect experimental testbed for studying the behavior of live animals and for testing artificial-intelligence (AI) techniques. Finally, legged robots are also used for social activities, including humanitarian assistance in de-mining and disabling bombs.

The subsections below briefly discuss some of the potential applications and Table 1.6 summarizes some of the walking robots that have already been built for carrying out these applications.

1.5.1 Military Applications

Military transport activities require vehicles that are highly efficient on a broad variety of terrain: irregular, inclined, sandy, muddy, paved, *etc.* In addition, these vehicles must drive over obstacles such as ditches and anti-tank obstacles. As we have seen in previous sections, legged vehicles are theoretically capable of walking on this type of terrain and obstacles. Thus, there have been attempts to build legged robots with the support of military institutions. Some examples include the prototype developed by Hutchinson and

Table 1.5. Complexity of a two-actuator wheeled robot and of a quadruped

	Wheeled robot (units)	Quadruped robot (units)
Actuators	2	12
Drivers	2	12
Controllers	2	12
Joint sensors	2	12
Wheel/foot sensors	0	4
Computers	1	1

sponsored by the British War Department, the Iron Mule Train sponsored by the US Army, and the ASV fully funded by The Defence Advanced Research Projects Agency (DARPA).

1.5.2 Inspection of Nuclear Power Plants

Another possible use for walking robots is the operation and inspection of nuclear power plants. Nuclear plants have areas that are not equipped for wheeled robots (with pipes on the floor, stairs, *etc.*), that are easily handled by walking robots.

As already mentioned, Odetics Inc. built an industrial version of the ODEX I legged robot specifically designed to work in this environment (Byrd and DeVries, 1990). In 1990, a Spanish consortium built the RIMHO four-legged robot to test the possible uses of these vehicles in nuclear environments (see Fig. 1.8) (Jimenez *et al.*, 1993). The Sherpa robot (French Atomic Energy Commission) is one more attempt to develop a six-legged robot for nuclear environments in Europe (Berns, 2005).

1.5.3 Land, Submarine and Planetary Exploration

The ability of legged robots to adapt to different unknown types of terrain, to overcome obstacles, and to use discrete contact points with the ground, makes

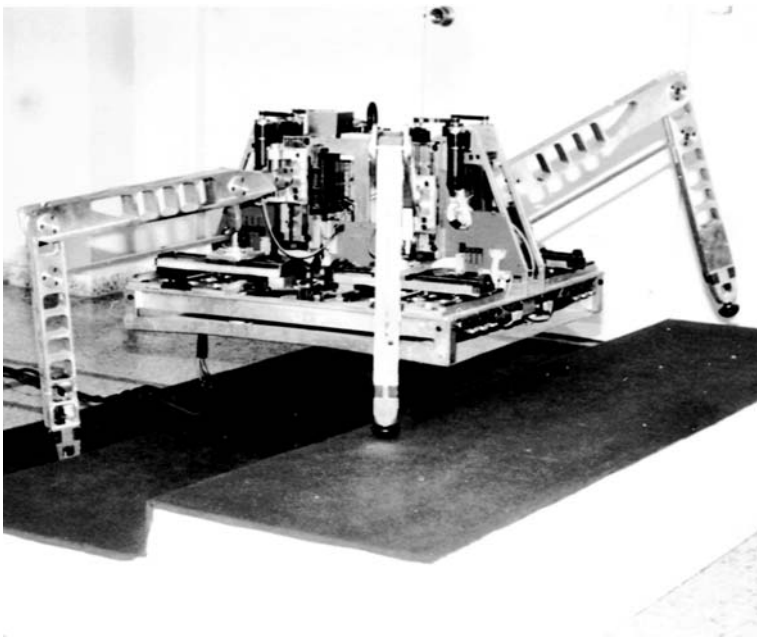


Fig. 1.8. The RIMHO walking robot

them perfect candidates for planetary, land and submarine exploration. Some robots have already been specifically built and tested for these uses; for example: (a) the AMBLER, developed by Carnegie-Mellon University under a NASA grant, which was developed as an experimental testbed for developing the technology required for a hypothetical mission to Mars (Bares and Whittaker, 1989, 1993); (b) the Dante II, developed by the same research group as the AMBLER, and sponsored by the same institution, which performed inspections of the inside of the crater of the Mount Spur volcano in Alaska (Bares and Wettergreen, 1999); and (c) the Aquarobot, built at the Robotics Laboratory of the Port and Harbor Research Institute in Japan, which is used for underwater surveying of seawalls (Akizono *et al.*, 1989). It is worth noting that over 50% of the land in the world is not viable for being covered by conventional vehicles (Bihari *et al.*, 1989) and must therefore by explored using animals or robots.

1.5.4 Forestry and Agricultural Tasks

Walking robots can be very useful in the forest, where it is necessary to move machinery or to chop down tree trunks. In this case, the trunks themselves are natural obstacles. Forests are normally sloped or mountainous. A legged robot can level its body, maintaining stability in this type of terrain. Wheeled robots do not have this ability, and they are prone to rolling on this type of terrain. Plustech Oy, a Finish company that is part of John Deere's Construction and Forestry Division, has developed a machine called the Timberjack for this type of terrain. The Timberjack is a six-legged robot that resembles an agricultural tractor and carries a manipulator to handle tools or grippers. The authors highly recommend watching the videos of this impressive walking machine (Plustech-Oy, 2005).

A similar machine would be very useful for agricultural tasks because it could move by simply using discrete contact points and therefore protecting the crops. On the other hand, an analogous wheeled or tracked vehicle would destroy crops along the wheel paths.

1.5.5 Construction

Construction is an important task for legged robots, especially for activities related to motion in complex environments. One such environment is that of ship building processes for connecting consecutive blocks at the dry-dock by welding together all of the longitudinal reinforcements and all of the vertical bulkheads. Figure 1.9 shows what a typical scenario looks like – closed cells that are $10 \times 4 \times 3\,\mathrm{m}$ around – where welding is traditionally performed by hand.

In order to improve productivity by increasing total arc time, improving weld quality, and creating improved operator working conditions, a team of shipyard companies and researchers designed and built an automatic welding

system (from 1994 to 1998) based on sponsorship from the European Community Commission (ECC).

The overall system consists of a mobile platform with a commercial welding system that is handled by a commercial manipulator. A four-legged robot was considered to be a candidate for the mobile platform. The leg was based on the SCARA structure, since it does not cause problems with shank-rocking, thus crashes between legs and stiffeners do not appear. However, this structure had problems with vertical link length, since it had to be 2 m high in order to place the manipulator close enough to the ceiling of the cell. This long link, and the heavy payload the machine had to carry (about 130 kg) made the vertical link highly prone to bend and vibrate, especially when the body was located at its highest position. This feature also jeopardized the machine's static stability. A modified structure, called ROWER, was proposed to solve this problem, to enable it to walk by grasping the stiffeners.

Figure 1.9 shows the ROWER walking robot inside a cell, with all four legs grasped to the stiffeners. In this position, the body can be moved forward, backward, up, down and sideways. The motion of the robot along the cell is performed by moving the legs and body sequentially. Body motions are performed with all four legs grasped to the stiffeners. The leg motions and grasping procedure are detailed in (Gonzalez de Santos *et al.*, 2000). This is an example of the use of legged robots to walk on special structures.

1.5.6 Civil Projects

Civil construction is an activity that requires the moving of special devices on uneven terrain and slopes. Motion on slopes is a normal part of road construction, where it is necessary to consolidate the terrain by inserting rods

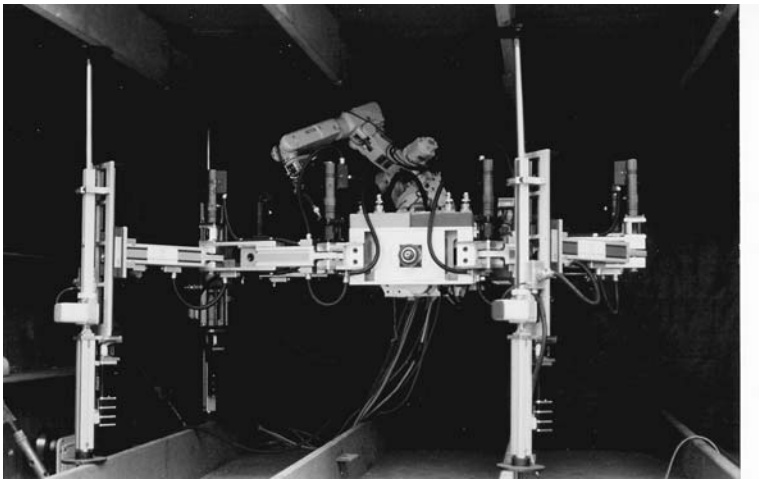


Fig. 1.9. The ROWER walking machine

and covering it with metallic nets in order to avoid moving earth and rolling stones. This work is performed with a frame containing the operator and tools. This frame is hung and dragged by cables attached to the upper part of the slope. To smooth the motion of the frame up and down the slope, Professor Hirose at TIT developed the TITAN VII legged robot (see Fig. 1.10), which hung from similar cables, using its legs to adapt to terrain irregularities (Hirose *et al.*, 1997). A similar system was recently developed by a European consortium with ECC funding (Nabulsi and Armada, 2004).

1.5.7 Help for Disabled People

The lives of handicapped people would surely be improved by legged wheelchairs. Although there is great societal interest in eliminating barriers for handicapped people, it is difficult to overcome the obstacles of buildings with stairs and the uneven terrain of the countryside. Legged robots could move the handicapped over these obstacles. A research team from the University of Illinois at Chicago attempted to create such a machine at the end of the 1990s, but no final results were reported (Zhang and Song, 1989). Nevertheless, this book's authors encountered some scepticism among potential users, based on the expected price of the chair and the amount of maintenance required. Once again, politicians and researchers should work together to solve this problem.

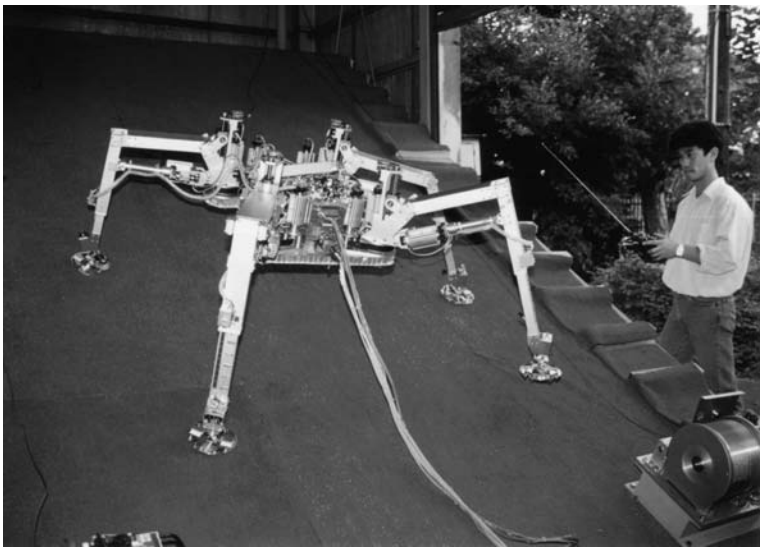


Fig. 1.10. TITAN VII: a walking robot for civil projects (photograph courtesy of Professor Hirose)

1.5.8 Support for AI Techniques

Significant artificial intelligence (AI) testing was conducted on robot manipulators some years ago, but mobile robots were extremely important to the development of these techniques. Some researchers believe that intelligence is a result of mobility (Moravec, 1988). Mobility allows us to learn and decide. Naturally a rover learns from more situations than a manipulator. Legged robots have the same major problems as rovers, as well as problems related to gait generation. Thus, many AI researchers have used legged robots to test their theories and methods. For example, Brooks built his own legged robot, called Attila, for this purpose (Angle and Brooks, 1990); the AIBO, by the Sony Corporation, has been used by many research groups, mainly for testing AI and sensor integration techniques (Fujita, 2001). The SILO4, developed by the Industrial Automation Institute (IAI-CSIC) in Spain, is another example of a legged robot planned for testing by researchers interested in developing AI, sensor integration, or gait generation techniques (Gonzalez de Santos *et al.*, 2003) (see Fig. 1.11). This robot has been offered as a common experimental testbed for comparative studies, and its developers have encouraged researchers to build their own prototypes using the drawings and indications available on the internet (SILO4, 2005).

1.5.9 Study of Living Creatures

Over the last two decades, zoologists and biologists have conducted a great deal of research for understanding the biological aspect of walking. Certain

Fig. 1.11. The SILO4 walking robot

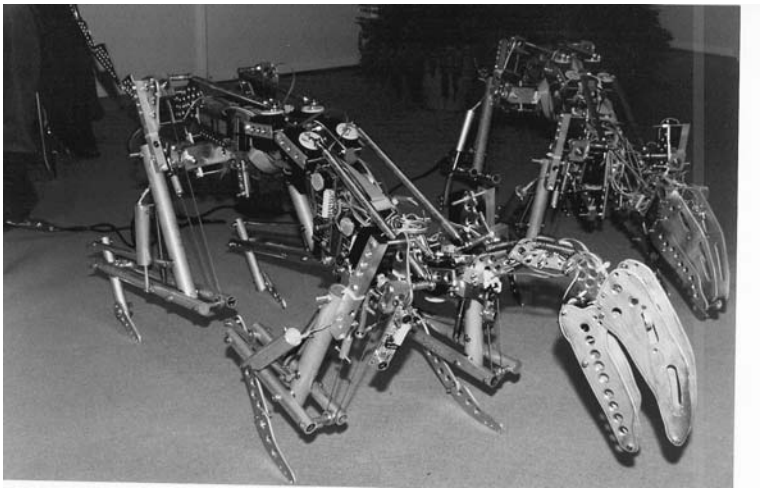


Fig. 1.12. Prototype for the PALAIOMATION project (reproduced with permission of Dr. Papantoniou)

researchers have concluded that a few simple rules are enough for defining a sequence of stable motions (Cruse *et al.*, 1998). These rules can be verified in simulations, but implementing them in real robots that emulate the behavior of insects is of paramount importance for these researchers. Thus, a few robots such as TUM (Pfeiffer and Weidemann, 1991) or Tarry (Frik *et al.*, 1999), whose legs imitate the leg structure of a stick insect, have been built for this purpose. It is also possible to find some robots that attempt to emulate extinct species. This has become very important for educational and entertainment purposes. Some examples are the robot developed in the PALAIOMATION project, funded by the ECC in 1994, which replicated an iguanodon at a quarter scale (see Fig. 1.12), and the new dino-robot called Butch, which resembles a protoceratops (Butch, 2005).

1.5.10 Humanitarian De-mining

Detection and removal of antipersonnel land-mines is an important concern worldwide. An enormous number of land-mines have been deployed over the last 20 years, and de-mining will take several more decades, even if no more mines are deployed in the future. An adequate mine-clearance rate can only be achieved by using new technology, such as improved sensors, efficient manipulators and mobile robots. Any potential vehicle can theoretically carry sensors over a mine-field; however, legged robots provide some potential advantages, such as those presented in Sect. 1.3.

The idea of using legged machines for humanitarian assistance for de-mining has been under development for about the last ten years, and some prototypes have already been tested. The TITAN VIII, developed at TIT,

was one of the first walking robots adapted for de-mining tasks (Hirose and Kato, 1998). AMRU-2, an electro pneumatic hexapod developed by the Free University of Brussels and the Royal Military Academy of Belgium (Baudoin *et al.*, 1999), and RIMHO2, developed at IAI-CSIC (see Fig. 1.13(a)), are two more examples of walking robots tested for humanitarian de-mining tasks.

The COMET-1 was perhaps the first legged robot developed specifically for de-mining tasks. It is a six-legged robot developed by a Japanese consortium, and it incorporates different sensors and location systems (Nonami *et al.*, 2000). The COMET team is currently working on developing a third version of this robot. ARIEL is another hexapod robot, developed by the IRobot Company, for mine recovery operations. DARPA and the US office of Naval Research are exploring methods for using this robot for underwater missions (Voth, 2002).

Following this tendency of developing legged robots specifically for humanitarian de-mining, the IAI-CSIC created the DYLEMA (a Spanish acronym that means "Efficient Detection and Location of Antipersonnel Landmines") system for detecting and locating land mines. It is based on the SILO6 legged robot, which carries a mine detecting set (see Fig. 1.13(b)). The main aim of the DYLEMA project is to develop an entire system for integrating the relevant technology in the fields of legged locomotion and sensor systems, in order to identify existing needs for humanitarian de-mining missions.

The six robots that were last mentioned are based on insect configurations, but there are also different legged robot configurations, such as sliding-frame systems, which are being tested as humanitarian de-mining robots (Habumuremyi, 1998; Marques *et al.*, 2002).

To sum up, there is a great deal of activity being carried out to develop walking robots for de-mining.

1.6 Quadrupeds *vs* Hexapods

Whenever a legged robot is being developed, the first feature to be defined is the number of legs. This number is the result of a trade-off between the many different features that the robot could have, and the specific application requirements. This balance must be studied in terms of stability, speed, reliability, weight, and price.

For example, a hexapod provides better static stability than a quadruped. Hexapods can perform static gaits by supporting the robot's body on five legs at any time, while quadrupeds can only walk statically on a minimum of three legs. This feature makes hexapods much more stable than quadrupeds, since they can use a bigger support polygon. Notice that stability is a fundamental issue for mobile robots.

Speed is also an extremely important factor for robot locomotion. It may be the main reason for the poor state of legged robots in industry and services. Waldron and his colleagues demonstrated that the speed of a legged robot, V,

(a)

(b)

Fig. 1.13. Robots for humanitarian de-mining: (**a**) RIMHO2; (**b**) SILO6

performing a wave gait depends on the leg stroke, R, the leg return time, τ, and the duty factor, β, which directly depends on the number of legs (Waldron *et al.*, 1984). That is (see Chap. 3):

$$V = \frac{R}{\tau}\left(\frac{1-\beta}{\beta}\right)$$

The minimum duty factor of an n-legged robot is $\beta_n = 3/n$. That is, $\beta_4 = 3/4$, $\beta_6 = 3/6$, and $\beta_8 = 3/8$, respectively. Therefore, the robot speed is determined by $V_4 = 0.333(R/\tau)$, $V_6 = R/\tau$ and $V_8 = 1.67(R/\tau)$. Hexapods can clearly achieve higher speeds than quadrupeds, and octopods are even faster.

Some researchers throughout the history of development of legged robots have pointed out that a hexapod can continue walking after the malfunctioning of up to two legs (one leg per side). This is an extreme method of obtaining a quadruped from a hexapod. In other words, a hexapod is a redundant statically stable walking robot, which is able to walk even with one or two failed

legs since we can define stable gaits by using either four of five legs. Although everyone has seen an insect walking on five legs, it is more difficult to imagine a legged robot walking after the failure of a leg which is stuck inside a hole, for example. It is understandable that the failure of a leg close to its maximum extension places tremendous limits on the mobility of the robot. Hence, the theoretical redundancy of hexapods cannot really be considered an advantage over quadrupeds.

Hexapods have other problems such as reliability, for example. More legs means a more complex mechanism and larger electronic and sensor system. Thus, the likelihood of failure is increased and therefore the likelihood of mission success is decreased.

The total robot weight is another inconvenience. Considering that the body weight of a robot, W_B, does not depend on the number of legs, n, a robot configured based on a leg weighing W_L has a total weight, W_T, of around

Table 1.6. Walking robots for specific applications (see (Berns, 2005))

Application	Robots
Military applications	GE Walking Truck Iron Mule Train ASV
Inspection of nuclear power plants	Odex I Sherpa RIMHO
Land, submarine and planetary exploration	ReCus AMBLER Dante Aquarobot
Forestry and agricultural tasks	Timberjack
Construction	ROWER
Civil projects	TITAN VII ROBOCLIMBER
Help for disabled persons	Walking chair
Support for AI techniques	Attila AIBO TUM TARRY SILO4
Study of living creatures	Palaiomation Butch
Humanitarian de-mining projects	TITAN VIII AMRU-2 COMET Ariel SILO6

$$W_T = W_B + nW_L$$

For robots configured around the SILO4 leg and body designs (see Appendix A), we get $W_B = 14\,\text{kg}$ and $W_L = 4\,\text{kg}$; therefore, the total weight is 30 kg for a quadruped – the real weight of the SILO4 - and 38 kg for a hexapod. This is an increase of about 27% in the total robot weight. Notice that in both robot configurations the total weight must be supported on three legs, the minimum number required to guarantee static stability; therefore, a hexapod must basically use the same type of leg as a quadruped of similar weight.

In nature, there is no land animal weighing more than 100 g that can walk on six legs (Williams, 2005). Therefore, it appears that quadrupeds are more efficient than hexapods for machines weighing 100 g to a few tons. As already pointed out, hexapods can only compete with quadrupeds for speed and stability under static conditions. However, stability could be greatly improved, providing more exact stability measurements and control algorithms. Speed would be significantly faster if it were possible to implement dynamic gaits and dynamic control laws for legged machines.

In summary, there are many reasons to continue to research quadruped robots. The main objective of this book is to encourage researchers to expand on four-legged robot techniques. And for this purpose, we have included in this book the main algorithms and techniques developed by us over the last ten years, which have all been tested on the SILO4 walking robot. Supplementary material such as pictures, videos, simulation software, construction drawings, *etc.*, is available on the SILO4 web site `http://www.iai.csic.es/users/silo4`.

2

Stability in Walking Robots

2.1 Introduction

Research on walking-robot stability began in the mid-1960s, when McGhee and Frank (1968) first defined the static stability of an ideal walking robot. Following their definition, an ideal robot is statically stable if the horizontal projection of its center of gravity (COG) lies inside the support pattern. The ideal robot is supposed to have massless legs, and system dynamics are assumed to be absent.

The idea of static stability was inspired by insects. These arthropods feature an exoskeleton composed of a segmented body and jointed appendages. Insects use their massless legs to simultaneously support their body during walking and provide propulsion. Hence, in order to move the body while maintaining balance, their sequence of steps is arranged to ensure static stability. The first generation of walking machines emulated this principle of locomotion (Kumar and Waldron, 1989). Early walking robots were huge mechanisms featuring heavy limbs too difficult to control (Song and Waldron, 1989). The adoption of statically stable gaits could simplify their control. However, during the motion of the heavy limbs and body some inertial effects and other dynamic components (friction, elasticity, *etc.*) were found to arise, restricting the robot's movements to low, constant velocities. Thus, the adoption of static stability facilitated motion control at the price of speed.

The only way to increase walking-robot speed is to consider the robot dynamics in the control of robot stability. The intrinsic complexity of considering the whole robot's dynamics (see Chap. 6) led some researchers to look for solutions by designing very mechanically simplified machines having only a few degrees of freedom (Raibert *et al.*, 1986; Wong and Orin, 1993; Buehler *et al.*, 1998), which adopt the stability criteria designed for bipeds (*i.e.* humanoid robots), extended to a couple of additional legs. The motion of these quadrupeds is limited to even terrain, because the stability criterion used – based on the Zero Moment Point (Vukobratovic and Juricic, 1969) – is only valid for that kind of surface (Goswami, 1999; Kimura *et al.*, 1990; Yoneda

et al., 1996). Also, the mechanical simplicity of these designs makes the robot useless for real applications, where manipulation and payload transportation are required. Little effort has been made to cope with the dynamic effects that limit statically stable machines' performance (Gonzalez de Santos *et al.*, 1998; Kang *et al.*, 1997; Lin and Song, 1993; Papadopoulos and Rey, 1996; Yoneda and Hirose, 1997; Garcia and Gonzalez de Santos, 2005). However, one of the goals of research on legged locomotion is the use of walking robots in industrial, field and service applications, and such robots are not meant always to trot or gallop but also to walk under static stability on uneven terrain.

The few dynamic stability criteria defined for quadrupedal walking seem to give different forms and names to a single idea: the sign of the moment around the edges of the support polygon caused by dynamic effects acting on the vehicle's center of mass. The suitability of each criterion for each particular application (*i.e.* manipulation forces and moment present, uneven terrain, *etc.*) is not clear at all. Nevertheless, the use of a stability criterion not suitable for the current application may prevent the task from succeeding. Therefore, in this chapter, static and dynamic stability criteria are briefly surveyed in Sects. 2.2 and 2.3. Then, a comparative study of their stability margins is carried out in Sect. 2.4 to analyze their suitability in different static and dynamic situations. This comparative study is realized through simulation using a quadruped robot as testbed. The simulation features are described in Appendix. B.

Controlling the robot's COG motion so as to guarantee a given stability level could be useful for gait generation. Section 2.5 presents stability-level curves that can be useful for such purposes.

2.2 Static Stability Criteria

The first static stability criterion for an ideal machine walking at constant speed along a constant direction and over flat, even terrain was proposed by McGhee and Frank (1968). The Center of Gravity Projection Method claims that the vehicle is statically stable if the horizontal projection of its COG lies inside the support polygon (defined as the convex polygon formed by connecting footprints). Later this criterion was extended to uneven terrain (McGhee and Iswandhi, 1979) by redefining the support polygon as the horizontal projection of the real support pattern. The Static Stability Margin (S_{SM}) was defined for a given support polygon as the smallest of the distances from the COG projection to the edges of the support polygon. S_{SM} is the optimum stability margin for an ideal machine on horizontal, even terrain. However, the equation for calculating S_{SM} is complex. Thus, Zhang and Song (1989) proposed the Longitudinal Stability Margin (S_{LSM}), defined as the smallest of the distances from the COG projection to the front and rear edges of the support polygon along the machine's longitudinal axis. S_{LSM} is a good approximation to S_{SM}, and it is simpler to calculate. However, considering the

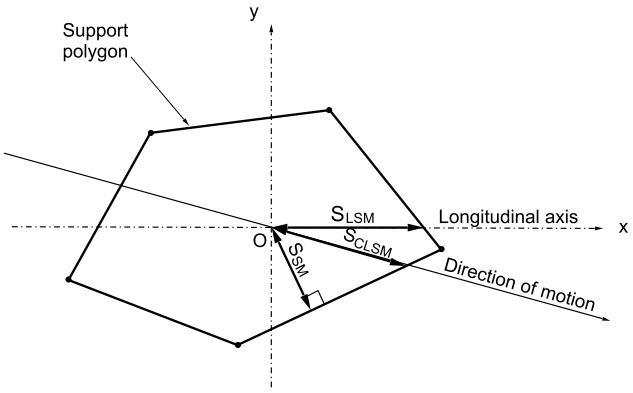


Fig. 2.1. Support polygon and different static stability margins

walking robot as a non-ideal vehicle where inertial effects arise during acceleration, the use of the Crab Longitudinal Stability Margin (S_{CLSM}) (Zhang and Song, 1990) will be more convenient. S_{CLSM} is the smallest of the distances from the *COG* projection to the front and rear edges of the support polygon along the machine's motion axis. Figure 2.1 shows S_{SM}, S_{LSM} and S_{CLSM} for a given support polygon.

Mahalingham *et al.* (1989) define the Conservative Support Polygon (*CSP*) as a subset of the support polygon, in order to limit the motion of the *COG* projection to guarantee system stability in the case of failure of any of the supporting legs. However, the use of the *CSP* is restricted to machines with six or more legs using a crawl gait.

The above stability criteria are all based on geometric concepts; S_{SM}, S_{LSM} and S_{CLSM} are independent of *COG* height and consider neither kinematic nor dynamic parameters. Intuitively speaking, the stability of a non-ideal walking machine should depend on those parameters.

A better stability measurement was proposed by Messuri (1985). He defined the Energy Stability Margin (S_{ESM}) as the minimum potential energy required to tumble the robot around the edges of the support polygon, that is:

$$S_{ESM} = \min_{i}^{n_s}(mgh_i) \tag{2.1}$$

where i denotes the segment of the support polygon considered the rotation axis, n_s is the number of supporting legs, and h_i is the variation of *COG* height during the tumble, which comes from

$$h_i = \mid \mathbf{R_i} \mid (1 - \cos\theta)\cos\psi \tag{2.2}$$

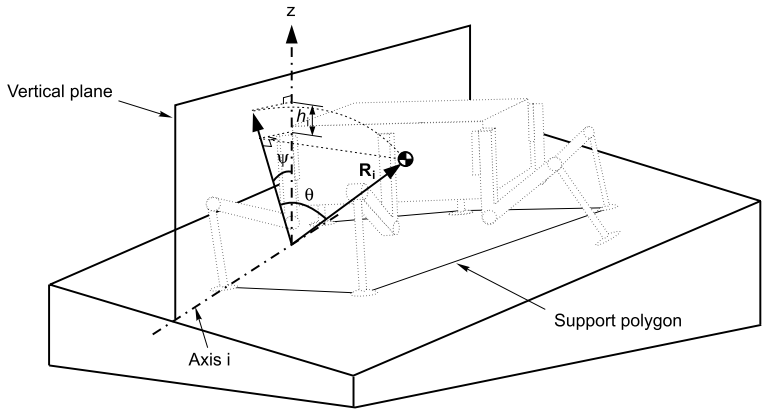


Fig. 2.2. Geometric outline for the computation of S_{ESM}

where $\mathbf{R_i}$ is the vector from the COG to the rotation axis, θ is the angle that $\mathbf{R_i}$ forms with the vertical plane, and ψ is the inclination angle of the rotation axis relative to the horizontal plane (see Fig. 2.2).

S_{ESM} is a more efficient static stability measurement. It gives a qualitative idea of the amount of impact energy the vehicle withstands and also considers the height of the COG. However, S_{ESM} still does not consider any dynamic effects that might disturb vehicle stability. S_{ESM} considers neither the effect of compliant terrains nor the stabilizing effect of a non-supporting leg. This is precisely what was proposed by Nagy (1991) as an extension of S_{ESM} that considers foot sinkage on soft and compliant terrain – the Compliant Energy Stability Margin (S_{CESM}), and Nagy also extended the concept to consider the stabilizing effect of a leg that is in the air – the Tipover Energy Stability Margin (S_{TESM}). For most walking machines, S_{ESM} and S_{TESM} coincide, because the non-supporting legs are too far from the floor to enhance stability. Only frame-based vehicles (Ishino *et al.*, 1983) will find this stability margin an advantage.

Finally, Hirose *et al.* (1998) normalized S_{ESM} to the robot weight and proposed the Normalized Energy Stability Margin (S_{NESM}), defined as

$$S_{NESM} = \frac{S_{ESM}}{mg} = \min_{i}^{n_s}(h_i). \tag{2.3}$$

S_{NESM} was shown to be the most efficient stability margin for statically stable walking machines. However, when dynamic effects arise during walking, machine stability cannot be judged precisely. Such situations exist in real walking robot applications, and therefore dynamic stability margins are more suitable.

2.3 Dynamic Stability Criteria

The first dynamic stability criterion for quadrupeds using crawl gaits was proposed by Orin (1976) as an extension of the Center of Gravity Projection Method. The Center of Pressure (*COP*) Method claims that a robot is dynamically stable if the projection of the *COG* along the direction of the resultant force acting on the *COG* lies inside the support polygon. The Dynamic Stability Margin is thus defined as the smallest distance from the *COP* to the edges of the support polygon – see also Gonzalez de Santos *et al.* (1998). The *COP* Method coincides with the Center of Gravity Projection Method under static conditions and uneven terrain. Thus, it presents the same limitations as mentioned in Sect. 2.2.

Kang *et al.* (1997) lately renamed the *COP* the Effective Mass Center (*EMC*), and redefined it as the point on the support plane where the resultant moment due to terrain-reaction forces and moments vanishes. Note that the definition of the *EMC* coincides with what in the literature of biped robots is commonly known as the Zero Moment Point (*ZMP*), first defined by Vukobratovic and Juricic (1969). However, the use of the *ZMP* in quadrupeds has been less extended than in bipeds because, as Yoneda and Hirose (1997) stated, the *EMC* or *ZMP* stability criterion is not valid for uneven terrain (to compute the *ZMP*, the support polygon has to be confined in a plane).

Some momentum-based stability criteria have been defined as well. Here only the most meaningful ones are reviewed. The statement is as follows. Given a robot-and-manipulator system as shown in Fig. 2.3(a), the forces and moments acting on the *COG* may destabilize it, making the system tumble. Dynamic equilibrium at the *COG* requires

$$\mathbf{F_I} = \mathbf{F_S} + \mathbf{F_G} + \mathbf{F_M} \tag{2.4}$$

$$\mathbf{M_I} = \mathbf{M_S} + \mathbf{M_G} + \mathbf{M_M} \tag{2.5}$$

where subscripts I, S, G and M denote inertia, support, gravitational and manipulation effects, respectively.

During the tumble the robot loses most of its support feet, leaving only those that conform a rotation axis (see Fig. 2.3(b)). A resultant interaction force $\mathbf{F_R}$ and moment $\mathbf{M_R}$ between robot and terrain result to counteract the addition of ground-reaction forces at every foot ($\mathbf{F_{r_i}}$) and the momentum they generate around the *COG*, respectively. For the robot to maintain stability a resultant force and moment generate a moment M_i about the rotation axis i that must compensate for the destabilizing forces and moments to ensure system stability. When such compensation is not enough, the system is said to be dynamically unstable.

Based on this statement, Lin and Song (1993) redefined the Dynamic Stability Margin (S_{DSM}) as the smallest of all moments M_i for every rotation axis in the support polygon, normalized to the weight of the system, that is:

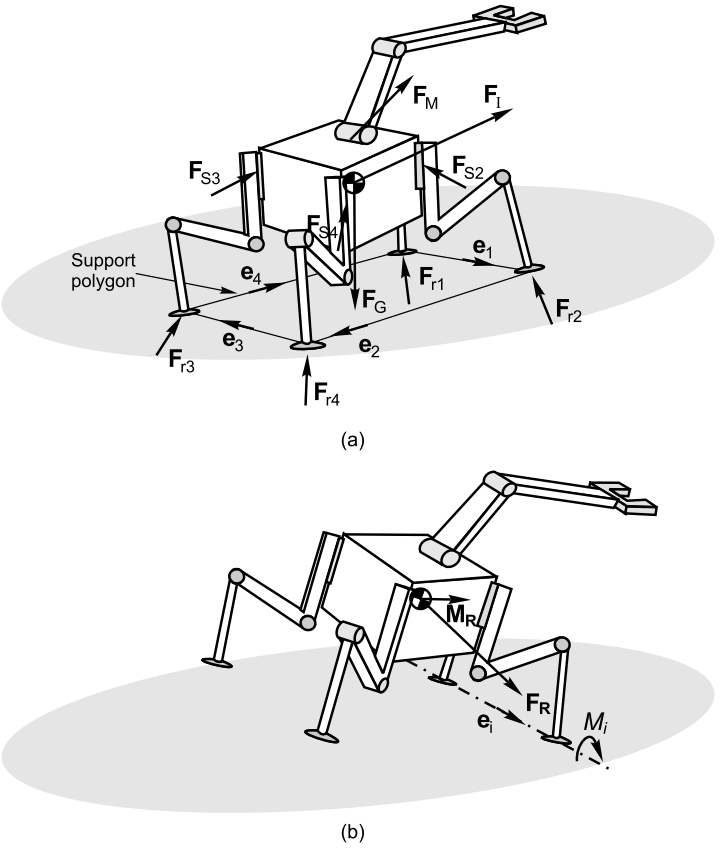


Fig. 2.3. (**a**) Forces acting on a robot + manipulator system; (**b**) Robot tumbling around the rotation axis

$$S_{DSM} = \min\left(\frac{M_i}{mg}\right) = \min\left(\frac{\mathbf{e_i} \cdot (\mathbf{F_R} \times \mathbf{P_i} + \mathbf{M_R})}{mg}\right) \tag{2.6}$$

where P_i is the position vector from the COG to the i-th support foot, and $\mathbf{e_i}$ is a unit vector that goes round the support polygon in the clockwise sense, as shown in Fig. 2.3(a). If all moments are positive (if they have the same direction and sense as $\mathbf{e_i}$), then the system is stable.

Note that the term S_{DSM} is used for both Orin's dynamic stability margin and Lin and Song's criterion but, in this chapter, the term S_{DSM} will be reserved for Lin and Song's criterion, while Orin's dynamic stability margin will be referred to as S_{ZMP}.

A few years later, Yoneda and Hirose (1997) proposed the Tumble Stability Judgment based on the same statement. In the dynamic equilibrium of the system, they assumed massless legs, so leg-support and foot-reaction forces coincide. In their study, Yoneda and Hirose use the resultant of ground-

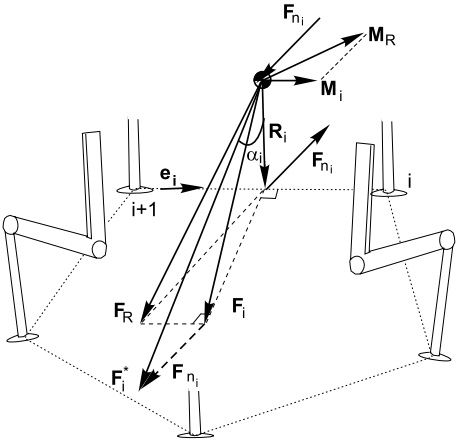


Fig. 2.4. Geometric problem of the Force-Angle stability margin

reaction forces $-\mathbf{F_R}$ and moments $-\mathbf{M_R}$ which are obtained from the dynamic equilibrium of the system as follows:

$$-\mathbf{F_R} = \mathbf{F_I} - \mathbf{F_G} - \mathbf{F_M} \tag{2.7}$$

$$-\mathbf{M_R} = \mathbf{M_I} - \mathbf{M_G} - \mathbf{M_M}. \tag{2.8}$$

Thus, the moment M_i around the rotation axis is calculated as follows:

$$M_i^{'} = -\mathbf{M_R} \cdot \mathbf{e_i} - \mathbf{F_R} \times \mathbf{p_i} \cdot \mathbf{e_i}. \tag{2.9}$$

Note that the moment in (2.9) is exactly the opposite of the moment used in (2.6).

The Tumble Stability Judgment states that the system is dynamically stable if there exists any support foot j in the direction of rotation that prevents the system from tumbling. Then, the Tumble Stability Margin, S_{TSM}, becomes

$$S_{TSM} = \min\left(\frac{|M_i^{'}|}{mg}\right). \tag{2.10}$$

Recently Zhou *et al.* (2000) proposed the Leg-end Supporting Moment criterion. Its stability margin is exactly the same as S_{TSM}, but its users obtain the resultant force $\mathbf{F_R}$ and moment $\mathbf{M_R}$ from force sensors at the feet. Therefore, S_{LESM} avoids the sort of errors that can appear in S_{TSM} from neglecting leg dynamics.

Apart from *ZMP*-based and momentum-based stability criteria, a different criterion was proposed by Papadopoulos and Rey (1996). The Force-Angle stability criterion finds the angle α_i between the resultant force acting from

the *COG* on the ground ($\mathbf{F_R}$) – the opposite to the reaction force – and the vector $\mathbf{R_i}$, normal to the rotation axis from the *COG* (see Fig. 2.4). The system becomes unstable when this angle becomes zero. The Force-Angle stability margin is the product of the angle times the module of resultant force $\mathbf{F_R}$, that is:

$$S_{FASM} = \min(\alpha_i)\|\mathbf{F_R}\|. \tag{2.11}$$

Later on, some researchers studied the measurement of robot stability from the energy viewpoint. Based on previous work on static stability margins that demonstrated that S_{ESM} is optimum under static conditions (Hirose *et al.*, 1998) – *e.g.* when the only significant force acting on the robot is gravity – some researches propose to extend S_{ESM} concept to the presence of other robot dynamics, like inertial forces or manipulation effects. S_{ESM} is computed from the increase of potential energy that the machine's *COG* experiences when pivoting around the edges of the support polygon (see Sect. 2.2). Therefore, the extension of S_{ESM} to the presence of other robot dynamics should compute the increase of mechanical energy that the *COG* experiences during the tumble. This idea was first proposed by Ghasempoor and Sepehri (1998) to measure robot stability in the application to wheel-based mobile manipulators. Later, Garcia and Gonzalez de Santos (2005) extended Ghasempoor and Sepehri's idea to walking machines, considering leg dynamics as a destabilizing effect, and normalized it to the robot's weight. The resulting Normalized Dynamic Energy Stability Margin, S_{NDESM}, was validated through simulation and it was shown to quantify robot stability accurately on uneven terrain in the presence of manipulation dynamics and external perturbations.

The Normalized Dynamic Energy Stability Margin is defined as the smallest of the stability levels required to tumble the robot around the support polygon, normalized to the robot mass, that is (see Fig. 2.5):

$$S_{NDESM} = \frac{\min(E_i)}{mg} \tag{2.12}$$

where E_i stands for the stability level of the i-th side of the support polygon, which physically means the increment of mechanical energy required to tumble the robot around the i-th side of the support polygon, computed from

$$E_i = mg|\mathbf{R}|\ (\cos\phi - \cos\varphi)\ \cos\Psi + (\mathbf{F_{RI}}\cdot\mathbf{t})|\mathbf{R}|\ \theta + (\mathbf{M_R}\cdot\mathbf{e_i})\ \theta - \frac{1}{2}I_i\omega_i^2 \tag{2.13}$$

where $\mathbf{R}$ is a vector orthogonal to the i-th side of the support polygon that points to the *COG* position, $\mathbf{F_{RI}}$ is the non-gravitational component of the resultant robot/ground interaction force $\mathbf{F_R}$, I_i is the moment of inertia around the rotation axis i, ω_i is the angular velocity of the *COG*, Ψ is the inclination angle of the i-th side of the support polygon, and ϕ, φ and θ are the rotation angles around the i axis. φ is the rotation angle required to position the *COG* inside the vertical plane (see Fig. 2.5); ϕ is the angle that the *COG* rotates

from the vertical plane to the critical plane, where the resultant moment acting on the COG vanishes. Finally, θ is the addition of both rotations. Unitary vector $\mathbf{t}$ is tangent to the COG trajectory and $\mathbf{e_i}$ is the unitary vector that goes around the support polygon in the clockwise sense. The first three terms on the right side of (2.13) represent the potential energy required for the tumble caused by gravitational and non-gravitational forces and moments, while the fourth term represents kinetic energy – see Garcia and Gonzalez de Santos (2005) for a more detailed explanation. S_{NDESM} is the only dynamic stability margin that considers the effect of impulsive, external disturbances on robot stability.

These are the main stability criteria used today in walking machine control. Some of them seem to define the same stability margins, and there are no clear differences between them. Furthermore, their suitability for judging stability in any real situation, *e.g.* on sloped terrain, or where there are manipulation forces and moments or dynamic effects during the transfer of the legs, cannot be extracted directly from the definitions.

The following section is devoted to coping with this lack of qualitative information about existing stability margins. For this purpose, a comparative analysis of stability margins is presented for different static and dynamic situations and different terrain profiles.

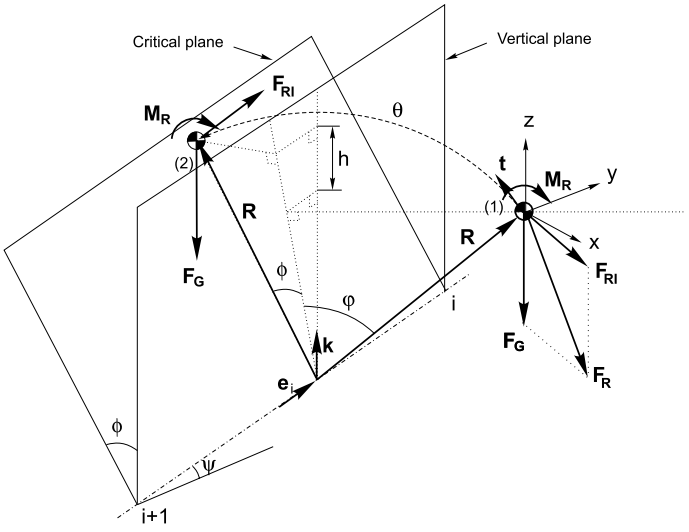


Fig. 2.5. Geometric outline for the computation of S_{NDESM}

2.4 A Comparative Study of Stability Margins

The goal of this comparative study is to produce a qualitative classification of stability margins to determine which is most suitable for each given application. The stability margins that have been selected for the analysis are S_{SM}, S_{NESM}, S_{DSM}, S_{TSM}, S_{FASM}, S_{ZMP} (also called S_{EMC}) and S_{NDESM} . They have been computed while the robot was walking using a two-phase discontinuous gait (see Sect. 3.4) in the following six different terrain and dynamic situations:

Case 1: Horizontal, even terrain in the absence of dynamics.
Case 2: Uneven terrain in the absence of dynamics.
Case 3: Horizontal, even terrain when inertial and elastic effects arise.
Case 4: Uneven terrain when inertial and elastic effects arise.
Case 5: Horizontal, even terrain when inertial, elastic and manipulation dynamics arise.
Case 6: Uneven terrain when inertial, elastic and manipulation dynamics arise.

The above six case studies represent different situations that quadruped robots can find in real applications. Due to the difficulty of utilizing various quadruped robots that comply the different specifications of each six cases (*i.e.* absence of robot dynamics in Cases 1 and 2, existence of manipulation dynamics in Cases 5 and 6, *etc.*), this analysis is carried out through simulation of a quadruped robot, using the simulation construction set that is described in Appendix B. Thus, different terrain profiles and different robot dynamics have been explored by using this simulation software. The two-phase discontinuous gait used in the simulations (see Sect. 3.4) is characterized by a sequence of leg and body motions. The leg sequence is performed by transferring one leg at a time, while the body is supported on the other three legs. The body is moved forward with all four legs on the ground (body motion).

Figures 2.6 and 2.7 show one-half of the gait cycle for Cases 1 and 2, respectively, that consist of the swing of a rear leg, the swing of the collateral front leg and a body motion. From both figures it can be observed that S_{SM}, S_{DSM}, S_{TSM} and S_{ZMP} coincide. The margins also coincide for different heights of the COG (dotted line). It is relevant that in the first case study (see Fig. 2.6), when the terrain is horizontal and even, these four margins do not vary with COG height. This is a drawback of all four criteria because, obviously, the increase of the COG height has a destabilizing effect. However, in the second case study, for uneven terrain (see Fig. 2.7), all six margins consider COG height. The vertical dashed line inside the body-support-phase interval points to the instant when S_{NDESM} is maximum; on horizontal terrain this instant is one-half of the support-phase interval. For the first case study that instant coincides for all margins.

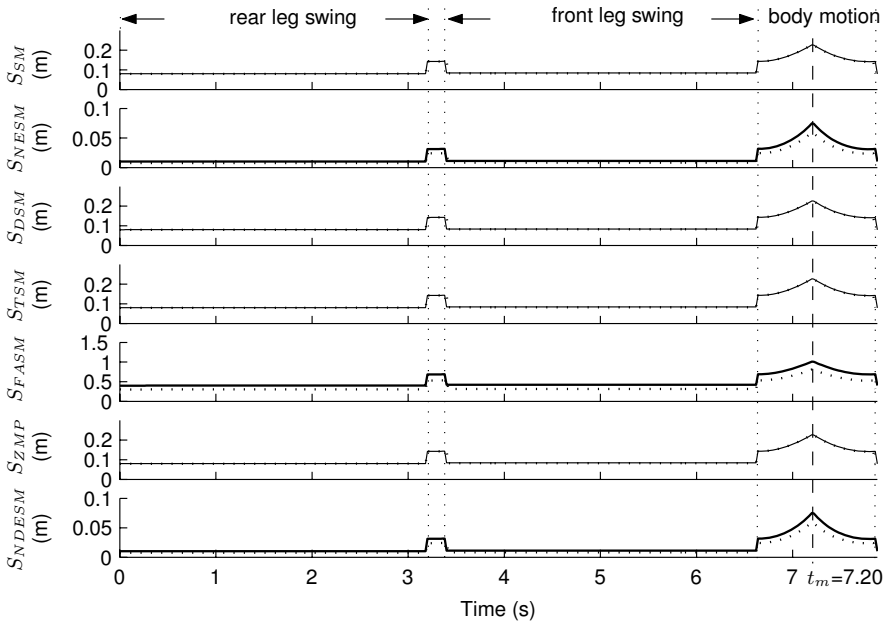


Fig. 2.6. Different stability margins in the absence of system dynamics on horizontal terrain (Case 1) for a robot height of 0.32 m (*solid line*) and for a robot height of 0.42 m (*dotted line*)

S_{NESM}, S_{FASM} and S_{NDESM} are the only margins that reflect the effect of body height increase on horizontal and even terrain. Thus, they are the only margins that give a successful stability measurement for Case 1.

On an inclined surface (see Fig. 2.7), S_{NESM}, S_{FASM} and S_{NDESM} differ from the others in their instant of maximum stability. S_{NESM} and S_{NDESM} coincide, and they reach maximum after S_{SM}, S_{DSM}, S_{TSM} and S_{ZMP} (which coincide too). Also the maximum S_{FASM} occurs even later than the maximum S_{NDESM}. The main question at this point seems to be which of the margins is the best for Case study 2? Hirose *et al.* (1998) carried out an experiment to determine which static stability margin was the most appropriate. They made a small model of walking robot and used a shock-generating device to produce an impact force against the robot model. The robot model was placed on inclined ground and the impact was exerted from both downhill and uphill sides respectively for the robot model's *COG* placed on each of the stability margin's maximum position. The experiment concluded that when the *COG* was placed at the maximum S_{NESM} point, the possibility of tumbling downhill was equal to uphill, and therefore, S_{NESM} is the most appropriate static-stability margin. S_{NESM} measures the impact energy that the system can absorb during the tumble. Considering that our second case study coincides

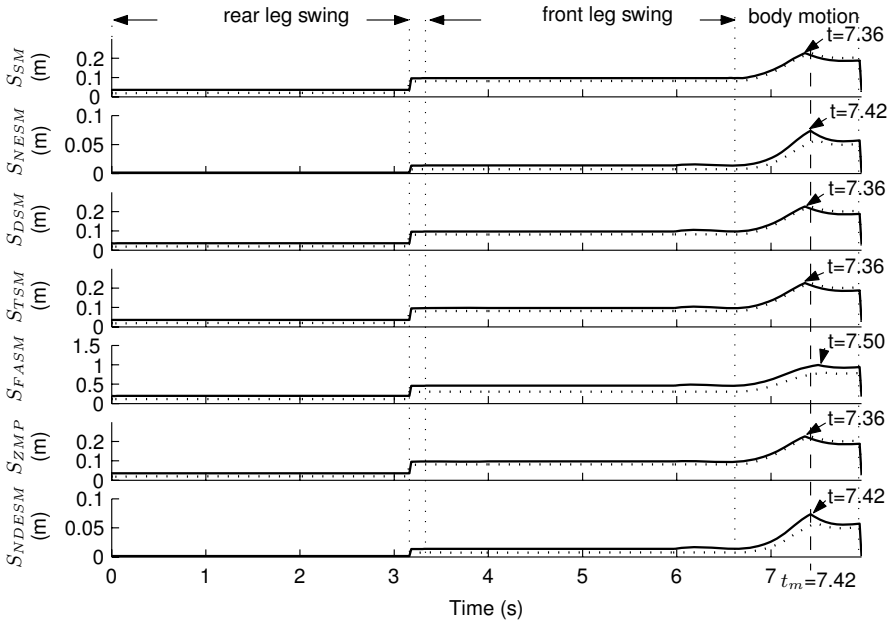


Fig. 2.7. Different stability margins in the absence of system dynamics on terrain inclined 10^o from the horizontal plane (Case 2) for a robot height of 0.32 m (*solid line*) and for a robot height of 0.42 m (*dotted line*)

with the experiment carried out by Hirose *et al.*, S_{NESM} and S_{NDESM} are the most suitable for a quadruped on inclined terrain when robot dynamics are negligible.

To show that the instant of maximum stability of the various stability margins always differ from the instant of maximum S_{NDESM} and S_{NESM}, and considering that S_{SM}, S_{DSM}, S_{TSM} and S_{ZMP} all coincide for this case study, and that S_{NESM} and S_{NDESM} coincide too, next we compare the stability measurements of S_{SM}, S_{NDESM} and S_{FASM} for different terrain inclinations. Figure 2.8(a) shows the difference between the instants of maximum S_{NDESM} and S_{SM} as a function of the terrain inclination angle, while Fig. 2.8(b) shows the difference between the instants of maximum S_{FASM} and S_{NDESM}. Figure 2.8(a) demonstrates that the instant of maximum S_{SM} always precedes the instant of maximum S_{NDESM} for different positive and negative terrain inclinations. Furthermore, the instant of maximum S_{FASM} always follows the instant of maximum S_{NDESM}, as shown in Fig. 2.8(b) for different terrain inclinations. Therefore, the instants of maximum S_{SM} and S_{FASM} only coincide with the instant of maximum S_{NDESM} when the terrain is horizontal and even. If there is a slope in the terrain, S_{SM} and S_{FASM} will never be the most suitable margins.

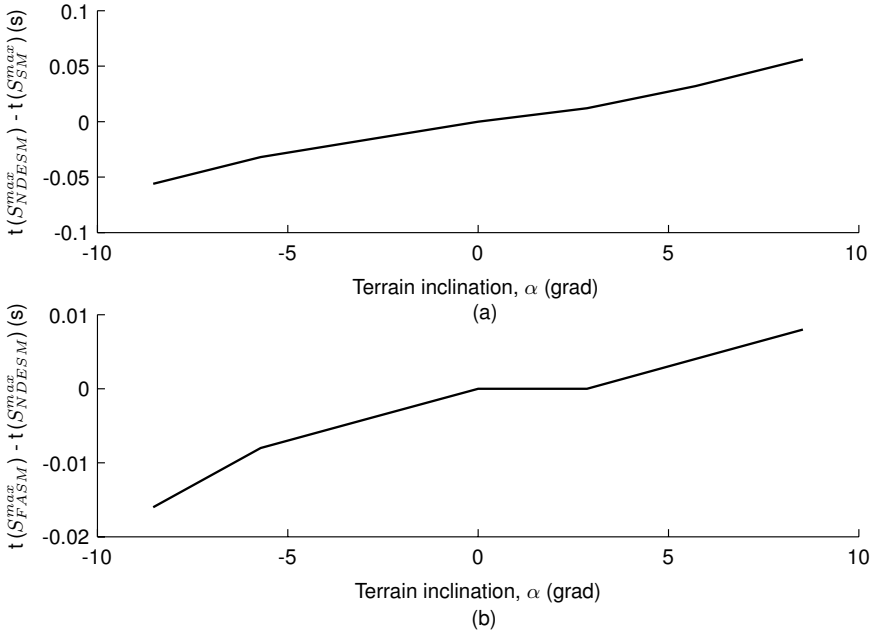


Fig. 2.8. Difference between instants of maximum stability for several terrain inclinations: (**a**) S_{NDESM} and S_{SM}; (**b**) S_{FASM} and S_{NDESM}

Figures 2.9 and 2.10 show one-half of the gait cycle for Cases 3 and 4 respectively, which correspond to the existence of inertial effects when the robot is walking over horizontal and inclined terrain, respectively. Elastic effects due to joint elasticity and ground contact are introduced as well.

On horizontal terrain (see Fig. 2.9), all the instants of maximum stability still coincide. However, S_{DSM}, S_{TSM}, S_{FASM}, S_{ZMP} and S_{NDESM} reflect some oscillation of the margin due to joint elasticity at leg lift, placement and body motion. Inertial effects during the leg transfer (swing) phase and body motion are reflected as well. These dynamic effects are not reflected by S_{SM} and S_{NESM}, because they are static stability margins. Figure 2.11 shows this difference, plotting a comparison between S_{SM} and S_{DSM}. S_{DSM} undergoes a decrease in stability due to inertial effects at leg lift and body propulsion. Vibrations due to joint elasticity are also reflected at leg lift, foot placement and body motion.

Therefore, only dynamic stability criteria are valid for judging stability when inertial and elastic effects are involved. However, S_{DSM}, S_{TSM} and S_{ZMP} have the same failing here as on horizontal terrain: they do not consider the effect of height changes. Only S_{FASM} and S_{NDESM} are suitable for Case study 3, for a quadruped walking on flat ground and robot dynamics are not negligible.

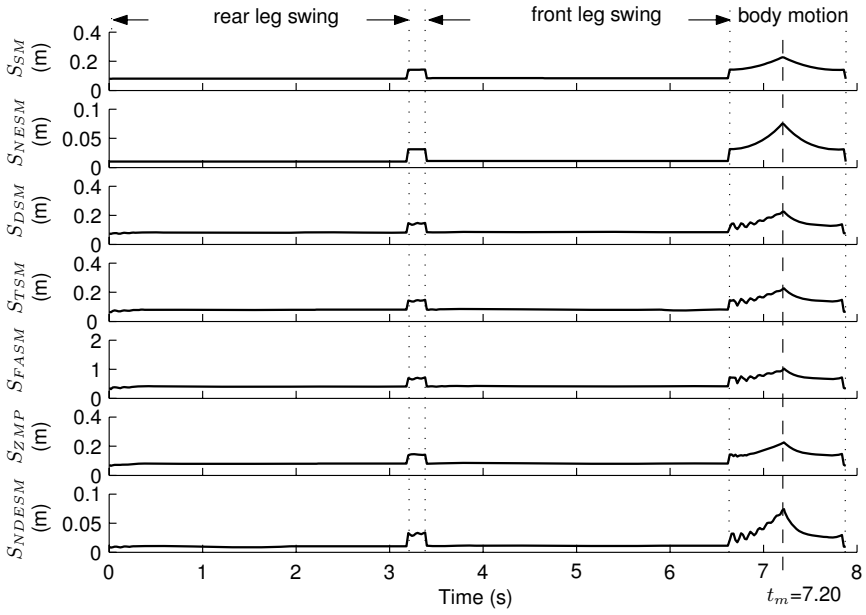


Fig. 2.9. Different stability margins when inertial and elastic effects arise on horizontal terrain (Case 3)

Figure 2.10 depicts stability margins for Case study 4, when the terrain is inclined and inertial and elastic effects become significant. While S_{SM} and S_{NESM} do not reflect any reduction of the stability margin due to dynamics (as they are the same as in Fig. 2.7), S_{DSM}, S_{TSM}, S_{FASM}, S_{ZMP} and S_{NDESM} reflect a decrease in stability. The maximum stability instant of S_{FASM} occurs later than that of S_{DSM}, S_{TSM}, S_{ZMP} and S_{NDESM}. However, the maximum S_{NDESM} takes place after S_{DSM}, S_{TSM} and S_{ZMP}. Once again, considering these differences between the stability margins, we would like to determine which stability margin is the most suitable for a quadruped walking on uneven terrain when robot dynamics are significant. To do so, we have simulated an experiment similar to the one that Hirose *et al.* (1998) performed to determine the most suitable static stability margin on uneven terrain. In this case a quadruped is simulated considering its inertia and elasticity. A 25-N external force opposing the robot's motion was simulated and it caused the robot to tumble down. Dimensionless stability margins have been computed and scaled in order to permit numerical comparison. For this purpose, S_{NDESM} has been divided by the robot height ($H = 0.34\,\mathrm{m}$), and S_{DSM} has been divided by half the stroke pitch ($P/2 = 0.5\,\mathrm{m}$). Afterwards, the three dimensionless numbers have been scaled in such a way that at the beginning of the motion (when no external disturbances exist and the robot is stopped)

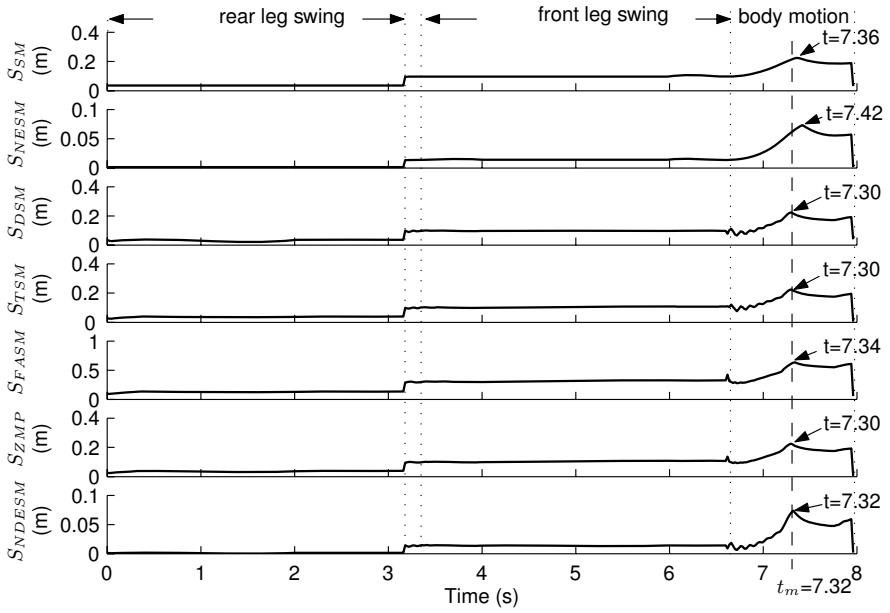


Fig. 2.10. Different stability margins when inertial and elastic effects arise on terrain inclined 10^o from the horizontal plane (Case 4)

they have the same value. It seems reasonable that under such normal conditions all dimensionless stability margins give the same value. Figure 2.12(a) shows the three dimensionless stability margins before and after the tumble occurs (at $t = 0.1$ s). After the tumble the three stability margins become zero, just because the robot becomes unstable, and that prevents any stability margin to be computed. However, before the tumble, the three stability margins behave differently. S_{FASM} reflects a delay in measuring the stability decrease just before the tumble, while S_{DSM} and S_{NDESM} show the stability decrease from the beginning of the motion. Nevertheless, S_{DSM} exhibits a discontinuity at the instant of tumble. This is clarified in Fig. 2.12(b), where derivatives of the three stability margins are shown. An impulse on derivatives of S_{DSM} and S_{FASM} reveals an error in the instability prediction. These discontinuities do not seem to be so, due to the fact that the data shown in the figure has been sampled at 0.02 s from the simulation data. However, stability margins become zero abruptly because when the robot becomes unstable no stability margin is computed (the support polygon disappears). However, if stability margins could have been computed, S_{FASM} and S_{DSM} surely would not have become zero, as shown by the tendency of both curves in Fig. 2.12(a) at that instant of time. As Fig. 2.12 shows, S_{NDESM} becomes zero continuously, and thus no prediction error exists. Therefore, S_{NDESM} has no error in the measurement

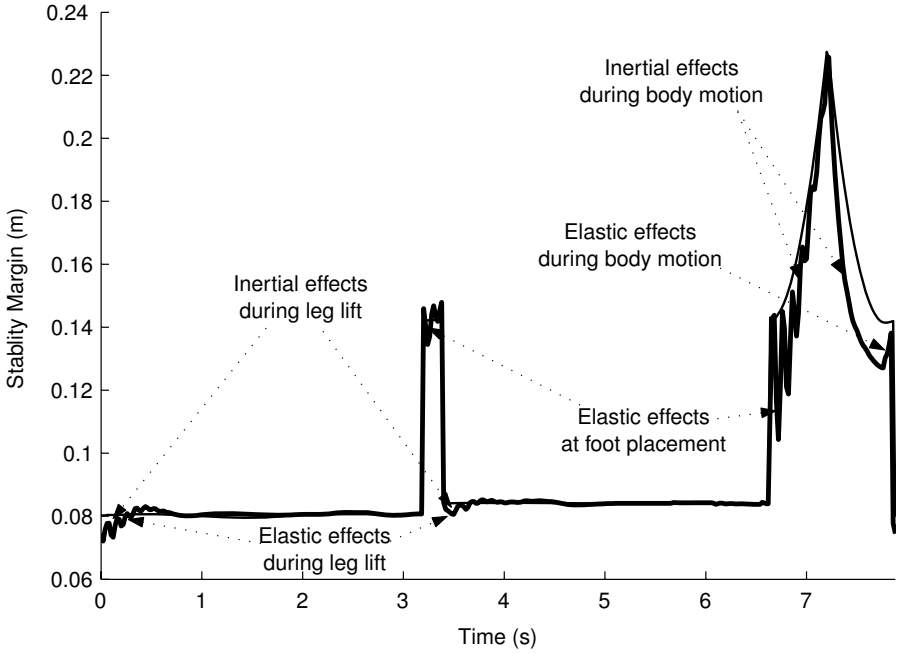


Fig. 2.11. S_{SM} (*thin line*) and S_{DSM} (*thick line*) during half gait cycle on horizontal terrain when inertial and elastic effects arise

of robot stability and can be used to predict robot instability precisely. This shows clearly the advantage of S_{NDESM} which has been shown to be the only exact stability measurement. Just before the instant $t = 0.12\,\mathrm{s}$ when the robot starts to fall, only $S_{NDESM} = 0$. The rest of stability margins would give a margin different from zero. This is critical for robot control. If a robot gait is controlled in such a way that the stability margin must always be over a certain value, the use of other stability margin different from S_{NDESM} will impose an error in the monitoring of the stability margin, and robot stability will be uncertain.

Figures 2.13 and 2.14 show one-half of the gait cycle for Cases 5 and 6, respectively, which correspond to the existence of manipulation effects when the robot walks over horizontal and inclined terrain, respectively. Inertial and elastic effects are considered as well. Both figures show that manipulation forces opposing motion cause a stability decrease at the rear leg's swing phase and an increase at the front leg's swing phase. Also, a delay of the maximum stability instant can be observed in S_{DSM}, S_{TSM}, S_{FASM}, S_{ZMP} and S_{NDESM}. It is obvious from the figure that if the manipulation force is increased the robot could be destabilized during the swing of the rear leg. This will never be foreseen by S_{SM} and S_{NESM}.

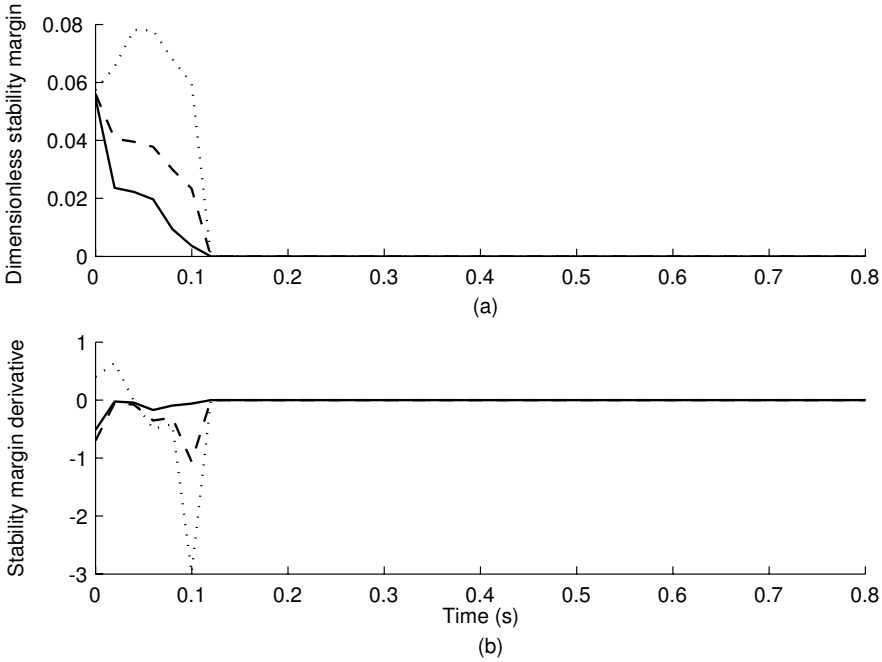


Fig. 2.12. (**a**) Dimensionless S_{DSM} (*dashed line*), S_{NDESM} (*solid line*) and S_{FASM} (*dotted line*) when instability takes place due to an external force of $-25\,\mathrm{N}$; (**b**) Dimensionless S_{DSM}, S_{NDESM} and S_{FASM} derivatives when instability takes place due to an external force of $-25\,\mathrm{N}$

Once again, the instant of maximum S_{FASM} takes place after the instant of maximum S_{DSM}, S_{TSM}, S_{ZMP} and S_{NDESM}. However, the maximum S_{NDESM} is reached after S_{DSM} and before S_{FASM}. This is shown in Fig. 2.15, where the instants of maximum S_{DSM} and S_{FASM} are compared with the maximum S_{NDESM} for different terrain inclination angles and different manipulation forces. As both figures show, neither S_{DSM} nor S_{FASM} coincide with S_{NDESM}, which results in the most suitable stability margin for a quadruped subject to manipulation dynamics.

Table 2.1 summarizes a classification of the stability margins studied herein. The symbol "$\surd$" denotes that the criterion is "valid," the symbol "$\times$" denotes "not valid," and the symbol "$*$" denotes "most suitable." S_{NESM} and S_{NDESM} are the most suitable measurements as static stability margins. However, all the rest are valid. As dynamic stability margins, S_{SM} and S_{NESM} are not valid. When the quadruped walks over horizontal terrain and robot inertia is significant, S_{FASM} and S_{NDESM} are the most adequate measurements, yet the rest of the dynamic stability criteria are valid. When any other dynamic effects are present, such as manipulation forces and moments, over horizontal or sloping terrain, S_{NDESM} is the most suitable. As a result

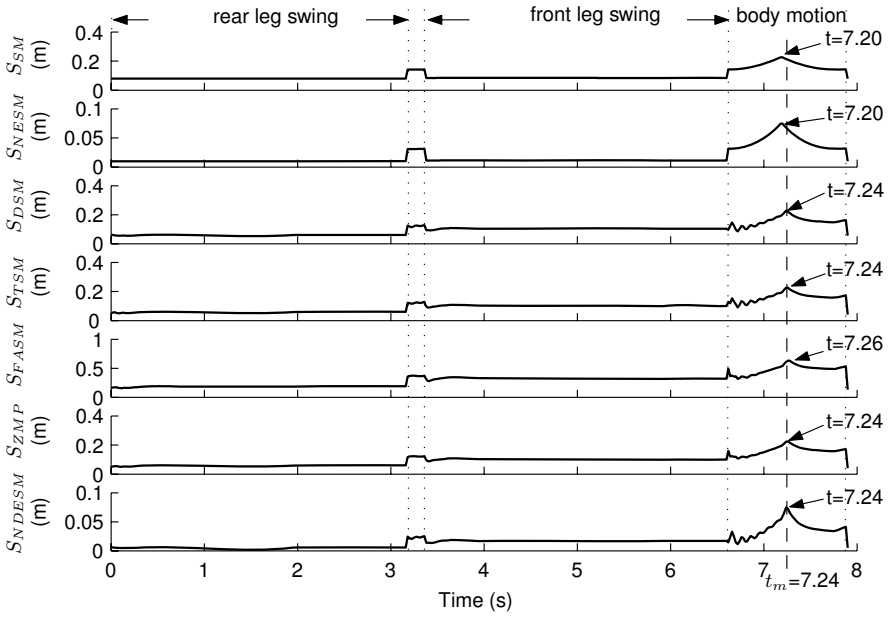


Fig. 2.13. Different stability margins when inertial, elastic and manipulation effects arise with a 20-N constant force opposing motion on horizontal terrain (Case 5)

Table 2.1. Classification of stability criteria

Uneven terrain	Robot dynamics	Manipulation dynamics	S_{SM}	S_{NESM}	S_{DSM}	S_{TSM}	S_{FASM}	S_{ZMP}	S_{NDESM}
No	No	No	$\surd$	*	$\surd$	$\surd$	*	$\surd$	*
No	Yes	No	$\times$	$\times$	$\surd$	$\surd$	*	$\surd$	*
No	Yes	Yes	$\times$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	*
Yes	No	No	$\surd$	*	$\surd$	$\surd$	$\surd$	$\surd$	*
Yes	Yes	No	$\times$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	*
Yes	Yes	Yes	$\times$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	*

of the comparative study, only S_{NDESM} is the most suitable margin for every case studied. Another conclusion of this study is that S_{DSM}, S_{TSM} and S_{ZMP} yield the same measurement for every situation studied herein.

The last comparison of the selected criteria according to their computational complexity was obtained by finding the number of mathematical operations required for simulation. Table 2.2 shows these data, represented as the number of additions, multiplications, trigonometric operations and square roots computed in each simulation step, considering a support polygon of n sides. In this calculation, foot reaction forces are assumed to be known. As

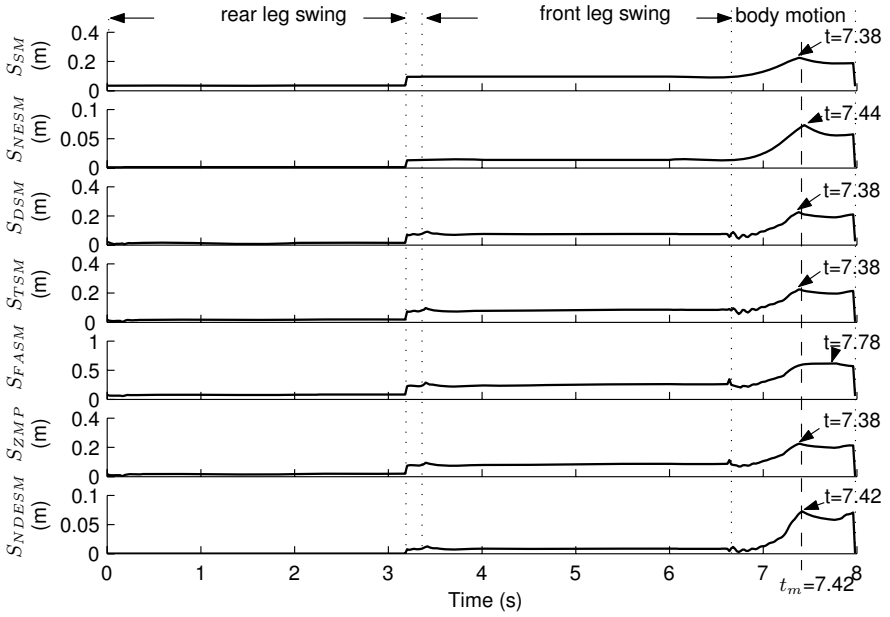


Fig. 2.14. Different stability margins when inertial, elastic and manipulation effects arise with a 20-N constant force opposing motion on terrain inclined 10^o from the horizontal plane (Case 6)

Table 2.2. Computational complexity of existing stability criteria

	S_{SM}	S_{NESM}	S_{DSM}	S_{TSM}	S_{FASM}	S_{ZMP}	S_{NDESM}
Additions	17n	33n	44n	86n	109n	67n	60n
Products	13n	23n	39n	90n	117n	70n	57n
Trigonometric	-	-	-	-	3n	-	3n
Square roots	n	2n	2n	3n	6n	2n	3n

the table shows, S_{FASM} is the most complex of the compared margins, while S_{DSM} is the least complex of the dynamic criteria and S_{SM} is the least complex of the static criteria. S_{NDESM}, which has been proved to be the most suitable stability margin for every studied situation, is one of the least complex of dynamic-stability margins.

2.5 Stability-level Curves

Based on the different stability margins defined in Sects. 2.2 and 2.3, the robot's *COG* trajectory can be controlled so as to guarantee a given stability

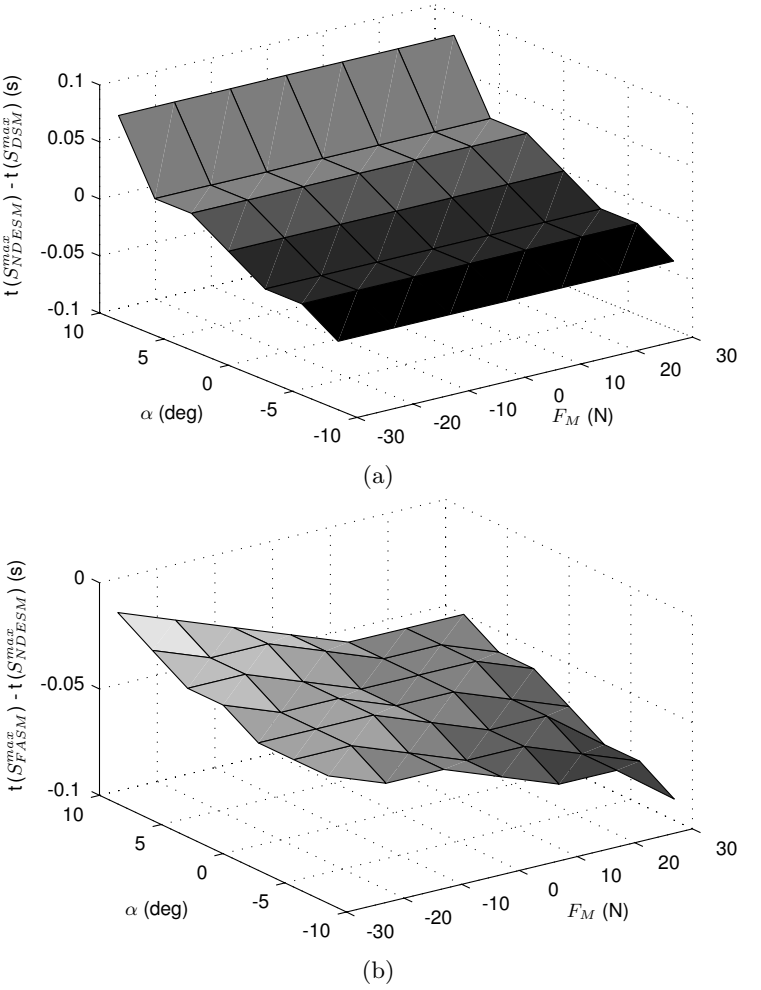


Fig. 2.15. Difference between instants of maximum stability for several terrain inclinations (α) and manipulation forces (F_M): (**a**) S_{DSM} *vs.* S_{NDESM}; (**b**) S_{FASM} *vs* S_{NDESM}

level. For this purpose, stability-level curves can be defined as the COG locus of equal stability margin inside the body plane (see Fig. 2.16(a)) which is defined by the longitudinal and transverse robot axes (x and y) and the COG position. Stability-level curves have been previously defined using S_{ESM} (Messuri, 1985) and S_{NESM} (Hirose *et al.*, 1998). Considering the results obtained in the comparative study of Sect. 2.4, in this section, stability-level curves are obtained using S_{NDESM}, which has been proved to be the most

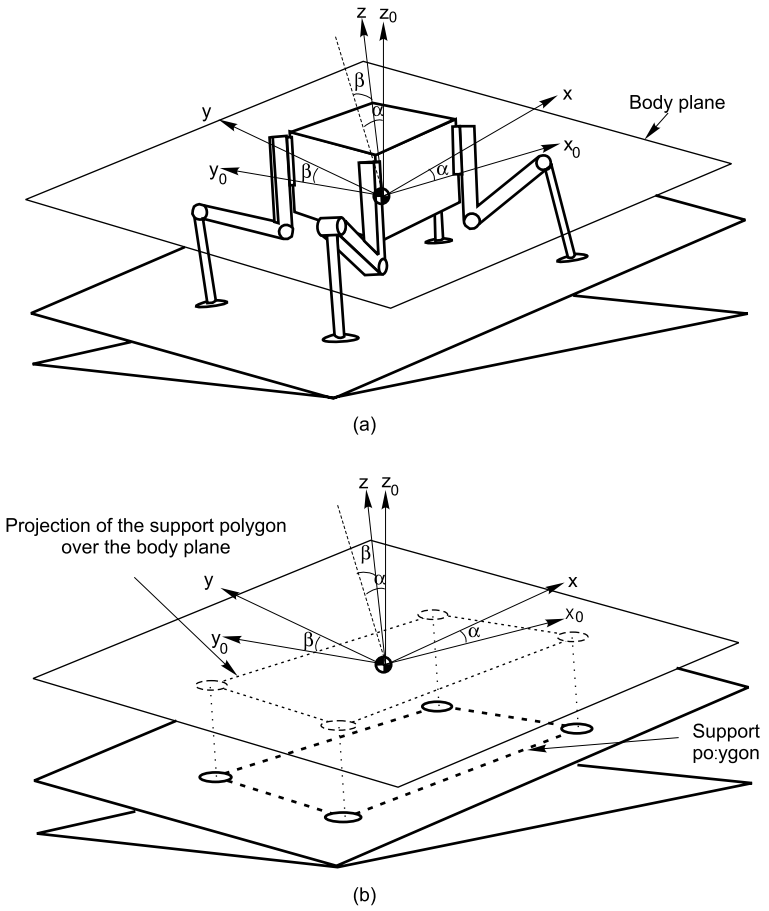


Fig. 2.16. Quadruped on support: (**a**) body reference frame $(x\,y\,z)$ and external reference frame $(x_0\,y_0\,z_0)$; (**b**) projection of the support polygon on the body plane

suitable stability margin. The stability-level curves are given by the following expression:

$$S_{NDESM}(COG_x, COG_y) = C \tag{2.14}$$

where COG_x and COG_y are COG coordinates with reference to a body reference frame $x\,y\,z$ (see Fig. 2.16 and C is a constant. The support polygon and the forces and moments acting on the robot are known with reference to an external reference frame $x_0\,y_0\,z_0$. Therefore, to solve (2.14) S_{NDESM} must be expressed in terms of variable COG-coordinates in the external reference frame and later mapped onto the body reference frame.

Let us name the initial-position vector of the body reference frame in the external reference frame $\mathbf{COG_{I0}}$, that is:

$$\mathbf{COG_{I0}} = (COG_{Ix0}\ COG_{Iy0}\ COG_{Iz0})^T. \tag{2.15}$$

Any motion of the COG to a different point $COG = (COG_x,\ COG_y,\ 0)$ in the body plane can be mapped into the external reference frame by means of the following homogeneous transformation:

$$\begin{pmatrix} COG_{x0} \\ COG_{y0} \\ COG_{z0} \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\alpha & \sin\alpha\ \sin\beta & \sin\alpha\ \cos\beta & COG_{Ix0} \\ 0 & \cos\beta & -\sin\beta & COG_{Iy0} \\ -\sin\alpha & \cos\alpha\ \sin\beta & \cos\alpha\ \cos\beta & COG_{Ixz} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} COG_x \\ COG_y \\ 0 \\ 1 \end{pmatrix} \tag{2.16}$$

where α is the angle between the x- and x_0-axes, and β is the angle between the y- and y_0-axes (see Fig. 2.16(b)).

To solve (2.14) S_{NDESM} must be expressed in terms of variable COG coordinates COG_{x0}, COG_{y0}, COG_{z0} and later mapped onto the body reference frame through (2.16). As a result, S_{NDESM} will be expressed in terms of body-plane coordinates COG_x, COG_y.

The analytic solution of (2.14) yields a complex expression. For the sake of clarity it has been solved numerically for different situations and results are plotted in Figs. 2.17 and 2.18, which show stability-level curves for a quadruped in its support phase when dynamic effects are considered. Footprint projections onto the body plane are marked with a cross.

Figure 2.17 shows stability-level curves for a robot subject to a 30-N manipulation force along the y_0-axis and a 20-Nm manipulation torque around the y_0-axis. The robot remains at rest. The resultant force causes a moment around the x-axis, and therefore the critical plane forms an angle ϕ with the vertical plane for the two robot sides parallel to the x-axis. The zero-stability curve shifts from the support polygon due to this effect. Likewise, the manipulation torque around the y-axis causes the angle between the critical plane and the vertical plane for the two robot sides parallel to the y-axis. Manipulation forces and torques also modify the gradient between stability-level curves.

Also, stability-level curves are plotted for the same situation as in Fig. 2.17 but while the robot is in motion, propelled by its four legs, that is, $\mathbf{v_{COG}} \neq 0$. Under such conditions an initial kinetic energy exists. Figure 2.18 shows an example where the COG moves at a constant speed of 0.2 m/s along the x-axis. As a result, stability-level curves are squeezed in the x direction. Therefore robot stability decreases when the body moves.

2.6 Conclusions

Several stability margins have been defined in the course of research on walking robots, yet none of their definitions directly suggest anything about their

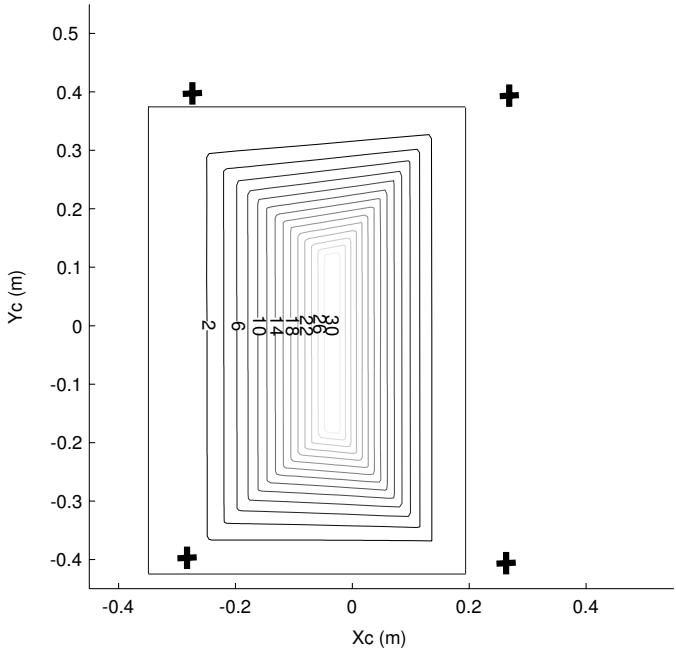


Fig. 2.17. Stability-level curves (m) over terrain inclined 20° in the x_0 direction, horizontal body plane. The robot is subject to a 30-N manipulation force along the y_0-axis and a manipulation moment of 20 Nm around the y_0-axis.

suitability to judge stability in any real situation, *e.g.* on sloped terrain, or in the presence of manipulation forces and moments or dynamic effects during leg transfer.

This chapter has been devoted to coping with this lack of qualitative information about existing stability margins. For this purpose, after surveying static and dynamic stability margins, a comparative study has been run on stability margins in different static and dynamic situations. This analysis has been carried out through simulation of a walking robot using a two-phase discontinuous gait in six different case studies where various terrain profiles and dynamic situations have been considered. These case studies cover all the situations that can occur during real industrial applications of legged robots.

As a result, a classification of stability criteria has been presented showing that the Normalized Dynamic Energy Stability Margin is the most suitable stability margin for every situation studied. Also, it has been shown that every momentum-based stability criterion provides the same stability margin. The selected criteria have been also compared in terms of their computational complexity. This classification enables the proper stability criterion to be chosen for each real application.

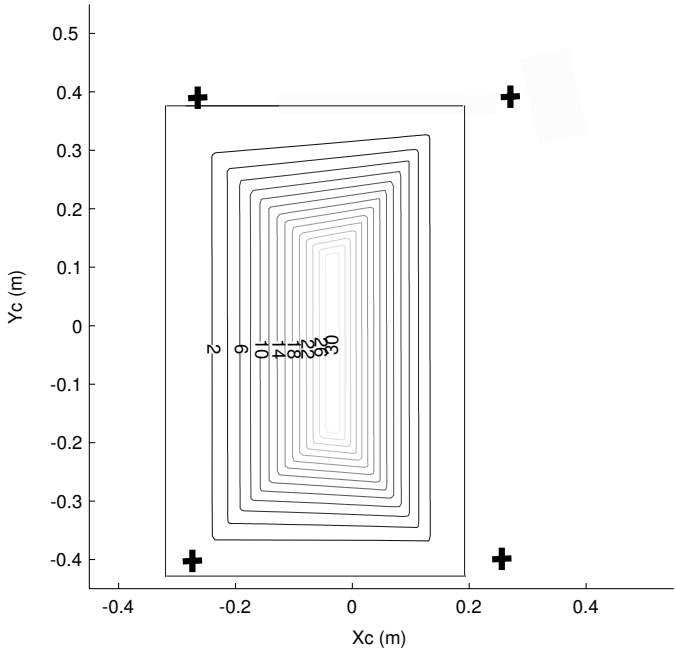


Fig. 2.18. Stability-level curves (m) over terrain inclined 20° in the x_0 direction, horizontal body plane. The robot is subject to a 30-N manipulation force along the y_0-axis and a manipulation moment of 20 Nm around the y_0-axis, and the initial body speed is 0.2 m/s.

Using the proper stability margin, stability-level curves have been obtained for a robot in different dynamic situations on inclined terrain. The computation of stability-level curves enables *COG* location to be controlled inside the body plane such as to achieve a certain stability level. The selection of the proper stability margin for a given application and the use of stability-level curves for gait control plays a major role in the successful generation of walking-robot tasks.

3

Generation of Periodic Gaits

3.1 Introduction

The first research into legged locomotion was focused on the observation, comprehension, and mathematical formulation of ordinary gaits found in nature. Many of these gaits feature leg motion sequences and footholds that are *periodic*. Another characteristic of these gaits is that the body is in constant motion, while all of the legs move simultaneously; thus they are termed *continuous* gaits. Wave gaits are a special kind of continuous gait used by mammals and insects at low speed over smooth, irregular terrain. Continuous wave gaits have been studied extensively. The formulation of a mathematical model to describe the leg sequence and the derivation of the model to perform crab and circular gaits were carried out between 1968 and 1989 (McGhee and Frank, 1968; Zhang and Song, 1989), but only for flat terrain conditions. The theoretical results for these gaits were significant, but they were inadequate for application to real legged robots walking on uneven ground. Therefore, some attempts were made to adapt these gaits to irregular terrain for specific cases (Kumar and Waldron, 1989; Jimenez and Gonzalez de Santos, 1997).

When the terrain irregularities are too sharp to maintain a continuous gait, mammals and insects can change their gait to a more secure gait. This gait is characterized by the sequential motion of legs and body. The body is propelled forward/backward with all of the feet securely placed on the ground and a leg is transferred with all other three legs and body halted. These gaits, called *discontinuous* gaits because they cause intermittent body motion, are beneficial to real legged machines: they are easier to implement, and they offer a better longitudinal stability margin than wave gaits. Also, they can achieve a faster velocity than wave gaits for some gait parameters. Finally, the fact that they move sequentially makes them appropriate for walking on very irregular terrain, since each moving leg is lowered, until it comes into contact with the ground, while the remaining legs and the body do not move. This is why it is said these gaits have intrinsic terrain adaptability.

Discontinuous gaits were studied in the 1990s. They were first implemented in realistic outdoor machines such as the AMBLER, which was developed at Carnegie-Mellon University. The AMBLER had a special discontinuous gait called a *circular* gait (Bares and Whittaker, 1989). The circular gait was a direct consequence of the specific topology of the machine. By the mid-1990s, discontinuous gaits applicable to mammal or insect-like robot configurations were created based on wave gaits (Gonzalez de Santos and Jimenez, 1995).

For terrain with impractical areas, such as large holes and protuberances, or dangerous zones such as land-mines, the gait can break its periodicity and become a different gait type, which selects the footholds and motion sequences on-line depending on the terrain conditions. This gait is termed as *non-periodic* gait, also known as a *free* gait. Therefore, gaits can be classified into two large groups, based on periodicity, as *periodic* and *non-periodic*. Thus, a gait is periodic if the different movement components (foot lifting, foot placement and body motion) occur at the same instants in a locomotion cycle. On the other hand, by observing the body motion there are gaits that move the body at a constant speed –*continuous*– or move the body intermittently –*discontinuous*–. This is the main gait classification that will be used in this book. A more detailed classification can be found in (Song and Waldron, 1989).

The aim of this chapter is to present the formulation of periodic gaits in two main groups: continuous (wave) and discontinuous. Non-periodic gaits will be studied in the next chapter. Apart from mathematical derivation, both gaits are compared in terms of stability, velocity and ease of implementation. To achieve these goals, continuous gaits are presented in Sects. 3.2 and 3.3. Discontinuous periodic gaits are formulated in Sect. 3.4, and their features are compared with those of continuous gaits. Sections 3.5 and 3.6 include calculations of discontinuous crab and discontinuous turning gaits. Section 3.7 presents some examples of how to combine the gaits to follow predefined paths. Finally, Sect. 3.8 concludes with a discussion of these issues.

3.2 Gait Generation

In the English language, gait is defined as a way or manner of moving on foot. In the field of legged locomotion, a gait is defined as a repetitive pattern of foot placements (Todd, 1985). A more precise description was made by Song and Waldron (1989), as follows.

Definition 3.1. *A gait is defined by the time and the location of the placing and lifting of each foot, coordinated with the motion of the body in its six degrees of freedom, in order to move the body from one place to another.*

The first attempts to define mathematical models for gaits were carried out by McGhee and Frank and was focused on quadrupeds (McGhee,

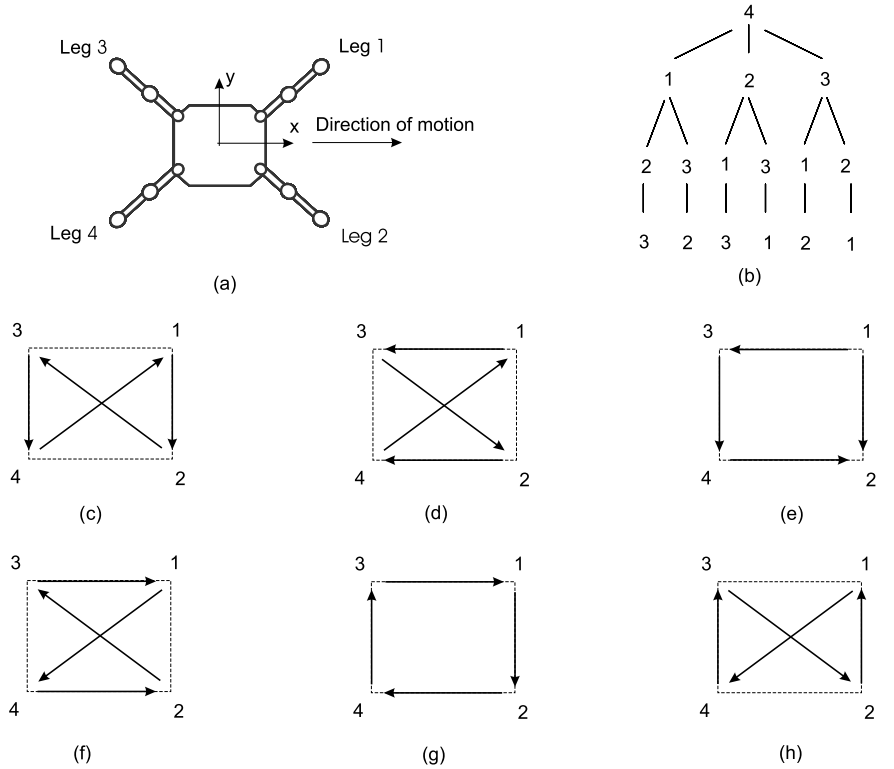


Fig. 3.1. Quadruped gaits: **(a)** top view of a robot; **(b)** graph of leg sequences; **(c)** – **(h)** sequences

1968; McGhee and Frank, 1968). These researchers tried to discover which quadruped gaits were able to maintain static stability. For this study, McGhee introduced a notation called the *event sequence*. An event is defined as a foot placement or a foot lifting. For an n-legged robot, the placement of foot i is denoted by event i, while the lifting of foot i is denoted by event $i + n$. Thus, a gait is expressed as a sequence of events such as 2-4-5-7-3-1-8-6, creating $2n$ different events. If two events occur at the same instant, the gait is termed *singular* gait, as opposed to totally ordered gaits, or *non-singular gaits*. The number of possible non-singular quadruped gaits is the permutation of $2n$ events; that is $2n!$. Considering the number of event sequences starting from a given one, the result is $N = (2n - 1)!$; for a quadruped $N = 5040$ (McGhee, 1968).

N represents the number of possible permutations of foot events. However, for a quadruped to maintain static stability, it must keep at least thee legs in support, *i.e.* just one leg in transfer. This means that after the lifting of leg i (event $i + n$) the placement of leg i must occur (event i). This feature drastically reduces the number of stable combinations to $N = (n - 1)!$, which for

a quadruped results in only six event sequences, demonstrated in Fig. 3.1(b) and illustrated in Fig. 3.1(c)–(h). In this example, the locomotion cycle starts with the movement of foot 4. Leg numbers are indicated in Fig. 3.1(a). The practice of numbering legs front to rear, using even numbers for right legs and odd numbers for left legs, is largely accepted.

Tomovic (1961) defined the gait of an n-legged robot as a *creeping* gait when every support pattern involves at least n–1 contact points. Hence, gaits in Fig. 3.1 are creeping gaits: there are always three feet in support. Creeping gaits may be either singular or non-singular. Notice that a singular gait can be obtained as the limit of a non-singular gait. Therefore, the event sequences in Fig. 3.1 are applicable to both singular and non-singular gaits. In this case, a singular gait means that the placement of a foot and the lifting of the next leg in the sequence occur at the same time.

Some years after this study, Hirose *et al.* (1986) found that these six event sequences could be applied to the formulation of quadruped turning gaits. They were classified as $\pm x$ type, $\pm y$ type (for motion along the x and y axes, respectively) and $\pm o$ type (for gaits that rotate the body around the z axis), where '+' means forward motion and '–' means reverse motion (see Fig. 3.1(c) – (h)).

McGhee and Frank (1968) studied the static stability of the six defined creeping gaits, and showed that the optimum static stability margin is achieved by a *regular* (every foot supported along the same fraction of the locomotion cycle), singular, $+x$ creeping gait (see Fig. 3.1(f)). This creeping gait is sometimes termed a *crawl* gait. Surprisingly, it is the unique gait used by quadruped animals at low velocities, thus it is known as the *standard gait*. Notice that there is no generally accepted definition for crawl gait, but crawl and creeping gaits are synonyms for quadrupeds.

At the beginning of the 1970s, Bessonov and Umnov (1973) came to the same conclusion when working on hexapods. They found through numerical experimentation that there is a periodic, regular, *symmetric* (the events of any right-left pair are exactly half a cycle out of phase) gait, that optimizes the static stability. This gait, and the one found for quadrupeds make the legs move back to front following a wave of stepping actions on each side of the body, such that the events of any right-left pair is shifted by half a locomotion cycle. Because of that leg-wave motion these gaits are termed *wave gaits*. This kind of gait is explained for quadrupeds in the next section.

3.3 Continuous Gaits

This section formulates the wave gait, the continuous gait that is the most widely used by natural and artificial quadrupeds. In this gait formulation, an ideal machine that assumes *massless* legs is used (see Chap. 2). The following definitions are also required for gait formulation.

Definition 3.2. *The duty factor, β_i, of leg i is the fraction of the cycle for which it is on the ground. If β_i is the same for all legs, the gait is regular.*

Definition 3.3. *The leg phase of leg i, Φ_i, is the normalized time by which the placement of leg i on the ground lags behind the placement of leg 1 (leg 1 is normally considered the reference leg).*

Definition 3.4. *The leg stroke, R, is the distance which a foot is moved relative to the body during the support phase. R must be within the leg workspace defined by R_x and R_y (see Fig. 3.2).*

Definition 3.5. *The stroke pitch, P, is the distance between stroke centers of the adjacent legs. P_x is the distance between stroke centers of collateral legs and P_y is the distance between stroke centers of contra-lateral legs (see Fig. 3.2).*

Definition 3.6. *The stride length, λ, of a gait is the distance travelled by the center of gravity COG of the body along a locomotion cycle. If the gait is periodic then*

$$\lambda = \frac{R}{\beta}. \tag{3.1}$$

With these definitions, the $+x$ type wave gait is defined by the following leg phases, assuming that the leg workspaces do not overlap, *i.e.* $R \leq P$,

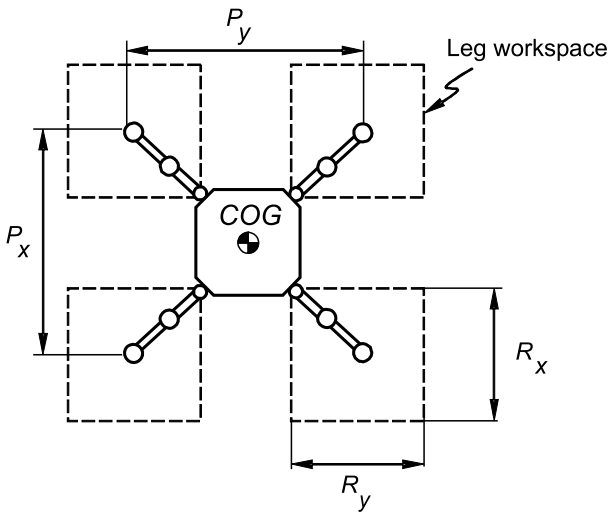


Fig. 3.2. Geometric definitions

$$\begin{aligned} \phi_1 &= 0 \\ \phi_2 &= \tfrac{1}{2} \\ \phi_3 &= \beta \\ \phi_4 &= F\left(\beta - \tfrac{1}{2}\right) \end{aligned} \tag{3.2}$$

where F is the *fractional function* defined as

Definition 3.7. *A fractional function $Y = F(X)$ of a real number X is defined as*

$$Y = \begin{cases} \textit{the fractional part of } X & \textit{if } X \geq 0 \\ 1 - \textit{the fractional part of } |X| & \textit{if } X < 0. \end{cases} \tag{3.3}$$

McGhee and Frank (1968) demonstrated that the S_{LSM}, defined in Sect. 2.2, of the wave gait is optimum, given by

$$S_{LSM} = \left(\beta - \frac{3}{4}\right)\lambda;\ 1 > \beta \geq \frac{3}{4}. \tag{3.4}$$

Note that the condition in (3.4), $1{>}\beta{\geq}3/4$, is mandatory for maintaining static stability: each leg is in support for at least for 3/4 of the period, which confirms that, for a wave gait, at least three legs are in support at any given time. Equation (3.2) can be understood better by using the *gait diagram*. The gait diagram is a method of depicting the time that a foot is in contact with the ground (represented by a solid line) or in the air (represented by a dashed line). This diagram also shows the time instants when a leg changes from support to transfer, and *vice versa*. The starting point for the solid segment is the moment when the foot is placed on the ground, and the end point represents the moment the leg is lifted. This diagram records the sequence of lifting and placing of legs, as well as the duration of the support and transfer phases of the legs. Figure 3.3 illustrates the gait diagram for different gait parameters.

Considering that a wave gait is symmetric, the definition of a gait can be simplified by defining leg phases for only one side of the robot. For the other side, a leg phase is the same as its pair increased in half cycle time. It is also possible to avoid defining Φ_1 because it is always defined as zero. Thus, a wave gait for a hexapod can be defined as

$$\phi_3 = \beta;\ \ \phi_5 = 2\beta - 1;\ \ \beta \geq \frac{1}{2} \tag{3.5}$$

The generalization of this formulation was performed Sun (1974), who stated that the wave gait for a $2n$-legged robot is defined by

$$\phi_{2m+1} = F(m\beta);\ m = 1,\ 2, ..., n-1;\ \frac{3}{2n} \leq \beta \leq 1. \tag{3.6}$$

Later, Zhang and Song (1989) formulated the *wave-turning* gait and *spinning* gait. Finally, in 1990 they derived the *wave-crab* gait (Zhang and Song,

1990). In the wave-crab gait, the body of the robot moves along a straight line which forms a constant crab angle with the longitudinal axis of the body. These formulations were derived for quadrupeds and practically ended the research into wave gaits for legged robots. This chapter does not include a description of those gaits, but interested readers are encouraged to study the material included in the references. Later sections will only focus on discontinuous gaits.

3.4 Discontinuous Gaits

Discontinuous gaits are characterized by the sequential motion of the legs and body (Gonzalez de Santos and Jimenez, 1995). One leg is transferred with all other legs in support and halted. The body is propelled with all legs in support and moving simultaneously, of course, maintaining their footprints.

In generating discontinuous-periodic gaits for quadrupeds, certain aspects should be considered:

1. If one leg in its support phase reaches the rear limit of its workspace (kinematic limit), this leg should change to the transfer phase to be placed at its front kinematic limit.
2. The body is propelled forward with all legs on the ground. After a body motion, at least one leg should stay in its rear kinematic limit to perform a transfer phase into the next leg motion.

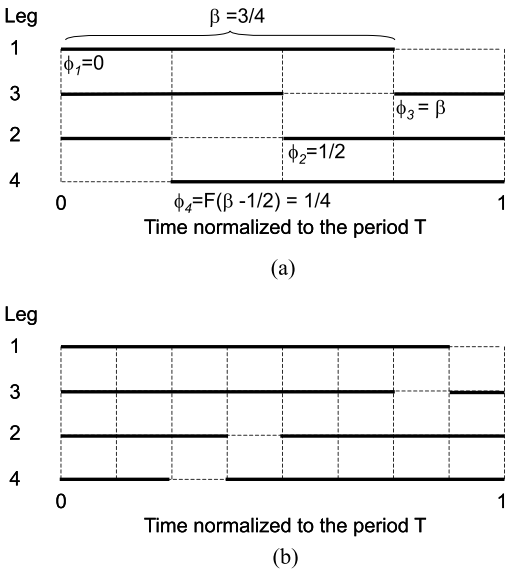


Fig. 3.3. Gait diagrams of wave gaits for quadrupeds: (**a**) β=3/4; (**b**) β=7/8

3. The leg that is contralateral and non-adjacent (CNA) to the present transfer leg should be placed at such a point that after the placement of the transferred leg, the COG stays on the other side of the line connecting the CNA leg with the transfer leg (see Fig. 3.4). In this way, it will be possible to lift another leg while maintaining the machine's stability.
4. The sequence of legs should be periodic; this will allow several locomotion cycles to be joined to follow a path.

In this section, a gait to move the machine straight forward along the longitudinal x-axis of the machine and under static stability will be considered. That means that the vertical projection of the COG is always inside the support polygon. The longitudinal stability margin, S_{LSM}, will be used as a stability measure.

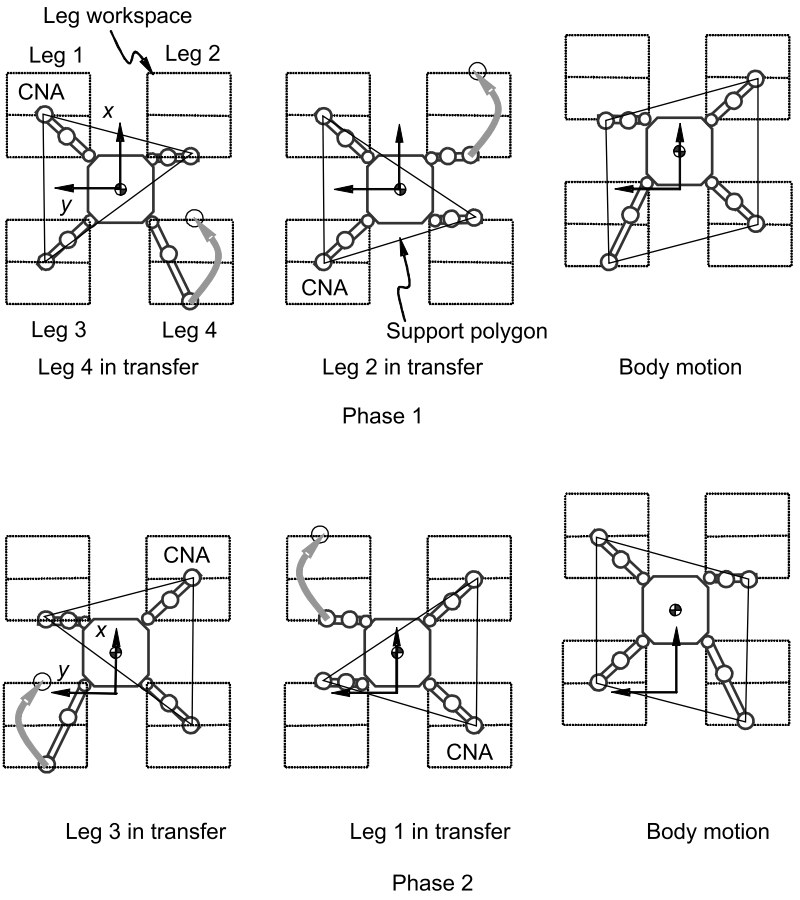


Fig. 3.4. Successive gait pattern of a two-phase discontinuous gait

3.4.1 Two-phase Discontinuous Gaits

We have seen in Sect. 3.2 that there are 4! different non-singular placing sequences for a quadruped. These sequences must be combined with leg locations to define a stable gait. The number of possible solutions is extremely high. To narrow this number of possibilities and find a single stable sequence, some restrictions may be considered. One could be the assumption of a fixed number of body motions per cycle. First, let us assume two body motions per cycle, or two phases. In this case, two legs must remain at their rear kinematic limit after each phase. These legs will be transferred sequentially toward their front kinematic limit. Hence, each leg moves the length of its stroke, and the number of placing leg points is reduced drastically. Let us assume that legs go to transfer phase following the sequence of the standard gait illustrated in Fig. 3.1(f). That means a rear leg is moved first and then the front leg is moved. After these leg motions, the body is propelled forward half a stroke. At the end of this motion, all legs that did not transfer must be located at their rear kinematic limits. To accomplish that, these legs need to stay in the middle of their workspace before the body motion, *i.e.* at the points that are the counterparts of the points where the other two legs are now located. Figure 3.4 shows the sequence of motions 4-2-B-3-1-B for the discontinuous gait upon which the following study will be based. B represents the body motion event. The earth reference frame (x, y) helps to illustrate the absolute displacement of the *COG* for every leg and body motion.

Longitudinal Stability Margin for a Discontinuous Gait

The longitudinal stability margin, S_{LSM}, (see Sect. 2.2) is determined by the diagonal defined by two contralateral non-adjacent feet as shown in Fig. 3.4. For all cases in this figure, the diagonal goes from a foot in the middle of its workspace to a foot placed at its kinematic limit closest to the *COG*. As can be seen in Fig. 3.4, the longitudinal stability margin (distance between the projection of *COG* and the diagonal along the x-axis) is the same for all cases, and its value is given by the absolute value of the diagonal abscissa at the origin. When leg 4 is in transfer, this value is given by (see Fig. 3.4)

$$S_{LSM_D} = \left| -y_2 \left(\frac{x_3 - x_2}{y_3 - y_2} \right) + x_2 \right| \tag{3.7}$$

where (x_2, y_2) and (x_3, y_3) are the end points of the diagonal, *i.e.* the position of feet that define the diagonal. The values of these points for leg 4 in transfer, *i.e.* the first machine posture in Fig. 3.4, are $(-P_x/2, P_y/2)$ and $(P_x/2 - R_x/2, -P_y/2)$, where P_x is the stroke pitch in the x-axis direction, P_y is the stroke pitch in the y-axis direction, and R_x and R_y are the dimensions of the leg workspace, which defines the stroke (see Fig. 3.2 for parameter definition). Substituting these values into (3.7) yields

$$S_{LSM_D} = \frac{R_x}{4}. \tag{3.8}$$

The longitudinal stability margin for a four-legged robot walking with a wave gait is given by (3.4), where λ is defined by (3.1). Substituting (3.1) into (3.4) yields that the S_{LSM} for a wave gait is

$$S_{LSM_C} = \left(\beta - \frac{3}{4}\right)\frac{R_x}{\beta}; \frac{3}{4} \leq \beta \leq 1 \tag{3.9}$$

and hence the S_{LSM_C} assumes values between zero and $\mathrm{R}_x/4$, while a discontinuous gait exhibits a constant S_{LSM_D} of $R_x/4$. Therefore, discontinuous gaits feature larger longitudinal stability margins than wave gaits as shown in Fig. 3.5, which represents the S_{LSM} normalized to the stroke, R_x. This is one important advantage of these gaits. Next, we will study what happens with velocity.

Velocity for the Two-phase Discontinuous Gait

The average velocity of a robot performing a discontinuous gait is determined by the stroke, R_x, and the period, T. In a discontinuous gait, the period is given by the sum of times for each leg subphase and times for body motions. Considering that a foot on transfer describes a rectangular trajectory in a perpendicular plane the cycle time is given by

$$T_D = 4(t_L + t_F + t_P) + 2t_{BP}$$

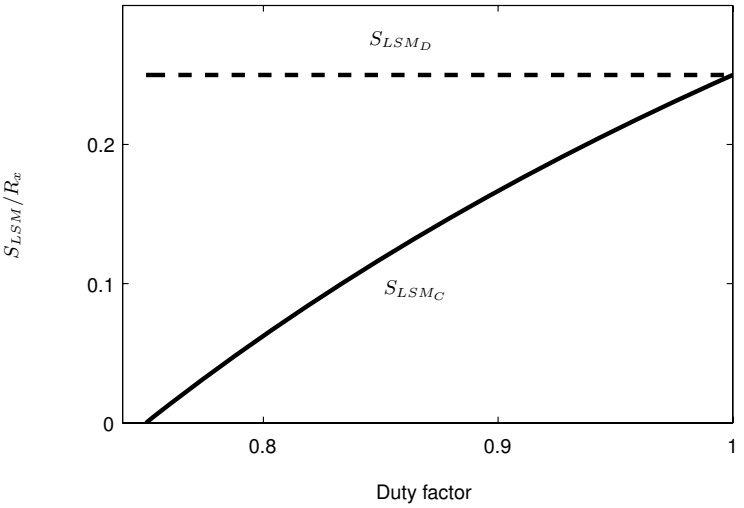


Fig. 3.5. S_{LSM} for a wave gait (*solid line*) and a discontinuous gait (*dashed line*)

where t_L is the leg lifting time, t_F is the leg motion forward time, t_P is the leg placement time and t_{BP} is the body propulsion time for each subphase.

If the leg step height is h, the leg stroke is R_x and the foot speed in axes x and z are V_x and V_z, respectively, then the period is given by

$$T_D = 4\left(\frac{h}{V_z} + \frac{R_x}{V_x} + \frac{h}{V_z}\right) + 2\left(\frac{R_x}{2V_x}\right) = \frac{8hV_x + 5R_xV_z}{V_xV_z}. \tag{3.10}$$

In a continuous gait, a leg is in support along a locomotion cycle by about $t_s = \beta T_C$ and in transfer by about $t_t = (1-\beta)T_C$, where T_C is the period of the cycle. Considering that the feet describe the same trajectory that discontinuous gaits do, the transfer leg time is

$$t_t = \left(2\frac{h}{V_z} + \frac{R_x}{V_x}\right)$$

and therefore

$$T_C = \frac{1}{1-\beta}\left(\frac{2h}{V_z} + \frac{R_x}{V_x}\right). \tag{3.11}$$

Equations (3.10) and (3.11) provide the period for each gait. Therefore, to compute gait velocity, the displacement of the body during one locomotion cycle must be known. This displacement is λ for a wave gait and R_x for a discontinuous gait; thus, the velocities for both continuous (wave) gaits, v_C, and discontinuous gaits, v_D, are given by

$$v_c = \frac{\lambda}{T_C} = \frac{\lambda(1-\beta)V_xV_z}{2hV_x + R_xV_z} \tag{3.12}$$

$$v_D = \frac{R_x}{T_D} = \frac{R_xV_xV_z}{8hV_x + 5R_xV_z}. \tag{3.13}$$

As a comparative example, let us consider that $V_z = V_x = V$, and $h = R_x/K$. In this case, Fig. 3.6 shows the velocity of the robot for both gaits, normalized to V, as a function of the duty factor, β, and the parameter K, which is the relationship between h and R. This figure shows how, for any value K, the velocity of the wave gait is greater than the velocity of the discontinuous gait for small duty factor values, and it becomes smaller for high duty factor values. Figure 3.6 also shows the curve that verifies $v_D = v_C$. This curve defines the frontier of the parameters for wave gaits that exhibit the same velocity that discontinuous gaits. For higher duty factors the wave gait becomes slower and less stable than a discontinuous gait with the same leg step parameters (see Fig. 3.5).

This study deals with the determination of the gait providing either the highest velocity or the greatest S_{LSM}; nevertheless, there are other interesting gait features that should be considered in choosing a suitable gait. For example, if the machine carries its operator or passengers, continuous gaits provide more comfort than discontinuous gaits, which move the body jerkily.

Note that in this theoretical study a rectangular profile of the velocity has been considered for axis and body motions, which means infinite acceleration. If axis and body acceleration are high, the velocity and the S_{LSM} shall be close to the theoretical one. In this case, the high acceleration will cause strong forces on the machine, jeopardizing its stability. Nevertheless, a discontinuous gait moves the body while all feet are on the ground; therefore, stability is really very high, and the only unstable effects could be if the robot rotates about one edge of the support boundary. In such a case, the dynamic stability measurements presented in Chap. 2 should be used.

Gait Diagram for a Two-phase Discontinuous Gait

Figure 3.7 shows the gait diagram of a two-phase discontinuous gait. Leg subphases and body motions are also displayed. This diagram has the same shape as the gait diagram for a wave gait with a duty factor of

$$\beta = \frac{(8h/V + 3hK/V)}{(10h/V + 4hK/V)} \tag{3.14}$$

and for $K = 2$ yields $\beta = 7/9$ (see Fig. 3.7).

Note that the gait diagram shows the time that a leg provides support as well as the leg transition instants. Moreover, in a wave gait the body moves continuously so that the elapsed time gives an idea about the distance travelled by the body normalized to the stride length, λ. However, for a discontinuous gait, the body moves during only two intervals and the distance travelled by the body is thus proportional to the number of body motions. The comparison

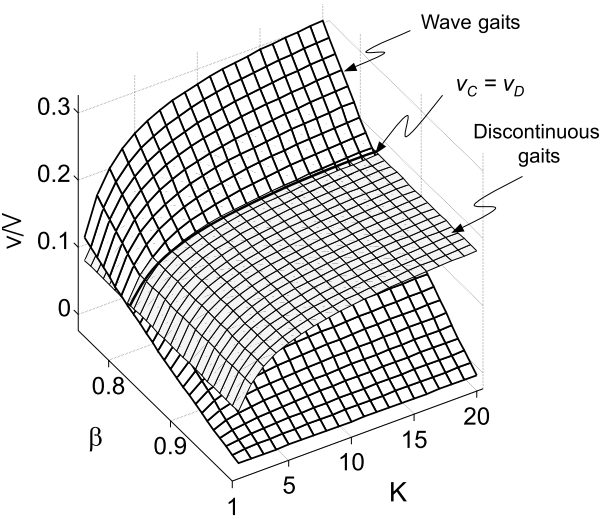


Fig. 3.6. Velocity for both a wave gait and a discontinuous gait

between wave gaits and discontinuous gaits in this section was made based upon an identical gait diagram for both gaits, as illustrated in Fig. 3.7. Note that the gait diagram should start in $\Phi_1 = 0$ as the gait (b) does. However, the gait diagram for the discontinuous gait (a) starts with the transference of leg 4, because it was defined in that way. Gait diagrams are the same in both cases although they have been plotted with a little delay illustrated in the figure.

3.4.2 Four-Phase Discontinuous Gaits

As mentioned above, in order to generate an efficient discontinuous gait, it is necessary to locate a leg at its rear kinematic limit before starting its transfer phase. For a two-phase gait, after each body motion, two collateral legs are placed at the end of their kinematic limit. However, it is possible to generate such a gait that only one leg stays at the end of its kinematic limit when a phase finishes. That requires performing four body motions per cycle (four phases), due to the fact that just one leg is at the end of its kinematic limit in each body motion.

The diagram for this gait is somewhat similar to the two-phase gait diagram, but each body displacement is divided into two additional displacements performed at the end of every leg motion. The velocity of this gait is the same as in the two-phase gait, because time and total displacement are

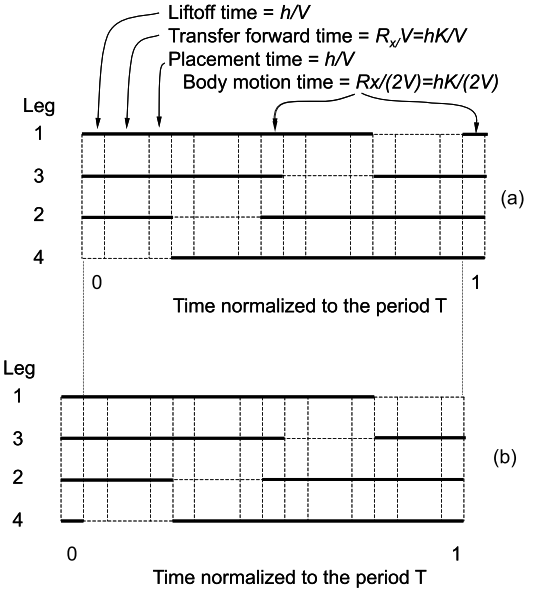


Fig. 3.7. (**a**) Gait diagram of a two-phase discontinuous gait; (**b**) Gait diagram of a wave gait. In both cases β =7/9

equal. Nevertheless, it is possible to show that the S_{LSM} for this gait is given by

$$S_{LSM_4} = \frac{Rx}{8}. \tag{3.15}$$

Consequently, the S_{LSM_4} is poorer than the S_{LSM} for a two-phase gait given by (3.8).

When the number of phases is increased, *i.e.* several body motions per leg motion, the S_{LSM} gets worse, while average velocity remains constant. Summarizing, two-phase discontinuous gaits offer the best S_{LSM} of any discontinuous gait while the velocity does not depend on the number of phases.

3.5 Two-phase Discontinuous Crab Gaits

Discontinuous crab gaits involve changing the foot locations and moving the body so that the direction of motion maintains a defined crab angle with the longitudinal axis of the body. To generate a two-phase discontinuous crab gait (TPDC) we use the same sequence of leg and body motion as in previously, *i.e.* the sequence of the standard gait, but the leg and body trajectories change. The foot positions may be observed in Fig. 3.8. Leg and body motion modify the S_{LSM}; therefore, there is a maximum crab angle achieved for the robot. Different possibilities of performing crab gaits, which vary depending on initial leg positions, are presented below.

3.5.1 TPDC Gait with No Change in Initial Position

The general formulation for the crab gait involves computing the foot displacements needed to follow a specified crab angle trajectory. Obviously, these new foot positions must be located inside the leg workspace and to perform leg steps as large as possible we will place the foothold on the workspace boundary. Thus, foot displacement along y axis will be restricted to its maximum, given by $|R_y/2|$, and the displacement along the x-axis will be related to the y displacement. Figure 3.9 illustrates two possible cases. The stroke, or increment in the foot position defined by its components L_x and L_y, may be defined as follows:

$$\text{Case A} \quad \left.\begin{aligned} L_x &= R_x \\ L_y &= R_x \tan\alpha \end{aligned}\right\} \; if \;\; |R_x \tan\alpha| \leq \tfrac{R_y}{2}$$

$$\text{Case B} \quad \left.\begin{aligned} L_x &= \left|\tfrac{R_y}{2}\cot\alpha\right| \\ L_y &= Sign(\alpha)\tfrac{R_y}{2} \end{aligned}\right\} \; if \;\; |R_x \tan\alpha| > \tfrac{R_y}{2}.$$

Figure 3.8 shows a sequence of workspace positions for the TPDC gait using the trajectory defined in Case A. For the crab angle, α, shown in this figure, each leg moves (R_x, R_x *tan* α) during a locomotion cycle, and the body moves ($R_x/2$, ($R_x/2$) tan α) in each phase. Figure 3.8 shows the diagonal that determines the stability margin for a crab gait (solid lines) and the diagonal for a non-crab body motion (dashed lines). Observe that for the first leg motion, that of leg 4, the S_{LSM} is the same in both cases (crab and non-crab gait); but when leg 2 moves, the S_{LSM} decreases for the crab gait motion. When leg 3 moves after a body motion, the S_{LSM} is again lower that in a non-crab motion. Finally, when leg 1 moves, the S_{LSM} is larger than the one for the non-crab gait (for the crab angle considered in the figure).

Figure 3.10 shows the S_{LSM} as a function of the crab angle for each leg in its transfer phase. In this figure, the following parameters have been considered: $P_x = 0.55$ m, $P_y = 0.55$ m, $R_x = 0.25$ m, $R_y = 0.25$ m, and α taking values from -45^o to 45^o. Leg 2 in transfer presents the smallest S_{LSM} for positive crab angles. Substituting leg position values into (3.7), when leg 2 is in transfer and α meets the case A constrains, yields

$$S_{LSM_{A+}} = \frac{R_x(P_y - 2P_x \tan\alpha)}{4(P_y - R_x \tan\alpha)}. \tag{3.16}$$

Considering a negative crab angle, the minimum S_{LSM} is achieved when leg 1 is in transfer. In this case, the value of the S_{LSM} is given by

$$S_{LSM_{A-}} = \frac{P_y R_x + 2P_x R_x \tan\alpha - R_x^2 \tan\alpha}{4P_y}. \tag{3.17}$$

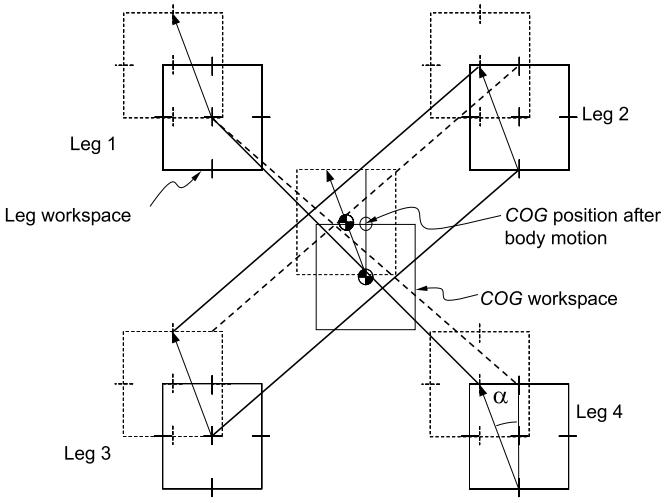


Fig. 3.8. Footholds and workspaces for a two-phase discontinuous crab gait. Initial workspaces are drawn in *solid line*. Workspaces at the end of the first phase are drawn in *dashed line*.

The S_{LSM_A} along the locomotion cycle, which is the minimum of the margins obtained for different legs in transfer, is depicted in thick-dashed line in Fig. 3.10.

When the crab angle meets case B constrains, Equation (3.7) provides the S_{LSM} expressions for both positive and negative crab angles given by

$$S_{LSM_{B+}} = \frac{-P_y R_x - P_x R_y + P_y R_y \cot\alpha}{2(2P_y - R_y)} \tag{3.18}$$

and

$$S_{LSM_{B-}} = \frac{-2P_y R_x - 2P_x R_y + R_x R_y - 2P_y R_y \cot\alpha}{8P_y}. \tag{3.19}$$

For those parameters considered, Fig. 3.10 shows that the permitted crab angle falls into the range running from -30^o to 26^o.

3.5.2 TPDC Gait with Change in Initial Position

The maximum crab angle that a quadruped can achieve with stability can be augmented if the foot trajectories of support legs pass over the center of the workspace. This is equivalent to locating the initial foot positions as shown in Fig. 3.11(a).

After the leg 2 and leg 4 transfer phases, the body will move half a cycle's displacement along the x and y axes. After this motion, legs 1 and 3 should stay in positions equivalent to those held by legs 2 and 4 at the beginning of the cycle; therefore, these legs do not need to accomplish initial foot displacement. The initial leg configuration for this gait is shown in Fig. 3.11(b).

The stroke defined by L_x and L_y, and initial foot displacements, expressed in the body reference frame, D_x, D_y, may be formulated as in following cases:

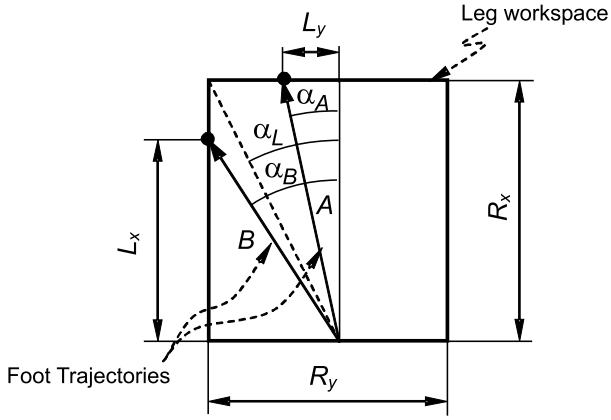


Fig. 3.9. Foot trajectories for a discontinuous crab gait (Cases A and B)

$$\left.\begin{array}{c}\text{Case C}\\ L_x = R_x\\ L_y = R_x \tan\alpha\\ D_x = 0\\ D_y = -\frac{R_x}{2}\tan\alpha\end{array}\right\} \; if \;\; |R_x \tan\alpha| \leq R_y$$

$$\left.\begin{array}{c}\text{Case D}\\ L_x = |R_y \cot\alpha|\\ L_y = Sign(\alpha) R_y\\ D_x = \frac{R_x}{2} - \left|\frac{R_y}{2}\cot\alpha\right|\\ D_y = -Sign(\alpha)\frac{R_y}{2}\end{array}\right\} \; if \;\; |R_x \tan\alpha| > R_y.$$

Figure 3.12 shows the S_{LSM} *vs* crab angle for the foot trajectories defined by both Cases C and D for the same parameters than in previous examples. The minimum S_{LSM} appears for either leg 2 or 3 in transfer when the crab angle is positive, and for legs 1 and 4 when the crab angle is negative. Repeating the process followed in the paragraph above, the expressions for the S_{LSM} in every case become

$$S_{LSM_{C+}} = \frac{R_x(P_y - P_x \tan\alpha)}{2(2P_y - R_x \tan\alpha)} \tag{3.20}$$

$$S_{LSM_{C-}} = \frac{R_x(P_y + P_x \tan\alpha)}{2(2P_y + R_x \tan\alpha)} \tag{3.21}$$

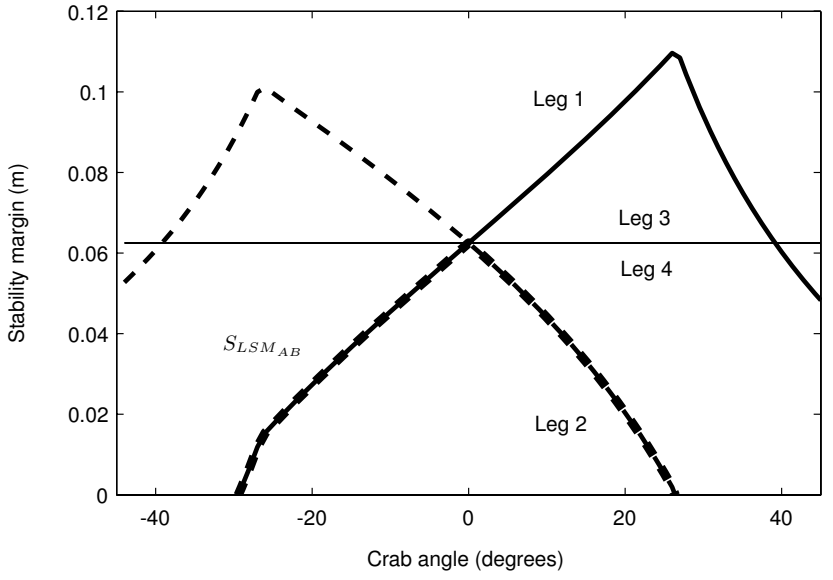


Fig. 3.10. Stability margin for a discontinuous crab gait

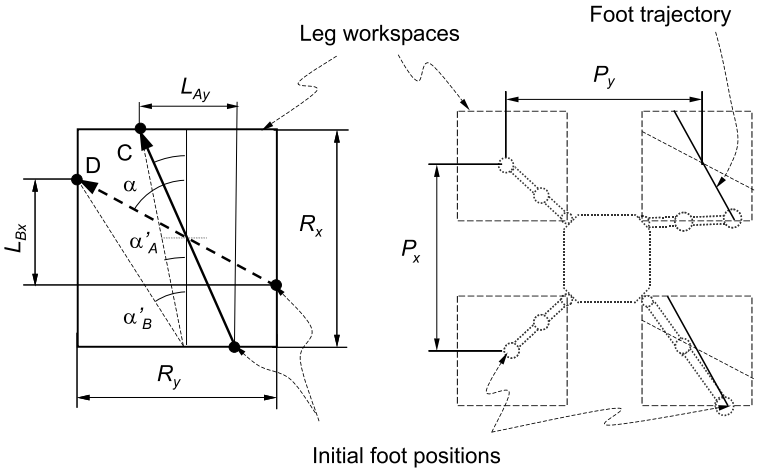


Fig. 3.11. Foot trajectories and initial foot positions for a discontinuous crab gait: Case C in *thick-solid line* and Case D in *thick-dashed line*

$$S_{LSM_{D+}} = \frac{-P_x R_y + P_y R_y \cot\alpha}{2(2P_y - R_y)} \tag{3.22}$$

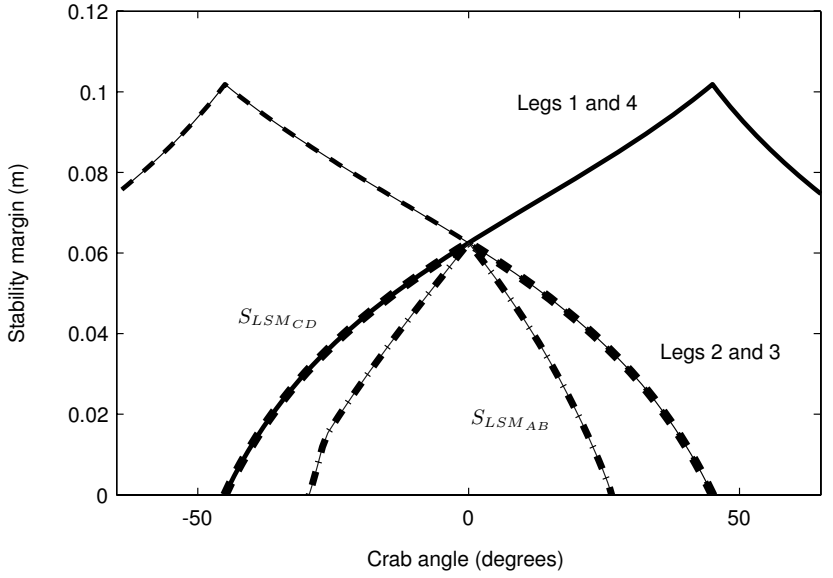


Fig. 3.12. Stability margin for a two phase-discontinuous gait with change in the initial position of right legs. The leg numbers indicate the leg in transfer. The S_{LSM} of the cycle is indicated in *thick-dashed line*. The S_{LSM} for cases A and B is indicated in *dotted line*.

$$S_{LSM_{D-}} = \frac{-R_y(P_y \cot\alpha + P_x)}{2(2P_y - R_y)}. \tag{3.23}$$

The $S_{LSM_{CD}}$ of the locomotion cycle is indicated in thick-dashed line in Fig. 3.12. The S_{LSM} for cases A and B, $S_{LSM_{AB}}$, have been superimposed in dotted line.

3.5.3 Strategy for Discontinuous Walking

The main interest in the gaits studied here is to connect them to follow a given trajectory. In this study, for the sake of simplicity, the initial and final leg locations of a cycle, in the body reference frame, will be the same for all gaits considered. Hence, it is quite simple to join different gaits at the end of a locomotion cycle.

Different possibilities for discontinuous crab walking have already been analyzed. Note that the two-phase discontinuous gait coincides with the zero crab angle discontinuous gait. The employment of one or the other depends on workspace constraints. Figures 3.10 and 3.12 show the S_{LSM} and the maximum crab angle achieved by each gait mode. Each graph shows one mode with two different cases defined by its foot trajectories. The switch from one trajectory to the other occurs at the point where the slope of the curve suddenly changes. The TPDC with a change in the initial position can achieve greater crab angles (from -45^o to 45^o, in our example) than the TPDC mode with no change in its initial position (which is restricted to between -30^o and 26^o). Furthermore, cases C and D (thick-solid line) exhibit larger S_{LSM} than cases A and B (dotted line), as can be observed in Fig. 3.12. Thus, the TPDC gait with a change in its initial position presents a better S_{LSM} and achieves greater crab angles, but at the cost of extra leg motions at the beginning of the gait to locate the right legs at their initial positions.

If the robot is starting a crab trajectory from the two-phase-gait initial position, leg initialization problems can be avoided. In such an instance, the initial motion and the first steps of legs 4 and 2 can be substituted by one initial step. This step moves legs 4 and 2, sequentially, from their two-phase-gait initial position to the final transfer leg position of either case C or case D. Figure 3.11 shows these points and the trajectory that legs 2 and 4 should follow in the initial step for each case (crab angles α'_A and α'_B, respectively). When leg 4 starts its transfer phase, the remaining legs are placed at the points they hold in a discontinuous two-phase gait; therefore, the S_{LSM}s coincide in both gaits. When leg 2 starts its transfer phase, the remaining legs are placed at the point corresponding to either case C or case D. Therefore, the S_{LSM} does not change when performing this initial step to avoid leg initialization.

Similarly, to change from a crab angle motion to a two-phase gait (null crab angle), a new right-leg initialization is required, but it could be avoided if the first step is made directly to the final transfer leg position of the two-phase gait. For this first step, during the transfer of leg 4, the remaining legs

are at the location they hold for either case C or case D, depending on the case in question. Therefore, the S_{LSM} is equal to the S_{LSM} for the trajectory accomplished. When leg 2 changes to transfer, the remaining legs stay in their corresponding two-phase initial position; thus the S_{LSM} is the same as the S_{LSM} for the two-phase case.

Summarizing, we see that the S_{LSM} for the initial step is always equal to the S_{LSM} for either case C or case D, and both these cases present larger S_{LSM} than cases A and B. Hence, if the body is going to follow a trajectory with at least two locomotion cycles, it is advisable to use either case C or case D, performing both an initial and a final step as noted before. Cases A and B are only useful when the motion is going to be shorter than two locomotion cycles.

Previous examples showed how the S_{LSM} varies when the motion changes from a two-phase gait to a crab gait, and vice versa. When the machine changes from a crab gait to another gait, it can be shown, in a similar way, that the S_{LSM} of the initial step is equal to the smaller S_{LSM} of the two connecting gaits.

3.6 Discontinuous Turning Gaits

Crab gaits move the robot's body along straight trajectories and by joining different crab gaits it is possible to follow easily complex trajectories (see Sect. 3.7). However, sometimes it is important either to move the robot along a non-straight trajectory maintaining the body tangent to that trajectory or to rotate the body about its perpendicular axis. Gaits that perform those kinds of motions are termed *turning* gaits.

A typical curved trajectory is the circumference and the gait that move a robot along this curve is called *circling* gait. Discontinuous quadruped circling gaits, like crab gaits, decrease the S_{LSM} as a function of the radius. When the robot describes a large radius circumference, the body turns about a small angle and the stability changes by a small quantity. When the radius of the circle is small, the body turns about a wider angle and the stability changes significantly. When this is carried to an extreme, the machine turns about the z-axis of the body frame, which means a null turning radius, and the resultant gait is also known as a *spinning* gait (Zhang and Song, 1989). These two gaits must be studied separately.

3.6.1 Circling Gaits

With the goal of linking different gaits to follow trajectories, the initial and final leg positions in the body reference frame in a locomotion cycle should be the same. These positions should also be the same as those in the gaits discussed above.

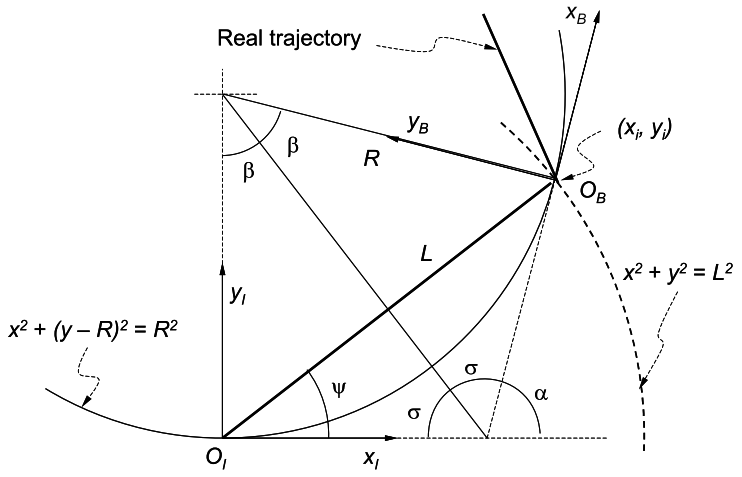


Fig. 3.13. A body trajectory for a discontinuous circling gait

Trajectory Definition

The trajectory of the robot in turning gaits should be circular, but for the sake of simplicity we will approximate the theoretical trajectory to a straight-segment trajectory. Thus, the body trajectory consists of segments whose two end points lie on the ideal circular trajectory of radius R. Each one of these segments will be followed by the COG of the body at the end of every phase. These straight motions will be combined with body turnings about the z-axis to maintain the longitudinal body axis tangent to the circular trajectory. Figure 3.13 shows both the ideal circular trajectory and the real segmented trajectory. This way of describing trajectories is good enough for many applications of walking robots.

If the coordinate frame (x_I, y_I) shown in Fig. 3.13 is the body reference frame at the beginning of the phase, the coordinates (x_i, y_i) represent the location point of the body COG at the end of the current phase. This point is determined by the intersection of the ideal trajectory of radius, R, defined by

$$x^2 + (y - R)^2 = R^2 \tag{3.24}$$

and the circumference of radius L, centered at the origin of the body frame and defined by

$$x^2 + y^2 = L^2$$

where L is the body displacement in every phase.

The simultaneous solution of both equations is

$$x_i = \pm \frac{L}{2} \sqrt{\left(4 - \frac{L^2}{R^2}\right)}$$
$$y_i = \frac{L^2}{2R}. \tag{3.25}$$

When the *COG* is placed on the point (x_i, y_i), the body should rotate about an angle α. In doing that, the longitudinal axis of the body will be tangent to the circular trajectory and the previous procedure can be repeated.

Referring to Fig. 3.13, the following equations can be written:

$$\begin{aligned} \beta &= \psi \\ \beta + \sigma &= \frac{\pi}{2} \\ 2\sigma + \alpha &= \pi. \end{aligned} \tag{3.26}$$

Therefore the angle rotated by the body is given by

$$\alpha = 2\beta$$

where angle β is given by

$$\beta = \arcsin\left(\frac{L}{2R}\right). \tag{3.27}$$

These parameters define the motion of the robot.

For discontinuous circling gaits, the same phasing leg as in discontinuous crab gaits will be considered. This circling gait should satisfy two conditions. First, leg locations have to be placed close to the position for the two-phase gait to maintain stability. Second, leg positions at the beginning and the end of one locomotion cycle have to be the same with respect to the body reference frame. The first requirement could be fulfilled by choosing L close to $R_x/2$, which is the displacement of the body in one phase for the discontinuous zero crab gait. To fulfil the second requirement, leg positions, in the initial reference frame (x_I, y_I), must be placed in positions with similar components into the reference frame at the end of the locomotion cycle (x_B, y_B).

In each phase, the body translates $L = (x_i, y_i)$ and rotates about an angle α. Hence the homogeneous matrix that transforms (x_B, y_B) into (x_I, y_I) is given by

$${}^I\mathbf{A}_B\,(\alpha, x_i, y_i) = \begin{pmatrix} \cos\alpha & -\sin\alpha & x_i \\ \sin\alpha & \cos\alpha & y_i \\ 0 & 0 & 1 \end{pmatrix}. \tag{3.28}$$

Legs 4 and 2 should be placed at the initial points of the gait in the body reference frame, $\mathbf{p_{04}}(x_{04}, y_{04})$ and $\mathbf{p_{02}}(x_{02}, y_{02})$, after the body has finished a locomotion cycle. Therefore, to compute these positions in the first reference

frame, two homogeneous transformations must be performed. Thus, the leg positions become

$$\begin{aligned} \mathbf{p_4} &= {}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right) {}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right) \mathbf{p_{04}} \\ \mathbf{p_2} &= {}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right) {}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right) \mathbf{p_{02}}. \end{aligned} \tag{3.29}$$

In this situation the body needs to be displaced a length, L, and rotated an angle, α. This motion is accomplished by moving the footholds expressed in the reference frame (x_I, y_I) to the reference frame (x_B, y_B), which indicates the body position after the body motion. The homogeneous transformation for this body motion becomes

$$\begin{aligned} &{}^{B}\mathbf{A}_{I}\left(\alpha, x_i, y_i\right) = \\ &{}^{I}\mathbf{A}_{B}^{-1}\left(\alpha, x_i, y_i\right) = \begin{pmatrix} \cos\alpha & \sin\alpha & -x_i\cos\alpha - y_i\sin\alpha \\ -\sin\alpha & \cos\alpha & x_i\sin\alpha - y_i\cos\alpha \\ 0 & 0 & 1 \end{pmatrix}. \end{aligned} \tag{3.30}$$

Now legs 3 and 1 have to be placed in their new location. Note that to compute the initial position of the gait in the body reference frame into the reference frame when the body is at the beginning of the second phase, it is only necessary to perform one transformation ${}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right)$. When the phasing leg sequence is completed, the body has to be displaced and rotated using the transformation ${}^{I}\mathbf{A}_{B}^{-1}\left(\alpha, x_i, y_i\right)$. The algorithm for a discontinuous circling gait may be summarized as follows:

Step 1. Place leg j on initial position $\mathbf{p}_{j0}$ for every j. Compute α, x_i, and y_i.
Step 2. Place leg 4 on $\mathbf{p_4} = {}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right) {}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right) \mathbf{p_{04}}$.
Step 3. Place leg 2 on $\mathbf{p_2} = {}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right) {}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right) \mathbf{p_{02}}$.
Step 4. Body motion: Place leg j on $\mathbf{p_j} = {}^{I}\mathbf{A}_{B}^{-1}\left(\alpha, x_i, y_i\right) \mathbf{p_j}\ \forall j$.
Step 5. Place leg 3 on $\mathbf{p_3} = {}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right) \mathbf{p_{03}}$.
Step 6. Place leg 1 on $\mathbf{p_1} = {}^{I}\mathbf{A}_{B}\left(\alpha, x_i, y_i\right) \mathbf{p_{01}}$.
Step 7. Body motion: Place leg j on $\mathbf{p_j} = {}^{I}\mathbf{A}_{B}^{-1}\left(\alpha, x_i, y_i\right) \mathbf{p_j}\ \forall j$.

Stability Margin

Figure 3.14 shows the S_{LSM} of a machine, with the same parameters as in previous examples, walking with a discontinuous circling gait. Leg 3 determines the S_{LSM} of the locomotion cycle, and the machine becomes unstable for circling radii of less than approximately 1 m.

In this circling gait, a distinction should be made between positive and negative circling, shown by the sign of the rotation vector. The phasing leg is similar for both cases, but the coordinates depend on the sign of the turning angle. For example, Fig. 3.13 shows positive circling, and x_i and y_i are positive; but for negative circling, y_i is negative while x_i remains positive. Leg positions on the right and left sides are not symmetrical; therefore, the S_{LSM} depends on the circling sign. Nevertheless, the influence of the sign of the

circling angle is evident in circling with a small radius, where the difference between leg positions is relatively significant. For large circling radii, the relative difference between leg positions is insignificant. S_{LSM} curves are similar for both positive and negative circling with small differences for small radii. A significant difference is that similar curves are obtained for diagonal legs in the transfer phase. Hence, for negative circling, the S_{LSM} is determined by leg 2, and the curve thus obtained is similar to the one for leg 3 in positive circling.

3.6.2 Spinning Gaits

Above it was shown how circling gaits become unstable for small radii. When the robot needs to follow a small radius circle, the phasing leg sequence must be changed. As the range of unstable radii for circling gaits is relatively short, a spinning gait, which means body rotation about the z axis, will be considered as the unique gait for that range.

Phasing Leg Sequence

In the previous gaits studied, the main consideration was to maintain leg positions at the same body coordinates after a locomotion cycle. This consideration will be kept in the spinning gaits. Figure 3.15(a) shows the leg positions at the beginning (dotted line) and the end (solid line) of the first phase. The robot and leg workspaces at the end of the first phase are rotated

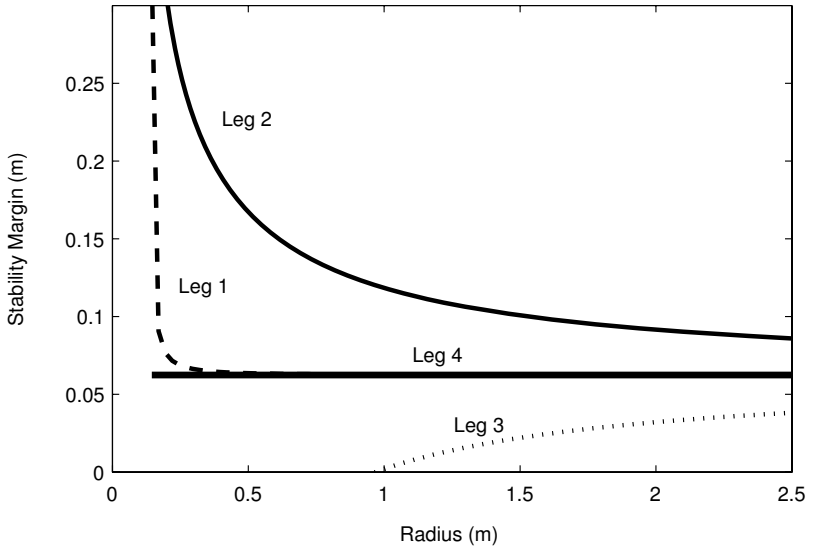
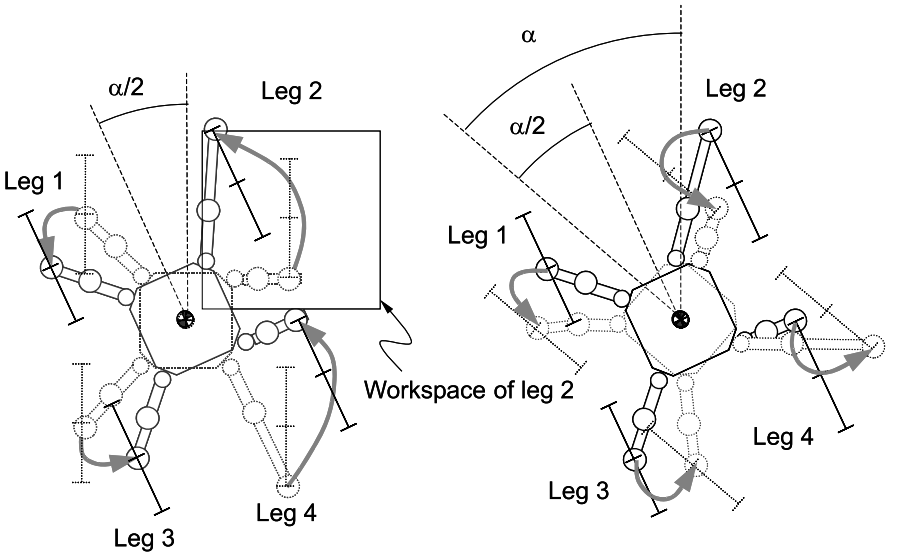


Fig. 3.14. S_{LSM} for a positive circling gait for the indicated leg in transfer



(a) (b)

Fig. 3.15. Gait pattern for a discontinuous spinning gait: (**a**) first phase; (**b**) second phase

$\alpha/2$. Legs 4 and 2 are placed in sequence in their location and then the body is rotated, but its rotation will not place the left legs at their right position at the end of the first phase (same positions as in the two-phase discontinuous gait). To place these legs in their positions, two more leg motions are required (legs 1 and 3). Thus, we see that in a discontinuous spinning gait one phase is determined by four leg motions instead of two, as in previous gaits. Notice that the foothold of a leg in transfer must be inside the initial leg workspace (before rotating the body). Therefore the leg stroke should be lower than the real leg workspace. To clarify this point the real leg workspace for leg 2 has been depicted in solid line in Fig. 3.15.

Figure 3.15(b) shows the initial position (solid line) and the final position (dotted line) of the second phase. Again, all four legs must be located in final positions before the final body rotation.

Leg locations at the end of the first phase are calculated by applying a homogeneous z-axis rotation of $\alpha/2$, ${}^{I}\mathbf{A}_{B}\,(\alpha/2, 0, 0)$, to the leg positions at the end of the first phase in the two-phase gait. The angle α through which the body rotates in a locomotion cycle must be a sub multiple of the angle rotated about in the whole trajectory, β; *i.e.* $\alpha = \beta/\mathrm{n}$, where n is the number of locomotion cycles required to complete the circular trajectory. The angle α should be chosen considering the following stability study.

With the location known, the leg sequence must be specified to define the gait. A geometric analysis of Fig. 3.15 reveals that the ordinary leg sequence 4-2-3-1 used in previous gaits is unstable in this case. Analyzing the 4! possible sequences, several stable gaits can be found. For instance, leg sequence 3-4-2-1 is stable for the first phase, and leg sequence 1-2-4-3 is stable for the second phase. These two leg sequences will be analyzed in the following sections.

Stability Margin

Figure 3.16 shows the S_{LSM} for the spinning gait defined above for every leg in its transfer subphase. Leg 4 in the first phase presents the smallest S_{LSM} for a negative rotation; for a positive rotation, the S_{LSM} is exhibited by leg 2 in the second phase. These two margins determine the S_{LSM} of the locomotion cycle (the minimum along a full locomotion cycle) indicated in solid line in Fig. 3.16.

The S_{LSM} for positive and negative spinning gaits are given by

$$S_{LSM_+} = \frac{P_y R_x \cos\frac{\alpha}{2} - \left(P_x^2 + P_y^2 - P_x R_x\right)\sin\frac{\alpha}{2}}{4\cos\frac{\alpha}{4}\left(P_y \cos\frac{\alpha}{4} + P_x \sin\frac{\alpha}{4}\right)} \tag{3.31}$$

and

$$S_{LSM_-} = \frac{P_y R_x \cos\frac{\alpha}{2} + \left(P_x^2 + P_y^2 - P_x R_x\right)\sin\frac{\alpha}{2}}{4\cos\frac{\alpha}{4}\left(P_y \cos\frac{\alpha}{4} - P_x \sin\frac{\alpha}{4}\right)} \tag{3.32}$$

respectively.

Section 3.6.2 shows that a possible stable sequence for a discontinuous spinning gait is 3-4-2-1-1-2-4-3. This means that leg 1 executes two consecutive motions. If leg 1 in the first phase is placed in its second-phase location, this avoids an extra motion. If the spinning motion consists of several locomotion cycles, the same procedure could be applied to leg 3. The final leg sequence could be 3-4-2-1-2-4. The S_{LSM} for this new sequence is provided by

$$S_{LSM_{R+}} = \frac{P_y R_x \cos\alpha - \left(P_x^2 + P_y^2 - P_x R_x\right)\sin\alpha}{4\cos\frac{\alpha}{2}\left(P_y \cos\frac{\alpha}{2} + P_x \sin\frac{\alpha}{2}\right)} \tag{3.33}$$

for a positive rotation and

$$S_{LSM_{R-}} = \frac{P_y R_x \cos\alpha + \left(P_x^2 + P_y^2 - P_x R_x\right)\sin\alpha}{2\left(P_y + P_y \cos\alpha - P_x \sin\alpha\right)} \tag{3.34}$$

for a negative rotation.

Figure 3.16 shows in dotted lines the S_{LSM} for the spinning gait with a reduction in the number of leg motions. Unfortunately, both S_{LSM} and maximum spinning angle in each cycle are smaller for the reduced spinning gait. Referring to Fig. 3.16, we may conclude that a reduced spinning gait is adequate for rotation angles smaller than $\pm 16^{\circ}$, but for greater angles, it is convenient to employ the full spinning gait. Notice that all these data depend on the geometric features of the example we are considering.

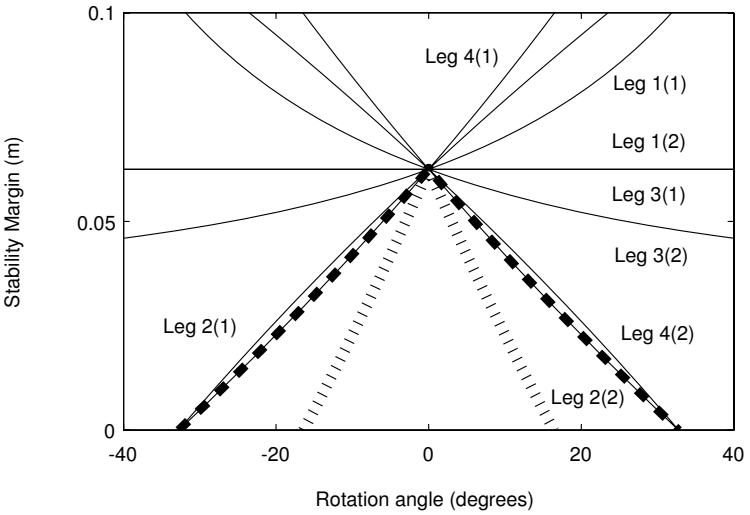


Fig. 3.16. S_{LSM} for a discontinuous spinning gait. Leg i(j) indicates the S_{LSM} for leg i in transfer along the j phase. The S_{LSM} of the locomotion cycle is indicated in *thick-dashed line*. In *thick-dotted line* is the S_{LSM} for the reduced gait.

3.7 Path Tracking with Discontinuous Gaits

Derivation of discontinuous gaits has been performed keeping in mind the periodicity of footholds to allow joining consecutive locomotion cycles to follow trajectories. This section considers the tracking of trajectories by using both crab and turning gaits.

Path planning and tracking for wheeled robots have been studied broadly and many different trajectories have been designed to enhance robot motion with a view to maneuverability, acceleration, *etc.* Good performance has been gained using β-splines, Bezier curves, and clothoids – plane curves whose curvature is a linear function of length. In this chapter's approach, a numeric solution will be considered; hence, any kind of mathematical function may be used to define the path.

3.7.1 Path Tracking with Crab Gaits

The definition of the discontinuous crab gait guarantees that foot and *COG* workspace are equal in size, and leg trajectory and *COG* trajectory are equal and parallel to each other as well. Let us assume that the *COG* is on the desired path and the body's longitudinal axis is aligned with the x-axis of the path reference system. If the path is given by $y = p(x)$, the problem is to place the *COG* over this trajectory as far as possible in every locomotion cycle, taking into account that this point must stay within the bounds of the *COG*

workspace. Therefore, the new COG position is defined by the intersection of the path function, $p(x)$, with the boundary of the COG workspace.

To find the new COG position, the first step is to test what type of trajectory the legs need to follow. Figure 3.9 shows how the angle separating the two trajectories is

$$\alpha_{_L} = \arctan\left(\frac{R_y}{2\,R_x}\right). \tag{3.35}$$

If the COG is located at x_m and the trajectory crosses the front limit of the COG workspace, then the trajectory verifies

$$\alpha_{_A} = \arctan\left(\frac{p(x_m + R_x) - p(x_m)}{R_x}\right) \leq \alpha_{_L}\,. \tag{3.36}$$

That means the legs will perform a type-A trajectory. Otherwise, a type-B trajectory will be achieved (see Fig. 3.9).

If it is possible to accomplish a type-A trajectory, the new COG position will be $(x_m + R_x, p(x_m + R_x))$. The body will travel to this position following a straight line from the initial position $(x_m, p(x_m))$. The procedure will then be repeated.

When the COG has to follow a type-B trajectory, the path crosses a lateral limit of the COG boundary. To discover this crossing point, the method of successive bisection can be used. This method computes the root of an equation $f(x) = 0$ in the interval $[x_1, x_2]$ where $f(x_1)f(x_2) < 0$. To locate the root, the interval $[x_1, x_2]$ is bisected at point $x' = (x_1 + x_2)/2$. If $|x_1 - x_2| \leq \varepsilon$, where ε is a small positive quantity, then x' is a root. Otherwise, $[x_1, x']$ could contain the root if $f(x_1)f(x') \leq 0$. If not, $[x', x_2]$ contains the root and $f(x')f(x_2) \leq 0$ is verified. This process is repeated over the interval containing x' until the condition $|x_1 - x_2| \leq \varepsilon$ is satisfied.

This iterative method is simple and ensures a root of the equation, but the number of iterations may be high if ε is too small. This number defines the accuracy of the solution. In the case of walking machines, a precision of half a centimeter could be adequate. For $\varepsilon = 0.005\,\mathrm{m}$ this algorithm finds a root in a few iterations without a large computational burden.

The above method computes a root of the equation $f(x) = 0$, but the problem here is to compute the intersection of $p(x)$ with a horizontal line passing through either the point $(x_m, p(x_m) + R_y/2)$ or $(x_m, p(x_m) - R_y/2)$, depending on the lateral limit intersecting the path. If the x-axis of the coordinate system is translated to the lateral limit position, the bisection method may be applied. This axis translation is equivalent to computing the root of the function

$$f(x) = p(x) - p(x_m) - \frac{\alpha_{_A}}{|\,\alpha_{_A}\,|}\frac{R_y}{2} \tag{3.37}$$

where x_m is the abscissa of the COG and α_A is an estimate of the foot trajectory angle that can be computed by (3.36).

With the new COG position known, the motion of the robot is performed with the gait algorithm described in Sect. 3.5.

3.7.2 Path Tracking with Turning Gaits

The discontinuous turning gait defined in Sect. 3.6 is characterized by the point (x_i, y_i) and the angle α_i. The point (x_i, y_i) is the new position of the *COG* at the end of a phase or a semi-cycle, and α is the angle the body has to rotate in each phase to maintain the body's longitudinal axis tangent to the desired path. The new *COG* position lies on the path at a distance L from the current *COG* position. Therefore, this point is the simultaneous solution of the function defining the path

$$y = p(x) \tag{3.38}$$

and the equation of the points whose distance is L from the current *COG* position, given by

$$(x - x_m)^2 + (y - p(x_m))^2 = L^2 . \tag{3.39}$$

Newton's method of solving non-linear equation systems has been used to compute the solution. This method solves a non-linear equation system such as

$$\begin{aligned} f(x, y) &= 0 \\ g(x, y) &= 0 \end{aligned} \tag{3.40}$$

provided that an initial approximation (x_0, y_0) of the solution is available. This method may be found in any elementary numerical method textbook. Usually, a few iterations of this process produce accurate solution values, providing that the initial approximation is sufficiently close to the true solution.

To apply Newton's method to Equations (3.38) and (3.39), they should be written as

$$\begin{aligned} f(x) &= y - p(x) = 0 \\ g(x) &= (x - x_m)^2 + (y - p(x_m))^2 - L^2 = 0. \end{aligned} \tag{3.41}$$

The solution of this system provides the new position for the *COG*. The rotation angle of the body to place its longitudinal axis tangent to the path is given by the derivative of the path at the new *COG* point, $p'(x_m)$.

This method requires an initial approximation of the solution. A rough approach may be the current *COG* position. A better approach is obtained by choosing a point in the direction of the tangent to the path at a distance L, *i.e.* $(x_m + (L/2) \cos \alpha_T, y_m + (L/2) \sin \alpha_T)$, where $\alpha_T = \tan^{-1}(p'(x_m))$. Using the rough approximation, the algorithm converges in an average of 20 iterations, while the second approximation can yield a solution in 2 iterations.

3.7.3 Path Tracking Examples

To illustrate the above methods, some simulations have been performed to follow a path with a discontinuous crab gait and a discontinuous turning gait, respectively. The function used to illustrate both cases is

$$p(x) = -\frac{1}{3}\sin\left(\frac{\pi}{2}x\right)^2. \tag{3.42}$$

Figure 3.17 shows the path and the segmented trajectory followed by the body when using the method described in Sect. 3.7. The rectangle superimposed on both the desired and the real path represents the *COG* workspace of the robot. The true *COG* workspace is advanced half a stroke with respect to the defined body workspace. Figure 3.17 shows in a solid line this area for the first pose of the machine. When a type-A trajectory can be run, the body is propelled forward by a full stroke. Thus, consecutive body workspaces lie tangent to each other. In the other case, a type-B trajectory is performed, and contiguous body workspaces overlap.

Figure 3.18 shows body poses when the path is followed using a discontinuous turning gait. In this case the body position at the end of each phase is drawn. Hence, there are two body plots per locomotion cycle in this figure.

The path in our example is followed quite well using crab gaits and it is perfectly followed using turning gaits. Of course, a path with large changes in its derivative may require either a crab or rotation angle, that makes the robot unstable. In such a case, the robot has to employ a different gait. For crab walking, the robot will need to change its leg trajectories (Cases C or D, for instance). For turning gaits, the robot may need to use spinning gaits.

3.8 Conclusion

Continuous and discontinuous periodic gaits for walking machines have been studied and compared. Some advantages of discontinuous gaits over their counterparts have been pointed out. Leading among these advantages are improved stability, simplicity of implementation in real machine controllers,

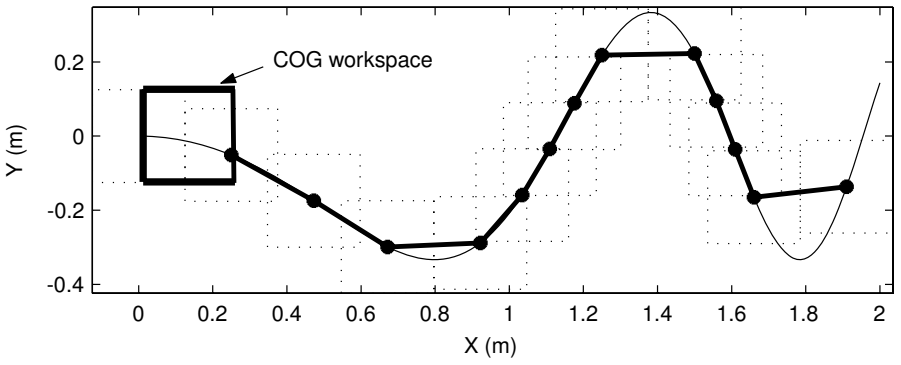


Fig. 3.17. Path tracking using a discontinuous crab gait: desired trajectory in *thick-solid line*, real trajectory in *solid line*, *COG* workspace in *dashed line*

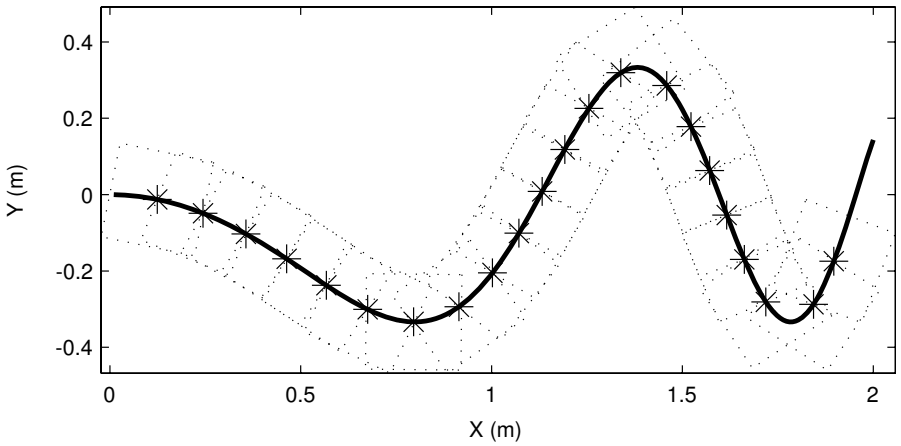


Fig. 3.18. Path tracking using a discontinuous turning gait

better maximum achievable velocity than wave gaits for median and large duty factors, *etc.* Continuous gait, in contrast, exhibit a smooth body motion while discontinuous gaits move the body jerkily.

After establishing the importance of discontinuous gaits, several variations in crab and turning walking have been investigated. Four different methods for discontinuous crab walking have been studied. They differ in stability and efficiency, and a strategy for selecting the most suitable methods has been addressed.

Turning gaits, divided according to turning radius into circling and spinning gaits, have also been analyzed. Discontinuous crab and circling gaits employ the same leg sequence as well as the same leg location at the beginning and end of a locomotion cycle. This allows different gaits to be joined to follow trajectories.

Spinning gaits lose stability when they use the same leg sequence as other gaits, but a new leg sequence has been defined to use the same footholds. In this way, spinning gaits can also be employed together with crab and circling gaits to follow defined paths. Two different leg sequences have been considered for spinning gaits in order to optimize either the number of leg motions or the turning angle.

Section 3.7 presents some algorithms to follow predefined paths by using discontinuous crab and turning gaits. The gaits presented in this chapter have been checked with the SILO4 walking robot.

4

Generation of Non-periodic Gaits

4.1 Introduction

Periodic gaits, introduced in Chap. 3, present several important advantages that soon generalized its use. However, they also present remarkable disadvantages, caused by the rigidity of their formulation, which hampers their adaptability.

As stated in Chap. 1, one gait algorithm problem is navigation over a given path. Periodic gaits can follow straight or circular trajectories but joining these elemental trajectories to follow complex paths is troublesome. This is because periodic gaits need certain initial foot positions (in the body reference frame) to carry out the motion, and these positions depend on the trajectory. Although several solutions have been proposed to connect different periodic gaits (see Chap. 3), a fully adaptable gait should be able to find the adequate footholds and appropriate sequences of leg transferences to follow any trajectory starting from any initial conditions.

Another limitation of periodic gaits is that they are ineffective on terrain containing forbidden areas, *i.e.*, regions unsuitable for the support of the walking machine. A forbidden area may correspond, for example, with a hole or a vertical edge on irregular terrain. Again, a fully adaptive gait should be able to change its support points, and even its leg sequence to traverse a very rough terrain to avoid stepping on such regions.

These problems motivated the study of free gaits since the first years of walking robots (Kugushev and Jaroshevskij, 1975). In a free gait, the leg sequence, footholds and body motions are planned in a non-fixed, flexible way as a function of the trajectory, the ground features and the machine's state. Hence, free gaits are more powerful than periodic and adaptive gaits when a functional level of mobility is required in terrain with forbidden areas.

A large number of free gaits for quadruped and hexapod robots have been developed to date. Free gaits have proven themselves effective at controlling hexapods (Wettergreen and Thorpe, 1992; Salmi and Halme, 1996), due to the high number of possible choices (footholds, body motions, *etc.*) that satisfy

the unavoidable kinematic and stability restrictions. In the case of quadruped robots, free gait algorithms are more susceptible to deadlocked states, defined as situations in which those basic restrictions cannot be satisfied jointly. The number of successful free gaits is considerably smaller for quadrupeds than for hexapods, and the results reported suggest that additional work is still needed to achieve an effective locomotion of quadrupeds on rough terrain with forbidden areas in realistic conditions.

There are two main methods used to generate free gaits: rule-based and search-based free gaits. Both methods have been tested mostly in simulation, yielding adequate results with quadruped and hexapod robots.

The rule-based method plans the robot's motions by the use of rules designed by the programmer (Hirose, 1984; Shih and Klein, 1993; Chen *et al.*, 1999b; Bai *et al.*, 1999), learned automatically (Maes and Brooks, 1990), or derived from biological mechanisms found in nature (Dean *et al.*, 1999). These rules incorporate some kind of knowledge about how the robot should move to achieve effective locomotion. For example, the deliberative approach proposed by Hirose (1984) employed a geometric method to restrict and select footholds in such a way that the standard gait sequence (see Sect. 3.2) could be maintained. A posterior work by Bai *et al.* (1999) combined that sequence with a method to generate alternative leg sequences when the standard gait sequence was not viable. Rule-based algorithms were the first approach employed to generate free gaits. However, in the case of quadruped robots rule-based algorithms present several drawbacks; one of them is the complexity of their formulation, which means excessively simplified models must perforce be used. For example, the use of Longitudinal Stability Margins (see Chap. 2) (Hirose, 1984; Bai *et al.*, 1999; Chen *et al.*, 1999b), simplifies the formulation, but this approach is not practical for a real robot walking on irregular terrain with arbitrary leg trajectories and crab angles.

In the search-based method, different series of robot actions are blindly generated and tested in simulation to determine if they would produce adequate vehicle motion (Pal and Jayarajan, 1991; Wettergreen and Thorpe, 1992; Chen *et al.*, 1999a; Eldershaw and Yim, 2001; Pack and Kang, 1999). Since a huge number of series of robot motions is possible, search methods must be employed to find a valid motion plan. Although search-based free gaits offer a simple strategy which is independent of the number of legs, some shortcomings may hinder its application to real quadruped robots. For example, the number of options considered for each robot action must be strictly limited to avoid an unmanageable number of possible motion series. Some authors (Pack and Kang, 1999; Pal and Jayarajan, 1991), consider three or four foothold candidates for the placement of each foot; this is a severe restriction for quadrupeds, whose footholds should be carefully planned to achieve adequate leg sequences while maintaining stability.

The weakest point of many deliberative free gaits is their typical difficulty of combining meticulously planned actions with the reactive behaviors needed for satisfactory performance in the real world. Reactive free gaits have been

tested successfully in hexapod (Brooks, 1989; Dean *et al.*, 1999) and eight-legged robots (Bares and Wettergreen, 1999), where stability restrictions can be formulated in a very simple way (*i.e.* considering just the states of the legs, transfer or support, and ignoring its positions), and foothold planning is not as strictly restricted by stability and sequencing conditions as it is in quadrupeds. Recently, reactive gaits based on rhythmic pattern generators (Fukuoka *et al.*, 2003; Lewis and Bekey, 2002) have yield impressive results at controlling quadruped robots. However, no pure reactive approaches have been successful, for example, in planning precise footholds on rough terrain with forbidden areas.

This chapter presents new free gaits which can be employed to navigate omnidirectionally over irregular terrain with a real four-legged robot (Estremera and Gonzalez de Santos, 2002). These gaits can plan the motion of the different parts of the robot to achieve a statically stable locomotion in which the body follows straight or circular elementary trajectories. The proposed deliberative rule-based algorithms can generate flexible leg sequences and select adequate footholds in order to enhance maneuverability and terrain adaptability. The free gaits can be combined by higher level software module to perform efficient path tracking. Also, a human operator can combine the three gaits to steer the robot in real time, as explained in Chap. 9. However, path planning is out of the scope of this chapter. These gaits have been tested on a real robot, and their efficiency has been validated, demonstrating the improvement in performance over previous similar gaits. The chapter is organized as follows: Sections 4.2, 4.3 and 4.4 present a free-crab gait, a free-turning gait and a free-spinning gait respectively. Section 4.5 presents some experiments with the SILO4 walking robot. Finally, some conclusions are reported in Sect. 4.6. The formulation and results of the free gaits are general for quadruped legged robots, but examples and implementation details have been particularized for the SILO4 walking robot.

4.2 Free-crab Gait

As outlined above, this new free gait is the result of an attempt to implement effective, efficient gaits in a real legged robot able to negotiate uneven terrain. Statically stable gaits were considered adequate to control simple walking machines equipped with slow actuators and a minimum set of sensors. Furthermore, many of the possible applications of walking machines (see Sect. 1.5) require heavy and slow moving machines which can be controlled considering static stability. Non-periodic free gaits were selected in principle because of their ability to change trajectory at any time and to progress on rough terrain with forbidden areas. Discontinuous gaits (see Sect. 3.4) were also considered good candidates because of their intrinsic features for terrain adaptation and the ease of implementation as well. In discontinuous gaits, the body of the vehicle stands still while one leg is in its transfer phase. This fact simplifies

the formulation of the gait as the leg coordination problem is avoided. The selection of footholds is only a matter of determining the x and y components of the foothold, and no temporal constraint must be imposed in this selection. This means that the transfer time from the old support point to the new foothold is irrelevant. For instance, the stability of the robot will not be imperiled by a longer than expected transference. Similarly, the supporting feet will never go out of their work volumes during a leg transference, no matter how long it lasts. The z component of the new foothold can be ignored also by the gait planning algorithms and can be determined *a posteriori* during the execution of the transference by using simple touch sensors.

The free gait presented here was inspired by the algorithm proposed by Hirose (1984), especially the method employed for foothold search. The leg sequence planning follows the approach proposed by Bai *et al.* (1999) in the sense that the standard leg sequence is adopted as a default, but flexible leg sequences can be generated when needed to lift legs with lower kinematic margins.

The gait is accomplished by the following modules: the *sequence planner* (in charge of imposing certain leg sequence criteria to coordinate foothold-searching and leg-lifting), the *foothold planner*, the *body motion planner* and the *leg-lifting planner*. A general sketch of the algorithm, extended with additional stability constraints, is presented in Fig. 5.10. Previous to the description of the gait modules, some basic concepts are defined in the next section.

4.2.1 Walking Machine Model and Basic Definitions

A simplified two-dimensional model of a quadruped walking machine is employed to derive the gait. The center of gravity is assumed to be located in the geometric center of the body, which is the origin of the body reference frame; this assumes massless legs. The body is assumed to be level during the locomotion. This is guaranteed by the use of a reactive attitude controller that rotates the body to level it by using inclinometers in the control loop. The foot workspace is bounded by two horizontal planes and an arbitrary-shaped vertical surface, and adjacent leg workspaces can overlap. For example, in the case of the SILO4, the leg workspace could be considered as a regular semi-cylinder contained in the true workspace of the leg (see Fig. 4.1).

As a measurement of the machine stability the Static Stability Margin (S_{SM}) is considered (see Sect. 2.2). The algorithm maintains the Static Stability Margin greater than a given minimum, $S_{SM}^{\min}$, as a way to improve the performance in realistic conditions as well as to cope with model imperfections. The machine is considered stable if this condition is fulfilled. As pointed out above, several previous works (Hirose, 1984; Bai *et al.*, 1999; Chen *et al.*, 1999b; Pack and Kang, 1999), employed the Longitudinal Stability Margin, L_{LSM} (see Sect. 2.2) to simplify the formulation. Thus, the use of the Static

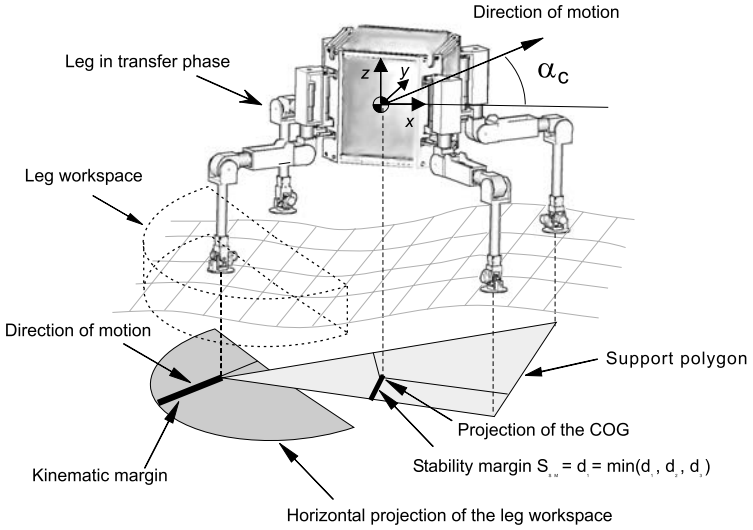


Fig. 4.1. Definition of magnitudes in a statically stable walking robot

Stability Margin can be considered the first improvement offered by the algorithm presented here. McGhee and Iswandhi (1979) introduced the *kinematic margin* (KM) of a leg as the distance that the foothold of a given leg can travel in the opposite direction of motion before reaching the boundary of its workspace. Figure 4.1 illustrates these basic concepts and definitions.

The following nomenclature is used for deriving the free gait:

- LT: Leg in its transfer phase.
- NLT: Next leg to go to transfer phase.
- $KM_{\min}$: Minimum kinematic margin of the supporting legs.
- $LKM_{\min}$: Leg with the minimum kinematic margin.
- LKM_i: Leg with the i–th smaller kinematic margin.
- $S_{SM}^{\min}$: Minimum stability margin demanded during locomotion.

The crab gait can be subdivided into four different types. Each type allows the body to move along a trajectory forming a certain angle with the x-axis of the body reference frame. This angle, α_c, is known as the *crab angle*, and the different types of gait are defined as:

- Type X+: when $315^o < \alpha_c < 45^o$.
- Type X-: when $135^o < \alpha_c < 225^o$.
- Type Y+: when $45^o < \alpha_c < 135^o$.
- Type Y-: when $225^o < \alpha_c < 315^o$.

These domains have been chosen by taking into consideration the symmetric properties of quadruped machines. After selecting the gait type as a function of the crab angle, the following duties are assigned to the legs:

- FRL: Front right leg.
- FLL: Front left leg.
- RRL: Rear right leg.
- RLL: Rear left leg.

Depending on this assignment, two legs can be classified as collateral (two right or two left legs) or contralateral; also, independently of the crab angle, two legs can be classified as adjacent or non-adjacent.

4.2.2 Terrain Model and Terrain Adaptation

The H-Type terrain described by Hirose (1984) was used to create a simulated environment. An H-Type terrain is defined as the ground on which regular walking can be realized without hindrance, but contains depressions (holes or ditches). In order to use a bi-dimensional gait planning the following possibilities for a foothold candidate should be considered:

1. The height of the foothold, referred to the robot's reference frame, is between the upper and lower limits of the workspace, so it is possible to place the foot in that foothold. In this case, that foothold will be an allowed point of the terrain, and ground adaptation is solved in a straightforward way by the discontinuous gait.
2. The foothold is below the lower limit of adaptation, that is, the foothold is placed in a hole. In this case, that foothold is considered a forbidden point of the terrain.
3. The foothold is above the upper limit of adaptation, that is, the foothold is placed in a protuberance of the terrain. In this case, this part of the terrain will be considered as an obstacle, since no part of the robot can pass above it.

The classification of the possible footholds for a leg into one of the two first possibilities is the only information needed by the gait generator about the environment to perform properly on practicable terrain. The third possibility represents impracticable terrain, and a higher level of autonomy should change the trajectory of the robot to avoid such regions.

The terrain is divided into square cells, which can be marked as forbidden or allowed terrain for a foothold. At this stage, the robot knows a complete map of the forbidden cells of the terrain. However, at each leg transference, only the cells contained in the leg's workspace are explored, since footholds are planned only one leg motion in advance. This is equivalent to exploring in real time the possible footholds for a particular leg (which are located in a near region of the terrain), by the use of sensors. The implementation of such a sensorial system is a very important task to achieve.

When a new foothold has been selected, the transfer foot is lifted vertically, placed above the foothold – *i.e.*, in its x and y coordinates - and lowered following a vertical trajectory. This trajectory is stopped when a contact sensor indicates the foot has touched the terrain surface. Knowing the relative height of the supporting points (or using exteroceptive terrain information, if available) an independent altitude controller regulates the vertical motion of the body to keep it at a convenient height above the terrain.

4.2.3 Leg Sequence Planner

The leg sequence planner determines the next leg that should be lifted and the conditions that a foothold should meet in order to bring about convenient leg sequences. These conditions will determine the shape of the foothold-searching zone for a transfer leg. We will consider two basic criteria that, merged together, will form the final algorithm.

Criterion N: Natural Sequence

In the first criterion, the leg sequence is planned based on the standard gait sequence used in wave gaits (see Sect. 3.2). Hirose (1984) adopted this sequence for his free gait, and it is also adopted by the two-phase discontinuous gait (see Sect. 3.4.1). The main reason for selecting this leg sequence is because optimal speed and stability can be achieved with it. This leg sequence combined with the body motions will be called the *natural sequence*, which is a periodic sequence defined in the following manner:

RRL transfer⇒ FRL transfer⇒ body motion ⇒
RLL transfer⇒FLL transfer⇒body motion

Given a leg in transfer phase, LT, the sequence planner will notify the foothold planner the conditions required for the new footholds to continue with this sequence. These sequencing conditions are listed below (see Fig. 4.2(h)):

Condition N1: *If LT is a rear leg, then its new foothold must permit the stable lifting of its collateral leg immediately after the placement of LT.*

Condition N2: *If LT is a rear leg, its foothold must enable the stable lifting of its contralateral rear leg after a body motion of length $KM_{\min}$. To calculate the $KM_{\min}$, the KM of the leg collateral to LT will not be taken into consideration, as that leg will be lifted immediately after LT is placed (Hirose, 1984).*

Condition N3: *If LT is a front leg its foothold must favor the formation of searching areas accomplishing conditions N1 and N2 for its contralateral rear leg.*

Before applying Condition N3, a check must be run to ascertain whether the body motion required to lift the rear leg contralateral to LT under static stability is smaller than $KM_{\min}$. If not, the search fails since the contralateral rear leg cannot be lifted with stability, regardless of the foothold of LT. Further description of this condition will be given in Sect. 4.2.4

Condition N4: *The KM of the new foothold must be greater than the $KM_{\min}$.*

This condition makes sure that the placement of LT does not result in a reduction of the overall KM.

If a foothold accomplishing these conditions is found, then the next leg of the natural sequence will be marked as the new LT.

Criterion K: Kinematic Margin

In the second criterion, the leg sequence is selected based on the leg kinematic margins used by McGhee and Iswandhi (1979) and will attempt to lift, under static stability, the legs with the lower KMs in order to increase the $KM_{\min}$ of the supporting legs.

Let us assume the machine has one leg in transfer phase, LT. To maximize the minimum kinematic margin of the legs in support, the leg with the lowest kinematic margin should be lifted first. So, we will assign $NLT = LKM_{\min}$, and depending on the assigned duties of the current LT and NLT, the following conditions are imposed to the foothold for LT:

Condition K1: *If LT and NLT are collateral and LT is a rear leg, then the foothold must permit the stable lifting of NLT immediately after the placement of LT (see Fig. 4.2(a)).*

This condition has proven to be useful for putting collateral legs in a similar phase, as needed in criterion N (see Sect. 4.2.3).

Condition K2: *If LT and NLT are collateral and LT is a front leg, then the foothold for LT must permit the stable lifting of NLT after a body motion of length $KM_{\min}$ (see Fig. 4.2(b)).*

In combination with condition K1, this condition helps to bring about a difference in the phases between contralateral legs.

Condition K3: *If LT and NLT are rear legs, then the foothold for LT must permit the stable lifting of NLT after a body motion of length $KM_{\min}$ (see Fig. 4.2(c)).*

This condition will favor body motions between rear-leg transfers, helping to produce a difference between contralateral-leg phases.

Condition K4: *If LT and NLT are front legs, then the foothold must permit the stable lifting of NLT immediately after the placement of LT (see Fig. 4.2(d)).*

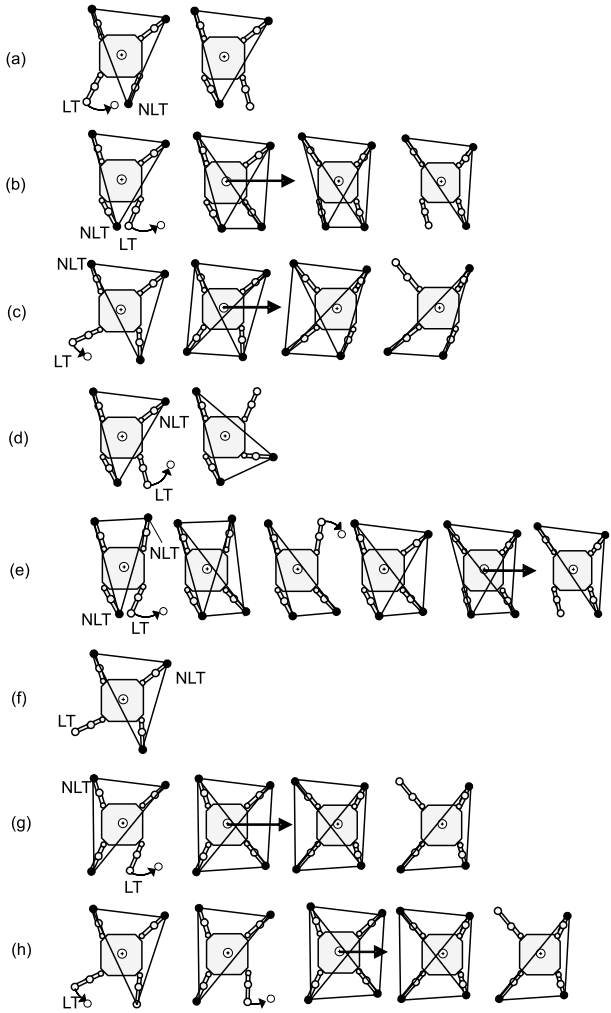


Fig. 4.2. Sequence conditions

Condition K5: *If LT and NLT are front legs, then the foothold for LT must permit the stable lifting of the leg collateral to LT after a body motion of length* $KM_{\min}$ *(see Fig. 4.2(e)).*

This is an optional constraint that helps to avoid deadlock and is ignored if it cannot be satisfied jointly with K4.

Condition K6: *If LT and NLT are non-adjacent and LT is a rear leg, then the search fails and a new leg is assigned as NLT (see Fig. 4.2(f)).*

This is because NLT cannot be lifted with stability, independently of the foothold for LT.

Condition K7: *If LT and NLT are non-adjacent and LT is a front leg and the body motion needed to lift the NLT (which only depends on the position of the two remaining legs) is shorter than the $KM_{\min}$, then the sequence planner will not impose any conditions but K8 on the foothold search (see Fig. 4.2(g)).*

Condition K8: *In any case, the KM of the foothold must be greater than the KM of LT.*

This condition increases the net $KM_{\min}$ when NLT goes to the transfer phase.

The set of conditions K1–K8 has been designed to avoid deadlock and has proven in simulation to facilitate the transition to the *natural sequence* described in Sect. 4.2.3.

If the search for a foothold fails under these conditions, then the leg with the second smallest KM is marked as the NLT, that is, it is assigned $NLT = LKM_2$ and the foothold search is repeated. If the foothold search still fails, then the procedure is repeated considering $NLT = LKM_3$. When a valid foothold is found, then the transference of LT is executed and the NLT is marked to be as the new LT by the leg-lifting planner. This strategy decreases the likelihood of leg deadlock because the foothold for the LT may be chosen from a variety of different searching zones, as we will see in Sect. 4.2.4.

Complete Algorithm

The gait algorithm will use criterion N as the starting point to obtain the leg sequence and search for footholds. That is, the natural sequence will be adopted to obtain high speed. This criterion normally works adequately in the following situations:

1. Initial foot positions are close to those of the two-phase discontinuous gait.
2. The direction of motion is close to the longitudinal or transversal axes of the body reference frame.
3. There are few or no forbidden areas.

If this criterion fails, *i.e.* the next leg cannot be lifted or conditions N1 to N4 cannot be met by any foothold, then criterion K is used, and conditions K1–K8 are employed to search for a new foothold. This second criterion works adequately for a higher number of situations and is able to search a higher number of options reducing the possibility of deadlock. If no valid foothold is found, the gait is deadlocked and the algorithm fails. Despite of the combined use of the two sequencing criteria in this method there are no separated gaits, nor are there any states like those described in (Bai *et al.*, 1999), which include complete predefined sequences of leg lifting and placements. The two criteria N and K only denote two different viewpoints about sequence planning. They are

used in this order to find every foothold, or to mark every new transferring leg, and they only indicate the order in which different possibilities are explored.

4.2.4 Foothold Planner

After selecting the basic conditions the foothold must meet, defined by the sequence planner, the exact points for placing the legs are calculated by the foothold planner. To do that, the foothold planner delimits a valid area for locating the foot, called the *Effective Searching Zone* (ESZ). The sections below will describe how the foothold planner interprets the conditions imposed by the sequence planner to define the ESZ, and the final algorithm used in the foothold search.

Foothold Search and Sequence Conditions

This section describes the way the ESZ of a transfer leg is limited in order to meet the sequencing conditions. Three foothold restricting areas inspired by the diagonal principles introduced by Hirose (1984) are employed to constrain the possible footholds. Depending on the sequencing conditions imposed to the foothold, one or more of the areas described below are used:

Area A: This area is defined as the portion of the plane in which the placement of LT will allow the lifting of an adjacent leg NLT immediately afterward. This restriction will favor consecutive leg transfers, without body motions between them. The area that satisfies this condition is limited by two straight lines, A1 and A2 (see Fig. 4.3).
Each of these lines passes through a given point (r_x, r_y) at a distance $S_{SM}^{\min}$ from a point (c_x, c_y). Thus, the equation of these lines can be obtained from the function L_A:

$$y = L_A(r_x, r_y, c_x, c_y, s, x) = \frac{(r_x - c_x)(r_y - c_y)}{(r_x - c_x)^2 - SM_{\min}^2} + \frac{sS_{SM}^{\min}\sqrt{(r_x - c_x)^2 + (r_y - c_y)^2 - SM_{\min}^2}}{(r_x - c_x)^2 - SM_{\min}^2}(x - r_x) + r_y \tag{4.1}$$

where x is the independent variable and the parameter s can be either $+1$ or -1, representing the two different solutions accomplishing the above definition. In order to determine this parameter to define lines A1 and A2, a clockwise relationship and a counterclockwise relationship among legs are defined:

Definition 4.1. *The relationship $L1 \Rightarrow L2$ between two adjacent legs is clockwise if it matches one of the following possibilities: $FLL \Rightarrow FRL$, $FRL \Rightarrow RRL$, $RRL \Rightarrow RLL$, $RLL \Rightarrow FLL$. On the other hand, the relationship is counterclockwise in the following cases: $FLL \Rightarrow RLL$, $RLL \Rightarrow RRL$, $RRL \Rightarrow FRL$, $FRL \Rightarrow FLL$.*

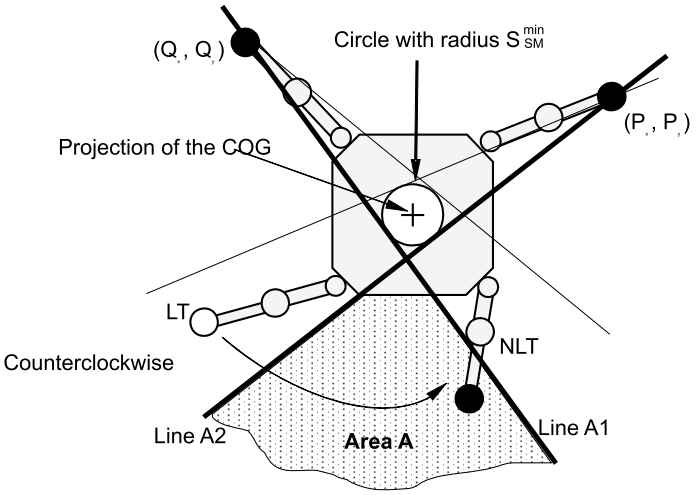


Fig. 4.3. Definition of Lines A1 and A2 and Area A (top view)

With these premises, lines limiting Area A can be defined as follows:
Line A1: This line passes through the support point (P_x, P_y) of the leg non-adjacent to LT and also passes at a distance $S_{SM}^{\min}$ from the projection of the COG. The equation of this line is then given by

$$\begin{aligned} y &= L_A(P_x, P_y, 0, 0, s, x) \\ s &= \begin{cases} 1 \text{ if } & LT \Rightarrow NLT \quad \text{is counterclockwise} \\ -1 \text{ if } & LT \Rightarrow NLT \quad \text{is clockwise.} \end{cases} \end{aligned} \tag{4.2}$$

Line A2: This line passes through the support point (Q_x, Q_y) of the leg non-adjacent to NLT, at a distance $S_{SM}^{\min}$ from the projection of the COG. Its equation is then given by

$$\begin{aligned} y &= L_A(Q_x, Q_y, 0, 0, s, x) \\ s &= \begin{cases} 1 \text{ if } & LT \Rightarrow NLT \quad \text{is clockwise} \\ -1 \text{ if } & LT \Rightarrow NLT \quad \text{is counterclockwise.} \end{cases} \end{aligned} \tag{4.3}$$

Lines A1 and A2 divide the plane into two semiplanes respectively, the Area A being the intersection of the semiplanes that do not contain the COG. Given two lines A1 and A2, in the form

$$\begin{aligned} y &= m_1 x + b_1 \\ y &= m_2 x + b_2. \end{aligned} \tag{4.4}$$

the Area A can be obtained from two of the following expressions satisfying the condition for b_i:

$$\begin{array}{ll} y < m_1 x + b_1 & if\ b_1 < 0 \\ y > m_1 x + b_1 & if\ b_1 > 0 \\ y < m_2 x + b_2 & if\ b_2 < 0 \\ y > m_2 x + b_2 & if\ b_2 > 0. \end{array} \tag{4.5}$$

Area B: This area is defined as the portion of the plane in which the placement of LT will permit the lifting of an adjacent leg NLT after a body motion of length M in the direction defined by the crab angle. Thus, the placement of transferring legs in this kind of area will favor body displacements between leg transfers. This area is limited by two Lines, B1 and B2, similar to Lines A1 and A2, but passing at a distance $S_{SM}^{\min}$ from the point where the COG will be located after a body motion of length M (see Fig. 4.4). This point, called COG_B, is given by

$$\begin{array}{l} COG_{Bx} = M \cos \alpha \\ COG_{By} = M \sin \alpha. \end{array} \tag{4.6}$$

With these premises, Lines B1 and B2 are given by the following equations:
Line B1:

$$\begin{array}{l} y = L_A(P_x, P_y, COG_{Bx}, COG_{By}, s, x) \\ s = \left\{ \begin{array}{ll} 1 \text{ if } & LT \Rightarrow NLT \quad \text{is counterclockwise} \\ -1 \text{ if } & LT \Rightarrow NLT \quad \text{is clockwise.} \end{array} \right. \end{array} \tag{4.7}$$

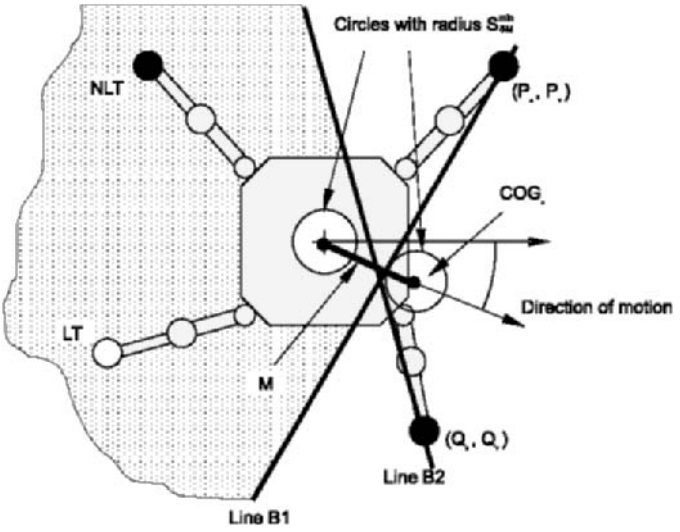


Fig. 4.4. Definition of Lines B1 and B2 and Area B (top view)

Line B2:

$$y = L_A(Q_x, Q_y, COG_{Bx}, COG_{By}, s, x)$$
$$s = \begin{cases} 1 \text{ if } & LT \Rightarrow NLT \quad \text{is clockwise} \\ -1 \text{ if } & LT \Rightarrow NLT \quad \text{is counterclockwise} \end{cases} \tag{4.8}$$

where (P_x, P_y) and (Q_x, Q_y) are the footholds of the legs non-adjacent to LT and NLT respectively. Given two Lines B1 and B2 as in (4.4), Area B is determined by the two expressions of (4.9) satisfying the condition for b_i:

$$\begin{array}{ll} y < m_1 x + b_1 & if\ b_1 < COG_{By} - m_1 COG_{Bx} \\ y > m_1 x + b_1 & if\ b_1 > COG_{By} - m_1 COG_{Bx} \\ y < m_2 x + b_2 & if\ b_2 < COG_{By} - m_2 COG_{Bx} \\ y > m_2 x + b_2 & if\ b_2 > COG_{By} - m_2 COG_{Bx}. \end{array} \tag{4.9}$$

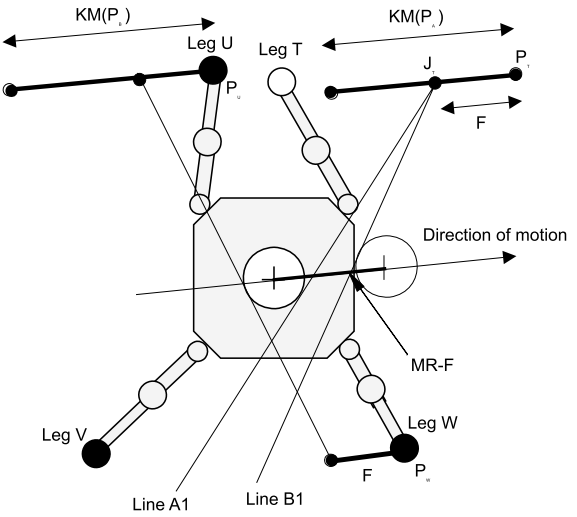


Fig. 4.5. Calculation of the points PA that define Area C

Area C: This area is defined as the region in which the placement of a front foot will originate, after a body motion, an ESZ for the non-adjacent leg accomplishing conditions N1 and N2 (see Sect. 4.2.3). Thus, this area constrains the ESZ of LT to get a foothold that facilitates the future searching of footholds for other legs. In order to find Area C, we will suppose a foothold $P_T = (P_{Tx}, P_{Ty})$ for the front leg in transfer T (see Fig. 4.5) and calculate the following parameters:

- F: Minimum displacement of the body required to permit the lifting of the rear leg V under static stability. This distance is only a function of the footholds of legs U and W (P_U and P_W respectively, see Fig. 4.5) and is given by

$$F = \sqrt{\left(D_c \cos\alpha - \frac{P_{Uy} - P_{Ux}m}{\tan\alpha - m}\right)^2 + \left(D_c \sin\alpha - \tan\alpha \frac{P_{Uy} - P_{Ux}m}{\tan\alpha - m}\right)^2} \tag{4.10}$$

where

$$D_c = \sqrt{((\tan\alpha)^2 + 1)\left(\frac{S_{SM}^{\min}\sqrt{m^2+1}}{\tan\alpha - m}\right)^2} \tag{4.11}$$

and

$$m = \frac{P_{Uy} - P_{Wy}}{P_{Ux} - P_{Wx}} \tag{4.12}$$

- MR: The minimum kinematic margin of legs T (placed in P_T) and U.
- J_T: The point where the front leg T will be placed when leg V can be lifted

$$\begin{aligned} J_{Tx} &= P_{Tx} - F\cos\alpha \\ J_{Ty} &= P_{Ty} - F\sin\alpha. \end{aligned} \tag{4.13}$$

Now, assuming that leg T is placed at J_T, we can calculate Line A1 for $LT = V$ and $NLT = W$, and Line B1 for $LT = V$, $NLT = U$ and $M = MR - F$. The limit of Area C is formed by the points P_T for which the calculated Lines A1 and B1 are superposed. Both lines pass through the point J_T, and thus their slopes must be equal in order to satisfy this condition. Those points are the solutions of the following equation

$$\frac{(x - F_x)(y - F_y) \pm S_{SM}^{\min}\sqrt{(x - F_x)^2 + (y - F_y)^2 - SM_{\min}^2}}{(x - F_x)^2 - SM_{\min}^2} = \frac{(x - MR_x)(y - MR_y)}{(x - MR_x)^2 - SM_{\min}^2} \pm \frac{S_{SM}^{\min}\sqrt{(x - MR_x)^2 + (y - MR_y)^2 - SM_{\min}^2}}{(x - MR_x)^2 - SM_{\min}^2} \tag{4.14}$$

where

$$\begin{aligned} MR_x &= MR\cos\alpha \\ MR_y &= MR\sin\alpha. \end{aligned} \tag{4.15}$$

The solution of this equation for a rectangular workspace, with $\alpha_c = 0$, $KM(P_B) = 0.3\,\text{m}$, $S_{SM}^{\min} = 0.08\,\text{m}$, and $F = 0.1\,\text{m}$ is presented in Fig. 4.6. The curve S1 is obtained from (4.14), considering that MR is equal to the KM of leg U, while the curve S2 is obtained considering that MR is equal to the KM of foothold P_T. Area C, which is limited by these two solutions, is also depicted in Fig. 4.6.

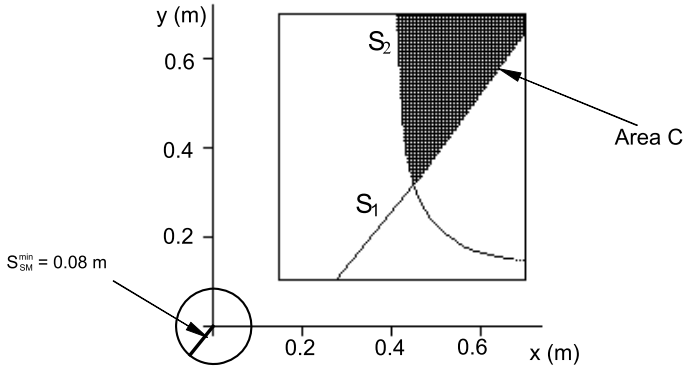


Fig. 4.6. Area C for a rectangular workspace

Once these areas have been described, the foothold conditions (see Sects. 4.2.3 and 4.2.3) imposed by the sequence planner can be translated in terms of restricting areas. The use of an Area A to restrict the footholds of a transferring leg LT will enable the immediate lifting of another leg (Conditions N1, K1 and K4). Similarly, the use of an Area B can guarantee that another leg can be lifted after a body motion of length $KM_{\min}$ (Conditions N2, K2, K3, K5). These areas are related with Diagonal Principles (DP) II and I respectively, proposed by Hirose (1984), but there are certain major differences. First, Hirose defines DP I and DP II considering longitudinal stability margins, while the described areas have been designed considering absolute stability margins, more appropriate for a real walking machine. Second, Hirose uses DP I and DP II only for rear legs, while Areas A and B have been defined and are used for all combinations of adjacent legs. Finally, Hirose describes lines and semiplanes, while in this chapter we describe areas in which the placement of a transferring leg actually does guarantee the static stability of the machine when another leg is lifted. This is more general and useful for unusual leg distributions in which workspaces overlap.

The meaning of Area C differs from that of Areas A and B. As we have seen above, when criterion N is considered, the foothold of a rear leg must satisfy two conditions, which may be incompatible. These conditions are imposed by an Area A and an Area B that depend on a front foothold. Therefore, a front foothold can determine the compatibility or incompatibility of both conditions when searching for a foothold for its contralateral rear leg. Hirose (1984) limits the foothold of a front leg using a line that passes through the projection of the COG and lies parallel to the direction of motion (DP III). However, if absolute margins are used, the placement of a front leg in some locations of the area described by Hirose would yield invalid solutions for Area

B when searching for a foothold for the contralateral rear leg. Furthermore, DP III does not ensure that DP I and DP II applied to the rear leg will be compatible. Area C was defined to limit the front footholds in such a way that rear legs find appropriate footholds accomplishing both Conditions N1 and N2.

Foothold Search Algorithm

Foothold selection is accomplished using the following five constraints. The first four constraints will define the ESZ for the new foothold, while the fifth constraint will look for a unique solution (see Fig. 4.7).

Constraint E1: *The foothold must be inside the leg workspace.*

Therefore, the horizontal projection of the workspace will define a boundary for the ESZ.

Constraint E2: *The foothold must accomplish the conditions imposed by the sequence planner to allow the lifting of other legs in subsequent locomotion cycles.*

These conditions were enumerated in Sect. 4.2.3 and are applied in the form of the foothold restricting areas described above in this section. If more than one condition is applied, the ESZ will be limited by the intersection of the corresponding restricting areas.

Constraint E3: *The foothold cannot be laid in a forbidden cell.*

Therefore, the robot's controller should be able to access a database of permitted/forbidden cells provided by an additional sensor system, as explained in Sect. 4.2.2.

Constraint E4: *The foothold must avoid collision among legs.*

The full leg should be tested to avoid collision, but for the sake of simplicity the final implementation in the walking robot considers only collisions among feet and knees of the LT and adjacent legs.

Constraint E5: *The foothold must provide the maximum KM.*

To find a foothold with a maximum KM, it is necessary to explore all the ESZ defined by Constraints E1, E2, E3 and E4. To accomplish the exploration, the horizontal projection of the workspace is divided into a discrete-point matrix, and points meeting Constraints E2, E3 and E4 are selected. The foothold will be given by Constraint E5 as the selected point with the greatest KM. The complete procedure employed for transferring a leg to a new foothold, which includes the algorithms of the sequence planner and the foothold planner, are summarized in Algorithms 4.1 to 4.3.

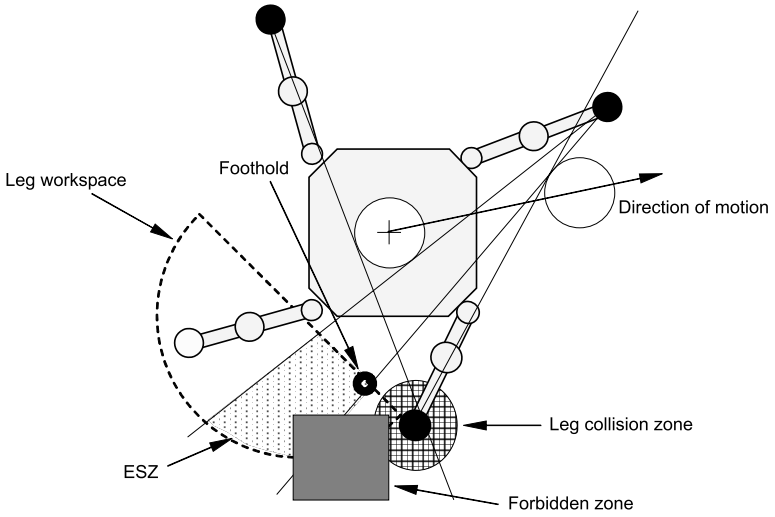


Fig. 4.7. Foothold search constraints

Algorithm 4.1. Foothold search

```
IF There is a leg lifted, LT, THEN
  SET NLT = leg that goes after LT in the natural sequence
  SET J=1
  Find a foothold for LT using CRITERION N
  WHILE A valid foothold has not been found AND J < 4
     SET NLT = LKM_J
     Find a foothold for LT using CRITERION K
     J=J+1
  ENDWHILE
  IF A valid foothold has been found THEN
     Place LT in the foothold
     Mark NLT as the new LT
  ELSE
     PROCEDURE FAILS
  ENDIF
ENDIF
EXIT
```

Algorithm 4.2. CRITERION N

```
IF LT is a front leg THEN
   IF the leg non-adjacent to LT can be lifted with
       stability > S_SM^min after a body motion of length < KM_min
       THEN SET constraint E2 = Area C, KM foothold > KM_min
   ENDIF
ELSEIF LT is a rear leg THEN
   SET NLT2 = rear leg contralateral to LT
   SET constraint E2 = Area A ∩ Area B (considering NLT2),
   KM foothold > KM_min
ENDIF
Find a foothold accomplishing Constraints E1-E5
EXIT
```

Algorithm 4.3. CRITERION K

```
IF LT is a front leg THEN
   IF NLT is a front leg THEN
       SET NLT2 = leg collateral to LT
       SET constraint E2 = Area A ∩ Area B (considering NLT2),
       KM foothold > KM of NLT
       Find a foothold accomplishing constraints E1-E5
       IF A valid foothold has not been found THEN
           SET constraint E2 = Area A, KM foothold > KM
           of NLT
       ENDIF
   ELSEIF NLT is collateral to LT THEN
       SET constraint E2 = Area B, KM foothold > KM of NLT
   ELSEIF NLT is non-adjacent to LT AND NLT will be able to go to
       transfer phase with stability > S_SM^min after a body motion of
       length < KM_min THEN
       SET constraint E2 = KM foothold > KM of NLT
   ELSE
       EXIT
   ENDIF
ELSEIF LT is a rear leg THEN
   IF NLT is a rear leg THEN
       SET constraint E2 = Area B, KM foothold > KM of NLT
   IF NLT is collateral to LT THEN
       SET constraint E2 = Area A, KM of foothold > KM of NLT
   ELSEIF NLT is non-adjacent to LT THEN
       EXIT
   ENDIF
ENDIF
Find a foothold accomplishing Constraints E1-E5
EXIT
```

4.2.5 Body Motion Planner

After studying the leg sequence and the foothold search, it is necessary to define the body motion and the leg lifting methods to characterize fully the gait. Body motion is performed in an iterative way. The body is moved a small distance at each iteration of the algorithm if the following conditions are true:

1. All legs are in their support phase.
2. All legs are able to move the body without reaching their kinematic limits.

Thus, Algorithm 4.4 shows the pseudo code of the body motion planner.

Algorithm 4.4. Pseudo code of the body motion planner

```
IF All legs are in support phase AND KM_min > d THEN
        Move the body a short distance d
ENDIF
```

Body motion appears naturally when there is a leg marked as LT by the sequence planner, but the stability condition still does not allow the transference of this leg, as explained in the next section.

4.2.6 Leg-lifting Planner

The leg-lifting planner is the module in charge of regulating which leg will be lifted when all legs are in their support phase. The algorithm is organized in the following steps:

1. Consider the following ordered list of legs: the leg marked for lifting by the leg sequence planner (if it exists), LKM_1, LKM_2, LKM_3 and LKM_4.
2. Find the first leg on the list, LL, which will be able to go to transfer phase with a stability margin greater than $S_{SM}^{\min}$ before $KM_{\min}$ vanishes. If none of the legs meet this condition, the gait is deadlocked.
3. If leg LL can be lifted currently with stability greater than $S_{SM}^{\min}$, then lift leg LL.

Thus, the leg marked for lifting by the leg sequence planner is the first leg considered for lifting. However, this leg might lose its ability to go to the transfer phase with stability due to a change in the crab angle, for instance; also, in the initial situation there is no leg marked for lifting. In these cases the other legs are considered for lifting, in an order given by their ascending kinematic margins. The pseudo code of the leg lifting planner is listed in Algorithm 4.5.

Algorithm 4.5. Pseudo code of the lifting planner

```
IF All legs are in support state THEN
   IF The sequence planner has marked a leg as LT AND LT will be
       able to go to transfer phase with stability > S_SM^min after a
       body motion of length < KM_min
   THEN SET LL = LT
   ELSE
       SET N = 1
       WHILE N <= 4
           IF LKM_N will be able to go to transfer phase with
               stability > S_SM^min after a body motion of length < KM_min
               THEN SET LL = LKM_N
               BREAK WHILE
           ELSE
               N=N+1
           ENDIF
       ENDWHILE
   ENDIF
   IF leg LL can be lifted currently with stability > S_SM^min
       THEN Lift leg LL
   ENDIF
ENDIF
EXIT
```

4.3 Free Turning Gaits

In this section we propose a turning gait derived from the free crab gait described above. In this turning gait the body reference frame follows a circular trajectory around a given point. A straight trajectory can be considered a turning gait of infinite radius. So, the general methods used to plan the gait should be valid in a turning gait of a long enough radius. However, it is necessary to redefine certain concepts and define new ones.

The parameters defining the trajectory (which should be given by the operator or a superior hierarchical level) are:

- **Turning center** (TC): This is the center of the circular trajectory of the body and should be defined by its components in the body reference frame.
- **Turning direction** (TD): Obviously, this is the direction of turn and can be clockwise or counterclockwise.

With these two parameters, the algorithm computes the following parameters:

- **Turning radius** (TR): This is the distance from the turning center (TC) to the origin of the body reference frame.

- **Turning-center angle (θ_{TC})**: This is the angle formed by the segment passing through the COG and the TC, and the x-axis of the body reference frame.
- **Turning-crab angle (α_T)**: This is the angle formed by the line tangent to the trajectory at the origin of the body reference frame and the x-axis. This angle, which is kept constant along the trajectory, becomes

$$\alpha_T = \begin{cases} \theta_{TC} + \frac{\pi}{2} & \text{if turning direction is clockwise} \\ \theta_{TC} - \frac{\pi}{2} & \text{if turning direction is counterclockwise.} \end{cases} \tag{4.16}$$

Also, the kinematic margin must be redefined:

- **Angular kinematic margin of a leg (μ_T)**: The angle that the foothold of a leg sweeps around TC in the opposite direction to TD before reaching the boundary of the leg workspace (see Fig. 4.8).

Some other concepts ($KM_{\min}$, LKM_i, *etc.*) should be redefined accordingly with the definition of μ_T. Additionally, some of the methods described for the free crab gait must be re-formulated to generate a turning gait. These modifications are described in the next two sections.

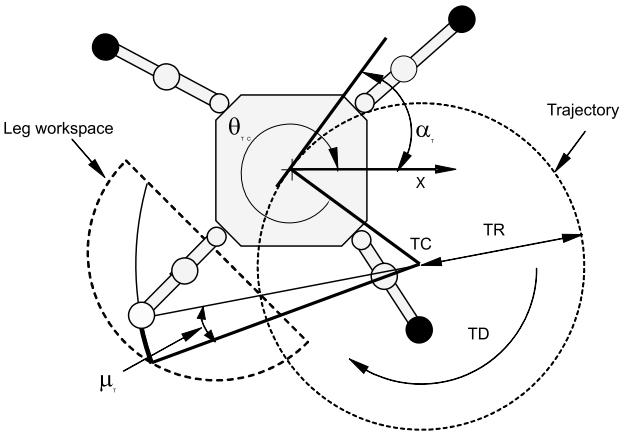


Fig. 4.8. Angular kinematic margin

4.3.1 Leg Sequence, Body Motion and Leg Lifting

All the methods employed to plan the leg sequence, the body motion and the leg lifting for the free-crab gait (Sects. 4.2.3, 4.2.5 and 4.2.6) are valid for the free-turning gait if KM is replaced by μ_T. The type of gait and the duty of each leg, *i.e.* right, left, front and rear, is determined as a function

of α_T instead of α_C. If the conditions of Sect. 4.2.5 are satisfied, then the body is rotated a fixed short distance around the TC at each iteration of the algorithm. To accomplish this motion, the legs must rotate in the opposite direction.

4.3.2 Foothold Planning

The methods employed to plan the new footholds for the free crab gait must be adapted partially for the turning gait. The foothold restricting Areas B and C described in Sect. 4.2.4 will be redefined as Area BT and Area CT so that they can be used for turning gaits.

Area BT: This area is equivalent to Area B, and it is calculated assuming that the COG is located at the point COG_{BT}, resulting from the rotation of the COG at an angle ν around the turning center in the direction of motion (see Fig. 4.9). This point is given by

$$\begin{aligned} COG_{BTx} &= TC_x - TR\cos(\theta_{TC} - \nu) \\ COG_{BTy} &= TC_y - TR\sin(\theta_{TC} - \nu). \end{aligned} \tag{4.17}$$

Lines BT1 and BT2 can be determined by inserting this new location of the COG in (4.7) and (4.8). Similarly, Area BT can be obtained from (4.9).

Area CT: The procedure used to determine Area CT is similar to the one described for Area C, with the following differences:

- φ : Minimum angular displacement of the body required to permit the stable lifting of the rear leg V (see Fig. 4.10). To accomplish this, the COG must be moved to the point COG_{CT}, defined by

$$\begin{aligned} COG_{CTx} &= \frac{(TC_x - mk)}{1+m^2} + \\ &\quad \frac{d_{t1}\sqrt{(TC_y - mk)^2 - (1+m^2)(TC_x + k^2 - TR^2)}}{1+m^2} \\ COG_{CTy} &= mCOG_{CTx} + k + TC_y \end{aligned} \tag{4.18}$$

where

$$k = P_{Uy} - mP_{Ux} + d_{t2}\mu_{T\min}\sqrt{m^2+1} - TC_y \tag{4.19}$$

$$\begin{cases} d_{t1} = d_{t2} & \text{if turning direction is clockwise} \\ d_{t1} = -d_{t2} & \text{if turning direction is counterclockwise} \\ |d_{t1}| = 1 \end{cases} \tag{4.20}$$

and m can be obtained from (4.12). The sign of parameter d_{t1} is chosen such that the solution meets one of the following conditions:

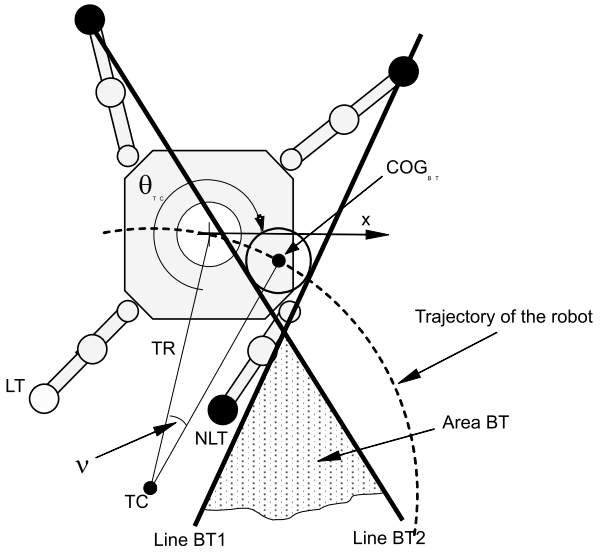


Fig. 4.9. Definition of Lines BT1 and BT2 and Area BT in the turning gait

$$\begin{cases} mCOG_{CTx} + b > COG_{CTy} & \text{if} \quad P_{Uy} - mP_{Ux} < 0 \\ mCOG_{CTx} + b < COG_{CTy} & \text{if} \quad P_{Uy} - mP_{Ux} > 0. \end{cases} \tag{4.21}$$

Once the point COG_{CT} has been determined, the angle φ is given by

$$\varphi = \arccos \frac{TR^2 - TC_x COG_{CTx} - TC_y COG_{CTy}}{TR^2}. \tag{4.22}$$

- ρ : Minimum μ_T of legs T (placed in P_T) and U.

The point J_T has now the following coordinates:

$$\begin{aligned} J_{Tx} &= (P_{Tx} - TC_x)\cos\varphi - (P_{Ty} - TC_y)\sin\varphi + TC_x \\ J_{Ty} &= (P_{Tx} - TC_x)\sin\varphi + (P_{Ty} - TC_y)\cos\varphi + TC_x \end{aligned} \tag{4.23}$$

Supposing that leg T is placed at J_T, we can calculate Line A1 as in Sect. 4.2.4 and Line BT1 for $LT = V$, $NLT = U$ and $\nu = \rho - \varphi$ (see Fig. 4.10). The limit of Area CT is formed by the points P_T for which the slopes of Lines A1 and BT1 are equal.

4.4 Free Spinning Gaits

The free spinning gait or zero-radius free turning gait rotates the body around the z-axis of the body reference frame, and therefore the turning centre is

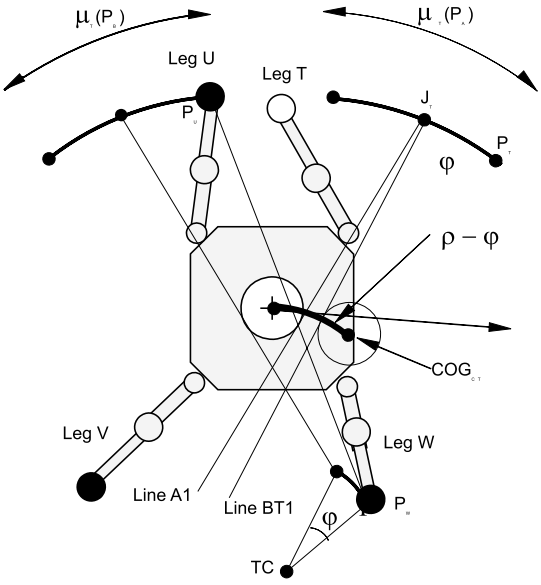


Fig. 4.10. Calculation of points PA that define Area CT

located at the *COG*, and the turning radius is zero. This gait provides an effective method to change the orientation of the robot, avoiding the deadlocked situations observed when the turning gait follows trajectories with small turning radius. As in previous gait algorithms, the gait is specified by the leg sequence, foothold selection and body motion.

4.4.1 Leg Sequence and Leg Lifting

To obtain large spinning angles with a minimum number of leg transfers, we chose a circular leg sequence in which the legs go to transfer phase in a clockwise (counterclockwise) sequence if the body turns clockwise (counterclockwise). Initially, the algorithm determines if it is possible to lift at least one of the legs with stability, that is, with a static stability margin greater than $S_{SM}^{\min}$. If it is not possible, the body is moved in the direction of the last crab angle until a leg can be lifted with stability. This operation is only necessary at the beginning of the execution of the gait algorithm, because, as we will see below, the footholds will always guarantee the stable lifting of the next leg. The transfer leg will be chosen imposing two constraints:

1. The leg can be lifted under static stability.
2. The contiguous leg in the direction of rotation cannot be lifted under static stability.

The latter condition means that the feet remain located in positions not excessively advanced in the direction of rotation. This will favor foothold searching, as explained in the next paragraph.

4.4.2 Foothold Planner and Body Motion Planner

The new foothold for a transferring leg LT will be limited by the sequencing conditions listed below.

Condition S1: *The foothold for LT must enable the stable lifting of the next leg in the direction of rotation.*

Thus, the ESZ will be limited by an Area A, being NLT leg after LT in the direction of rotation.

Condition S2: *The foothold for LT must be placed so that when searching for a foothold for the leg non-adjacent to LT (*i.e.*, two leg transferences later), the condition S1 will be compatible with the constraint imposed by the workspace.*

As pointed out above, Condition S1 implies that the foothold for LT must be within an Area A. This area depends on the position of the leg non-adjacent to LT. If the leg non-adjacent to LT is located in certain positions then the Area A and the constraint imposed by the leg workspace can be incompatible. Typically, this incompatibility will appear if leg non-adjacent to LT is located on excessively advanced positions in the direction of rotation. Condition S2 is designed to limit the footholds for a transferring leg in such a way that this incompatibility will not appear two transferences later. This condition will be fulfilled by the use of a foothold restricting Area DS, described below.

Area DS: This is the semiplane not containing the COG defined by a Line DS given by

$$\begin{aligned} y &= L_A(G_x, G_y, 0, 0, s, x) \\ s &= \begin{cases} 1 & \text{if turning direction is clockwise} \\ -1 & \text{if turning direction is counterclockwise.} \end{cases} \end{aligned} \tag{4.24}$$

where (G_x, G_y) is a random point chosen in such a way that the size of the intersection of Area DS and the workspace of the leg non-adjacent to LT is equal to a predefined minimum searching area (see Fig. 4.11). This will guarantee that, when searching for a foothold for the leg non-adjacent to LT two transferences later, the application of Condition S1 and the workspace constraint will generate a minimum effective searching area, although no body rotation is produced between those transfers.

Areas A and DS, the leg workspace, the forbidden areas and the leg collision areas define an ESZ quite similar to the ESZ for the crab gait. The procedure for selecting the foothold is the same as for crab gaits, except that it searches for a maximum μ_T instead of a maximum KM.

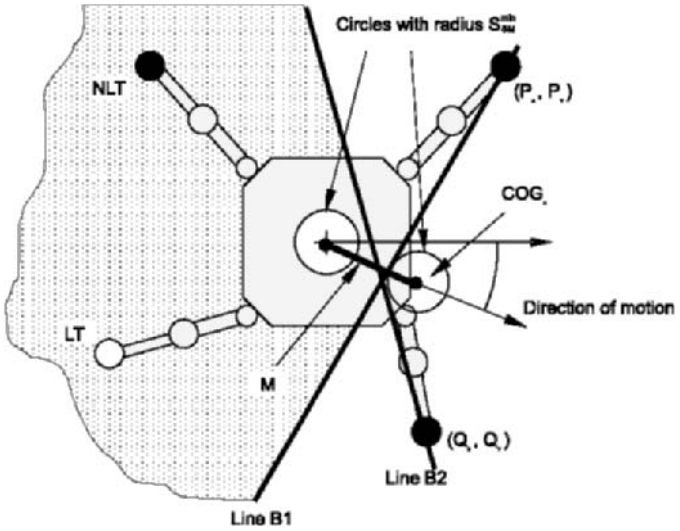


Fig. 4.11. Area DS and Effective Searching Zone (ESZ) for the spinning gait

If under these conditions it is not possible to find a foothold (see Fig. 4.12(a)), then two methods are employed to facilitate foothold search and avoid deadlock:

1. If a valid foothold is not found, then the transfer leg is lifted and the body is rotated on three legs until the leg with the minimum μ_T reaches its kinematic limit (see Fig. 4.12(b)), and then a new search is performed (see Fig. 4.12(c)).
2. If such a body motion is not possible, then condition S2 is ignored, since its aim is to obtain good results for subsequent locomotion cycles, and a new search is performed (see Fig. 4.12(d)).

After each leg transfer, the body is rotated as much as possible, *i.e.* until the leg with the minimum $\mu_{T\min}$ reaches its kinematic limit. In doing that, the legs will be located in points less advanced in the direction of rotation, a position which facilitates the search for new footholds and reduces the possibility of deadlock. The spinning gait's complete algorithm is presented in Algorithm 4.6.

4.5 Experimental Results

The algorithms presented in this chapter have been intensively simulated and tested on the SILO4 walking robot; however, only two experiments are reported here, each consisting in following a predefined path under different

conditions. Although this path has been pre-programmed the experiments are equivalent to the case in which a human operator steers the robot while it is walking. The predefined path consists of four straight-line segments defined in Table 4.1 by their initial points and crab angles with the x-axis of the world reference frame (see Fig. 4.13). The robot's initial foot positions in the body reference frame are

$$\begin{aligned}(x_{FLL}, y_{FLL}) &= (0.3\ m, 0.3\ m)\\ (x_{FRL}, y_{FRL}) &= (0.3\ m, -0.3\ m)\\ (x_{RLL}, y_{RLL}) &= (-0.3\ m, 0.3\ m)\\ (x_{RRL}, y_{RRL}) &= (-0.3\ m, -0.3\ m)\end{aligned}$$

and the body reference frame is parallel to the world reference frame. The $S_{SM}^{\min}$ required in the experiments is 0.04 m.

The robot's position is obtained by odometry, *i.e.* the controller only takes into account leg and body motions to compute the position in the world reference frame. However, an external measurement system based on a digital camera is used for recording foot and body positions and reporting results.

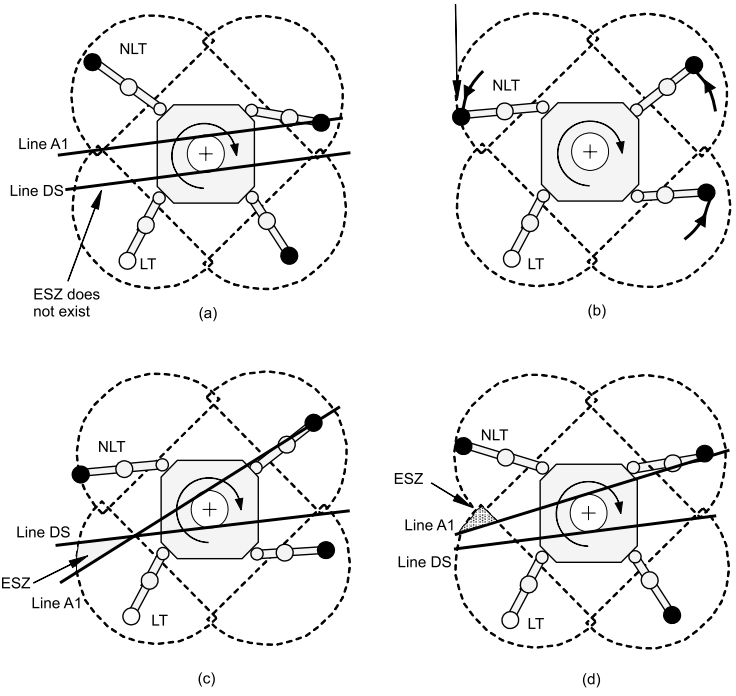


Fig. 4.12. Searching for footholds by rotating the body on three legs

Algorithm 4.6. Pseudo code of the spinning gait

```
LABEL A:
    Rotate the body an angle equal to μ_Tmin
    SET LT= a leg that can be lifted with stability and the leg after
        it in the direction of rotation cannot
    Find a foothold for LT satisfying conditions S1 and S2
    IF a valid foothold has not been found THEN
        IF μ_T > 0 THEN
            Lift leg LT
            GO TO LABEL A
        ENDIF
    Find a foothold for LT satisfying conditions S1
        IF a valid foothold has not been found THEN
            PROCEDURE FAILS
            EXIT
        ENDIF
    ENDIF
    Place LT at the foothold found
    EXIT
```

Table 4.1. Trajectory features

Stretch	Initial foot positions (m)	Crab angle (degrees)	Theoretical length (m)	Real length (m)	Average speed (m/s)
1	(0.400, 0.600)	0	1.500	1.551	0.0097
2	(1.900, 0.600)	23	0.760	0.774	0.0053
3	(2.559, 0.897)	90	0.900	0.948	0.0059
4	(2.559, 1.797)	0.205	0.800	0.777	0.0049

The robot is on flat leveled terrain for these specific experiments. It is worth noting at this point that adaptability to irregular terrain is an intrinsic property of discontinuous gaits.

In the first experiment, the robot follows the trajectory using the free-crab gait; therefore the x-axis of the body reference frame is always in the same direction. The robot moves along a full trajectory of 3.96 m that takes 7.5 min. The external measurement system takes a picture every 15 s. After processing, that picture enables the trajectory to be drawn as shown in Fig. 4.13. The dashed line shows the original trajectory to be followed by the robot. The real trajectory departs from the theoretical one by about 0.10 m at the end of the trajectory (which is about 4 m long). This error occurs because the odometry-based estimation of the position of the robot only considers theoretical displacements. The phenomena associated with body and leg flexure and foot slippage are not taken into account. Additional sensors are required

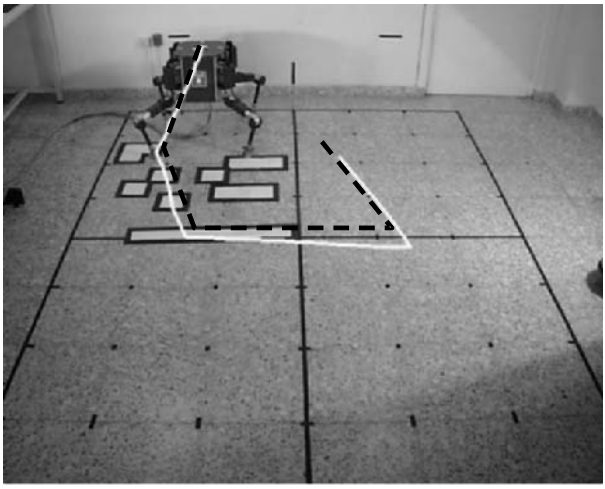


Fig. 4.13. SILO4 robot following a predefined path

to solve this problem, which is beyond the scope of this chapter. Table 4.1 summarizes real body displacement and speed during the experiment.

The second experiment consists in following the same path with additional forbidden areas depicted in Fig. 4.14. The robot controller knows the size and position of all these areas, which have been considered for gait generation. The robot cannot step on the shaded area. Therefore, the effective forbidden area for the feet should be enlarged by the diameter of the foot sole in every component. This effective area is indicated with a thin line in Fig. 4.14.

To show how the algorithm works, Fig. 4.14 shows the foot support points for three different situations. Figure 4.14(a) presents the foot support points for the first stretch of the first experiment, *i.e.* the robot follows a straight line with no forbidden areas. Figure 4.14(b) presents the foot support points obtained in simulation when performing the same trajectory with forbidden areas. In this case it is easy to observe that there is no support at all over the forbidden areas. The last case (Fig. 4.14(c)) illustrates the true support points when the robot performs the trajectory over the defined forbidden areas. The foot support points stray slightly from the points obtained in simulations (Fig. 4.14(b)). This is because of the problems involved in the odometry-based estimate of the position of the robot.

4.6 Conclusions

This chapter has presented the derivation of three new gaits for real walking machines capable of negotiating irregular terrain. The crab gait, the circular gait and the spinning gait can be joined together to follow complex paths

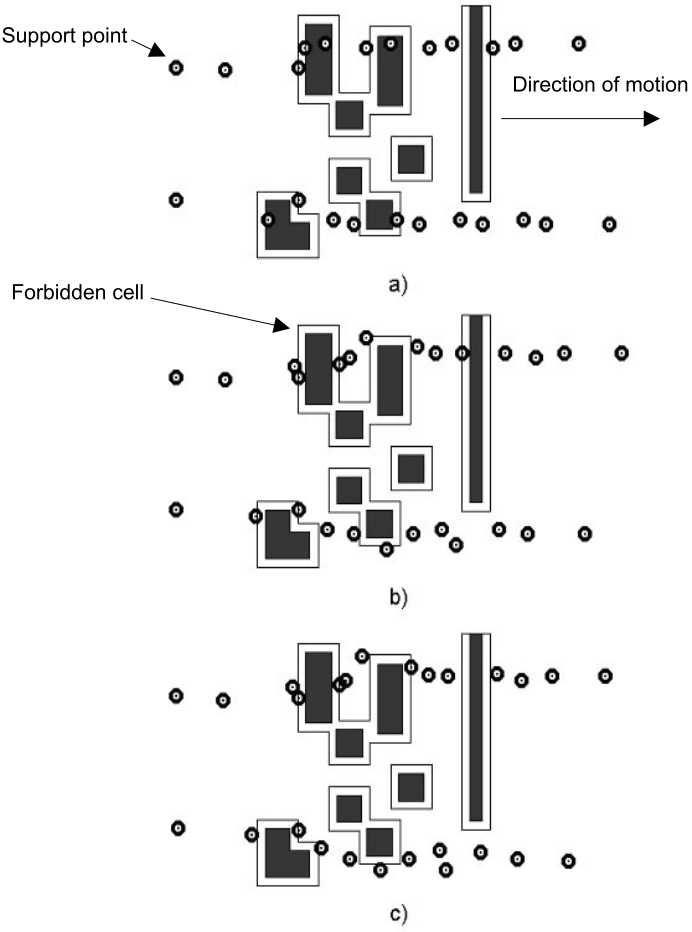


Fig. 4.14. Foot support points over forbidden areas for different situations

efficiently. These new gaits are based on free and discontinuous gaits. Free gaits are adequate for following trajectories, and discontinuous gaits exhibit good intrinsic features for terrain adaptability.

The free gait algorithms are based on a heuristic rules focused on maintaining the static stability of the machine and avoiding leg deadlocking while maintaining a convenient leg sequence. New foothold restricting areas, which are based on Hirose's work and consider absolute stability margins, are used to plan footholds in such a way that some leg sequencing criteria are accomplished. In particular, the areas employed to restrict the footholds of fore legs in crab and turning gaits represent a new and effective method.

The new gait's theoretical features are superior to those of previous gaits, due to the use of absolute stability margins, which allow a real machine to walk in a stable way regardless of the crab angle. Finally, the use of a discontinuous gait furnishes a straightforward solution to the problem of adapting to irregular terrain.

Some experimental results are reported to validate the theoretical and practical features of the proposed gaits. The experiments were conducted on the SILO4 walking robot, and the algorithms have been proved efficient in real time.

5

New Approaches to Stability

5.1 Introduction

In previous chapters, we have studied different static and dynamic stability margins to infer how far a legged robot is from instability. Those margins were based on either geometric or energetic measurements and definitions assumed implicitly ideal actuators and ideal power supplies *i.e.* actuators capable of supplying any requested torque and batteries capable of providing any requested current. However, real motors are torque limited and real power supplies can only provided a limited maximum current. Therefore, measuring stability by using only the geometric parameters of the robot, neglecting the influence of motor-torque and power-consumption limitations of real systems, is a hard inconvenience for robots working in real applications.

This chapter reviews static stability theory for walking robots, illustrates real problems through simulation and experiments using real walking machines, and proposes a new concept of static stability that takes into consideration some of the robot's intrinsic parameters. The resulting stability measurement can improve efficiency in terms of robot design and power consumption, two aspects that are of paramount importance in autonomous walking robots for real applications.

The need for including actuator features and power-supply limits arises because legs have normally been designed as 3-DOF manipulators. A robot manipulator defines a given workspace, and motor power is selected for features such as maximum payload and speed. Motor weight for manipulators is not a major problem; motors are easily balanced within the structure, and the total weight is supported by the ground directly or through a stiff structure (Gonzalez de Santos *et al.*, 2005).

Walking robots present quite a different problem. Legs need:

- To have a workspace that ensures them a foothold (taking forbidden zones into consideration).

- To support both body and payload.
- To cross over obstacles.

That means a leg could need long links, not for the sake of enjoying a wide workspace, but in order to cross over obstacles. As a consequence, legs do not need to ensure large torques over their whole workspace, but rather requested torques within a reduced volume of their whole workspace. This phenomenon is also observed in nature, where legged animals do not use their entire leg workspace for normal walking, and the ordinary leg workspace is smaller than the achievable workspace. A human being, for instance, possesses legs with a big workspace but uses only a reduced workspace during walking, because he or she cannot withstand the muscle effort required for some extreme positions. This fact has encouraged researchers to develop mechanisms and algorithms to reduce power in ankle joints, for instance Yi and Zheng (1997).

Legged robots suffer the same problem. Their actuators are selected to exert a given force in a normal leg configuration. Yet, when a foothold lies outside the normal configuration, the requested joint torques might be greater than the maximum torques exerted by actuators, and so the leg might not support the body weight properly, and the robot might fall down. This problem does not normally appear in walking robots performing periodic gaits on even terrain (see Chap. 3). Periodic gaits use foot trajectories that lie within a favorable reduced area of their workspace. Free gaits, however, use the entire width and breadth of the leg workspace, and free-gait foot trajectories may run in any direction in order to achieve better speed and omni-directionality (see Chap. 4). Reducing the workspace is not an acceptable solution, because it reduces the robot's average speed (which depends on the leg stroke), and speed must be kept as high as possible, because walking robots are intrinsically slow machines. Moreover, this is far from an optimum solution, because the force exerted by a foot (the force is in fact produced by the joint torques) depends not only on the position of that foot, but also on the position of all other feet as well. The optimum solution seems to be to consider the real joint torques of all the robot's joints, and to avoid those leg configurations in which one joint torque is higher that the maximum allowed torque. There is one more constraint: even in the case of a robot configuration with all joint torques below maximum, the power consumption could be higher than the maximum allowed by either the power supply or the onboard electronic devices. Note that power is a very limited resource in autonomous robots, and power-consumption optimization is imperative.

As we have seen in Chap. 2, for statically stable walking machines, stability is defined by a geometric approach: a walking machine is stable if the projection of its center of gravity lies inside the machine's support polygon. This definition assumes that the actuators can provide any requested torque. Real actuators, power supplies and electronic devices, however, are another matter, and the concept of stability ought to be redefined accordingly.

This chapter illustrates this realistic problem and proposes a new concept of static stability to improve the efficiency of real walking robots. The chapter is organized as follows: First, Sect. 5.2 presents the problem approach. Then, Sects. 5.3 and 5.4 study some effects of using the geometric stability criterion both in simulation and in a real walking robot, respectively. Sect. 5.5 proposes the global stability criterion. Finally, Sect. 5.6 summarizes some conclusions.

5.2 Geometric Stability and Required Torques

The leg forces exerted by a robot depend on the number of legs in support and their footholds as well. Quadrupeds walking under static stability must alternate groups of three legs in support ($\beta = 3/4$) or sequences of three and four legs in support ($\beta > 3/4$) (see Chap. 3).

The equilibrium equations that balance forces and moments in the robot's body are given by (Klein and Chung, 1987)

$$\mathbf{A}_{3\times4}\,\mathbf{F} = \mathbf{W} \tag{5.1}$$

where

$$\mathbf{A}_{3\times4} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ 1 & 1 & 1 & 1 \end{pmatrix} \tag{5.2}$$

$$\mathbf{F} = (F_1, F_2, F_3, F_4)^T \tag{5.3}$$

and

$$\mathbf{W} = (0, 0, -W)^T. \tag{5.4}$$

F_i is the vertical ground-reaction force in foot i in support ($-F_i$ is the force that the foot i must exert against the ground), (x_i, y_i) are the position components of foot i in the robot's reference frame (x, y, z), which is located at the body centre of gravity, and W is the robot's weight.

Equations (5.1) is an underdetermined system because it consists of three equation and four unknowns (F_i) and its solutions can be found through the psedoinverse of $\mathbf{A}_{3\times4}$, $\mathbf{A}^+_{3\times4}$, given by

$$\mathbf{A}^+_{3\times4} = \mathbf{A}^T_{3\times4}(\mathbf{A}_{3\times4}\mathbf{A}^T_{3\times4})^{-1} \tag{5.5}$$

where $\mathbf{A}^T$ represents the transposed matrix of $\mathbf{A}$. Therefore, if all feet are in support the foot forces are given by

$$\mathbf{F} = \mathbf{A}^+_{3\times4}\mathbf{W} = \mathbf{A}^T_{3\times4}(\mathbf{A}_{3\times4}\mathbf{A}^T_{3\times4})^{-1}\mathbf{W}. \tag{5.6}$$

If the robot has three legs in support, equation (5.1) can be written as

$$\mathbf{A_{3\times3}}\,\mathbf{F} = \mathbf{W} \tag{5.7}$$

where

$$\mathbf{A_{3\times 3}} = \begin{pmatrix} x_q & x_r & x_s \\ y_q & y_r & y_s \\ 1 & 1 & 1 \end{pmatrix} \tag{5.8}$$

$$\mathbf{F} = (F_q,\ F_r,\ F_s)^T \tag{5.9}$$

and

$$\mathbf{W} = (0,\ 0,\ -W)^T. \tag{5.10}$$

The inverse matrix of $\mathbf{A}$ exists if and only if $\det(\mathrm{A}) \neq 0$ and then the system has a unique solution for foot forces given by

$$\begin{pmatrix} F_q \\ F_r \\ F_s \end{pmatrix} = \begin{pmatrix} x_q & x_r & x_s \\ y_q & y_r & y_s \\ 1 & 1 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ 0 \\ -W \end{pmatrix} \tag{5.11}$$

$$F_t = 0$$

where q, r, and s represent the legs in support and t is the leg in transfer.

Thus for a quadruped discontinuous gait, foot forces for body motions are computed by (5.6) and foot forces when a leg is in transfer are computed by (5.11). Figure 5.1 plots the forces exerted by each leg when the SILO4 robot performs a discontinuous gait with $\beta = 7/9$, which gait diagram is shown in Fig. 3.7(a). A null leg force means the leg is in transfer phase and the linear transition of forces (immediately before $t = 0.5$ and $t = 1$) occurs during the body motion.

Support forces must be exerted in fact by leg joint torques that depend on the leg configuration. That means that a leg can exert a given force with different joint torques at different leg configurations. Joint torque vector as a function of the ground reaction forces is given by (Spong and Vidyasagar, 1989)

$$\boldsymbol{\tau}_i = \mathbf{J}^T \mathbf{F}_i \tag{5.12}$$

where $\boldsymbol{\tau}_i = (\tau_{i1}\ \tau_{i2}\ \tau_{i3})^T$ is the joint torque vector for leg i, $\mathbf{F}_i$ is the ground reaction force vector for foot i and $\mathbf{J}$ is the Jacobian matrix of the leg type (see Appendix A). Torques τ_{ij} must be provided by the actuator to maintain the robot, otherwise, it will fall down to the ground.

Figure 5.2 plots the motor torques exerted by all the joints along a locomotion cycle for the discontinuous gait of last example ($\beta = 7/9$). Notice that the time is normalized to the period of the locomotion cycle. In this case, the robot is levelled and the joint 1, indicated in dotted line, of each leg does not need to exert any torque. Torques in joints 2 and 3, solid line and dashed line respectively, support the body in different rates depending on the phase of the gait. Joint torques to exert a given foot force depend on the leg configuration, as advanced above; therefore, joint torques also depend strongly of the foothold positions. Leg workspaces are large enough and they are fully used

when required to enhance stability or avoid forbidden zones. In such a case, torques can increase as shown in Fig. 5.3. This figure plots the torque of joint 2 for front legs (legs 1 and 2) in three different cases. Rear legs repeat these force patterns with half a cycle delay. In Case A footholds lie along trajectories located 0.45 m off the sagittal plane of the body. In Case B feet 1 and 3 are located 0.5 m off the sagittal plane of the body, while feet 2 and 4 are at 0.35 m (the closest limit of the workspace to the sagittal plane). Finally, in Case C, all feet are located 0.5 m off the sagittal plane (see Fig. 5.4).

For a better understanding of these plots, Table 5.1 indicates the maximum torques in every joint of the front legs (1 and 2) for all three cases. From Table 5.1 we can see that in case C the torque in joint 2 is bigger than in cases A and B. In case A, torques are low because the footholds are closer to the sagittal plane. In case B the torques in joint 2 of right legs (2 and 4) could be even lower than in case A because of the particular force distribution. Therefore, we can conclude that if the machine is designed to support maximum torques as in case A (see Fig. 5.4(a)), then the leg workspace will be very small. It will then be impossible to get a leg configuration like the one in case B (see Fig. 5.4(b)), which uses a larger leg extension for left legs and requires lower torques for joint 2. In contrast, selecting motors to exert torques as in case C (see Fig. 5.4(c)) can require heavy motors and gears that curtail the robot's features. Note that continuous motor-power selection is not possible. Motors are offered in a discrete collection, and increasing torques can demand

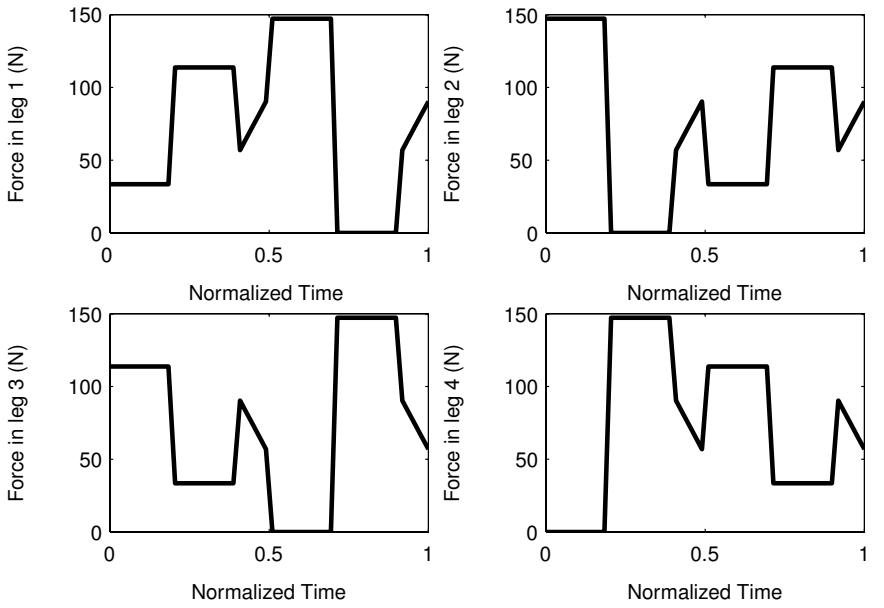


Fig. 5.1. Foot forces along a discontinuous gait ($\beta = 7/9$)

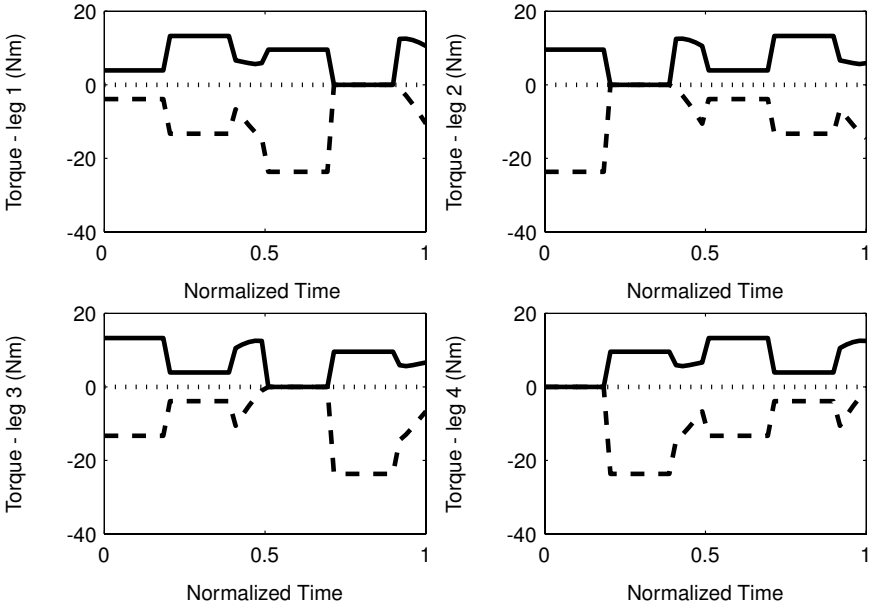


Fig. 5.2. Joint torques along a discontinuous gait ($\beta = 7/9$): joint 1 in *dotted line*, joint 2 in *solid line*, and joint 3 in *dashed line*

higher motor types, thus also increasing weight. Yet another problem arises if the walking machine is intended to carry different payloads. The machine might be stable from the geometric point of view, but if the payload is high either the requested torques or the power consumption may prove to be higher than the maximum allowed. The sections below illustrate such effects.

5.3 Effects of Considering a Limited Motor Torque: Simulation Study

The importance of considering stability due to limitations in torques provided by actuators (torque-limit stability) separately from geometric stability is il-

Table 5.1. Maximum joint torques for three different foot trajectories (see Fig. 5.4)

	Case A		Case B		Case C	
Legs	1, 3	2, 4	1, 3	2, 4	1, 3	2, 4
Foot trajectory position (m)	0.45	0.45	0.5	0.35	0.5	0.5
Torque in joint 2 (Nm)	35.32	35.32	35.14	24.24	42.67	42.67
Torque in joint 3 (Nm)	5.38	5.38	8.07	16.59	8.51	8.51

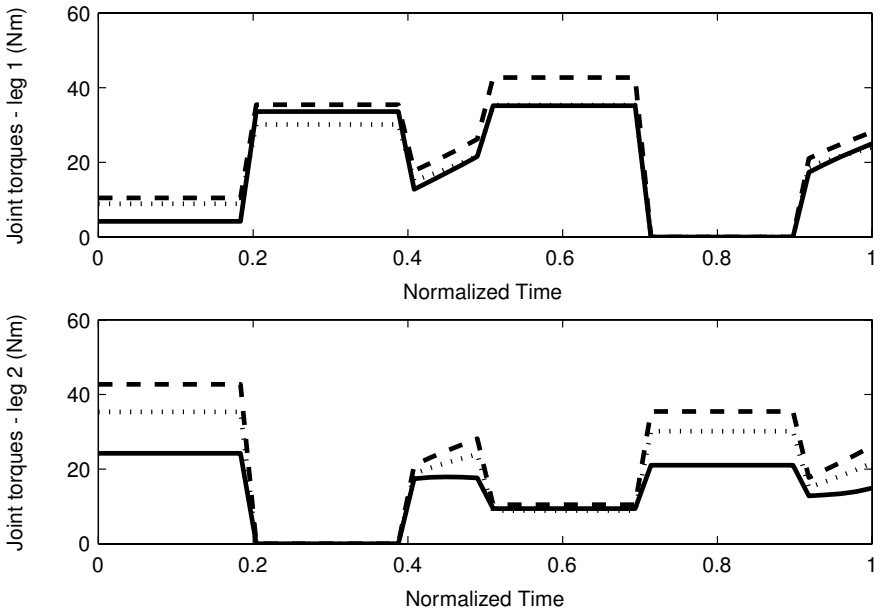


Fig. 5.3. Torques in joints 2 of front legs for different footholds: Case A in *dotted line*, case B in *solid line*, and Case C in *dashed line*. For the sake of comparison the absolute values of torques are plotted.

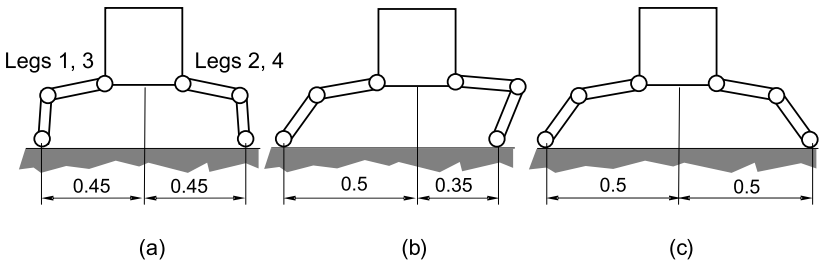


Fig. 5.4. Different leg configurations for the SILO4 walking robot

lustrated in this section through simulation. The Simulation Construction Set software package was used here for simulation purposes (see Appendix B). The main idea was to observe the robot's behavior while traversing a straight line without torque limitation in the robot's joints (ideal actuator), and to compare this with the robot's behavior while performing the same trajectory with torque limitation in all its joints (real actuator).

The SILO4 simulation model was commanded to follow a straight line along the x-axis while remaining at a constant height (z-component) and

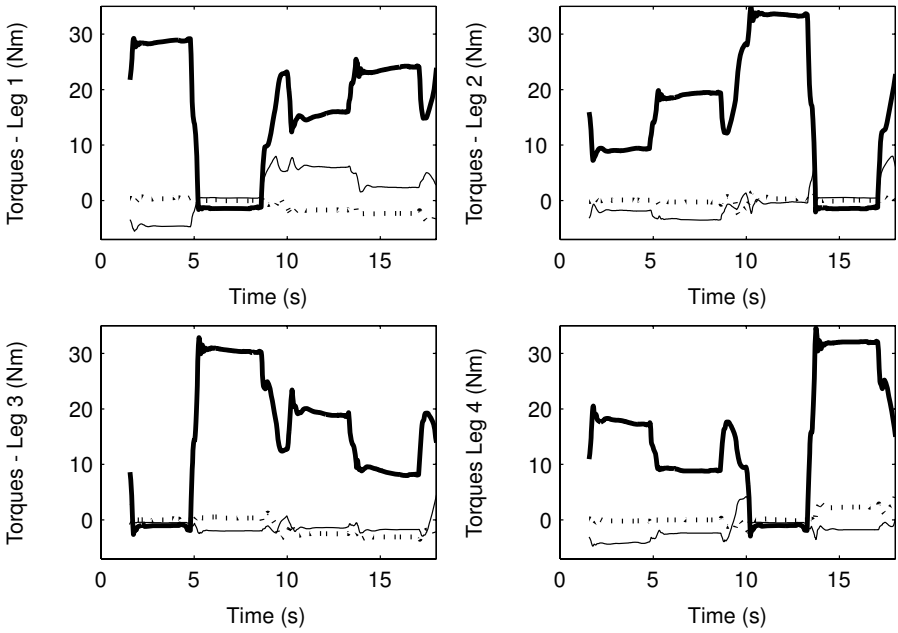


Fig. 5.5. Joint torques in the SILO4 simulation model performing a discontinuous gait with foot trajectories as in case C: joint 1 in *dotted line*, joint 2 in *thick line*, and joint 3 in *thin line*

performing a discontinuous gait. Foot trajectories are as indicated in case C in Fig. 5.4.

Step length was 0.4 m, and step height was 0.1 m. Transfer-leg speed was about 0.15 m/s, and support-leg speed was about 0.14 m/s. These parameters yielded an average body speed of about 0.025 m/s.

Figure 5.5 plots the torques exerted by every joint during the defined trajectory. The controller could provide every requested torque, and the trajectory was as specified, as shown in Fig. 5.6 (thin line).

When the walking robot tried to follow the same body and leg trajectories but with joint torques limited to 35.32 Nm (the maximum torque for Case A), motors in central legs reach saturation and fail to provide the required torques; therefore, the real body trajectory differed from the specified path, as shown in Fig. 5.6 (thick line). Under these conditions, the robot fails to maintain the requested trajectory, and it falls down within a few seconds (z-component = 0), because it cannot even maintain its geometric stability.

Thus, when torques are limited, the robot cannot accomplish the requested task with the imposed constraints. That is, the robot cannot follow the requested body trajectory with the defined foothold trajectories. However, the robot's main task is to follow the body trajectory, which can be done just by using foothold trajectories as specified in Cases A or B in Fig. 5.4. For these

foothold trajectories, the exerted torques are lower than the maximum torque, as shown in Table 5.1 for case A, and the body trajectory can be followed. Note that in case B, legs 1 and 3 use the same area of the leg workspace as in case C; therefore, limiting the leg workspace to avoid foot trajectories as in case C also limits the possibility of using foot trajectories as in case B (for legs 1 and 3), thus losing the possibility of following the desired trajectory. Therefore, using any kind of information about torques (in advance or just by direct measurements) the gait algorithm can avoid those robot's poses demanding unstable torques and can look for new footholds that satisfy stability conditions keeping body trajectories, thus improving the robot's mobility.

5.4 Effects of Limiting Motor Torque in Real Robots

This section illustrates how the static stability of real robots can decrease for certain body motions even while maintaining the same geometric stability margins. Figure 5.7(b) shows the SILO4 walking robot (see Appendix A) in a pose at the beginning of a new crab trajectory. In this pose, the robot is under static stability, and the operator or navigator could decide to lift the body by pursuing a trajectory parallel to the z-axis of the body reference frame (which is assumed to be levelled) up to the pose shown in Fig. 5.7(a). This motion does not change the geometric static stability, because the projection of the

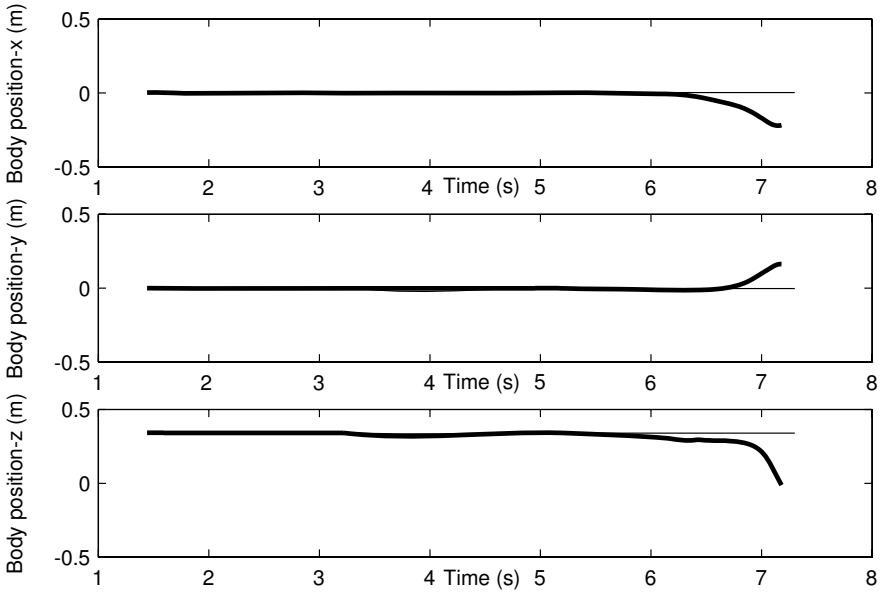
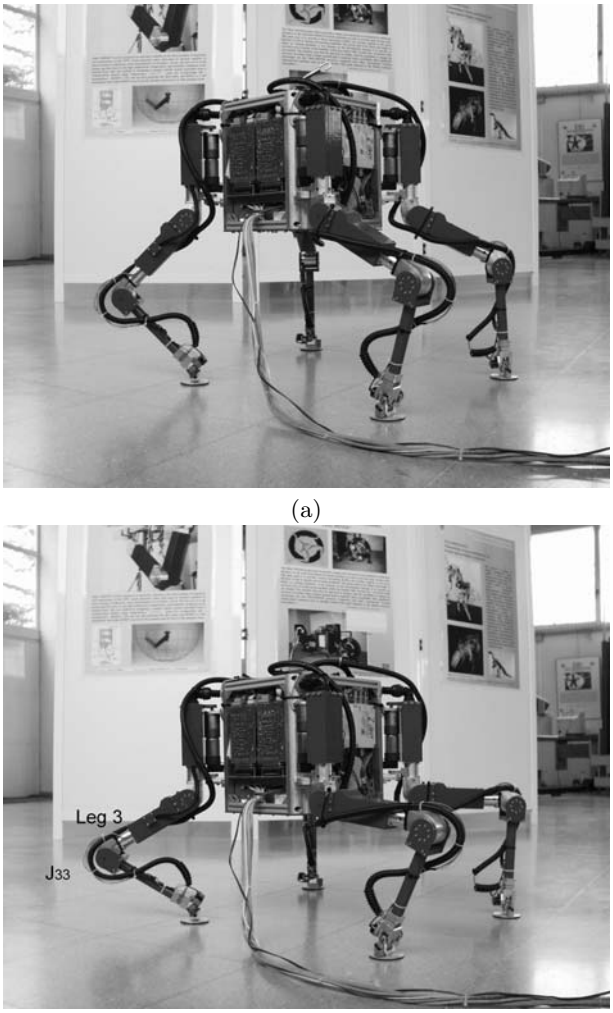


Fig. 5.6. Components of the body trajectory with limited joint torques (*thick line*) and without limited joint torques (*thin line*) along about 1/3 of the locomotion cycle



(a)

(b)

Fig. 5.7. Initial robot poses for first and third experiments

center of gravity and foot contact points remain the same. However, joint 3 in leg 3, J_{33}, in pose (b) is in an overloaded position – the third link of leg 3 is almost horizontal and thus the force exerted in foot 3 produces a big torque in joint 3 (T_{33}=D F_3), and the joint torques the pose demands could rise to beyond maximum level, throwing trajectories off and decreasing the robot's stability (see Fig. 5.8).

Some experiments have been conducted to illustrate this effect. They consist in lifting the body while maintaining the x and y foot components the

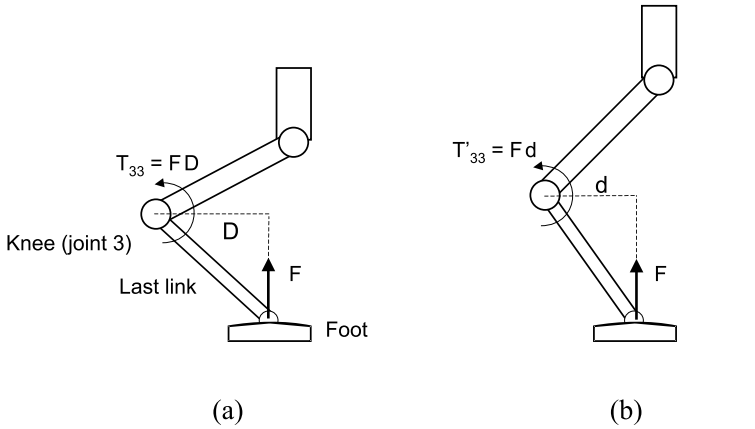


Fig. 5.8. Torque exerted in joint 3

robot has in the two poses shown in Fig. 5.7. At the beginning of the motion, the body is levelled, and the initial foot positions in the body reference frame are

$$\begin{aligned}(x_{foot}, y_{foot})_{leg_1} &= (0.35\,\text{m}, 0.30\,\text{m})\\ (x_{foot}, y_{foot})_{leg_2} &= (0.35\,\text{m}, -0.35\,\text{m})\\ (x_{foot}, y_{foot})_{leg_3} &= (-0.20\,\text{m}, 0.20\,\text{m})\\ (x_{foot}, y_{foot})_{leg_4} &= (-0.15\,\text{m}, -0.42\,\text{m})\end{aligned} \tag{5.13}$$

In the first experiment, the body is at a height of about 0.36 m (see Fig. 5.7(a)); that means

$$(z_{foot})_{leg_i} = -0.36\,\text{m}; \text{for } i = 1...4 \tag{5.14}$$

and the body is lifted about 0.03 m at a speed of 0.01 m/s. During the experiment, the x and y components maintain their values, while the body's z components increase to the final z value. Figure 5.9(a) shows how the z-foot components change. Leg 3 reaches its final position but follows a slightly different trajectory than the other legs.

The second experiment repeats the previous procedure but starts at a body height of 0.32 m, which means $z_i = -0.32$ m for all legs (0.04 m lower than in the first experiment). Figure 5.9(b) shows how the z foot components vary. In this case, feet 1, 2, and 4 move correctly; foot 3, however, follows an irregular trajectory, because leg 3's joints cannot provide the required torques. At the end of the experiment, leg 3 fails to achieve its final position, and the robot becomes unlevelled, worsening its stability margin.

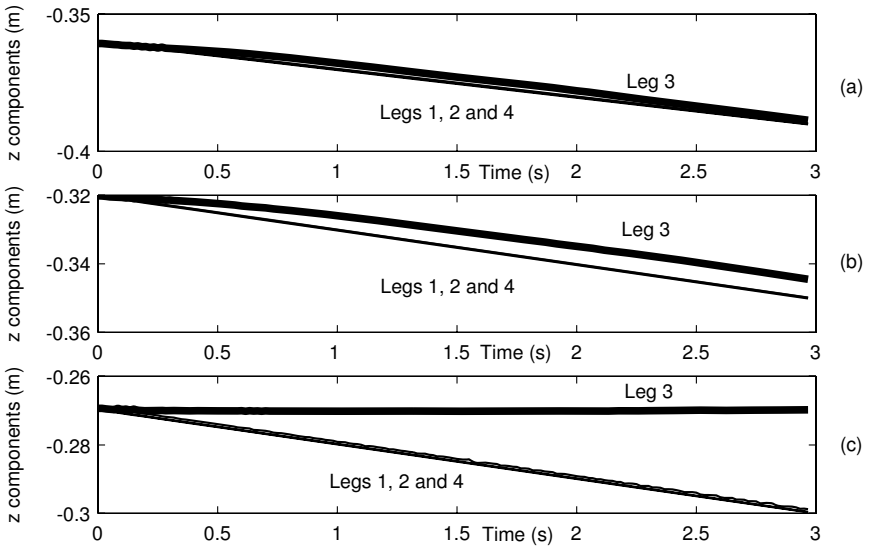


Fig. 5.9. z-Foot components for three different experiments

The last experiment follows the same procedure again, starting at a height of about 0.27 m as in Fig. 5.7(b). Figure 5.9(c) shows the results. In this case leg 3 cannot move the body upwards (the z component remains constant), and the body tilts, reducing the geometric-stability margin and jeopardizing static stability. Thus, a suitable measurement or estimate of joint torques might prevent unstable motion and improve the manoeuvrability of walking robots by incorporating this measurement in the gait generation algorithm (see Sect. 5.5.2).

Summarizing the last three sections, it is unwise to confine the leg workspace, because some achievable robot configurations might thus be ruled out, decreasing the robot's ability. In addition, leg workspace could be reduced or enlarged depending on the payload; humans and animals operate in a similar way. Therefore, real motor torques and power consumption must be taken into account to optimize the functionality of walking robots. These two features can be found very simply. Many robot controllers and drivers provide a measurement of the motor armature current. The torque exerted by the motor is proportional to this current, so armature current provides a direct measurement of the motor torque. Finally, the total power consumption can be obtained by adding up independent motor currents.

Real walking robots can benefit by considering torque and power-consumption stability margins and improve their mobility, especially when using free gaits. In this type of gait, either the operator or, in autonomous systems, the navigator can command a new direction of motion at any time. The gait generator will respond with new foot trajectories using the whole leg workspace, and con-

sequently foot positions might prove unsuitable from the torque-requirement point of view. The incorporation of joint torques and power consumption in gait planning can prevent this kind of malfunction.

This preliminary discussion is an invitation to revise the concept of static stability, extending it, based on geometric considerations, to a new definition that includes both torque limits and power-consumption limits.

5.5 Global-stability Criterion

5.5.1 Definition of Global Criterion

We have seen up to now that to keep a legged robot statically stable the following conditions must be satisfied:

1. The robot must be supported on at least three legs.
2. The stability margin must be positive.
3. The actuators must exert the required joint torques.

There are many stability margins to measure a robot's stability and the most adequate measurement depends on the working conditions (see Chap. 2). However, the margin used does not influence the final stability criterion presented bellow. Therefore, for the sake of simplicity, we have chosen the absolute stability margin (S_{SM}).

The torque-limit-stability margin is a new concept defined as follows.

Definition 5.1. *The torque-limit-stability margin ($\mathbf{S}_{TSM}$) of an n-legged robot with three joints per leg is a vector of dimension $3n$ defined as*

$$\mathbf{S}_{TSM} = \begin{pmatrix} \tau_{\max_1} - |\tau_{req_1}| \\ \dots \\ \tau_{\max_{3n}} - |\tau_{req_{3n}}| \end{pmatrix} \tag{5.15}$$

where $\tau_{\max_i}$ is the maximum torque (positive) exerted by joint i and τ_{req_i} is the maximum required torque to drive joint i.

The torque, τ_i, in a DC motor, i, depends directly on its armature current, I_i; that is, $\tau_i = K_{M_i} I_i$, where K_{M_i} is the torque constant of the motor. Therefore, equation (5.15) can be re-written as

$$\mathbf{S}_{TSM} = \begin{pmatrix} \tau_{\max_1} & -K_{M_1} |I_1| \\ \dots & \dots \\ \tau_{\max_{3n}} & -K_{M_{3n}} |I_{3n}| \end{pmatrix}. \tag{5.16}$$

Motors and actuators in real systems can only provide limited torques normally determined by the motor itself, saturation in drivers and power supply. If a given joint requires a torque greater than the maximum, then the

joint will exert less torque than required. In such a case, the leg could fail, and the robot could become unstable.

One more limitation of a robotic system is the maximum current the power supply can provide. There exists a possible leg configuration in which the $\mathbf{S}_{TSM}$ is positive, but the total current through the system is higher than that supplied by the power supply. In such a case, some joints would be unable to provide the required torques, and an unstable situation would arise. Thus, we define the following stability margin.

Definition 5.2. *The current stability margin (S_{CSM}) is the difference between the maximum electrical current supplied by the power supply and the electrical current required by all the motors simultaneously. That is*

$$S_{CSM} = I_{\max} - \sum_{i=1}^{i=3n} |I_i| \tag{5.17}$$

where $I_{\max}$ is the maximum power supply current and I_i is the current in motor i.

With these premises, we define the global static-stability margin as

Definition 5.3. *The global static-stability margin is the vector defined by*

$$\mathbf{S}_{GSSM} = \begin{pmatrix} S_{SM} \\ \mathbf{S}_{TSM} \\ S_{CSM} \end{pmatrix} \tag{5.18}$$

and then static stability can be defined as below.

Definition 5.4. *A legged robot is statically stable if its global static-stability margin ($\mathbf{S}_{GSSM}$) is definite positive.*

Note that to apply this stability criterion the system must know:

1. The torque exerted by each robot's joint.
2. The maximum torque provided by each joint (we assume here it is the same for all robot's joints).
3. The instant current.
4. The maximum current provided by the power supply.
5. The stability margin, which can easily be compute from the foot positions.

The $\mathbf{S}_{GSSM}$ has $3n+2$ components ($3n$ is the number of joints) of different magnitudes and turns out to be difficult to handle. By normalizing the $\mathbf{S}_{GSSM}$'s components to their highest value, we can obtain non-dimensional components, all of which can be expressed as a fraction of their maximum value. For this purpose, we define the normalized geometric stability margin as

$$S_{GSM_N} = \frac{S_{SM}}{L} \tag{5.19}$$

where L is the maximum geometric stability margin the robot can achieve. Then, the robot will be statically stable if $S_{SM_N} \in (0\ 1]$. Note that quadrupeds get this maximum margin with a robot pose in which the robot is blocked: (a) the robot cannot lift a leg because that would cause it to lose its stability and (b) the robot cannot move the body, because all its legs are at their maximum extension. Hexapods do not get blocked in a pose with all legs fully extended, because they can move three legs while standing on the other three legs.

The normalized torque-limit stability margin is similarly defined as a vector with the generic element i given by

$$S_{TSM_{N_i}} = 1 - \frac{|\tau_{req_i}|}{\tau_{\max_i}}. \tag{5.20}$$

Thus, the robot will be stable if $S_{TSM_{N_i}} \in (0\ 1]$, $\forall i$.

Finally, the normalized current stability margin becomes

$$S_{CSM_N} = 1 - \frac{\sum_{i=1}^{i=3n} |I_i|}{I_{\max}}. \tag{5.21}$$

The robot will be stable if $S_{CSM_N} \in (0\ 1]$.

With these normalized margins, the $\mathbf{S}_{GSSM}$ can be re-written as

$$\mathrm{S_{GSSMN}} = M \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}_{(3n+2)\times 1} \tag{5.22}$$

where M is referred to as the static stability margin and gives an idea of how far we want to move the robot from instability. Static stability is assured if and only if $M \in (0\ 1]$. Note that an M stability margin for the S_{SM} means the margin must be greater than $M\,L$, while for the $\mathbf{S}_{TSM}$ and the S_{CSM}, a margin M means τ_{req_i} and $\sum_{i=1}^{i=3n} |I_i|$ must be below $\tau_{\max_i}$ and $I_{\max}$ in about $M\,\tau_{\max_i}$ and $M\,I_{\max}$, respectively. That is, if we specify a static margin $M = 0.1$, that means we want to perform a gait with a stability-safe zone of 0.1 times the maximum value from the unstable values. For instance, if the maximum geometric margin is 0.25 m, the maximum torque allowed in each joint, $\tau_{\max_i}$, is 80 Nm and the maximum total current allowed in the power supply, $\sum_{i=1}^{i=3n} |I_i|$, is 5 A, then the gait will be performed with a geometric stability margin higher than 0.025 m, joint torques below 72 Nm and total current below 4.5 A. In this case the robot will be able to walk even in the face of an undetermined disturbance.

There are two ways of finding the torque and current stability margins: direct measurement and estimation. The direct measurement of joint torques has the advantage of simplicity. Normally it does not need additional sensors installed on the robot. Motor drives and controllers usually provide an analogue signal proportional to the motor torque (current). However, this method requires each next supporting robot pose to be specifically tried to determine whether it will fall within a stable margin or not. This is one clear shortcoming of the method.

The second method, based on *a priori* estimation of the torques required for the next robot pose, is more efficient, because instability can be detected in advance. However, computing needed torques demands an accurate dynamic model of the robot and the solution of the force-distribution problem. This is a complex problem, which is being solved at present using optimization methods and is far from real-time solutions (Pfeiffer and Weidemann, 1991; Lin and Song, 1993). One more drawback is that this algorithm will depend on the robot's payload, which has to be reported to the controller or measured by the robot itself, a procedure that requires additional sensors.

By obtaining the global stability margin through either measuring or estimation, we can improve the robot's gait. Free gaits are characterized by selecting footholds and leg sequences as a function of features such as stability measurements, terrain conditions and direction of motion (see Chap. 4). Joint torques and power consumption can be considered two more conditions/restrictions in the foothold-selection algorithm. If the stability margin is estimated, then it can be taken into consideration in the foothold-selection process. This is, of course, the best solution. However, if the stability margin is obtained by direct measurements, the robot will have to be placed in the out-of-stability-margin pose (note that this does not mean the robot will become unstable). The solution in this case is to re-compute the foothold after detecting an unstable margin. Normally, the robot will be stopped, and some trajectory features could be changed, such as speed; nevertheless, this method is the easiest for real-time implementation.

Including the stability criterion introduced in this chapter in gait generation algorithms is a simple, straightforward step sketched in the following section.

5.5.2 Gait Based on the Global Criterion

To illustrate how to use the global stability margin in a gait algorithm, let us consider the discontinuous free gait developed in Chap. 4. In that algorithm, the foothold planner had five constraints: E1 to E5. Now, it must exhibit two additional constraints: E6 and E7. That is as follows.

Constraint E6: *The foothold must accomplish the torque-limit stability condition, written as*

$$\tau_i < \tau_{\max_i} - M\,\tau_{\max} \quad \forall i \in [1,\ 3n] \tag{5.23}$$

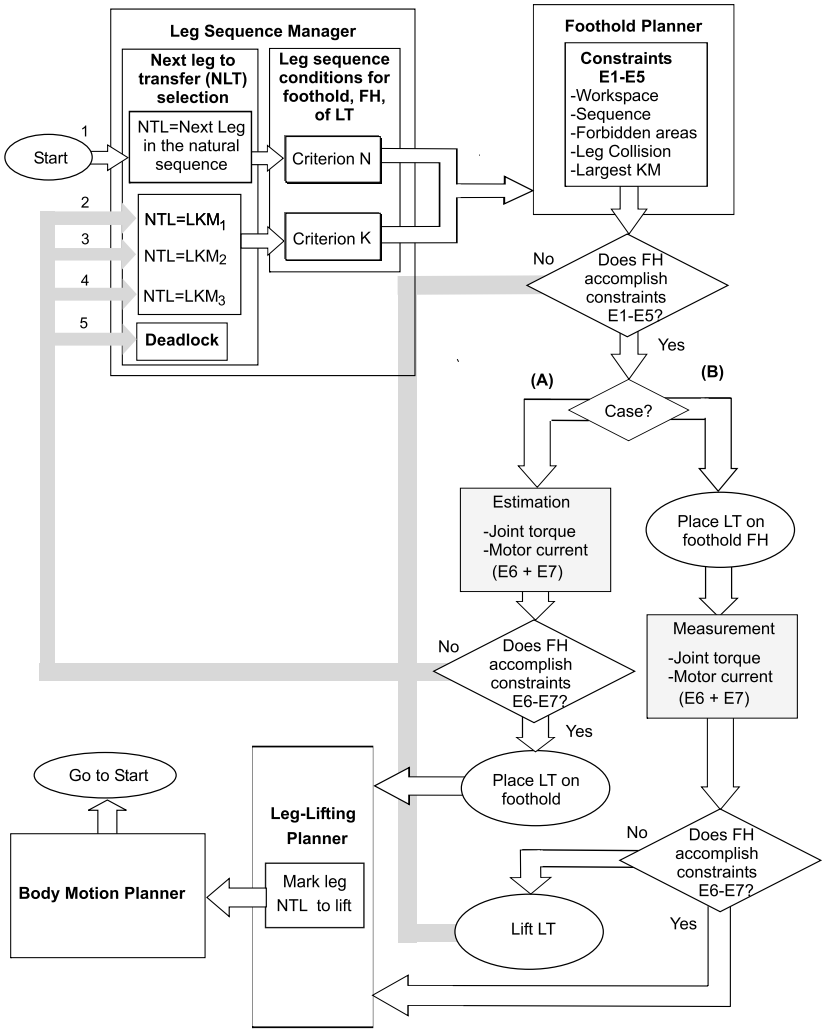


Fig. 5.10. Free gait algorithm including joint torque and motor current criteria

where i denotes the joint and M is the quasi-static stability margin define for the desired motion.

Constraint E7: *The foothold must accomplish the condition for the total current contribution, written as*

$$\sum_{i=1}^{i=3n} |I_i| < I_{\max} - M\, I_{\max} \tag{5.24}$$

If Constraints E1 – E5 are not satisfied (see Fig. 5.10) the algorithm is repeated starting with the selection of next leg (in the sequence defined by criteria N or K in Chap. 4). If the robot's controller can estimate the joint torques and total current (Case A) and constraints E6 and E7 are satisfied, then the leg is placed in the selected foothold. If the robot's controller cannot estimate the joint torques and total current, but it can measure the actual values, then Constraints E6 and E7 are checked during foot placement (Case B in Fig. 5.10). If the selected global margin is violated (Constraints E6 and E7), the leg LT is lifted and a new foothold is sought.

The leg sequence manager, body motion planner and leg-lifting planner of the algorithm remain the same as in Chap. 4.

5.6 Conclusions

Static-stability measurements for walking robots have traditionally focused on geometric aspects. This chapter reports that considering geometric parameters alone in robot stability leads to inefficient robot design and robot malfunction. The effects of using this limited measurement are shown by means of computer simulation and experiments using a real walking robot. This chapter states that those problems can be overcome by taking into account joint torques and power consumption in the design of gait algorithms. A new global stability criterion is proposed to factor these new magnitudes into the static-stability margin of a walking robot, taking into consideration geometric and torque/power measurements. The global stability criterion is based on the well-known geometric stability margin (scalar) and on two new criteria, the torque-limit stability margin (vector) and the current-limit stability margin (scalar). This chapter states that it is possible to improve features and the functionality of walking robots by taking global static stability into account in the design of both robots and locomotion gaits.

Part II

Control Techniques

6

Kinematics and Dynamics

6.1 Introduction

Walking robots are very complex mechanical systems, featuring a variable structure defined by its number of degrees of freedom (DOF). For designing algorithms for the control of walking robots it is important to have good models describing the kinematic and dynamic behaviour of the robot.

The kinematic model describes the relationship between joint variables and foot position and its derivatives, while the dynamic model relates joint motion with the forces involved on it.

The kinematics of walking robots can be reduced to the kinematics of its legs and body. Legs of traditional multi-legged robots exhibit up to three DOF and it is unusual to find a legged robot based on legs with more than three DOF. Bipeds and humanoids are, again, an exception in this book.

Regarding the kinematics of the body, it can be reduced to knowing its orientation in space, which is normally accomplished by using a two-axis inclinometer and an electromagnetic compass. If the support plane of the robot is known, leg kinematics can determine the body position and orientation. On irregular unstructured terrain, the body orientation can only be obtained by direct sensor readings as mentioned above.

The kinematics of a leg is that of a manipulator; therefore, the methods to derive the kinematic relationships are the same. One possible method consists in using trigonometric relationships. However, the kinematic analysis can be extremely complex and the introduction of some conventions simplifies the final equations greatly. The most popular convention was defined by Denavit and Hartenberg[1] in 1955, which is a helpful, systematic way of choosing the reference frame associated with joints and links to derive the kinematic model of an open-loop articulated chain. This method is especially useful when the number of DOF is high. For a 3-DOF mechanism such as a leg, the kinematic model could be easily derived by using trigonometric relationships. However,

[1] We will refer to the Denavit-Hartenberg convention as the D-H convention.

specifying the kinematic model in terms of the D-H method also helps in deriving the dynamic model.

The D-H convention and procedure to derive the kinematic model of an open-loop articulated mechanism can be found in books on robot manipulators (Fu *et al.*, 1987; Paul, 1981; Spong and Vidyasagar, 1989; Craig, 1989). In this chapter we will just mention the basic procedure of the D-H method. Interested readers are encouraged to read the referred books.

By contrast, obtaining a quadruped's equations of motion is extremely time-consuming and yields an indeterminate system of equations, which must be solved using some optimization criterion, *e.g.* optimal force distribution, employing the Lagrange-multipliers method (Bennani and Giri, 1996; Pfeiffer and Weidemann, 1991). To simplify the problem, dynamic models of quadrupeds do not usually consider the dynamics of the legs, based on the assumption of high gearing and massless legs. However, this assumption should be ensured by means of a dynamic analysis of the robot to avoid errors due to an unreasonable simplification. In order to make dynamic equations reflect the reality of the physical system, it is important to model the most significant effects acting on the system. A trade-off has to be established between an accurate model of the system and the viability of its real-time implementation for dynamic control. Therefore, robot-dynamics analyzing methods are needed that permit the simplification of the equations of motion without yielding significant errors during real-time control. As equations of motion are normally used for trajectory generation and control, the analysis should reflect which dynamic effects arise during the different robot movements.

Methods already exist for experimental identification of robot dynamics for model generation (Armstrong, 1989; Mayeda *et al.*, 1984; Swevers *et al.*, 2000). The equations of motion are derived through an experimental parameter-identification process based on an initial guess in the dynamic response of the system. However, the correctness of the initial guess is critical, and these methods do not give any insight into the conceptual and physical understanding of robot dynamics. Garcia *et al.* (2003) propose a method for experimental dynamic analysis of a walking robot depending on trajectory parameters. The envisaged application is model-based robot control, *e.g.* computed torque control. The accuracy of these controllers relies highly on the ability of the model to accurately predict the required actuator torques. Therefore, the method ascertains the main dynamic components affecting the system during different real leg trajectories, including actuator dynamics and friction. The mathematical model thus obtained results in an accurate simplified representation of system dynamics.

This chapter is structured as follows: first the kinematic and dynamic models for a quadruped robot are presented in Sects. 6.2 and 6.3. Then a model analysis method is shown as a function of trajectory parameters in Sect. 6.4. To show the use of this method, an application to the SILO4 quadruped robot is detailed in Sect. 6.5, and finally, relevant conclusions are found in Sect. 6.6.

6.2 Kinematics of Walking Robots

The kinematic model describes the relationship between the joint variables of the leg, $(q_1\ ,\ \ldots,\ q_n)^T$, and foot position and orientation, $(x,\ y,\ z,\ r,\ p,\ y)^T$. In the case of a rotary joint, the joint variable is the angle between the links joined in that joint. In the case of a prismatic or sliding joints, the joint variable is the link extension.

The kinematic model of an open-loop articulated chain can be divided into two problems: forward kinematic problem and inverse kinematic problem. The forward kinematic problem gives the position and orientation of the foot, $(x,\ y,\ z,\ r,\ p,\ y)^T$, in terms of the joint variables, $(q_1,\ \ldots,\ q_n)^T$. In contrast, the inverse problem gives the joint variables in terms of the position and orientation of the foot.

6.2.1 Forward Kinematics: The Denavit–Hartenberg Convention

This section derives the forward kinematic model by using the D-H convention. In fact, this is the description of the procedure following Spong and Vidyasagar (1989).

Let us assume an n-DOF leg. Then:

- The leg has n joints.
- The leg has $n+1$ links numbered from 0 (hip) to n (foot).
- The i-th joint connects link $i-1$ and link i.
- The joint variable of the i-th joint is denoted by q_i. In a rotary joint q_i is the angle between link $i-1$ and link i. In a prismatic joint q_i is the displacement between link $i-1$ and link i.
- Every link i has attached rigidly a reference frame i, which is associated to the joint $i-1$. It is assumed that the hip has attached the inertial frame 0.

With these assumptions, we have that the coordinates of a point $\mathbf{p}_i$ in the link i are constant with respect to the reference frame i and they do not depend on the leg motion. However, if a rotation occurs between link i and $i-1$, point $\mathbf{p}_i$ is expressed in the reference frame $i-1$ as

$$\mathbf{p_{i-1}} = {}^{\mathbf{i-1}}\mathbf{R_i}\mathbf{p_i} \tag{6.1}$$

where ${}^{\mathbf{i-1}}\mathbf{R_i}$ is the rotation matrix (3×3) that transform a point in the i-th reference frame into the *(i-1)*th reference frame when origins on both frames coincide.

In a more general case, if there is a rotation and translation between reference frames, then (6.1) becomes (see Fig. 6.1)

$$\mathbf{p_{i-1}} = {}^{\mathbf{i-1}}\mathbf{R_i}\mathbf{p_i} + {}^{\mathbf{i-1}}\mathbf{d_i} \tag{6.2}$$

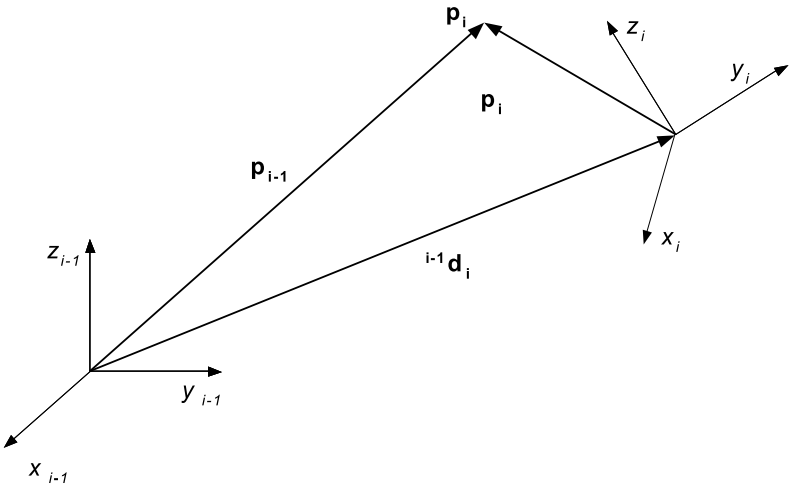


Fig. 6.1. Coordinate frames

where $^{\mathbf{i-1}}\mathbf{d_i}$ is the vector (3×1) that represents the origin of the reference frame i into the reference frame $i-1$. Equation (6.2) can be expressed as a matrix product as

$$\begin{pmatrix} \mathbf{p_{i-1}} \\ \mathbf{1} \end{pmatrix} = \begin{pmatrix} ^{\mathbf{i-1}}\mathbf{R_i} & ^{\mathbf{i-1}}\mathbf{d_i} \\ \mathbf{0}_{(1\times 3)} & \mathbf{1} \end{pmatrix} \begin{pmatrix} \mathbf{p_i} \\ \mathbf{1} \end{pmatrix}. \tag{6.3}$$

The matrix

$$^{\mathbf{i-1}}\mathbf{A_i} = \begin{pmatrix} ^{\mathbf{i-1}}\mathbf{R_i} & ^{\mathbf{i-1}}\mathbf{d_i} \\ \mathbf{0}_{(1\times 3)} & \mathbf{1} \end{pmatrix} \tag{6.4}$$

is termed the homogeneous matrix. Vectors with the form $(p_x\, p_y\, p_z\, 1)^T$ are termed homogeneous vectors. Matrix $^{\mathbf{i-1}}\mathbf{A_i}$ transforms the coordinates of a point from reference frame i into reference frame $i-1$. This matrix changes with the configuration of the leg but it only depends on the joint variable q_i, *i.e.*

$$^{\mathbf{i-1}}\mathbf{A_i} = {}^{\mathbf{i-1}}\mathbf{A_i}(q_i). \tag{6.5}$$

The homogeneous transformation matrix associated to the joint i in the most general case can be expressed by (Fu *et al.*, 1987; Paul, 1981; Spong and Vidyasagar, 1989; Craig, 1989)

$$^{\mathbf{i-1}}\mathbf{A_i} = \begin{pmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.6}$$

where a_i, α_i, θ_i, and d_i are the link parameters. Parameters a_i and α_i are constant for every link, θ_i is the joint variable for a rotary joint and d_i is the joint variable for a prismatic or sliding joint, otherwise they are also constants. These joint parameters are defined below.

Finally, the homogeneous matrix that transforms the coordinates of a point from reference frame n (foot) into the reference frame 0 (hip) is given by

$$^0A_n = {}^0A_1{}^1A_2....^{n-1}A_n. \tag{6.7}$$

Denavit–Hartenberg Procedure

The D-H procedure can be stated in the following steps.

Step 1: Locate the joint axes $z_0, \ldots, z_{n-1}$ along the joint shaft.

- If joint i is a rotary joint, z_i lies along the joint revolution axis.
- If joint i is a prismatic one, z_i lies along the joint translation axis.

Step 2: Set up the hip reference frame. Locate the origin anywhere in the z_0 axis. The x_0 and y_0 axes must be chosen to form a right-hand frame.

Step 3: Perform Steps 4 to 6 for i= 1, ..., n−1.

Step 4: Locate the origin o_i.

- If z_i intersects z_{i-1}, then locate o_i at this axis intersection.
- If z_i is parallel to z_{i-1}, then locate o_i at joint $i + 1$.
- If z_i and z_{i-1} are not in the same plane, then locate o_i where the common normal to z_i and to z_{i-1} intersects z_i.

Step 5: Locate x_i-axis.

- If z_i intersects z_{i-1}, then locate x_i in the direction normal to $z_i - z_{i-1}$ plane.
- If z_i does not intersect z_{i-1}, then locate x_i along the common normal between z_i and z_{i-1}through o_i.

Step 6: Define y_i to form a right-hand frame.

Step 7: Establish the foot reference frame (x_n, y_n, z_n).

- z_n lies along z_{n-1}.
- x_n must be normal to z_{n-1} and z_n (x_n intersects z_{n-1}).
- y_i must form a right-hand frame.

Step 8: Get the link parameters a_i, α_i, θ_i, and d_i for every i.

- a_i is the **link length** or distance along x_i from o_i to the intersection of x_i and z_{i-1}.
- α_i is the **link rotation** or the angle that z_{i-1} must rotate about x_i to coincide with z_i.
- d_i is the **distance between adjacent links** or distance along z_{i-1} from o_{i-1} to the intersection of x_i and z_{i-1}. For a prismatic joint, d_i is the joint variable.
- θ_i is the **angle between adjacent links** or the angle that x_{i-1} must rotate about z_{i-1} to coincide with x_i. For a rotary joint, θ_i is the joint variable.

Step 9: Form the matrices $\mathbf{^{i-1}A_i}$ for $i = 1, ..., n$.

Step 10: Form the homogeneous matrix

$$\mathbf{^0A_n} = \mathbf{^0A_1\,^1A_2}....\mathbf{^{n-1}A_n} = \begin{pmatrix} \mathbf{^{i-1}R_i} & \mathbf{^{i-1}d_i} \\ \mathbf{0}_{(1\times 3)} & \mathbf{1} \end{pmatrix} \tag{6.8}$$

where vector $\mathbf{^{i-1}d_i}$ gives the foot position in the hip reference frame and $\mathbf{^{i-1}R_i}$ gives the orientation of the foot reference frame in the hip reference frame.

Example of Forward Kinematics

This section derives the kinematic forward solution of the leg of the SILO4 walking robot, used in this book as a testbed model for simulation and experiments. This leg consists of three rotary joints as indicated in Fig. 6.2.

The application of Steps 1 to 7 of the D-H procedure presented above in this section produces the reference frames plotted in Fig. 6.2. Step 8 defines the link parameters indicated in Table 6.1 and joint variables θ_1, θ_2 and θ_3 illustrated in Fig. 6.2.

Table 6.1. D-H parameter for the SILO4 leg

Parameters	Links		
a_i	a_1	a_2	a_3
α_i	$\pi/2$	0	0
d_i	0	0	0
θ_i	θ_1	θ_2	θ_3

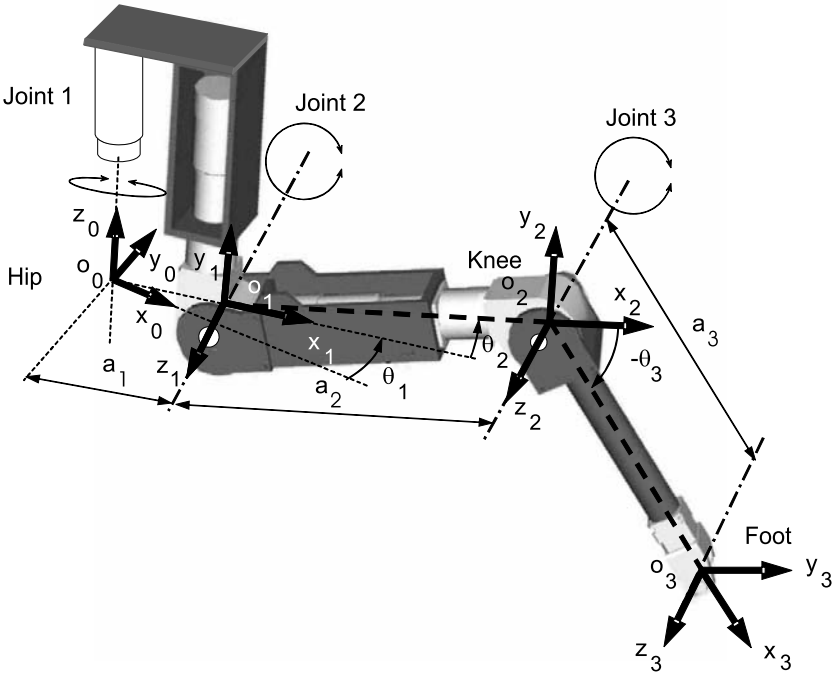


Fig. 6.2. D-H parameters of the SILO4 leg

For the link parameters obtained, (6.6) defines the homogeneous matrices associated to the joints. These matrices are[2]

$$ {}^{\mathbf{0}}\mathbf{A}_{\mathbf{1}} = \begin{pmatrix} C_1 & 0 & S_1 & a_1 C_1 \\ S_1 & 0 & -C_1 & a_1 S_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.9} $$

$$ {}^{1}\mathbf{A}_{2} = \begin{pmatrix} C_2 & -S_2 & 0 & a_2 C_2 \\ S_2 & C_2 & 0 & a_2 S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.10} $$

$$ {}^{2}\mathbf{A}_{3} = \begin{pmatrix} C_3 & -S_3 & 0 & a_3 C_3 \\ S_3 & C_3 & 0 & a_3 S_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{6.11} $$

Finally, Step 10 provides the matrix that translates the foot reference frame into the hip reference frame.

[2] The following notation is assumed: $S_i = \sin\theta_i$, $C_i = \cos\theta_i$, $S_{ij} = \sin(\theta_i + \theta_j)$, $C_{ij} = \cos(\theta_i + \theta_j)$.

$$
{}^0\mathbf{A}_3 = {}^0\mathbf{A}_1\,{}^1\mathbf{A}_2\,{}^2\mathbf{A}_3 = \begin{pmatrix} R_{3\times3} & d_{3\times1} \\ 0_{1\times3} & 1 \end{pmatrix} = \\
\begin{pmatrix} C_1C_{23} & -S_{23}C_1 & S_1 & C_1(a_3C_{23}+a_2C_2+a_1) \\ S_1C_{23} & -S_{23}S_1 & -C_1 & S_1(a_3C_{23}+a_2C_2+a_1) \\ S_{23} & C_{23} & 0 & a_3S_{23}+a_2S_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.12}
$$

where $d = (x, y, z)^T$. Therefore, the position of the foot is given by

$$x = C_1(a_3C_{23} + a_2C_2 + a_1) \tag{6.13}$$

$$y = S_1(a_3C_{23} + a_2C_2 + a_1) \tag{6.14}$$

$$z = a_3S_{23} + a_2S_2. \tag{6.15}$$

6.2.2 Inverse Kinematics

The inverse kinematics consists in determining the joint variables $(q_1, \ldots, q_n)^T$ in terms of the foot position and orientation. For a 3-DOF leg the problems can be stated as

$$
\begin{pmatrix} a_{11} & a_{12} & a_{13} & x \\ a_{21} & a_{22} & a_{23} & y \\ a_{31} & a_{32} & a_{33} & z \\ 0 & 0 & 0 & 1 \end{pmatrix} = {}^0A_1(q_1)\,{}^1A_2(q_2)\,{}^2A_3(q_3) \tag{6.16}
$$

where $(x, y, z)^T$ is the foot position and $(a_{11}, a_{21}, a_{31})^T$, $(a_{12}, a_{22}, a_{32})^T$ and $(a_{13}, a_{23}, a_{33})^T$ are the orientation vectors of the foot. Equation (6.16) represents a system of 12 equations in 3 unknowns, which is difficult to solve directly in closed form and, when the direct kinematic has a unique solution, the inverse problem may or may not have a solution. Furthermore, if a solution exits it may or may not be unique.

Closed form solutions rather than numerical solutions are preferable for two reasons: first, they can be solved at a quicker rate and, second, it is easier to choose a particular solution among several possible solutions. There exist several methods to solve the inverse kinematics of an open-loop chain: geometric approach, algebraic method, inverse transform technique. The following section solves the inverse kinematics of the SILO4 leg by using an algebraic approach.

An Algebraic Approach to the Solution of the Inverse

Equations (6.13)–(6.15) relate foot positions and joint variables. From (6.13) and (6.14) we obtain

$$xS_1 - yC_1 = 0 \tag{6.17}$$

that is

$$\tan\theta_1 = \frac{S_1}{C_1} = \frac{y}{x}. \tag{6.18}$$

Therefore if $x \neq 0$ and $y \neq 0$ a solution for θ_1 is

$$\theta_1 = \arctan 2(y, x)\,. \tag{6.19}$$

If x=0 and y=0, an infinite number of solutions exits for θ_1. In such a case we say the leg is in a singular configuration.

From (6.13) and (6.14) and using the trigonometric relationship

$$S_j^2 + C_j^2 = 1 \tag{6.20}$$

we can obtain

$$C_{23} = \frac{xC_1 + yS_1 - a_2C_2 - a_1}{a_3}. \tag{6.21}$$

From (6.15) we get

$$S_{23} = \frac{z - a_2S_2}{a_3} \tag{6.22}$$

and substituting for C_{23} and S_{23} from (6.21) and (6.22), respectively, into (6.20) we obtain

$$AS_2 + BC_2 = D \tag{6.23}$$

where

$$\begin{aligned} A &= -z \\ B &= a_1 - (xC_1 + yS_1) \\ D &= \frac{2a_1(xC_1 + yS_1) + a_3^2 - a_2^2 - a_1^2 - z^2 - (xC_1 + yS_1)^2}{2a_2}. \end{aligned} \tag{6.24}$$

The equation form at (6.23) is traditionally solved by performing the following change of variables (Craig, 1989):

$$\begin{aligned} A &= r\cos\phi \\ B &= r\sin\phi \end{aligned} \tag{6.25}$$

then

$$\begin{aligned} \phi &= \arctan 2(B, A) \\ r &= +\sqrt{A^2 + B^2} \end{aligned}\,. \tag{6.26}$$

Equation (6.23) can now be written as

$$\cos\phi\sin\theta_2 + \sin\phi\cos\theta_2 = \frac{D}{r} \tag{6.27}$$

or

$$\sin(\phi + \theta_2) = \frac{D}{r}. \tag{6.28}$$

Using the trigonometric relationship (6.20), we get

$$\cos(\phi + \theta_2) = \pm\sqrt{1 - \sin^2(\phi + \theta_2)} = \frac{\pm\sqrt{r^2 - D^2}}{r}. \tag{6.29}$$

Thus,

$$\tan(\phi + \theta_2) = \frac{D}{\pm\sqrt{r^2 - D^2}} \tag{6.30}$$

and so

$$\theta_2 = -\phi + \arctan 2(D, \pm\sqrt{r^2 - D^2}) \tag{6.31}$$

and substituting the values of ϕ and r defined in (6.26) we obtain

$$\theta_2 = -\arctan 2(B, A) + \arctan 2(D, \pm\sqrt{A^2 + B^2 - D^2}). \tag{6.32}$$

Notice that (6.32) has two solutions. The right solution must be analyzed in terms of mechanical constraints.

Finally, θ_3 can be obtained from (6.21) and (6.22) as

$$\theta_3 = \arctan 2(z - a_2 S_2, xC_1 + yS_1 - a_2 C_2 - a_1) - \theta_2. \tag{6.33}$$

Thus, equations (6.19), (6.32) and (6.33) provide the inverse kinematic model of the SILO4 leg.

6.2.3 A Geometric Approach to Solve Kinematics

As it was mentioned above, D-H convention is just a method to solve the forward kinematics in a systematic manner. However, especially for legs or manipulators with few DOF, it could be straightforward to use traditional geometric methods. In this section, the pantographic mechanism is used to illustrate this method although the author's main aim is to provide the kinematic relationships of the pantograph, a device broadly used as a leg in a large number of walking robots (see Figs. 1.5 and 1.8).

The pantograph is a four-bar mechanism with four passive joints (pj) as indicated in Fig. 6.3. Point A is moved along the z axis by using a prismatic device. Point B is also moved along the x axis by using an additional prismatic device. Motions of points A and B produce the motion of point C, which is linked with the foot when the device is used as a leg. This mechanism is known as planar pantograph and provides 2 DOF. There are two manners of providing the third DOF. The first and maybe the most broadly used pantographic configuration exhibits a rotary joint which shaft coincides with the z_0 axis. In this way, the planar pantograph rotates about the z_0 axis. The ASV used this mechanism with the z-axis parallel to the longitudinal axis of the body (see Fig. 1.4). The second configuration provides in point B one more prismatic device that moves point B parallel to the y axis (perpendicularly to the sheet). This mechanism is called Cartesian pantograph and provides three independent linear motions in the foot (see Figs. 1.5 and 1.8).

From Fig. 6.3, we can compute the x-foot component as

$$x = L_1 \cos\alpha + L_2 \cos\beta \tag{6.34}$$

and

$$\cos\alpha = \frac{d_x - a_2 \cos\beta}{a_1} \tag{6.35}$$

and thus

$$x = L_1 \frac{d_x - a_2 \cos\beta}{a_1} + L_2 \cos\beta = \frac{L_1}{a_1} d_x + (L_2 - \frac{L_1}{a_1} a_2) \cos\beta. \tag{6.36}$$

If the four main bars satisfy

$$\begin{aligned} L_1 &= L_2 = L \\ a_1 &= a_2 = a \end{aligned} \tag{6.37}$$

then (6.36) yields

$$x = \frac{L}{a} d_x. \tag{6.38}$$

From Fig. 6.3, the z_0 foot component is[3]

$$z_0 = L_1 \sin\alpha + d_{z0} - L_2 \sin\beta \tag{6.39}$$

and

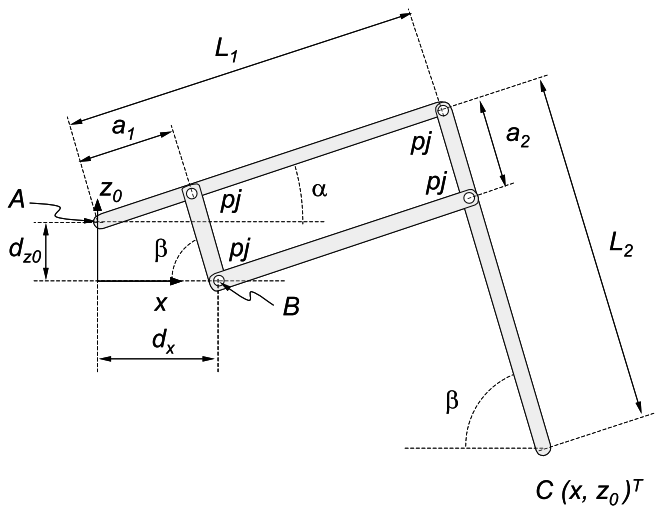


Fig. 6.3. Planar pantographic mechanism

[3] Notice that z_0 component is referred to the fixed reference frame, while the x component is referred to the reference frame (x, z_0), which lies on the plane of the planar pantograph.

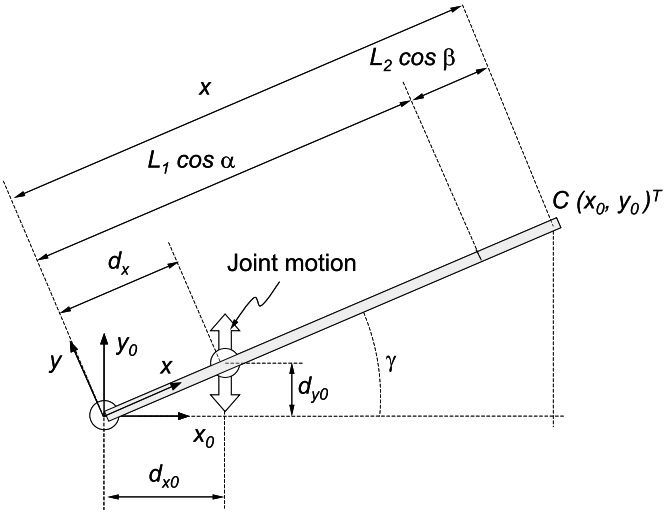


Fig. 6.4. Top view of the Cartesian pantographic mechanism

$$\sin\alpha = \frac{a_2 \sin\beta - d_{z0}}{a_1} \tag{6.40}$$

and thus

$$z_0 = (1 - \frac{L_1}{a_1})d_{z0} + (\frac{L_1}{a_1}a_2 - L_2)\sin\beta \tag{6.41}$$

and for the conditions at equation (6.37) we get

$$z_0 = (1 - \frac{L}{a})d_{z0}. \tag{6.42}$$

Equations (6.38) and (6.42) give the kinematic relationships of the planar pantograph. For a Cartesian pantograph, the y component is actuated by one more prismatic joint. In this case, the actuator moves the point B (passive joint) parallel to the y_0 axis (no matter what the x component is). Figure 6.4 shows a top view of the Cartesian pantograph mechanism. In this figure, we can compute that

$$\sin\gamma = \frac{d_{y0}}{d_x} \tag{6.43}$$

and

$$y_0 = x\sin\gamma = x\frac{d_{y0}}{d_x} \tag{6.44}$$

and using (6.38) we obtain

$$y_0 = \frac{L}{a}d_{y0}. \tag{6.45}$$

Foot components are $(x_0, y_0, z_0)^T$ and joint variables are $(d_{x_0}, d_{y_0}, d_{z_0})^T$; however, (6.38) gives the x-component of the system (x, z_0) contained in the

plane of the planar pantograph, where d_x is the joint variable. To compute x_0 as a function of d_{x0}, we have

$$x_0 = x\cos\gamma = \frac{L}{a} d_x \cos\gamma. \tag{6.46}$$

From Fig. 6.4 we obtain

$$d_x = \sqrt{d_{x0} + d_{y0}} \tag{6.47}$$

and

$$\cos\gamma = \frac{d_{x0}}{\sqrt{d_{x0} + d_{y0}}} \tag{6.48}$$

and then

$$x_0 = \frac{L}{a} d_{x0}. \tag{6.49}$$

Equations (6.49), (6.45) and (6.42) can be written in matrix form as

$$\begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} = \begin{pmatrix} \frac{L}{a} & 0 & 0 \\ 0 & \frac{L}{a} & 0 \\ 0 & 0 & 1 - \frac{L}{a} \end{pmatrix} \begin{pmatrix} d_{x0} \\ d_{y0} \\ d_{z0} \end{pmatrix}. \tag{6.50}$$

Thus, the components of a Cartesian pantograph are decoupled and each external component only depends on its internal variable. This is one of the advantages of the pantograph mechanism. Another advantage is that the motion of a foot component is the motion of its joint (d_{x0} or d_{y0}) times the factor L/a for x_0 and y_0 components, and d_{z0} times the factor $(1 - L/a)$ for z_0 component.

As the components are decoupled the inverse kinematic can be easily computed, yielding

$$\begin{pmatrix} d_{x0} \\ d_{y0} \\ d_{z0} \end{pmatrix} = \begin{pmatrix} \frac{a}{L} & 0 & 0 \\ 0 & \frac{a}{L} & 0 \\ 0 & 0 & \frac{a}{a-L} \end{pmatrix} \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix}. \tag{6.51}$$

Interested readers should try to compute the kinematic relationships of a pantograph leg with a rotary joint as the third DOF.

6.3 Dynamics of Walking Robots

Robot dynamics state the relationship between robot motion and the forces involved therein. Specifically, the dynamic model of a robot manipulator finds mathematical relationships among:

1. Robot location and its derivatives, velocity and acceleration.
2. Forces and torques applied at the robot joints or end-effector.
3. Dimensional parameters of the robot manipulator, such as link length, mass and inertia.

Walking robots are very complex mechanical systems. The legs of a walking robot are connected to one another through the body and also through the ground, forming closed kinematic chains. Forces and moments propagate through the kinematic chain from one leg to another, and therefore dynamic coupling exists. The equations of motion of such a complex system with m legs each of n DOF are derived from d'Alembert's principle and are given in equation (6.52), where $\boldsymbol{\tau} \epsilon \mathbf{R}^{n \times m}$ is the vector of active joint torques, $\mathbf{F} \epsilon \mathbf{R}^{m \times 3}$ is the vector of ground-contact forces, $\mathbf{D} \epsilon \mathbf{R}^{n \times m+6, n \times m+6}$ denotes the mass matrix, $\mathbf{q} \epsilon \mathbf{R}^{n \times m+6}$ is the vector of generalized coordinates, $\mathbf{H} \epsilon \mathbf{R}^{n \times m+6}$ denotes Coriolis and centrifugal effects, $\mathbf{C} \epsilon \mathbf{R}^{n \times m+6}$ denotes the generalized influence of gravitation, and $\mathbf{J_M} \epsilon \mathbf{R}^{n \times m, n \times m+6}$ and $\mathbf{J_F} \epsilon \mathbf{R}^{n \times m, n \times m+6}$ project the torques and contact forces, respectively, to the space of the generalized coordinates $\mathbf{q}$:

$$\mathbf{J_M}^T \boldsymbol{\tau} - \mathbf{J_F}^T \mathbf{F} = \mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{H}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{C}(\mathbf{q}). \tag{6.52}$$

Computing the above equations of motion is extremely time-consuming and yields a system of $2 \times n \times m$ unknown variables ($\boldsymbol{\tau}$ and $\mathbf{F}$) with $n \times m + 6$ equations, which must be solved using some optimization criterion, *e.g.* optimal force distribution, employing the Lagrange-multipliers method (Bennani and Giri, 1996; Pfeiffer and Weidemann, 1991). To simplify the problem, some assumptions are usually made based on the property of the robots' legs, which are high-geared robotic systems. Using high reduction ratios in the joints of the legs makes it possible to neglect the coupling effect of Coriolis and centrifugal forces and therefore to decouple the dynamic Equation (6.52). However, high-geared mechanisms feature other specific dynamics, such as friction, backlash and elasticity (Garcia *et al.*, 2002; Pfeiffer and Rossmann, 2000; Shing, 1994; Spong, 1987). Therefore, if high reduction ratios are used in the robot joints, the dynamic equations can be decoupled, but other dynamic effects must be modeled instead.

A robotic leg can be studied from the dynamics point of view as a 3-DOF manipulator with a foot as end-effector. The dynamic model of a manipulator consists of the model of the mechanical part and the model of its actuators and transmission systems. The dynamic model of the mechanical part states the mathematical relationships between manipulator motion and the forces and torques causing it. On the other hand, the dynamic model of actuators and transmission systems finds relationships between control signals and forces and torques required for motion. We will derive the dynamic model of the actuators and the mechanical part of the legs of a walking robot separately in the following subsections.

6.3.1 Dynamic Model of the Mechanical Part

The mechanical part of a 3-DOF leg is the chain of serial links that conform the leg, excluding actuators and transmission systems. For deriving the dynamic

equations of the mechanical part of the leg, the Lagrange-Euler formulation has been chosen (Fu *et al.*, 1987). The direct application of the Lagrangian dynamics formulation together with the Denavit-Hartenberg link-coordinate representation results in a convenient, compact, systematic algorithmic description of the leg equations of motion. Although real-time computation of the Newton-Euler formulation (Fu *et al.*, 1987) is still more efficient than the Lagrange-Euler equations in open-loop control, the fact is that today's processors are fast enough to compute efficiently the 4×4 homogeneous transformation matrices of the Lagrangian formulation. The Lagrange-Euler formulation is a simple, secure method for deriving the mathematical expressions. Later analysis of the dynamic model of the leg will result in simplifications that ensure the real-time computation of the final equations of motion.

Systematic derivation of the Lagrange-Euler equations yields a dynamic expression that can be written in the form

$$\boldsymbol{\tau}_{\mathbf{e}} - \mathbf{J}^T\mathbf{F} = \mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{H}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{C}(\mathbf{q}). \tag{6.53}$$

where $\mathbf{D}(\mathbf{q})$ is the 3×3 mass matrix of the leg, $\mathbf{H}$ is a 3×1 vector of centrifugal and Coriolis terms, and $\mathbf{C}(\mathbf{q})$ is a 3×1 vector of gravity terms.

The mass matrix is symmetric, positive-defined, and configuration-dependent. Its diagonal elements d_{ii} are the inertia moments of the mechanical part around joint i when all the rest of the joints are blocked. The d_{ij} elements represent the effect of the acceleration of joint j on joint i.

The vector of centrifugal and Coriolis terms is configuration-dependent and rate-dependent. Its h_i elements are the sum of quadratic terms in joint speed and represent the effects induced on joint i due to the speed of the rest of the joints.

Lastly, the vector of gravity terms is configuration-dependent. The c_i terms show the moments around joint i caused by gravity.

The first term in (6.53) consists of torques and forces required for trajectory tracking, where $\boldsymbol{\tau}_{\mathbf{e}}$ is the 3×1 vector of active joint torques and $\mathbf{F}$ is the 3×1 vector of ground-contact forces. During the leg-transfer phase, there is no foot/terrain interaction, and $\mathbf{F}$ becomes zero. However, during the support phase, ground contact exists, and (6.53) becomes undetermined and should be solved in one of these ways:

- Using Lagrange multipliers to minimize some energy function (Dettman, 1988).
- Modeling foot/terrain interaction (Manko, 1992). The relationship between contact forces and foot positions is established, therefore adding to the number of equations required to solve (6.53).
- Using force sensors at the feet to measure $\mathbf{F}$ (Zhou *et al.*, 2000).

6.3.2 Dynamic Model of Actuators and Transmission Systems

Actuators and transmission systems play a relevant role in the computation of robot dynamics. Actuators are mainly of three types, electric, pneumatic,

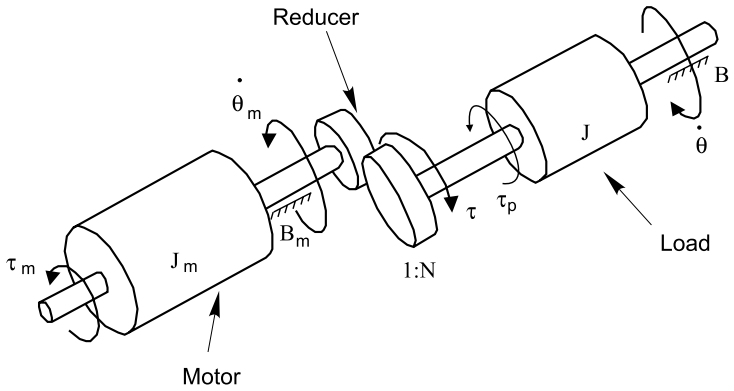


Fig. 6.5. Mechanical model of a DC torque motor connected through gearing to an inertial load

and hydraulic. Electric actuators are more precisely controlled, however, and thus are more widely used. In the operation of DC motors, inertia and friction arise during rotor spin that must be balanced through the motor torque. Also, if high reduction gearing is used, as in walking robots, friction, elasticity and backlash are introduced.

Here only DC motors will be considered, because they are the type of motor most usually found in the joints of legged robots. Dynamic models of hydraulic actuators can be found in Craig (1989) and Sciavicco and Siciliano (2000) while different models of pneumatic actuators are explained in Ogata (1996). Figure 6.5 shows the mechanical model of a DC torque motor connected through gear reduction to an inertial load. The torque applied to the rotor, τ_m, must balance both rotor and load inertias, which here we denote as equivalent inertia, J_{eq}. Likewise it must balance damping effects due to motor and load friction, which we denote as equivalent damping, B_{eq}, that is

$$\tau_m - \frac{1}{N}\tau_p = J_{eq}\ddot{\theta}_m + B_{eq}\dot{\theta}_m \tag{6.54}$$

where θ_m are actuator positions and N is the reduction coefficient. The equivalent inertia and damping are obtained from

$$J_{eq} = J_m + \frac{1}{N^2}J \tag{6.55}$$

$$B_{eq} = B_m + \frac{1}{N^2}B. \tag{6.56}$$

Transmission systems are another source of friction. Viscous friction is usually present in lubricated contacts; therefore this friction should be included in the equivalent damping term, B_{eq}. However, other friction components, like

Coulomb friction, can also exist. This friction component is responsible for energy losses in the transmission and can be modeled using the mechanical efficiency of the reducer, η. Therefore, Coulomb and viscous friction in the transmission system can be included in the dynamic model as follows:

$$J_{eq} = J_m + \frac{1}{N^2\eta} J \tag{6.57}$$

$$B_{eq} = B_m + \frac{1}{N^2\eta} B. \tag{6.58}$$

Nevertheless, this model of transmission-system friction might not be precise enough for every system. The model considers neither static friction nor meshing friction (Garcia *et al.*, 2002), which is especially dominant in high-geared systems. In such cases, a friction model is required. A complete friction model for high-geared robotic systems can be found in Garcia *et al.* (2002).

6.3.3 The Complete Dynamic Model

The dynamic model of the leg consists of the dynamic model of the mechanical part and the dynamic model of the actuators and transmission systems. Considering the mechanical part of the leg as a configuration-dependent load that the actuator torque must balance, the active actuator torques needed to move the mechanical part are taken from (6.53):

$$\boldsymbol{\tau}_{\mathbf{a}} = \mathbf{N}^{-1}\boldsymbol{\tau}_{\mathbf{e}} \tag{6.59}$$

where $\mathbf{N}$ is a 3×3 diagonal matrix of joint-reduction ratios. Actuator position, velocity and acceleration are related to joint position, velocity and acceleration:

$$\boldsymbol{\theta}_m = \mathbf{N}\mathbf{q}\,; \quad \dot{\boldsymbol{\theta}}_m = \mathbf{N}\dot{\mathbf{q}}\,; \quad \ddot{\boldsymbol{\theta}}_m = \mathbf{N}\ddot{\mathbf{q}} \tag{6.60}$$

where $\boldsymbol{\theta}_m$, $\dot{\boldsymbol{\theta}}_m$, and $\ddot{\boldsymbol{\theta}}_m$ are actuator position, velocity, and acceleration respectively, and $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$ are joint position, velocity, and acceleration respectively.

The mass matrix $\mathbf{D}$ can be written as the addition of two matrices:

$$\mathbf{D}(\mathbf{q}) = \mathbf{D_1} + \mathbf{D_2}(\mathbf{q}) \tag{6.61}$$

where $\mathbf{D_1}$ is the 3×3 diagonal matrix of the constant terms in $\mathbf{D}(\mathbf{q})$. Substituting (6.53), (6.60) and (6.61) in (6.59):

$$\boldsymbol{\tau}_{\mathbf{a}} = \mathbf{N}^{-1}\mathbf{D_1}\mathbf{N}^{-1}\ddot{\boldsymbol{\theta}}_m + \boldsymbol{\tau}_{\mathbf{p}} \tag{6.62}$$

where

$$\boldsymbol{\tau}_{\mathbf{p}} = \mathbf{N}^{-1}\mathbf{D_2}(\boldsymbol{\theta}_m)\mathbf{N}^{-1}\ddot{\boldsymbol{\theta}}_m + \mathbf{N}^{-1}\mathbf{H}(\boldsymbol{\theta}_m, \dot{\boldsymbol{\theta}}_m) + \mathbf{N}^{-1}\mathbf{C}(\boldsymbol{\theta}_m) + \mathbf{N}^{-1}\mathbf{J}^T\mathbf{F}. \tag{6.63}$$

Thus, the dynamics of the mechanical part of the leg can be considered as the sum of two terms: a constant inertia given by $\mathbf{J} = \mathbf{N}^{-1}\mathbf{D_1}\mathbf{N}^{-1}$ and a perturbation $\boldsymbol{\tau}_\mathbf{p}$ given by the variable terms in (6.53). Considering that the actuator torque must balance the leg dynamics, the dynamic model of the actuators can be expressed in a matrix form:

$$\boldsymbol{\tau}_\mathbf{m} = \mathbf{J_{eq}}\ddot{\boldsymbol{\theta}}_m + \mathbf{B_{eq}}\dot{\boldsymbol{\theta}}_m + \boldsymbol{\tau}_\mathbf{p} + \boldsymbol{\tau}_\mathbf{F} \tag{6.64}$$

where $\boldsymbol{\tau}_\mathbf{m}$ is a 3×1 vector of actuator torques. The equivalent inertia and damping are obtained from

$$\mathbf{J_{eq}} = \mathbf{J_m} + \mathbf{N}^{-1}\mathbf{D_1}\mathbf{N}^{-1} \tag{6.65}$$

$$\mathbf{B_{eq}} = \mathbf{B_m} + \mathbf{F_v} \tag{6.66}$$

where $\mathbf{J_m}$ is a 3×3 diagonal matrix whose element $J_{m_{ii}}$ is the rotor inertia of actuator i. Likewise, $\mathbf{B_m}$ is the 3×3 diagonal matrix whose element $B_{m_{ii}}$ is the coefficient of viscous friction of actuator i. $\mathbf{F_v}$ is the 3×3 diagonal matrix whose element $F_{v_{ii}}$ is the coefficient of viscous friction in the transmission system of joint i. $\boldsymbol{\tau}_\mathbf{F}$ is the 3×1 vector of the dynamic model of friction in each joint, excluding the viscous friction, $F_{v_{ii}}$.

Equations (6.63) to (6.66) complete the dynamic model of a robot leg or manipulator. Notice that τ_p and τ_F are non-linear terms. Also the terms in τ_p are coupled between joints. Only if high reduction is used could this term be neglected and the model in (6.64), decoupled. However, in such cases the τ_F term will increase its relevance, and the model will become more complex. Therefore it is necessary to study and analyze the model to obtain a precise simplified expression.

6.4 A Method for Dynamic Model Analysis

The inherent complexity of the dynamic model of a robot leg usually converges to an improper model simplification to enable real-time motion control. As a result, control will become imprecise due to a careless simplification procedure.

To enable an accurate simplification of the dynamic model, a method for dynamic analysis is here proposed consisting of four steps.

Step 1: Computation of each term in dynamic equation (6.64) during real robot trajectories covering the whole workspace.

Step 2: Analysis of the torque contribution of each computed term.

Step 3: If the torque contribution of a term in the model is less than 5% for every trajectory, then that term is considered non-significant and can be neglected.

Step 4: The remaining terms reflect the relevant dynamics. Then the evolution of these significant terms during different trajectories is studied as follows:

- Evolution of torque contributions as a function of end-effector position.
- Evolution of torque contributions as a function of linear-trajectory speed.
- Evolution of torque contributions as a function of linear-trajectory acceleration.

As a result of the analysis, the variation of the relevant robot dynamics during real robot tasks is identified. The relationship between robot dynamics and trajectory parameters can be used in a real-time control algorithm or in a maximum-speed trajectory generation algorithm (see Chap. 7).

6.5 Application to the SILO4 Walking Robot

This section is aimed at showing the usage of the above dynamic-model analysis method. For this purpose, the SILO4 quadruped described in Appendix A is used again to test the performance of the analysis technique.

6.5.1 Dynamic Model of the Mechanical Part

The Lagrange-Euler formulation was used to derive the dynamic equations of the mechanical part of the SILO4 leg. Direct application of the Lagrangian dynamics formulation together with the Denavit-Hartenberg link coordinate representation resulted in a convenient, compact, systematic algorithmic description of the SILO4 leg's equations of motion. Table 6.2 lists all the dynamic parameters of the SILO4 leg used for the derivation of the dynamic equations of motion. Accurate values of inertial moments and centre-of-mass positions were computed using Pro/ENGINEER mechanical design software (Lamit, 2001). Mass values were checked experimentally.

Systematic derivation of the Lagrange-Euler equations yielded dynamic equation (6.53) for the mechanical part of the leg. Matrices D, H and C for the SILO4 leg are presented in the following paragraphs. The Maple V software package was used for symbolic simplification of the results (Monagan *et al.*, 1998).

Mass Matrix for the SILO4 Leg (D)

The mass matrix is a 3×3 matrix containing inertia forces between two links of the leg. The general form of this matrix is

$$\mathbf{D} = \begin{pmatrix} D_{11} & D_{12} & D_{13} \\ D_{21} & D_{22} & D_{23} \\ D_{31} & D_{32} & D_{33} \end{pmatrix}. \tag{6.67}$$

Table 6.2. Dynamic parameters of the SILO4 leg referred to Denavit-Hartenberg link coordinate representation

Link parameter		Link 1	Link 2	Link 3 + foot
Mass (kg)		1.22	1.26	0.63
Length (m)		0.06	0.24	0.24
Position of the c.o.m. (10^{-3} m)	x_{cm}	−12.2	−109.4	−84.5
	y_{cm}	101.0	11.4	−2.5
	z_{cm}	0.4	−0.8	3.9
Inertia tensor (10^{-3} kg m^2)	I_{xx}	18.2	0.6	0.3
	I_{xy}	1.7	1.8	−0.01
	I_{xz}	0.002	−0.17	0.17
	I_{yy}	0.6	22.4	10.8
	I_{yz}	−0.03	0.01	0.0
	I_{zz}	18.4	22.5	10.8

The contribution of every term of each element of this matrix has been analyzed for different foot trajectories, and finally non-significant terms, whose contribution is less than 10^{-4}, have been omitted. Thus, after these mathematical simplifications, each element of the mass matrix has the following final form:

$$\begin{aligned} D_{11} &= aC_2 + bS_2 + cC_3 + dC_{23} + c\,\cos(q_3 + 2q_2) \\ &\quad + e\,\sin(2q_2) + f\,\cos(2q_2) + g\,\cos(2q_3 + 2q_2) + h \\ D_{12} &= 0 \\ D_{13} &= 0 \\ D_{22} &= kC_3 + l \\ D_{23} &= cC_3 + m \\ D_{33} &= m \end{aligned} \tag{6.68}$$

where $S_i = \sin(q_i)$, $C_i = \cos(q_i)$, $S_{ij} = \sin(q_i + q_j)$, and $C_{ij} = \cos(q_i + q_j)$. The mass matrix $\mathbf{D}$ is usually separated into two matrices:

$$\mathbf{D} = \mathbf{D_1} + \mathbf{D_2} \tag{6.69}$$

where $\mathbf{D_1}$ is a constant, diagonal matrix whose elements are the constant terms of the diagonal of matrix $\mathbf{D}$:

$$\mathbf{D_1} = \begin{pmatrix} h & 0 & 0 \\ 0 & l & 0 \\ 0 & 0 & m \end{pmatrix} \tag{6.70}$$

and $\mathbf{D_2}$ is obtained from (6.69). Constants a to m are listed in Table 6.3.

Table 6.3. Constant values in SI units for the dynamic model of the SILO4 leg

a	0.0376	h	0.0532	r	0.00527
b	−0.00173	k	0.0462	s	0.00581
c	0.0231	l	0.0856	t	0.0115
d	0.0116	m	0.0213	u	3.077
e	−0.00528	n	−0.0635	v	−0.142
f	0.0317	p	−0.0210	w	0.951
g	0.0105	q	0.0188	x	0.0152

Vector of Centrifugal and Coriolis Terms (H)

The vector of centrifugal and Coriolis terms is of the form

$$\mathbf{H} = (h_1 \; h_2 \; h_3)^T \tag{6.71}$$

where, after analysis and simplification, each element results as

$$\begin{aligned} h_1 &= h_{112}\dot{q}_1\dot{q}_2 + h_{113}\dot{q}_1\dot{q}_3 \\ h_2 &= h_{211}\dot{q}_1^2 + h_{223}\dot{q}_2\dot{q}_3 + h_{233}\dot{q}_3^2 \\ h3 &= h_{311}\dot{q}_1^2 + h_{322}\dot{q}_2^2 \end{aligned} \tag{6.72}$$

where

$$\begin{aligned} h_{112} &= -aS_2 + n\,\sin(2q_2) - g\,\cos(2q_2) \\ &\quad -dS_{23} - k\,\sin(2q_2 + q_3) + p\,\sin(2q_2 + 2q_3) \\ h_{113} &= -cS_3 - dS_{23} - c\,\sin(2q_2 + q_3) + p\sin(2q_2 + 2q_3) \\ h_{211} &= qS_2 + f\,\sin(2q_2) + r\,\cos(2q_2) \\ &\quad +sS_{23} + c\,\sin(2q_2 + q_3) + g\,\sin(2q_2 + 2q_3) \\ h_{223} &= -kS_3 \\ h_{233} &= -cS_3 \\ h_{311} &= tS_3 + sS_{23} + t\,\sin(2q_2 + q_3) + g\,\sin(2q_2 + 2q_3) \\ h_{322} &= cS_3. \end{aligned} \tag{6.73}$$

Constants a to t are listed in Table 6.3.

Vector of Gravity Terms (G)

The vector of gravity terms is of the form

$$C = (g_1 \; g_2 \; g_3)^T \tag{6.74}$$

where, after numerical simplification:

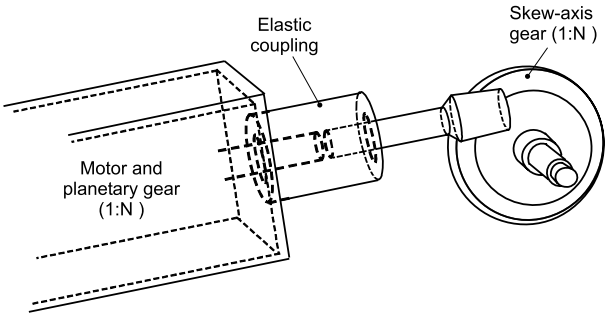


Fig. 6.6. Transmission and gearing of 2nd and 3rd joints of the SILO4 leg

$$\begin{aligned} g_1 &= 0 \\ g_2 &= uC_2 + vS_2 + wC_{23} \\ g_3 &= wC_{23} + xS_{23}. \end{aligned} \tag{6.75}$$

Constants u to x are listed in Table 6.3.

6.5.2 Dynamic Model of the Actuators

The actuators of the SILO4 leg are three low-inertia DC motors, located at each joint and connected through gear reduction to the load. The first joint actuator is connected through a planetary gear; joints 2 and 3, however, have a planetary gear plus a skew-axis gear (see Fig. 6.6). Thus, the first joint-motor assembly will match the model in Fig. 6.5, while the joint-motor assemblies of joints 2 and 3 have two gear stages and thus will have a more complex model. If we want to achieve an accurate model of these actuators, we should bear in mind that they are non-ideal actuators. Each gear stage has torque losses due to Coulomb, viscous, and meshing friction, included in the dynamic model of the leg using the friction model proposed in Garcia *et al.* (2002), which is

$$\begin{aligned} \tau_{F_i} = [\tau_C + (\tau_E - \tau_C)e^{-|\dot{\theta}_{m_i}|/\dot{\theta}_S} + A_1 \sin(\omega_1 \theta_{m_i} + \phi_1) \\ + A_2 e^{-\beta|\dot{\theta}_{m_i}|} \sin(\omega_2 \theta_{m_i} + \phi_2)] sign(\dot{\theta}_{m_i}) \end{aligned} \tag{6.76}$$

where subindex i denotes the joint number. This friction model includes a static-friction value, τ_E, a Coulomb-friction value, τ_C, a Stribeck effect, represented by the Stribeck velocity, $\dot{\theta}_S$, a position-dependent friction of amplitude A_1 and frequency ω_1, and a meshing-friction component of variable amplitude and frequency ω_2. The viscous friction in the transmission, F_{vi}, has been included in the equivalent damping term of the actuator. These friction parameters have been identified for the SILO4 leg and are shown in Tables 6.4–6.6. Parameter identification has been carried out by the least square

Table 6.4. Friction parameters identified in the first joint of the SILO4 leg

Rotation	τ_E (10^{-3} Nm)	τ_C (10^{-3} Nm)	$\dot{\theta}_S$ (rpm-motor)	B (10^{-3} Nm/rpm)
Positive	3.019	2.97	38.4	0.00264
Negative	3.019	2.97	38.4	0.00264
	A_1 (10^{-3} Nm)	ω_1 (rad/s)	ϕ_1 (rad)	
Positive	0	3.5×10^{-3}	0	
Negative	0	3.5×10^{-3}	0	
	A_2 (10^{-3} Nm)	β_2 (rpm^{-1})	ω_2 (rad/s)	ϕ_2 (rad)
Positive	0.21	6.5×10^{-8}	1	0.03
Negative	0.23	5.7×10^{-8}	1	0.03

Table 6.5. Friction parameters identified in the second joint of the SILO4 leg

Rotation	τ_E (10^{-3} Nm)	τ_C (10^{-3} Nm)	$\dot{\theta}_S$ (rpm-motor)	B (10^{-3} Nm/rpm)
Positive	34.91	34.48	5691	0.00123
Negative	34.99	34.53	5702	0.00086
	A_1 (10^{-3} Nm)	ω_1 (rad/s)	ϕ_1 (rad)	
Positive	2.50	3.5×10^{-3}	0.2	
Negative	2.50	3.5×10^{-3}	0.2	
	A_2 (10^{-3} Nm)	β_2 (rpm^{-1})	ω_2 (rad/s)	ϕ_2 (rad)
Positive	1.02	3.1×10^{-11}	0.071	1.2
Negative	1.40	1.2×10^{-11}	0.071	1.2
	A_3 (10^{-3} Nm)	β_3 (rpm^{-1})	ω_3 (rad/s)	ϕ_3 (rad)
Positive	0.25	5.8×10^{-15}	1	$-\pi/3$
Negative	0.23	0.9×10^{-15}	1	$-\pi/3$

method, and it is detailed in Garcia *et al.* (2002). Then let us name the rotor inertia and damping for joint i J_{mi} and B_{mi} respectively, and let us also name the inertia and damping of the elastic coupling element between the planetary gear and the skew-axis gear J_{ei} and B_{ei} respectively. The torque balance of (6.64) for the three joint-motor assemblies of the leg is as follows:

$$\tau_{m1} - \tau_{p1} - \tau_{F1} = (J_{m1} + N_{p1}^{-2} h)\ddot{\theta}_{m1} + (B_{m1} + F_{v1})\dot{\theta}_{m1} \tag{6.77}$$

Table 6.6. Friction parameters identified in the third joint of the SILO4 leg

Rotation	τ_E (10^{-3} Nm)	τ_C (10^{-3} Nm)	$\dot{\theta}_S$ (rpm-motor)	B (10^{-3} Nm/rpm)
Positive	8.58	7.106	28.18	0.0134
Negative	9.41	7.909	26.58	0.0138
	A_1 (10^{-3} Nm)	ω_1 (rad/s)	ϕ_1 (rad)	
Positive	0.3	3.5×10^{-3}	$\pi/2$	
Negative	0.3	3.5×10^{-3}	$\pi/2$	
	A_2 (10^{-3} Nm)	β_2 (rpm^{-1})	ω_2 (rad/s)	ϕ_2 (rad)
Positive	0.576	1.27×10^{-4}	0.071	$\pi/2$
Negative	0.526	1.12×10^{-4}	0.071	$\pi/2$
	A_3 (10^{-3} Nm)	β_3 (rpm^{-1})	ω_3 (rad/s)	ϕ_3 (rad)
Positive	2.99	5.28×10^{-5}	1	π
Negative	3.27	0.75×10^{-5}	1	π

Table 6.7. Actuator parameters

		Actuator 1	Actuator 2	Actuator 3
J_m (10^{-6} kgm^2)		2.3	6.4	4.9
B_m (10^{-4} Nm/rad/s)		1.77	9.14	3.0
R (Ω)		10.5	2.0	5.5
L (10^{-3} H)		0.94	0.27	0.85
K_M (10^{-3} Nm/A)		46.81	42.88	41.05
K_E (V/rad/s)		0.039	0.043	0.041
Planetary gear	N_p	246	14	14
	η_p (%)	60	80	80
Skew-axis gear	N_s		20.5	20.5
	η_s (%)		70	70
B_e			0.0	0.0
J_e (10^{-6} kgm^2)			6.5	6.5

$$\tau_{m2} - \tau_{p2} - \tau_{F2} = (J_{m2} + N_{p2}^{-2} J_{e2} + N_{p2}^{-2} N_{s2}^{-2} l)\ddot{\theta}_{m2} + (B_{m2} + N_{p2}^{-2} B_{e2} + F_{v2})\dot{\theta}_{m2} \quad (6.78)$$

$$\tau_{m3} - \tau_{p3} - \tau_{F3} = (J_{m3} + N_{p3}^{-2} J_{e3} + N_{p3}^{-2} N_{s3}^{-2} m)\ddot{\theta}_{m3} + (B_{m3} + N_{p3}^{-2} B_{e3} + F_{v3})\dot{\theta}_{m3}. \quad (6.79)$$

Actuator dynamic parameters are listed in Table 6.7. Perturbation torques τ_{p1}, τ_{p2}, and τ_{p3} are the torques required in joints 1, 2, and 3 of the leg respectively to follow a given trajectory and are obtained from (6.63):

$$\boldsymbol{\tau}_{\mathbf{p}} = \mathbf{N}^{-1}\mathbf{D}_{\mathbf{2}}(\boldsymbol{\theta}_m)\mathbf{N}^{-1}\ddot{\boldsymbol{\theta}}_m + \mathbf{N}^{-1}\mathbf{H}(\boldsymbol{\theta}_m, \dot{\boldsymbol{\theta}}_m) + \mathbf{N}^{-1}\mathbf{C}(\boldsymbol{\theta}_m) + \mathbf{N}^{-1}\mathbf{J}^T\mathbf{F} \quad (6.80)$$

where

$$\boldsymbol{\tau}_{\mathbf{p}} = (\tau_{p1}\ \tau_{p2}\ \tau_{p3})^T \quad (6.81)$$

$$\mathbf{N} = \begin{pmatrix} N_{p1} & 0 & 0 \\ 0 & N_{p2}N_{s2} & 0 \\ 0 & 0 & N_{p3}N_{s3} \end{pmatrix}. \quad (6.82)$$

6.5.3 Model Analysis

Once the mathematical model of the SILO4 leg has been obtained, the model is analyzed. First, a prototype of the SILO4 leg (shown in Fig. 6.7) is used to analyze the torque contributions of the mechanical part during real leg trajectories, and later the torque contributions of the actuator dynamics are compared. It is well known that accurate and efficient model analysis requires the chosen trajectories to be sufficiently exciting. The trajectories chosen for the analysis of the SILO4 leg model have the highest acceleration to provide sufficiently dynamic excitement. The algorithm used for generation of such improved trajectories is explained in detail in Chap. 7. Then the experiments are carried out to obtain the torque contributions of the mechanical part of the leg and its actuators while each joint is PID controlled.

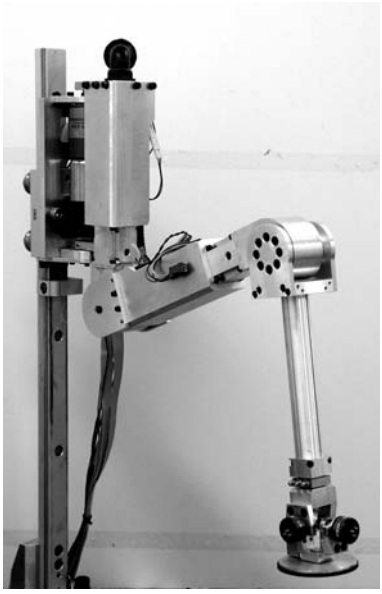


Fig. 6.7. Prototype of the SILO4 leg used for the experiments

Torque Contribution of the Mechanical Part

To analyze the dynamics of the mechanical part of the leg, the torque contributions of each term in the mathematical model are compared during real leg-transfer trajectories. Fig. 6.8 shows the torque contributions corresponding to the four terms in the model of the mechanical part, which are

$$\begin{aligned} \boldsymbol{\tau}_{\mathbf{D1}} &= \mathbf{D_1}(\mathbf{q})\ddot{\mathbf{q}} \\ \boldsymbol{\tau}_{\mathbf{D2}} &= \mathbf{D_2}(\mathbf{q})\ddot{\mathbf{q}} \\ \boldsymbol{\tau}_{\mathbf{H}} &= \mathbf{H}(\mathbf{q}, \dot{\mathbf{q}}) \\ \boldsymbol{\tau}_{\mathbf{C}} &= \mathbf{C}(\mathbf{q}) \end{aligned} \tag{6.83}$$

where the total torque contribution of the mechanical part of the leg is

$$\boldsymbol{\tau}_{\mathbf{e}} = \boldsymbol{\tau}_{\mathbf{D1}} + \boldsymbol{\tau}_{\mathbf{D2}} + \boldsymbol{\tau}_{\mathbf{H}} + \boldsymbol{\tau}_{\mathbf{C}}. \tag{6.84}$$

The terms that contribute to the perturbation torque, τ_p, are

$$\boldsymbol{\tau}_{\mathbf{p}} = \boldsymbol{\tau}_{\mathbf{D2}} + \boldsymbol{\tau}_{\mathbf{H}} + \boldsymbol{\tau}_{\mathbf{C}} \tag{6.85}$$

and the term $\boldsymbol{\tau}_{\mathbf{D1}}$ contributes to the actuator equivalent inertia. The time evolution of the torque contribution of this term along a given leg trajectory

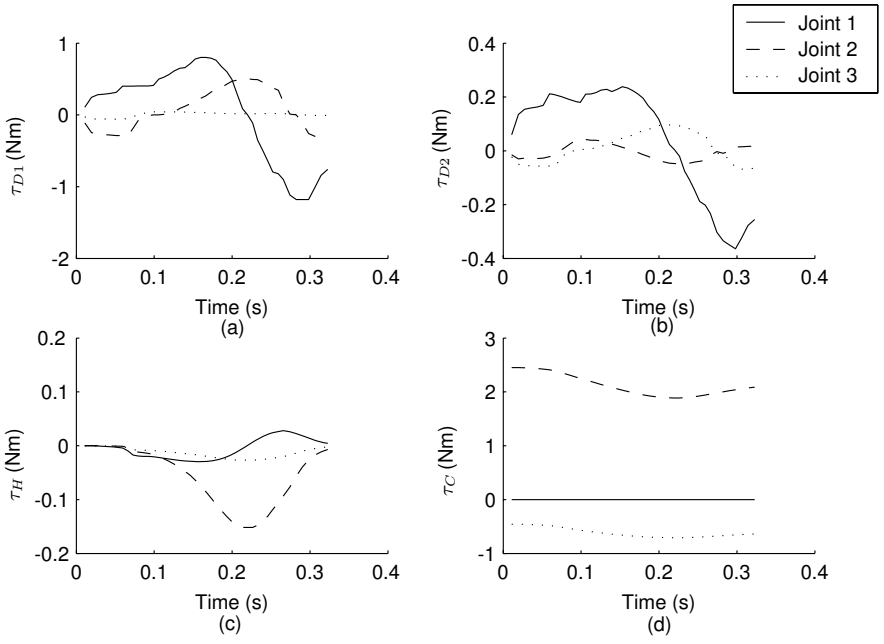


Fig. 6.8. Torque contributions of the mechanical part of the SILO4 leg

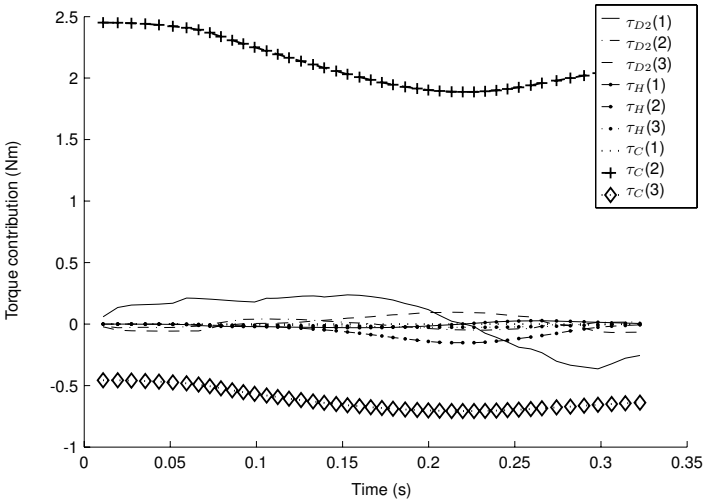


Fig. 6.9. Numerical comparison of torque contributions of the mechanical part of the SILO4 leg

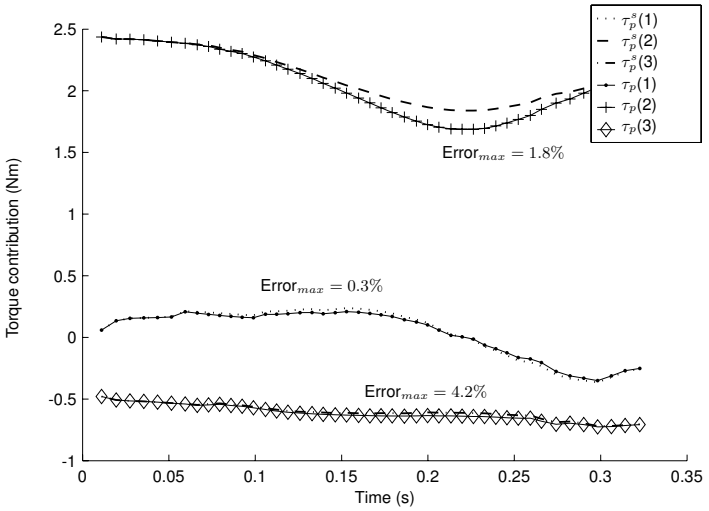


Fig. 6.10. Perturbation torque (τ_p) and simplified perturbation torque (τ_p^s) for the dynamic model of the SILO4 leg

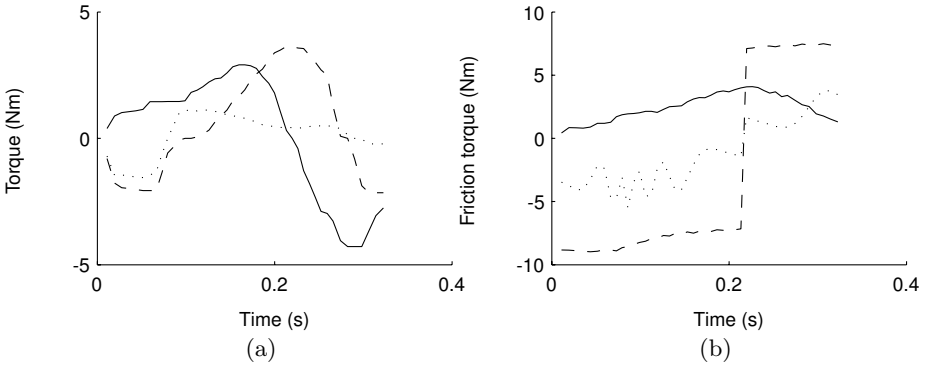


Fig. 6.11. (**a**) Torque contribution of actuator equivalent inertia; (**b**) Torque contribution of friction in actuator-transmission system. Joint 1 in *solid line*, joint 2 in *dashed line*, joint 3 in *dotted line*.

is represented in Fig. 6.8(a), where each line shows the torque required in each joint to move the constant inertia of its own link (vector τ_{D_1}). Likewise, Fig. 6.8(b) shows the time evolution of the torque contribution of non-constant and non-diagonal inertia terms (vector τ_{D_2}), Fig. 6.8(c) shows the torque contribution of Coriolis and centrifugal effects (vector τ_H), and Fig. 6.8(d) shows the torque contribution of gravitational effects (vector τ_C).

Detailed numerical comparison of the contributions of the four terms in the dynamic model shows that Coriolis and centrifugal effects play a very little role in leg dynamics (see Fig. 6.8(c)). This can be observed more clearly in Fig. 6.9, where all the torque contributions have been plotted together. Now it is clear that $\boldsymbol{\tau_H}$ can be neglected. The significance of gravitational terms in the motion of joints 2 and 3 is visible, as well as the relevance of constant inertia in joint 1. Figure 6.10 shows the total perturbation torque of each joint τ_{p_i} in (6.85) and compares it with the simplified perturbation torque obtained by extracting $\boldsymbol{\tau_H}$ from (6.85), that is

$$\boldsymbol{\tau_p}^s = \boldsymbol{\tau_{D2}} + \boldsymbol{\tau_C}. \tag{6.86}$$

Figure 6.10 also shows the maximum error made in the simplification, which is 4.2% in the worst case.

To conclude the analysis of the mechanical part of the leg, we state that the relevant dynamics affecting the first joint of the SILO4 leg are inertias, while the dynamics that mainly affect joints 2 and 3 are gravitational effects. Inertias play a secondary role in these two joints.

Torque Contribution of Actuators and Transmission Systems

Figure 6.11 shows torque contributions due to actuator equivalent inertia and friction during a given leg trajectory. Actuator equivalent inertia includes the

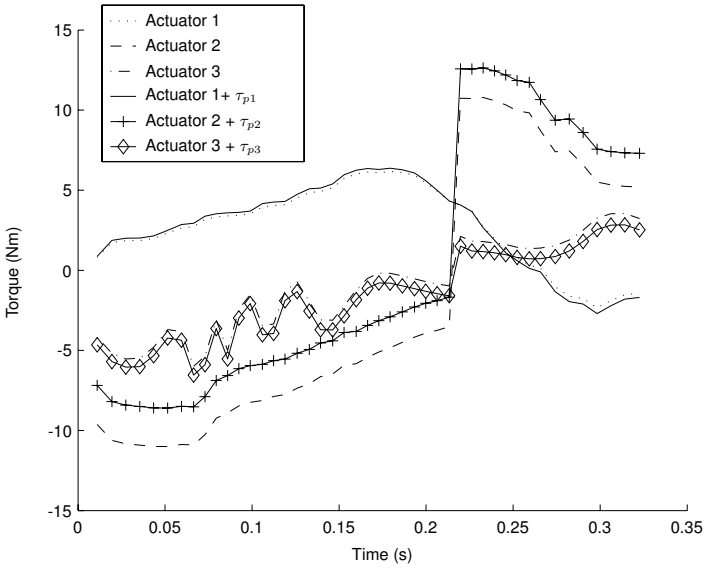


Fig. 6.12. Torque contribution of the actuator model and effect of the perturbation due to the mechanical part of the SILO4 leg

effect of constant inertia of the mechanical part of the leg, $\mathbf{D_1}$, as in (6.77)–(6.79). The friction torque has been computed using (6.76) during real leg trajectories. The comparison of the two figures allows us to state that friction in joints 2 and 3 is twice the inertial contribution. In the first joint, however, inertia is more relevant than friction, yet in any case friction is never negligible. Therefore, the initial guess that friction in high-geared robotic systems is relevant enough to hamper model simplification is here verified. In fact, friction dominates the dynamics of actuators 2 and 3. Therefore, simplification of the dynamic model of walking robots assuming that no friction exists will surely yield significant errors during motion control.

Figure 6.12 is intended to show the perturbing effect of the mechanical part of the leg on actuator dynamics. For the example trajectory, this perturbation, $\boldsymbol{\tau_p}$, can be considered as constant for actuators 2 and 3. However, the perturbation in actuator 1 is not constant, due to the variable inertia $\mathbf{D_2}$, although it is almost negligible. Figure 6.13(a) shows the maximum error of neglecting the perturbation in actuator 1 relative to the inertial torque in the actuator for different leg trajectories. The figure shows that this error is always less than 0.5%.

Let us define the **horizontal lengthening of the leg,** R_h, as the horizontal projection of the distance from the end-effector to the origin of the leg reference frame (see Fig. 6.14). For the SILO4 leg this becomes

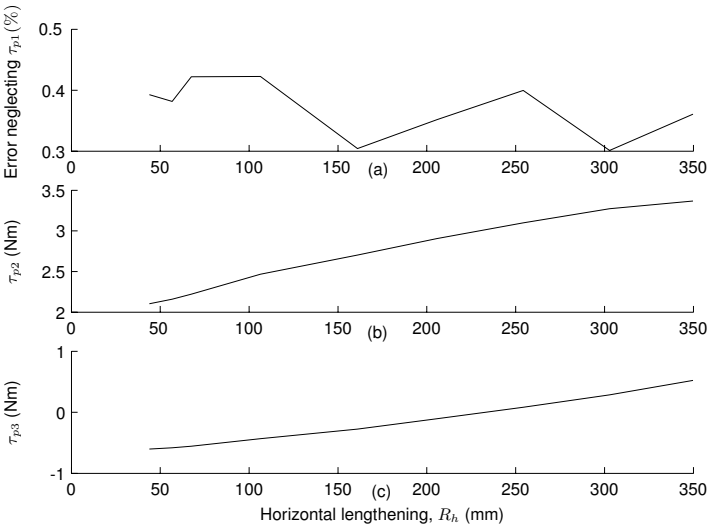


Fig. 6.13. Perturbation torques due to the mechanical part of the SILO4 leg for trajectories of different horizontal length: (**a**) maximum error when neglecting τ_{p1}; (**b**) evolution of τ_{p2} when increasing the horizontal lengthening; (**c**) evolution of τ_{p3} when increasing the horizontal lengthening

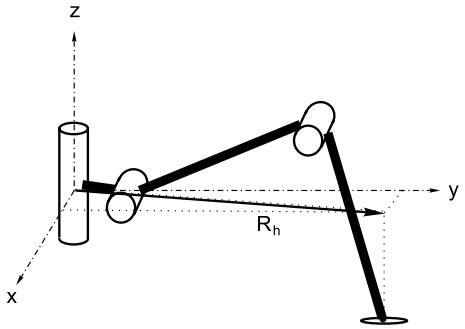


Fig. 6.14. Horizontal lengthening of a robot leg

$$R_h = a_3 \cos(q_2 + q_3) + a_2 \cos(q_2). \tag{6.87}$$

The dominant perturbation of the dynamics of joints 2 and 3 is due to gravity, and this effect increases as the leg stretches horizontally, as shown in Fig. 6.13(b),(c).

This analysis of the dynamic model of the SILO4 leg concludes that the effect of the dynamics of the mechanical part on the first joint can be neglected. On the other hand, the perturbation of the mechanical part on joints 2 and 3 varies with the horizontal lengthening of the leg. Based on this consideration,

the model can be simplified without losing accuracy. Then the simplified perturbation torques on the three actuators due to the mechanical part of the leg, τ^s_{p1}, τ^s_{p2}, and τ^s_{p3}, are given by

$$\tau^s_{p1} = 0 \tag{6.88}$$

$$\tau^s_{p2} = m_2 R_h(q_2, q_3) + b_2 \tag{6.89}$$

$$\tau^s_{p3} = m_3 R_h(q_2, q_3) + b_3 \tag{6.90}$$

where the relationships between τ^s_{p2}, τ^s_{p3}, and R_h have been linearized.

The analysis of the dynamic model of the SILO4 leg has produced a simplified model where joints 2 and 3 are coupled while joint 1 is independent. This analysis enables any model-based control strategy to be employed more efficiently. As an example, the results of the model analysis will be used for controlling high-speed trajectories of the SILO4 leg in the next chapter of this book.

6.6 Conclusions

Many authors recommend not taking leg dynamics into account in the control of walking robots. The high gearing employed is often the reason for neglecting the effect of leg dynamics on trajectory control. However, the use of a gear reduction high enough to ignore leg dynamics implies a significant increase in backlash, friction and elasticity in the transmission system. These undesired additional effects are much more difficult to model than the dynamics of the mechanical part. One main conclusion of this chapter is that considering robot legs as massless systems is not always the best option. The effect of leg dynamics can be appreciable, and moreover it can be used to improve the control system. To ilustrate this, we have derived a precise, accurate model of a robotic leg. Experiments have been done using a real leg prototype to analyze the torque contributions of different dynamic components during real leg trajectories. Detailed analysis of leg dynamics has led us to a simplified, accurate model of the dynamic effect of the leg on motion control.

7

Improving Leg Speed by Soft Computing Techniques

7.1 Introduction

In order to accomplish successful real applications of walking robots, the current performance of walking machines needs to be improved. Detractors of legged robots usually point out the machine speed as one of the major shortcomings of these vehicles. A walking machine's leg must be designed to work under worse load condition than a manipulator, with a weaker mechanical structure and achieving good accuracy. For instance, a given commercial manipulator designed to carry a 3-kg payload weighs about 50 kg and exhibits a strong structure. However, a leg for the SILO4 walking robot, used for experiments in this book, weighs 4 kg and must support, for some foot positions, up to 15 kg (half the machine's weight). Normally, the big payloads of legs are carried using strong gearing that increases output torque at the price of speed; thus legs become slower than manipulators. Therefore, leg velocity is a feature that must be improved by means of approaching the drive-speed limits at the time of trajectory generation.

Walking robots that are designed to work outdoors, on natural ground, usually need to negotiate obstacles like rocks or trees that can interrupt a leg's transfer trajectory. To manage this inconvenience, leg trajectories are generated on-line, to avoid the computational burden of repeatedly generating a complete trajectory each time an obstacle has to be avoided. Therefore, improving leg speed in on-line trajectory generation is required. This chapter is focused on improving leg velocity for on-line trajectory generation. Section 7.2 states the problem using some real examples and enumerates some methods. The problem is solved by a fuzzy inference system in Sect. 7.3 and finally, experimental results are reported in Sect. 7.4.

7.2 Improving Leg Speed in On-line Trajectory Generation

Every robotic leg of n rotary joints exhibits torque constraints that might make it imposible to obtain high-speed cartesian trajectories of the foot based on the syncronization of angular joint speeds.

Figure 7.1 shows an outline of a 3-DOF leg in the initial and final positions of a straight trajectory $\bar{IF}$ of the foot. Let us define a plane formed by the trajectory $\bar{IF}$ and the origin of the leg reference frame, O. Let us label it plane $I\hat{O}F$. Trajectory $\bar{IF}$ could also be executed by using a virtual 2-DOF leg, with one rotational joint placed at the origin O, orthogonal to plane $I\hat{O}F$, and a telescopic joint extending the length of the leg a distance $R(t)$. Thus, the Cartesian motion of the foot, referred to the leg's reference frame $x_l\, y_l\, z_l$, and expressed in terms of variables $R(t)$ and $\theta(t)$ is given by

$$x_l(t) = R(t)\, \cos\theta(t) \tag{7.1}$$

$$y_l(t) = R(t)\, \sin\theta(t). \tag{7.2}$$

From (7.1) and (7.2) the Cartesian components of the foot speed are

$$\dot{x}_l(t) = \dot{R}(t)\, \cos\theta(t) - \dot{\theta}(t)\, R(t)\, \sin\theta(t) \tag{7.3}$$

$$\dot{y}_l(t) = \dot{R}(t)\, \sin\theta(t) + \dot{\theta}(t)\, R(t)\, \cos\theta(t). \tag{7.4}$$

The linear speed of the foot is obtained from (7.3) and (7.4):

$$v = \sqrt{\dot{R}(t)^2 + R(t)^2\, \dot{\theta}(t)^2}. \tag{7.5}$$

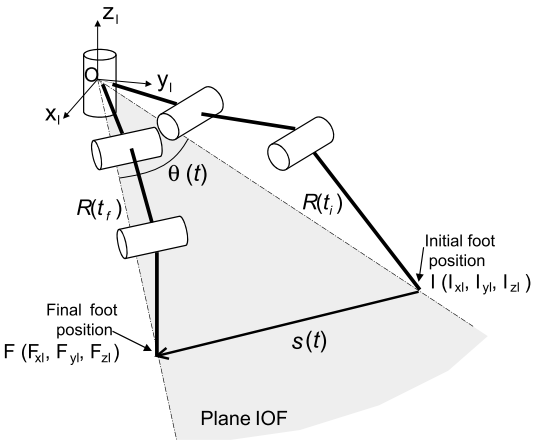


Fig. 7.1. Foot trajectory and plane of motion

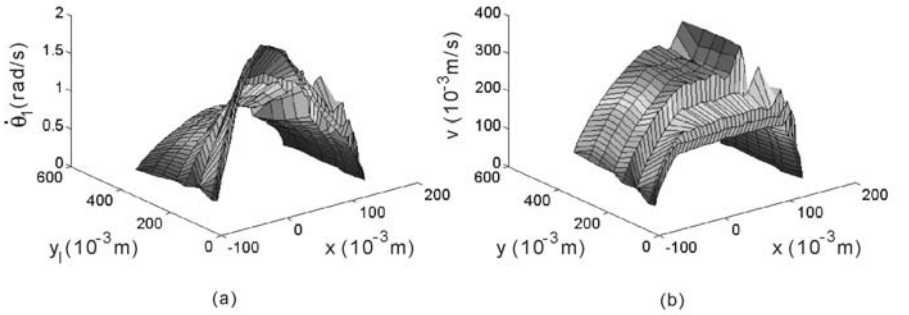


Fig. 7.2. Foot trajectories parallel to the leg $x_l\, y_l$ plane: (**a**) first angular joint speed profile evolution; (**b**) foot velocity profile evolution

Based on (7.5), maximizing v requires maximizing $\dot{\theta}(t)$, $\dot{R}(t)$ and $R(t)$. Maximizing $R(t)$ will not be always possible, depending on the task. Maximizing $\dot{\theta}(t)$ and $\dot{R}(t)$ is limited by maximum joint speeds, because both $\dot{\theta}(t)$ and $\dot{R}(t)$ are defined in terms of joint angles and joint speeds of the real 3-DOF leg, $\dot{\theta}_i(t)$, where $i = 1, ..., n$:

$$\dot{\theta}(t) = \mathcal{F}(\dot{\theta}_1(t), \ldots, \dot{\theta}_n(t), \theta_1(t), \ldots, \theta_n(t)) \tag{7.6}$$

$$\dot{R}(t) = \mathcal{G}(\dot{\theta}_1(t), \ldots, \dot{\theta}_n(t), \theta_1(t), \ldots, \theta_n(t)). \tag{7.7}$$

To illustrate this explanation, Fig. 7.2 shows how angular joint speeds affect the average foot speed in an articulated leg such as the SILO4 leg example. Trapezoidal foot-speed profiles have been used for straight-line trajectory generation and are shown in Fig. 7.2(b) for trajectories that are parallel to the leg's x axis, for different values of the y coordinate. Figure 7.2(a) shows the resultant angular-speed profiles of the first joint of the SILO4 leg, which is the joint that first reaches its speed limit for the same trajectories. This figure shows that joint speed reaches its maximum driving speed, given by maximum actuator torque, for trajectories closer to the leg's origin O, that is, for short values of $R(t)$, and this prevents the desired average foot speed from being reached, as shown in Fig. 7.2(b). In contrast, the angular speed required for achieving the same average foot speed decreases for trajectories that are far from O, that is, with higher values of $R(t)$. Therefore, foot speed for higher values of $R(t)$ could be increased until the actuator of the first joint reaches its speed limit.

This experiment illustrates that improving the linear foot speed requires the modification of the foot-velocity profile to adjust to each desired trajectory, characterized by the workspace of the leg. The modification of the foot-velocity profile will be based on trying to reach the speed limit of at least one actuator

to maximize $\dot{\theta}(t)$ and $\dot{R}(t)$. Solutions to this problem come from minimum-time control techniques (Bobrow *et al.*, 1985; Shin and McKay, 1985; Yang and Slotine, 1994). However, these methods are not applicable in on-line trajectory generation due to its computational burden. Moreover, the minimum-time control theory assumes that a perfectly accurate model of the leg is available and that there are no external disturbances. In practice, however, it is not possible to obtain such ideal model.

A different approach based fuzzy reasoning has been proposed recently to improve foot speed in on-line trajectory generation in walking machines (Garcia and Gonzalez de Santos, 2001). The application of fuzzy set theory is specially recommended when a mathematical model of the system is not available. Moreover, fuzzy rules provide a good representation of parameter uncertainties existing on real machines and is an efficient representation of the real dynamic effects over the system motion, avoiding time-consuming mathematical models. The acceleration tuning approach, described in the following sections is based on generating the trapezoidal velocity profile that better fits the trajectory parameters reaching at least one actuator's speed limit. However, the characteristic of using a given velocity profile implies that the resultant linear foot velocity is lower that the theoretical speed achieved using minimum-time algorithms. Figure 7.3 compares two trajectories in the phase plane generated theoretically using minimum-time control and acceleration tuning approaches respectively. The speed-limit curve imposed by actuator torque limits and leg dynamics has been also represented. As the figure shows, the trajectory based on trapezoidal velocity profiles is tangent to the limit curve in its lowest point, maintaining that velocity value constant along the trajectory between acceleration and deceleration. In contrast, the

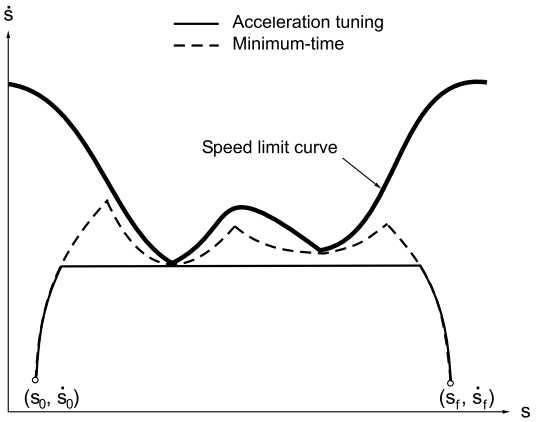


Fig. 7.3. Minimum-time trajectory and acceleration-tuning based trajectory in the phase plane

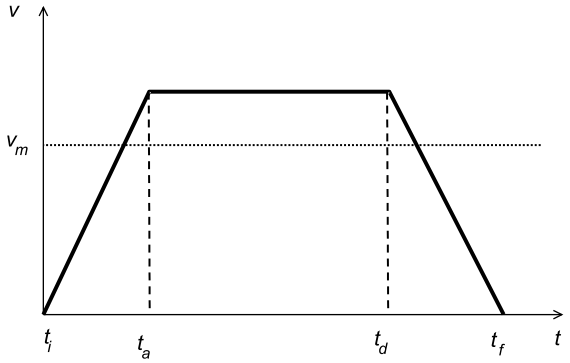


Fig. 7.4. Trapezoidal foot velocity profile for trajectory generation. v_m is the average foot velocity and t_a and t_d are acceleration and deceleration times respectively.

trajectory generated using minimum-time schemes is a tangent to the speed-limit curve at several points, thus adapting to the drive limits during the whole trajectory. Therefore, the average speed achieved using minimum-time techniques is higher that the obtained by the acceleration tuning approach. Nevertheless, the acceleration tuning approach is able to generate on-line the highest-speed trajectory achievable using trapezoidal velocity profiles.

7.3 The Acceleration Tuning Approach

Let us consider two types of trajectories inside the leg workspace. The first type will be trajectories of small $R(t)$, whose maximum average foot speed will depend largely on motor speed, as deduced from (7.5). The drive limits will be found for low linear foot speeds and there is no other possibility for increasing average foot speed for such trajectories than incorporating more powerful drives. The second type of trajectory will be trajectories of higher $R(t)$, which could reach very high foot speeds if drive speed is raised up to its limit. If the trajectory is generated using a trapezoidal velocity profile with a given foot acceleration (see Fig. 7.4), then the only magnitude that can limit foot speed for the second type of trajectory is the acceleration of the velocity profile. Hence, increasing the average foot speed in that type of trajectory can be achieved by increasing foot acceleration. However, increasing the acceleration of the velocity profile is not always an admissible solution in real mechanical systems. Leg dynamics could prevent the desired trajectory for a step velocity profile with infinite acceleration from being followed. Therefore, finding out the appropriate velocity profile for each particular type of trajectory, taking into account the effects of leg dynamics without using complex mathematical models to guarantee on-line trajectory generation is the main goal of the acceleration tuning approach.

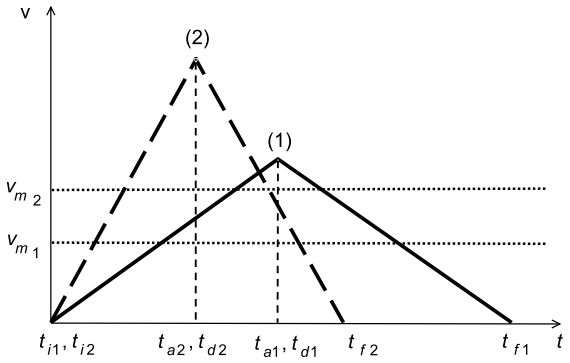


Fig. 7.5. Two foot velocity profiles for the same distance traveled during straight line generation

Using foot velocity profiles with higher acceleration might help achieve higher foot speed in another situation. Foot velocity profile becomes triangular for short trajectories (small $s(t)$), and hence, only small average foot speeds can be achieved. Increasing foot acceleration would allow short trajectories to be performed at high speeds. Figure 7.5 shows two foot velocity profiles for the same distance traveled. Profile (1) has lower foot acceleration than profile (2), and as a result it will never reach the average speed value v_{m2} that the second profile does. However, there are some cases where the effect of leg dynamics critically limits foot acceleration. Section 6.5.3 shows that the relevant dynamics affecting the motion of a 3-DOF articulated leg such as the SILO4 leg are inertial effects over the first joint of the leg due to its own acceleration, and gravitational effects over the second and third joints respectively. These dynamic effects over the leg motion correspond to the following practical observations: When the foot trajectory has a big z-component increment, the plane $I\hat{O}F$ is nearly vertical and the angle θ in Fig. 7.1 is mostly defined by the second and third joints of the leg, especially by the third joint, which reaches its maximum absolute angular speed during acceleration time in the foot velocity profile (the maximum drive torque of joint 3 is smaller than the second one; see Fig. 7.6). This means that raising the foot at high speeds amplifies the requested internal acceleration for the third axis, and thus, leg dynamics could prevent such a reference from being followed (due to its own weight, note from Sect. 6.5.3 that joints 2 and 3 are mainly affected by gravity). The same effect could affect the downward movement of the leg if it is bearing the robot's weight. Also notice that the second and third joints of the SILO4 leg have skew-axis gears (see Appendix A), which present nonlinearities due to frictional forces and backlash. The constant perturbation of gravity added to non-linear frictional and backlash effects could produce errors in the on-line generation algorithm that cause oscillations at the beginning of the upward movement if the requested speed

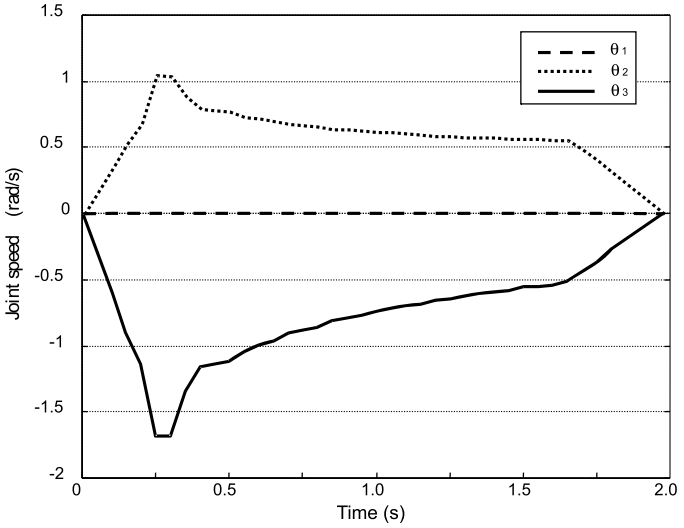


Fig. 7.6. SILO4 leg angular joint speeds when lifting the foot from leg position $(0, 0.350, -0.400)$ to $(0, 0.350, -0.200)$ at 0.1 m/s

is high. Finding an accurate and computationally cheap mathematical model of such dynamic effects over the leg motion is unavoidable. Fuzzy theory is an adequate tool for solving nonlinear system problems where a mathematical model is absent. Therefore, this soft computing technique is employed to introduce the dynamics affecting the leg motion into a foot acceleration tuning algorithm that provides the best acceleration value of the foot for each trajectory. Following this reflection we may conclude that fuzzy foot acceleration tuning as a function of the trajectory parameters and leg dynamics could be an adequate technique for improving performance in legged locomotion. Thus, the main goal of this chapter is to study how acceleration variations on the foot velocity profile modify leg performance.

The following steps summarize the acceleration tuning approach:

1. **Experimental workspace partitioning** taking limitation of drives into account. Experimental determination of the range of values for the trajectory parameters. That is, trajectory distance, average speed, and acceleration ranges must be identified.
2. **Fuzzy sets and rules**. Trajectory parameter transfer to fuzzy linguistic variables, and finding of fuzzy sets by partitioning the universe of discourse of each variable.
3. **Inference map** calculation. Direct application of the fuzzy inference system to the linguistic variables.

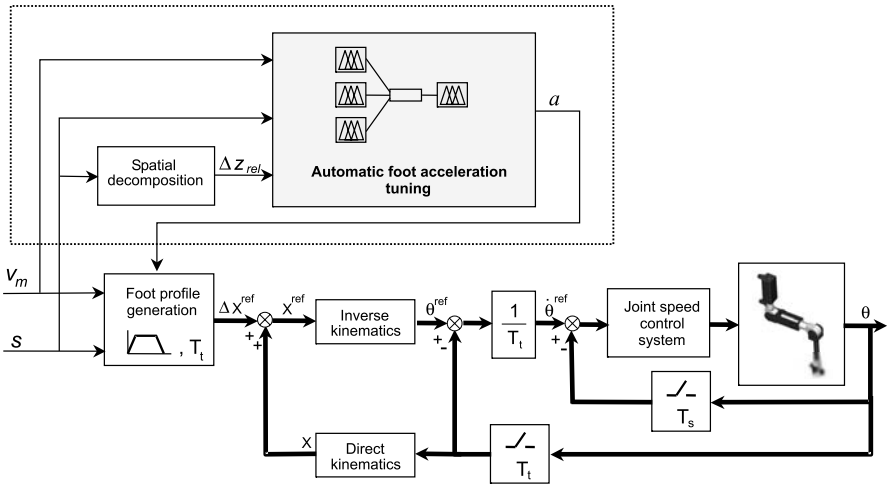


Fig. 7.7. Block diagram of the foot-trajectory control system

The solution will be a relationship between the acceleration of the foot and the trajectory parameters.

Figure 7.7 shows a block diagram of the foot trajectory control system. The foot positioning system fits into the classical scheme. Blocks inside the dotted rectangle correspond to the foot acceleration tuning method.

The following sections describe the three steps of the acceleration tuning approach in the velocity improvement of the SILO4 leg.

7.3.1 Experimental Workspace Partitioning

The first step of the acceleration tuning approach is the identification of trajectories inside the leg workspace whose linear foot speed is drive-limited. Figure 7.8(a) shows trapezoidal foot velocity profiles for trajectories parallel to the SILO4 leg's x_l axis. All trajectories have the same length and the same average foot speed. Figures 7.8(b)–(d) show angular speed profiles of the first, second and third joints of the leg respectively. One can conclude from the figures that for trajectories close to the leg's origin, the first joint of the leg reaches the actuator torque limit for foot speeds lower than the foot speeds at which joints 2 and 3 reach their drive limits. Therefore, the acceleration tuning approach should adjust foot acceleration for these type of trajectories to raise the foot speed until this actuator finds its torque limit, for every value of coordinate y_l.

The same experiment is carried out for foot trajectories parallel to the SILO4 leg's y_l and z_l axis. For both types of trajectories, the third joint limits the foot speed (see Fig. 7.9). Therefore, the acceleration tuning approach

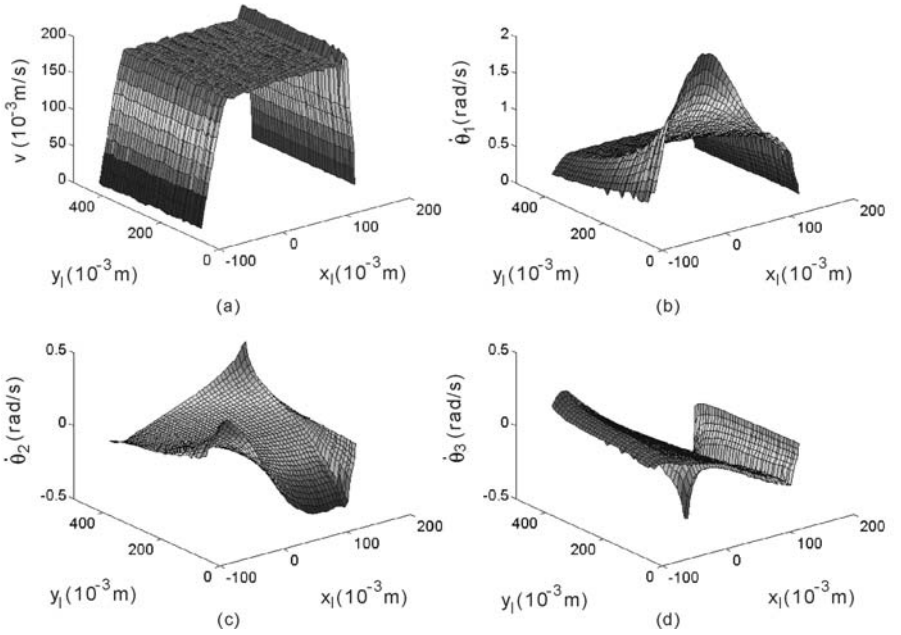


Fig. 7.8. Velocity profiles for trajectories parallel to the leg's x_l axis: (**a**) linear foot speed; (**b**) angular speed of joint 1; (**c**) angular speed of joint 2; (d) angular speed of joint 3

should adjust the foot acceleration to rise foot speed until the third actuator finds its torque limit.

However, Fig 7.9(d) shows that the third joint reaches the torque limit during acceleration time, which prevents the foot acceleration from being increased during vertical trajectories, due to the errors that can arise caused by leg dynamics. This case must be considered in the acceleration tuning approach.

This experimental study is summarized in the following considerations:

1. It is necessary to increase foot acceleration for short trajectories to obtain higher foot speeds.
2. Foot acceleration should be moderate when trajectories are large. A very high foot acceleration value is not adequate in the case of long trajectories because the drive limits of other joints are reached for stretched leg postures. An oscillatory response in the foot will be avoided with moderate acceleration values.
3. Foot acceleration should be decreased for vertical movements of the foot to avoid undesired errors due to dynamic effects.

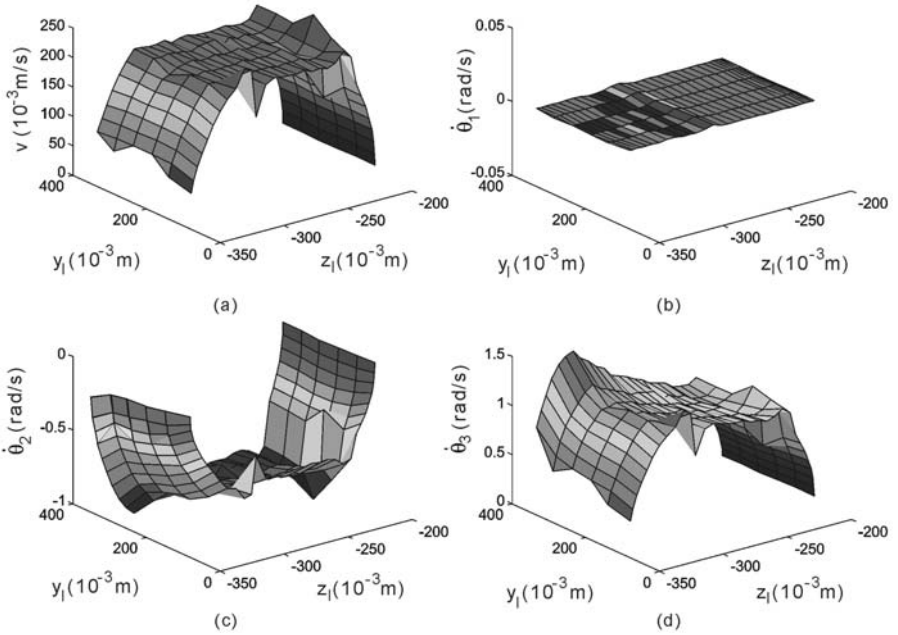


Fig. 7.9. Velocity profiles for trajectories parallel to the leg's z_l axis: (**a**) linear foot speed; (**b**) angular speed of joint 1; (**c**) angular speed of joint 2; (d) angular speed of joint 3

These considerations are vague rules that relate trajectory parameters with one another, although with some degree of uncertainty. In the following, the above conditions are translated into fuzzy rules that manage the relationships between short and long distance trajectories, required average speed and whether or not the foot is moving upwards or downwards as input variables to infer an appropriate foot acceleration.

7.3.2 Fuzzy Sets and Rules

The problem of finding the best value of foot acceleration for a given foot trajectory is overcome by using a very simple Mamdani fuzzy inference system (Mamdani, 1981). Three input linguistic variables define foot trajectory, which are: desired average foot speed (v_m), distance from initial to final position (s), and relative z increment (Δz_{rel}), which is the ratio between z increment and distance traveled for a given trajectory, that is

$$\Delta z_{\text{rel}} = \frac{|z_l(t_f) - z_l(t_i)|}{s} \leq 1. \tag{7.8}$$

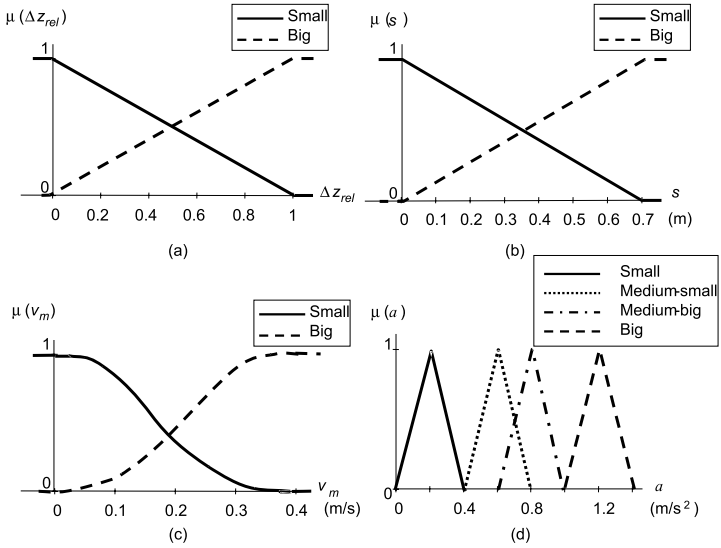


Fig. 7.10. Membership functions of the fuzzy inference system input and output variables: (**a**) relative z increment; (**b**) trajectory distance; (**c**) average foot speed; (**d**) foot acceleration

The output variable of the fuzzy inference system is the foot acceleration, which is needed for foot velocity profile determination and thus foot trajectory generation. Input and output fuzzy variables are represented by fuzzy sets (for example, distance is BIG, or foot speed is SMALL), and the degree of membership of each variable ($\mu(x)$ for the variable x) to the fuzzy sets is given by membership functions. The shape of these membership functions is chosen for each linguistic variable to adjust the relationship between speed, distance and acceleration in a triangular or trapezoidal velocity profile, which makes acceleration inversely proportional to distance and directly proportional to the square of the speed, which is expressed thus in analytical form:

$$a = K\frac{v_m^2}{s} \tag{7.9}$$

where K is a constant value for each profile: $K = 4$ for a triangular velocity profile, $K > 4$ for a trapezoidal velocity profile. Guidelines on fuzzy controller design have been followed (Matia *et al.*, 1992). They state that the inference map shape matches the shape of membership functions of the input variables, provided that membership functions are normal, symmetrical and overlapped by pairs, and the membership functions defined over the output variables have the same area. Taking this into consideration, the following assumptions have been made in order to design the fuzzy system:

1. Let us assume that the relative z increment in a trajectory, Δz_{rel}, is represented by two fuzzy sets {SMALL, BIG}. Membership functions of this input variable are trapezoidal and are shown in Fig. 7.10(a) where the abscissa is the value of the relative z increment and the ordinate $\mu(\Delta z_{\text{rel}})$ is the degree of membership. The relationship between Δz_{rel}, and a in (7.9) requires a negative inclination of the resulting inference map along this variable edge, what will be expressed by means of fuzzy rules in the next subsection.
2. Trajectory distance (s) is also represented by two fuzzy sets {SMALL, BIG}. Two trapezoidal distance membership functions are shown in Fig. 7.10(b), where the abscissa is the value of the trajectory distance and the ordinate $\mu(s)$ is the degree of membership. Their limit values were obtained experimentally for the SILO4 leg workspace, where the maximum linear distance for a trajectory is 0.7 m.
3. Average foot speed (v_m) is also represented by two fuzzy sets as the first two variables, {SMALL, BIG}. However, membership functions are parabolic rather than trapezoidal (see Fig. 7.10(c)) just to adjust to the relationship in (7.9). Their limit values were found experimentally for the SILO4 leg example. The average foot speed is limited to 0.4 m/s due to maximum motor speed. A foot speed of 0.2 m/s seems to be an intermediate value appropriate for straight-line motion (see Fig. 7.10(c)).
4. The output of this fuzzy inference system is the foot acceleration (a), which is represented by four fuzzy sets {SMALL, MEDIUM-SMALL, MEDIUM-BIG, BIG}, and membership functions are shown in Fig. 7.10(d). These membership functions are triangular and the limit values are obtained experimentally for the SILO4 leg example.

The fuzzy-inference mechanism is based on the following five rules, which represent the fuzzy dependence of foot acceleration on foot speed, trajectory distance and relative z increment:

1. If v_m is SMALL and s is SMALL and Δz_{rel} is SMALL, then a is MEDIUM-BIG.
2. If v_m is SMALL and s is BIG and Δz_{rel} is SMALL, then a is SMALL.
3. If v_m is BIG and s is SMALL and Δz_{rel} is SMALL, then a is BIG.
4. If v_m is BIG and s is BIG and Δz_{rel} is SMALL, then a is MEDIUM-BIG.
5. If Δz_{rel} is BIG, then a es MEDIUM-SMALL.

7.3.3 Fuzzy Inference Map

MATLAB and its Fuzzy Toolbox were used to solve the fuzzy problem, where *min* represents the *and* method and implication, *max* represents aggregation, and *centroid* is used for defuzzification. The inference map for the foot acceleration problem, is a hypersurface which we have represented by three bidimensional inference maps for the sake of clarity. Figure 7.11 shows foot

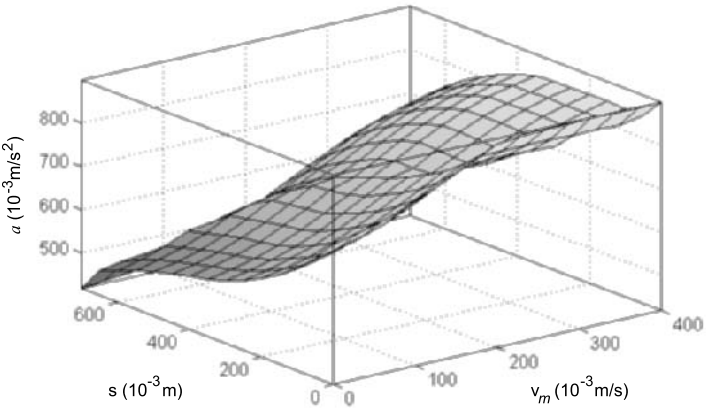


Fig. 7.11. Inference map of foot acceleration *vs* trajectory distance and average foot speed for $\Delta z_{\text{rel}} = 0$

acceleration as a function of the distance traveled and the average foot speed for any trajectory with almost no z increment in the foot trajectory, that is, when the relative z increment is very close to zero. Figure 7.12 shows the limiting effect of any z increment in foot acceleration. Figure 7.12(a) shows the foot acceleration output as a function of the relative z increment and distance traveled, when fixing the average foot velocity at a value of 0.200 m/s, and Fig. 7.12(b) shows acceleration *vs* relative z increment and average foot speed, for a distance traveled of 0.350 m. Once the foot acceleration function has been obtained, optimization methods for real-time implementation of the fuzzy reasoning process can be used (Matia and Jimenez, 1996). The number of fuzzy rules imposed on the foot behavior to solve the speed improvement problem can be modified in terms of dynamic complexity, however, it is independent on the number of degrees of freedom of the robot leg.

7.4 Experimental Results

Different experiments have been conducted to show the improvement on online trajectory generation by foot acceleration tuning. For this purpose the SILO4 leg has been used. The first experiment shows the effect of foot acceleration tuning when executing straight-line trajectories of several lengths. Figure 7.13 illustrates this experiment, depicting the maximum achievable average foot speed for different trajectory distances. Every trajectory was parallel to the leg's x_l axis and for $y_l = 0.215\,\text{m}$ and $z_l = -0.250\,\text{m}$. Each thin curve in this graph represents maximum average foot speeds that could be reached with a foot velocity profile of constant acceleration assuming that

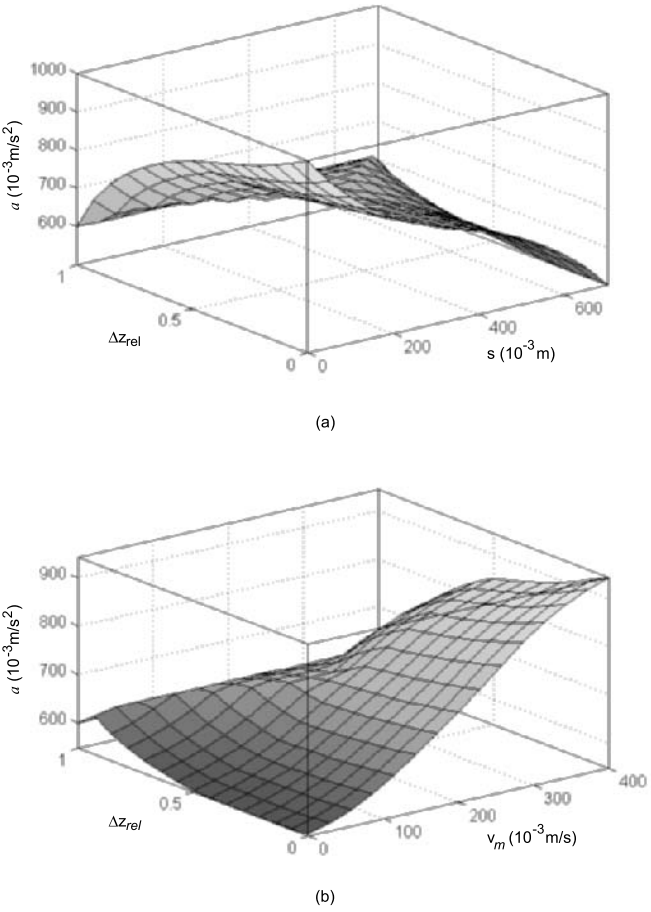


Fig. 7.12. Inference map of foot acceleration *vs* relative z-increment and: (**a**) distance traveled; (**b**) foot speed

no dynamics perturb the motion, and the thick curve represents the maximum achievable foot speed during the same trajectories, considering leg dynamics, using foot acceleration tuning. If the maximum constant acceleration curves were used, the dynamics affecting the leg motion would prevent the leg from following the specified path (dotted lines in Fig. 7.13) and oscillations and non-desired effects will appear during on-line trajectory generation. To ensure that leg dynamics will not perturb the motion, a conservative acceleration value should be chosen (*i.e.* $0.6\,\mathrm{m/s^2}$), and low speed trajectories would be performed for distances where leg dynamics and limitation of drives does not prevent from achieving higher speeds. Figure 7.13 clearly shows the improvement of foot acceleration tuning on this problem. The maximum acceleration that is achievable by using this method is very close to the curve

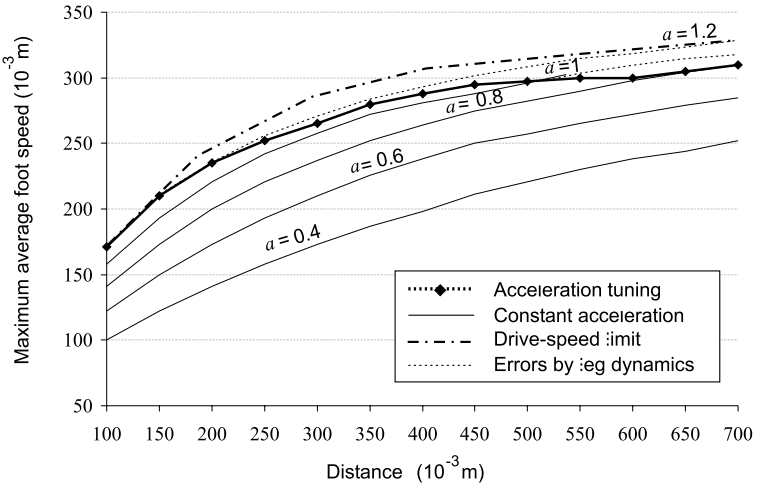


Fig. 7.13. Curves of maximum average foot speed *vs* trajectory distance with constant foot acceleration profile and with foot acceleration tuning

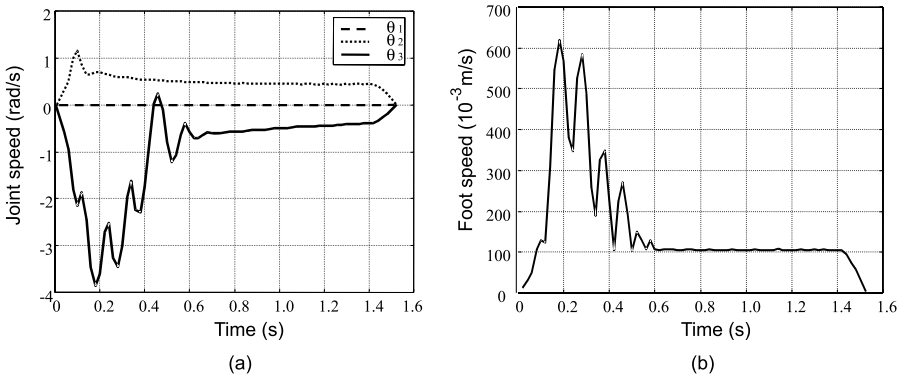


Fig. 7.14. Leg oscillations during lifting experiments with foot acceleration of $1\,\mathrm{m/s^2}$: (**a**) articular joint velocity profiles; (**b**) foot velocity profile

of drive-limited speed (dash-dotted line), but avoiding non-desired dynamic effects.

Another important feature can be observed from Fig. 7.13. The maximum average speed curve that is achieved using acceleration tuning is nearly constant for BIG distance trajectories ($s > 0.35\,\mathrm{m}$). If this walking robot control system always manages to walk over these BIG step trajectories, the maximum robot speed will always be achieved, independently of the trajectory

Table 7.1. Comparison of maximum achievable average foot speeds

$\min_{t_i \le t \le t_f}\{R(t)\}$ (10^{-3} m)	s (10^{-3} m)	v_m^{max} (10^{-3} m/s)		Improvement (%)
		Acceleration tuning	$a = 0.6\,\mathrm{m/s^2}$	
120	330	230	180	27.7
150	500	290	220	31.8
200	580	310	250	24.0
300	510	340	270	25.9
350	400	340	240	41.6
400	200	320	160	100.0

distance, s. This is not possible using a velocity profile with constant acceleration, where maximum average speed changes with trajectory distance. The conclusion of this first experiment is that foot acceleration tuning finds the acceleration values that provide higher foot speeds, avoiding the use of very high acceleration values that could impose oscillatory behavior. It also helps maintain the maximum average foot speed as a constant specification for the control system.

A second experiment was run lifting the foot at high speed. The foot acceleration tuning approach limits the acceleration value to $0.6\,\mathrm{m/s^2}$ when lifting the foot, as a result of applying the fifth fuzzy rule to $\Delta z_{\mathrm{rel}} \simeq 1$ (see Fig. 7.12). Thus, oscillations due to the leg's dynamic effects on on-line trajectory generation were avoided by using a moderate acceleration value. Without acceleration tuning, a foot acceleration value higher than $0.6\,\mathrm{m/s^2}$ produces oscillations during foot lifting, due to errors in trajectory tracking caused mainly by leg weight. Figure 7.14(a) shows velocity profiles of the three joints, and Fig. 7.14(b) shows the foot velocity profile, while the foot tried to lift its own weight at a speed of 0.09 m/s with a foot acceleration of $1\,\mathrm{m/s^2}$. The angular speed of the third joint of the leg reaches its maximum achievable value (limitation of the drive) during acceleration time, which causes errors in on-line trajectory tracking that make the leg oscillate.

The last experiment showing the acceptability of this method studied the behavior of acceleration tuning through a variety of trajectories covering the leg workspace. Figure 7.15 illustrates this experiment, where trajectories of different length, $s(t)$, and different radius, $R(t)$, were executed by the leg with and without acceleration tuning at its maximum achievable foot speed (thick and thin lines respectively). Every trajectory is confined to the plane $z_l = -0.300\,\mathrm{m}$. Behavior was the same within different z-component planes. Average foot speed values as well as distance traveled and minimum radius $R(t)$ values are listed in Table 7.1 for each trajectory of the experiment in Fig. 7.15. The improvement offered by the acceleration tuning algorithm is relevant for short trajectories ($s < 0.350\,\mathrm{m}$) and acceptable for long ones, as shown in Table 7.1. Improvement reaches higher values for trajectories of the same length ($s(t)$) with higher $R(t)$, as stated in (7.5). As can be observed

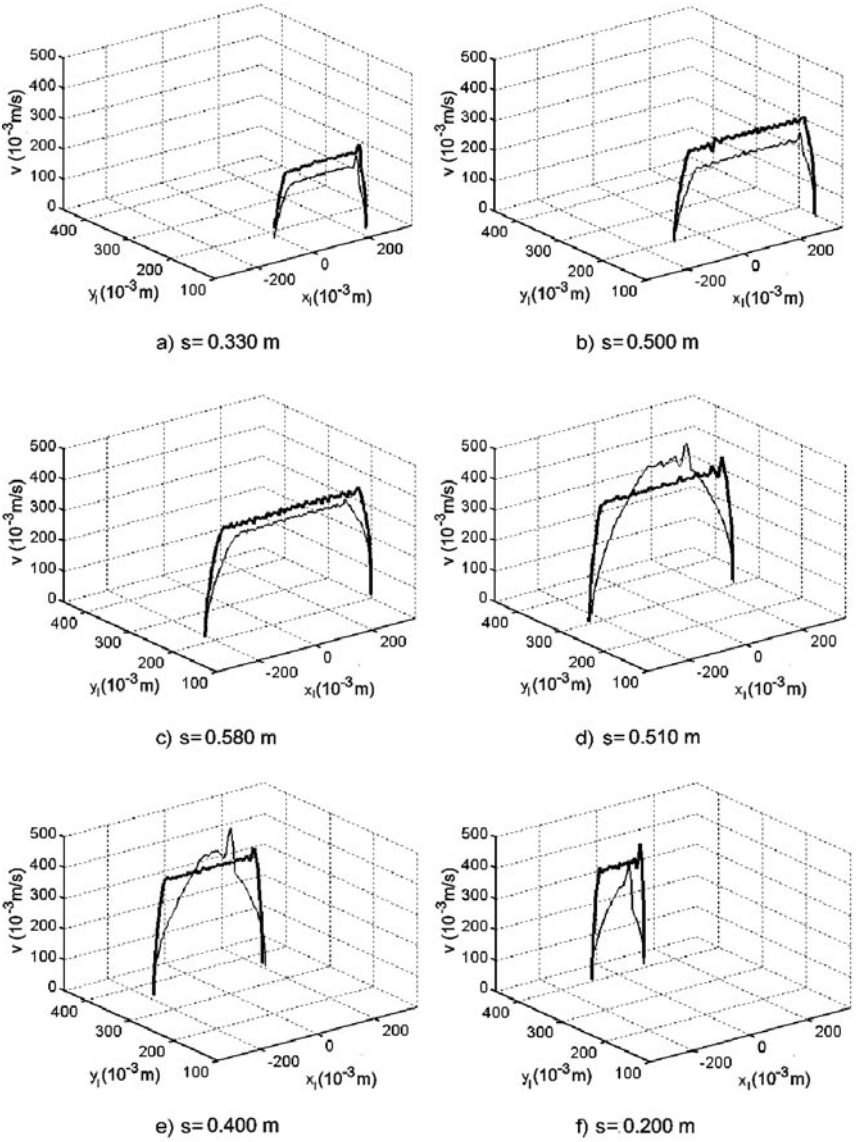


Fig. 7.15. Comparison between foot velocity profiles with acceleration tuning (*thick line*) and with constant acceleration of $0.6\,\mathrm{m/s^2}$ (*thin line*)

from Fig. 7.15, leg behavior with acceleration tuning is less oscillatory for every trajectory because high average speed values can be achieved without reaching the drive limits and avoiding dynamic effects. Also, higher average foot speeds can be achieved. Table 7.1 reveals that the acceleration tuning method increases the average foot speed by 27–100% over the maximum achievable speed when acceleration tuning is not used. It may be concluded that the foot acceleration tuning approach for on-line trajectory generation is a very suitable method for achieving precise, smooth, and fast foot movements, which is highly important for legged locomotion. All the experiments in this chapter show on-line-generated straight-line trajectories of the foot. The on-line path planning computation was realized at a sampling period of 4 ms on a 486 processor at 50 MHz, while the fuzzy foot acceleration tuning calculation required 0.5 ms. This ensures the real-time application of the proposed acceleration tuning algorithm in the locomotion system of a walking robot.

7.5 Discussion and Conclusions

The work presented in this chapter focuses on improving leg speed in on-line trajectory generation of a walking robot. Two different techniques have been considered such as minimum-time control algorithms and the foot acceleration tuning approach. However, the requisite of on-line trajectory generation makes classical approaches fail in this goal, and therefore, the acceleration tuning approach has been chosen. Several experiments have been conducted using the SILO4 robot to study how average foot speed can be increased. These experiments have been illustrated in detail, and the phenomenon has been thoroughly explained. The acceleration tuning approach targets the acceleration of the speed profile as the proper magnitude to be tuned. To avoid problems stemming from the robot's parameter uncertainties, fuzzy techniques are used. For this purpose, fuzzy rules are defined based on experiments, and the most suitable acceleration for every given trajectory is found. A simple Mamdani fuzzy inference system is used to compute the required acceleration. It is based on five rules using three linguistic variables. The number of fuzzy rules imposed on the foot behavior can be modified in terms of dynamic complexity, however, it is independent on the number of degrees of freedom of the robot leg.

Some experiments have been carried out to validate the algorithm. These experiments concluded that the foot acceleration tuning method finds the acceleration values that provide fast and smooth foot trajectories, avoiding perturbing effects due to saturation of the actuators and leg dynamics. Finally, some experiments were performed to compare the behavior of a leg executing the same trajectory both with and without the acceleration tuning method. This comparative study reveals that, depending on the distance traveled, the acceleration tuning method increases average foot speed by 27–100% over the maximum achievable speed when acceleration tuning is not used.

8

Virtual Sensors for Walking Robots

8.1 Introduction

One of the main advantages of legged robots is their great adaptability to irregular, unstructured terrain. However, the great complexity of the control algorithms and the mechanical and electronic hardware that go into these robots is a negative aspect of this essential characteristic. As stated in Chap. 1 this complexity is one of the main drawbacks that have kept walking machines from expanding to real applications.

One essential part of the information needed to perform terrain adaptation is the detection of the contact between the foot and the terrain at the end of leg transference (ground detection). The detection of impacts between the leg and obstacles during leg transference is also necessary when traversing highly irregular terrain. In addition, the monitoring of foot forces during the support phase is useful for enhancing locomotion features, since it enables foot traction to be improved and weight distribution optimized. Traditionally, these aspects of terrain adaptation have been accomplished by the use of different kinds of sensors, usually installed in the robot's feet. Some walking-robot developments include in their leg design devices such as switches (Jimenez *et al.*, 1993), whiskers (Hirose *et al.*, 1990), strain gauges (Schneider and Schmucker, 2000; Rossmann and Pfeiffer, 1998), load cells (Gonzalez de Santos *et al.*, 2003), spring-potentiometer arrangements (Riddestrom *et al.*, 2000) and inductive sensors (Grieco *et al.*, 1998).

However, these solutions present some drawbacks, which can be summarized as an increment in the complexity of the machine and a reduction in its robustness. The installation of sensors in the feet involves the installation of wiring and sometimes electronic equipment at a point of the robot that is constantly exposed to interaction with different kinds of terrain and obstacles. Locomotion on sandy, wet or abrasive terrain can be an important source of failures in the detection process and damage to the sensor system as well. Furthermore, the installation of this additional hardware increases the total price of the robot as well as maintenance.

An alternative to the use of sensors installed in the robot's feet is the analysis of signals present in the servomotor system. For instance, the monitoring of motor-current in order to estimate joint torques for ground-adaptation purposes can be considered an usual practice (Kepplin and Berns, 1999). However, other less well-known sources of information can be used to achieve the required perception of the environment. The information provided by joint-position sensors can serve this purpose, as shown in Sect. 8.2.

Furthermore, a specific sensor can only partially characterize the internal state of the machine or its environment. The consideration of multiple and interrelated sources of information can be used to provide the walking machine with inherent sensorial redundancy without increasing its hardware. This redundancy can prevent drastic system failures, and generate more accurate, consistent sensorial information for determining the behaviour of the walking machine in unstructured terrain. Some recent works (Kepplin and Berns, 1999; Luo *et al.*, 2001; Shimizu *et al.*, 2002) give examples of the increasing interest in sensor-fusion systems in the field of walking machines.

This chapter addresses the design, development and testing of virtual sensors, which can be defined as systems that infer sensorial magnitudes by monitoring of other available magnitudes. The objective is either to substitute or to complement the information on terrain given normally by physical sensors. These objectives have been motivated by the necessity of simplifying the hardware of walking robots and reducing their design, construction and maintenance costs while enhancing the robustness of the machine and the reliability of its behaviour. These virtual sensors are based on neural networks and can estimate the forces exerted by the feet from data extracted from joint-position sensors, which are mandatory in all robotic systems. These estimates can emulate the information given by either a switch, a one-axis force sensor or a three-axis force sensor installed on the foot. The force estimates are to be employed to detect the contact between the foot and the ground at the end of leg transference.

8.2 Problem Approach

As outlined in the introduction of this chapter, the acquisition of sensorial information about the terrain can be accomplished by monitoring and processing certain magnitudes available in the servo-controlled system of the robot. Here we propose to analyze the information provided by joint-position sensors, which are available in any robotic system, to estimate the forces exerted by the leg. The immediate objective of this analysis is to detect the contact between the foot and the ground at the end of leg transference; however, these estimates could be used for other purposes during locomotion, such as to detect leg crashes and to monitor weight distributions, or even in manipulation tasks, although the calibration procedure should be carefully adapted to each case.

The most relevant magnitude that can be drawn from the joint-position sensors is the position error of the joint, defined as the difference between the desired position of a joint, given by the trajectory generator, and the actual position, given by the joint-position sensor. The correct interpretation of this signal, which is accessible in a wide range of servo-controlled systems, can yield an estimate of the joint torques and, consequently, the foot forces. The position sensors, which are initially installed for the sole, necessary purpose of closing the control loop, are used in this manner to extract information about the environment. This can result in either the simplification of the walking machine, due to the elimination of sensors and associated hardware installed in the leg, or improved perception of the environment, due to the larger variety of information available. For example, the estimates can inform about the forces exerted not only by the foot, but by any part of the leg structure, a fact that can be useful for detecting unexpected collisions during leg transference. Thus this estimate can complement the information provided by a traditional force sensor installed on the foot.

To ascertain the viability of this method, some preliminary experiments were performed with a walking robot (see Sect. 8.5). In these experiments one of the robot's feet moved towards the ground while the position errors of all three joints of the leg were sampled. Figure 8.1(a) depicts the position errors when the leg moves freely in the air; Figure 8.1(b) represents the evolution of joint-position errors when, in the same conditions, the foot collides with the ground (around T = 2 s.). The appreciable rise in position errors in this case points to the viability of the detection of foot/ground contact and the estimation of foot forces from this kind of data.

However, the relationship between joint-position errors and foot forces can be very complex, as it depends on a wide variety of factors, which are described below.

- Mainly, this relationship depends on certain variables of leg motion, as described below. First, joint-position errors are heavily dependent on internal joint speed. This dependence is due principally to the friction effects caused by the reduction gears and transmissions, which can be a relevant and complex phenomenon, as identified in (Garcia *et al.*, 2002). The magnitude of position errors caused merely by friction when the foot moves freely is significant and comparable to the magnitude of position errors measured when the foot collides against the ground. Second, the position errors depend also on the position of the foot. This dependence is given essentially by gravity's effects and by the transformation of foot forces into joint torques through leg kinematics, two contributions that can be easily modelled mathematically. But position-dependent friction effects may be present as well (Garcia *et al.*, 2002), being a complex phenomenon caused by transmission wearing, shaft misalignments, *etc.* Finally, position error depends on joint acceleration. During the acceleration of a motor, the

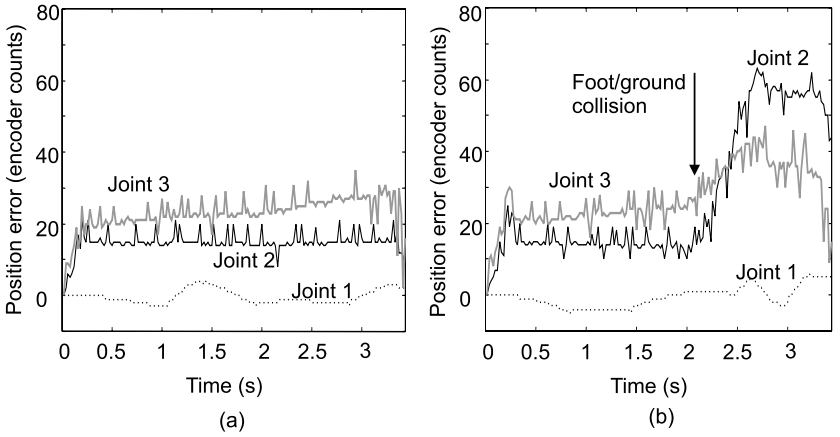


Fig. 8.1. Examples of evolution of joint-position errors during leg motion: (**a**) the leg moves freely in the air; (**b**) the foot collides with the ground

joint-position errors depend on the controller's ability to track the velocity profile. Friction is also affected by variations in the joint speed; an increasing joint speed can yield higher friction than a decreasing speed, and the hysteresis loop becomes wider as the velocity variations become faster.

- Position errors depend also on factors that might be considered constant, although they can vary over time. As examples of these factors, we might highlight changes in the machine's mechanical properties due to wear and maintenance, temperature, changes in the electrical features due to battery depletion and even deliberate changes in the parameters of the motion controllers.
- Finally, position error depends on other factors that, albeit constant and characteristic for each robot, must be considered when designing a versatile sensing system applicable to different servo-controlled systems. The magnitude of position errors depends on the characteristics of the motion controller and power drives, – whose control law is not always known –, the transmissions and reductions used (worm gears, spur gears, lead screws, belts, harmonic drives, *etc.*), the features of the power supply, *etc.*

To sum up, the relationship between foot forces and joint-position errors depends on many factors and is consequently determined by a complex pattern. In order to obtain a functional system to estimate foot forces, it is necessary to develop a model that identifies these effects, enabling the unknown magnitude to be correlated with the available magnitudes. This idea matches the definition of the virtual sensor, a technique that has become more popular in recent years, as discussed in the next section.

8.3 Virtual Sensors Based on Neural Networks

Virtual sensors, also known as software sensors or estimators, are methods used to measure sensorial magnitudes indirectly through the analysis of available related sensorial data. One of the most frequent uses of virtual sensors is as a substitute for physical sensors, when physical sensors are not feasible due to price, size, weight, weakness, the quality of the measurement they provide (Wickstrom *et al.*, 1997; Masson *et al.*, 1999), or when no specific physical sensors are available (Valentin and Denoeux, 2001; Leal *et al.*, 1997). Additionally, virtual sensors form the basis for a number of fault-tolerant systems, fault-detection systems, and sensor-fusion systems. One important feature of virtual sensors is that they are intrinsically robust because of their inherent redundancy features, one consequence of the consideration of different interrelated sensorial magnitudes without the use of additional hardware.

Data-driven (empirical) virtual sensors (Masson *et al.*, 1999) estimate the dependence between known and unknown sensorial magnitudes statistically, from a training set including representative examples of operation. This kind of virtual sensor is used when an analytical model of the system is unknown, as well as when model development costs must be kept down (Hanzevack *et al.*, 1997). The most recent implementations of data-driven virtual sensors are frequently based on artificial neural networks (Ablameyko *et al.*, 2003). The main advantage of neural networks is their capability to establish highly complex, non-linear, multidimensional, dynamic and even adaptive associations between input and output magnitudes. Virtual sensors based on neural networks are particularly well-suited for the implementation of sensor-fusion systems (Leal *et al.*, 1997) data validation and fault detection (Hines *et al.*, 1998) systems and for the reconstruction of incomplete sensorial information (Valentin and Denoeux, 2001). Other qualities of neural networks are the possibility of autonomous on-line learning – which allows the construction of adaptive models (Hanzevack *et al.*, 1997) – and prediction of the values of unknown magnitudes (Yuan and Vanrollehghem, 1998).

The main drawback of virtual sensors based on neural networks is that they are reliable only in working conditions included in the training region, so they must be trained carefully with representative samples accurately describing the process throughout the whole range of situations that could be encountered in practice. Additionally, neural networks require much more extensive, careful validation than physical models. Finally, virtual sensors require a processing system, such as a computer, microcontroller or FPGA to calculate their estimates.

8.4 Virtual-sensor Design

Section 8.2 pointed out the necessity of a model of the position errors in the overall electro-mechanical system to estimate foot forces. Here we propose the

use of neural networks to correlate the foot forces with the available sensorial magnitudes, and so implementing a virtual sensor. The general advantages of neural networks, described in Sect. 8.3, yield interesting properties in this particular application.

As stated, neural black-box modelling renders the development of mathematical models unnecessary, a fact that makes this technique easily extendable to other servo-controlled systems. This is a very interesting feature, as this virtual sensor is meant to be adapted to different walking robots. Additionally, the behaviour of the complete system can be modelled in one step, a feature that accelerates the calibration process, since it can be quite similar to the ordinary locomotion process, and simplifies the experimental setup needed (see Sect. 8.5.2). However, although neural modelling reduces the required *a priori* knowledge about the system, an appropriate network architecture still needs to be selected to model the system correctly, so the neural model is, in this sense, a grey box. Furthermore, the use of these devices will facilitate the addition of complementary sensorial sources in future, integrating data from physical sensors installed in the robot foot (force sensors, switches, *etc.*) and other estimates (motor current). Finally, most walking robots are provided with a computing system able to calculate the estimates, so the implementation of virtual sensors does not result in an increase in hardware.

Although all the factors described in Sect. 8.2 affect the magnitude of joint-position errors, only the joint position and speed have been specifically considered here in the design of the virtual sensors. As stated, joint position and speed are the main motion variables that determine the system's working conditions during leg motion. Consequently, these magnitudes have been selected along with joint-position errors as the neural network's inputs, allowing the modelling of viscous friction, position dependent friction effects, leg kinematics, gravitational effects, *etc.* The inclusion of joint accelerations as network inputs did not prove to enhance the performance of the virtual sensor in the experimental platform used in this work (see Sect. 8.5). Relevant dynamic effects due to joint moment of inertia are observed only in start-stop conditions (*i.e.* at high accelerations), which are not found during the last phase of the leg transference, when the virtual sensors are to be used. Similarly, dynamic friction effects such as hysteresis have not been found to be significant in this system. The rest of the factors that affect the estimate (see Sect. 8.2) are considered constant for each machine and have been taken into account in the general design of the virtual sensors: the existence of all these complex effects, which can differ widely in different machines, is why we used a general (black box) approach to model them, thus facilitating the adaptation to different servo-controlled systems.

Three possible desired output responses have been tested in this work:

1. Classification of the leg state into one of the following options: collision with terrain or free motion.

2. Estimation of the module of the force exerted by the foot.
3. Estimation of the three foot force components.

These three possibilities have been considered with the intention of evaluating and comparing different levels of output-information quality from the point of view of experimental hardware requirements and the quality of the estimates obtained.

Some variations of the virtual sensor described above have been tested in preliminary experiments to analyze the influence of joint-position inputs on the estimates. In these modified virtual sensors, the neural network was trained to estimate the three joint torques. The transformation from foot forces to joint torques (and *vice versa*) was computed through an analytical dynamic model of the leg (see Chap. 6). Position signals were not included in as inputs to the network, assuming that all position-dependent effects were considered in this model. The experiments carried out with this modified virtual sensor showed a significantly poorer performance than the original design; this fact indicates the existence of other position-dependent effects not included in the dynamic model, probably related to reduction friction and misalignments in the transmission shafts. The inclusion of joint-position signals as network inputs in this modified design resulted in a sensor performance similar to that obtained with the original virtual sensor. Thus, we can conclude that:

1. Position network inputs are necessary for modelling unidentified position-dependent effects, and they substantially enhance the quality of the estimate
2. The performance of the virtual-sensor design was not enhanced by including an analytical model including some position dependent effects. Consequently, these variations were finally ruled out.

8.5 Using Virtual Sensors in Real Walking Machines

The virtual sensor-design proposed in the previous section is to be applied to any kind of robot; however, preliminary experiments have been conducted using the SILO4 quadruped (see Appendix A), in which the validity of the estimates has been characterized experimentally. As explained in Sect. A.2.4, a three-axis piezoelectric force sensor is placed in the foot for ground adaptation purposes. Here, this force sensor is used to calibrate the virtual sensors. The standard error of the measurements provided by this sensor is 2 N, about 2% of the force range considered in this work (from 0 to 100 N). Although some characteristics of the virtual-sensor implementation and calibration procedure may vary, the method we propose can be extended to any walking machine and in general to any servo-controlled system with similar characteristics. In this section we shall describe the neural-network architecture and the calibration procedure, ending with an account of the experiments and their results.

8.5.1 The Neural Network

To solve the ground-detection problem, a non-linear feed-forward neural network (multilayer perceptron) with one hidden layer and sigmoidal activation functions was selected. This network architecture is widely extended in the modelling of non-linear static systems and in the implementation of virtual sensors (Valentin and Denoeux, 2001; Wickstrom *et al.*, 1997; Leal *et al.*, 1997; Yuan and Vanrollehghem, 1998; Masson *et al.*, 1999).

Previous experience showed that dynamic effects due to joint inertia or friction hysteresis were not relevant in the SILO4 platform's normal range of operation. Accordingly, the use of some well-known dynamic network architectures (such as Elman networks and multilayer perceptrons with feedback) or the inclusion of past inputs in a static network (tapped delay inputs) did not prove to enhance the system's performance. The static approach of the feed-forward network was found to be sufficient to model the system for this particular platform.

As stated in Sect. 8.4, the position error, the position and the speed of each leg joint have been selected as input magnitudes for the network. Since the leg consists of three joints, the network has a total of nine input neurons. The number of hidden neurons has been fixed at five empirically, a selection based on the quality of the results and the training cost. The number of output neurons is just one in the case of the virtual switch and the virtual one-axis force sensor, and three in the case of the virtual three-axis force sensor.

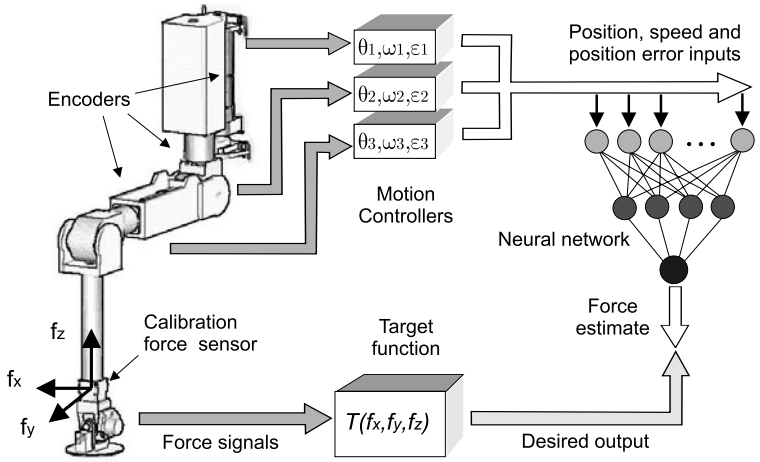


Fig. 8.2. Calibration procedure

8.5.2 Network Calibration Example Sets

In order to calibrate the virtual sensors, the neural network must be trained with examples of input magnitudes and desired output during the normal functioning of the walking machine. Hence, the examples consist of series of samples of the positions, speeds and position errors of the three leg joints, together with measurements from the three-axis piezoelectric force sensor used for calibration. For robots not equipped with force sensors, a provisional calibration sensor should be installed during the data-acquisition process. Depending on the desired virtual-sensor response, this calibration sensor may be a switch, a one-axis force sensor or a three-axis force sensor. As the purpose of the virtual sensors is to detect foot/ground contact, each example contains data taken while the foot moves down towards the ground and eventually collides with it, in a process similar to normal locomotion. The sensor-sampling frequency used in the experiments is 50 Hz. During the experiments, the robot is supported on its other three legs in a pose similar to those found in the discontinuous gaits the robot is currently using (see Chap. 3). In this manner, the foot forces obtained (which depend on the total weight of the robot, the leg flexiblity, the positions of supporting feet, *etc.*) are similar to those exerted in normal conditions.

The Training-example Set

The examples used to train the network must accurately represent all the possibilities that can be found, but kept as low as possible to accelerate the calibration process. In order to delimit the problem, the possible foot trajectories have been limited to the trajectories in the discontinuous gaits the robot currently uses. Therefore, the trajectories used in the training examples are vertical and rectilinear (Estremera *et al.*, 2005). The training examples are then selected attempting to represent all the possible foot speeds and all the vertical trajectories (*i.e.*, all the operating points) found in normal locomotion. The vertical foot trajectories and speed values were selected with the aim of uniformly covering the whole leg workspace and the whole speed range (0.025 – 0.1 m/s) respectively. The total number of examples used to train the network (the result of the combination of those foot speeds and trajectories) was 210, summing up to 30,000 samples.

The Test-example Set

New experiments were performed in order to obtain the test example set, which is designed to validate the virtual sensors in different situations that could be encountered in practice. Thus, the examples represent rectilinear and vertical trajectories executed within the leg workspace and the speed range employed in normal locomotion. Other working conditions were discarded as they are not expected to appear during the ground detection process. This

example set was divided into three subsets, each intended to evaluate a specific aspect of the virtual sensors' performance:

- **Test set A.** This set was designed to test the accuracy of the estimates in operating conditions considered in the training process. Therefore, the examples it included represented the same operating points as those included in the training set.
- **Test set B.** This set was used to test the repeatability of the system at a single operating point. All the examples corresponded to a single randomly chosen operating point.
- **Test set C.** This set was intended to test the accuracy of the estimates in working conditions not represented in the training process. It included examples for the foot speeds and trajectories most dissimilar to those found in the training set, while remaining within the considered speed range and leg workspace.

8.5.3 Training Procedure

The sensorial magnitudes recorded in the experiments described above are correlated in the training process so as to produce the appropriate output. As pointed out in Sect. 8.4, three possibilities have been tested:

- **Virtual switch:** the network output classifies the leg state as either free motion or ground contact. The threshold force, F_T, defined as the value of the module of the force for which we consider that the foot has collided and is firmly placed on the ground, was considered to determine leg state. Taking this into account, the target output is defined as the following step function:

$$T(f_x, f_y, f_z, F_T) = \begin{cases} 0 \text{ if} \sqrt{f_x^2 + f_y^2 + f_z^2} < F_T, \\ 1 \text{ otherwise.} \end{cases}$$

  where f_x, f_y and f_z are the three force components provided by the calibration sensor. In subsequent training processes, the target function was generated considering different force thresholds to evaluate the viability of regulating the sensitivity of the virtual switch.
- **Virtual one-axis force sensor:** the network output is the estimate of the module of the foot force. The target function used is thus

$$T(f_x, f_y, f_z) = \sqrt{f_x^2 + f_y^2 + f_z^2}.$$

- **Virtual three-axis force sensor:** a neural network with three output neurons is trained to approximate each one of the three signals provided by the calibration force sensor

$$\mathbf{T}(f_x, f_y, f_z) = (f_x, f_y, f_z).$$

  Hence, it is possible to estimate the direction of the forces exerted; this is the most general case of estimate of foot/ground interaction forces.

The training procedure was accomplished with the help of Matlab 5.0 neural network toolbox (MATLAB, 1992) in an external computer. The learning algorithm selected was Levenberg-Marquardt backpropagation (Hagan *et al.*, 1996), a selection based on training time and the quality of the results obtained. The quadratic error is practically stabilized in about 200 epochs, and further training does not appreciably enhance the results obtained. The training process was completed in about 10 min on a Pentium 800-MHz computer.

8.5.4 Network-performance Testing

Several experiments were performed to test the operation of the ground detection system. In these experiments the network output was calculated while the foot was moving towards the ground in the working conditions included in the test sets (see Sect. 8.5.2). The time needed to compute the estimates was less than 1 ms in all cases. The network output was compared with the data taken from the calibration force sensor to characterize the precision of the virtual sensors. This section presents the results obtained in the experimental tests. The discussion of these results can be found in the next section.

Virtual-switch Test

In this case, the detection system's performance is evaluated by the use of the contact force, F_C, defined as the force measured by the calibration force sensor at the instant the network output is activated (*i.e.*, when the network estimates that the force exerted exceeds the force threshold, F_T). The difference between the force threshold and the contact force is the system's force error F_E. Figure 8.3 illustrates these magnitudes in a representative test experiment. The test has been performed with several network weight sets (obtained in different training processes), corresponding to different force thresholds. Figure 8.4 depicts by the use of error bars the mean F_C and the standard deviation about the mean F_C, for the different operating points included in test set A and for several force thresholds, F_T. This graph also shows the straight line that represents the ideal behavior of the system. In these experiments the standard error about F_T, measured for different operating points, varies from 4 to 8%, depending on F_T. Figure 8.5 shows the same data for the examples in test set C. In this case the standard error varies from 3 to 14% of the total force range. Figure 8.6 shows the F_C measured in several experiments for the same operating point (*i.e.* for all the examples included in test set B). The standard error about the mean F_C, which is used to characterize the repeatability of the virtual sensor, is between 2 and 4% of the force range, depending on F_T.

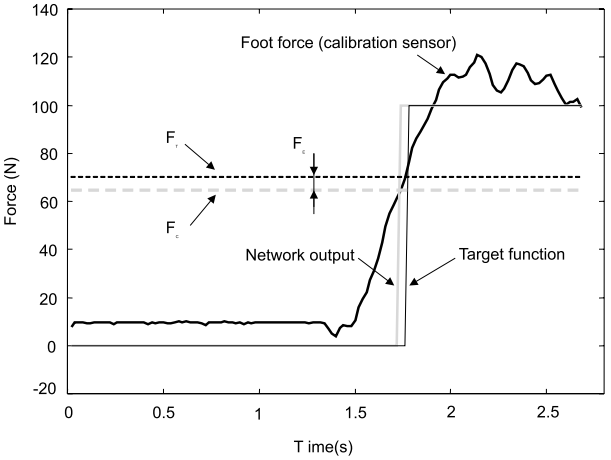


Fig. 8.3. Magnitudes tracked in virtual-switch testing

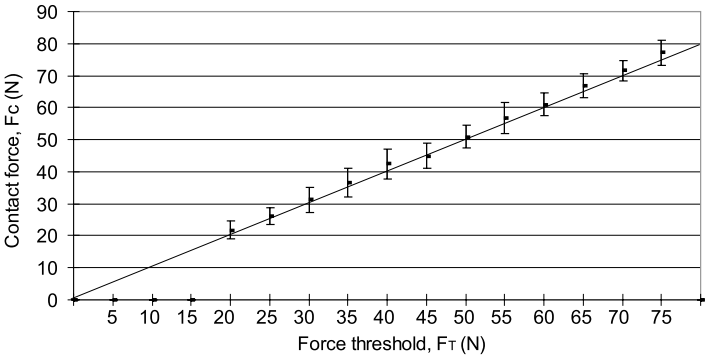


Fig. 8.4. Contact forces measured in virtual-switch testing (test set A)

Virtual One-axis Force-sensor Test

Figure 8.7 illustrates with a representative example the accuracy of the estimate obtained with the virtual one-axis force sensor. The standard error of the estimate, the mean force error and the coefficient of determination (multiple correlation coefficient) have been used to characterize the accuracy of the estimate for each test example. The estimate's standard error is below 5% of the total measurement range in all the test examples studied, while the maximum force errors observed are always below 10% of this range. Figure 8.8 shows these statistics for all the examples included in test set A. In all

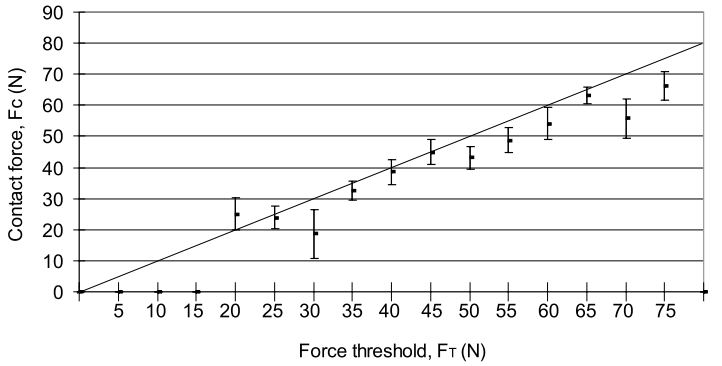


Fig. 8.5. Contact forces measured in virtual-switch testing (test set C)

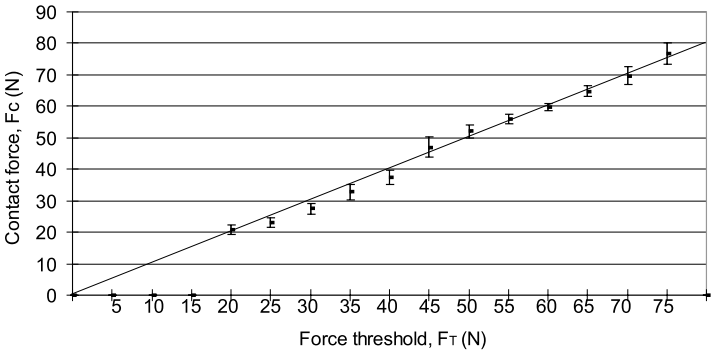


Fig. 8.6. Contact forces measured in virtual-switch testing(test set B)

of these cases, the mean error is around 2% (and it is always positive), and the standard error is between 2 and 4%. Figure 8.9 shows these statistics for all the examples in test set C. As depicted in this figure, the mean error and the standard error are quite similar to those calculated for test set A. Figure 8.10 shows the results obtained when the virtual sensor was evaluated in the same way as the virtual switch for all the examples in test set C. Figure 8.11 represents the dependence of the coefficient of determination on the operating point.

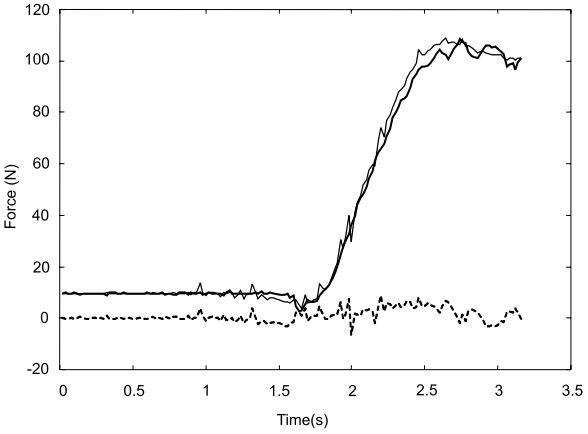


Fig. 8.7. Example of one-axis force estimation. Foot force measured by the calibration sensor (*thick solid line*), network estimate (*thin solid line*), and force error (*dashed line*).

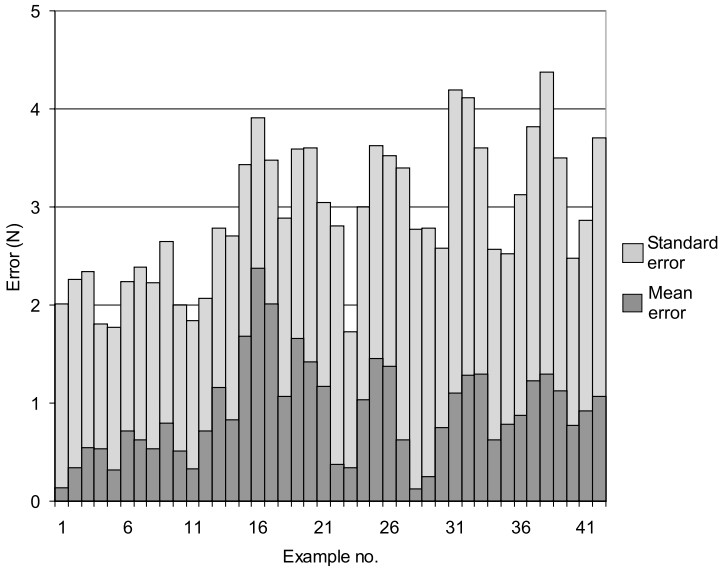


Fig. 8.8. Standard and mean error of the one-axis force estimate in the examples in test set A

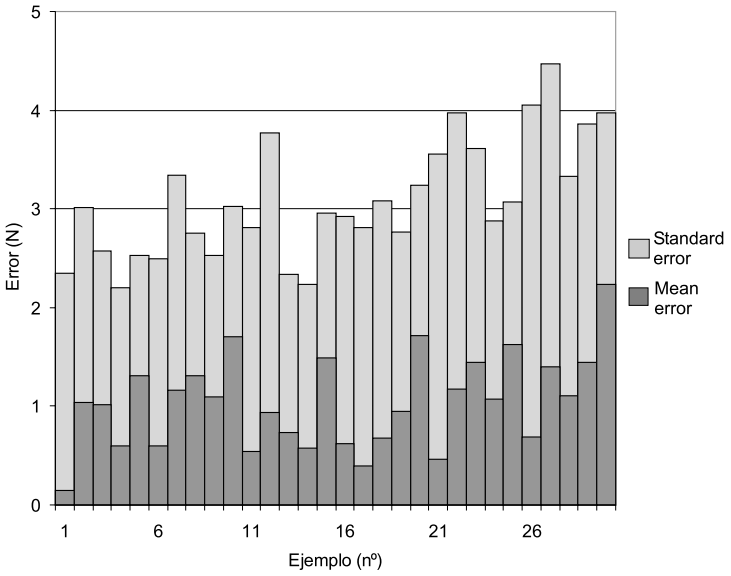


Fig. 8.9. Standard and mean error of the one-axis force estimate in the examples in test set C

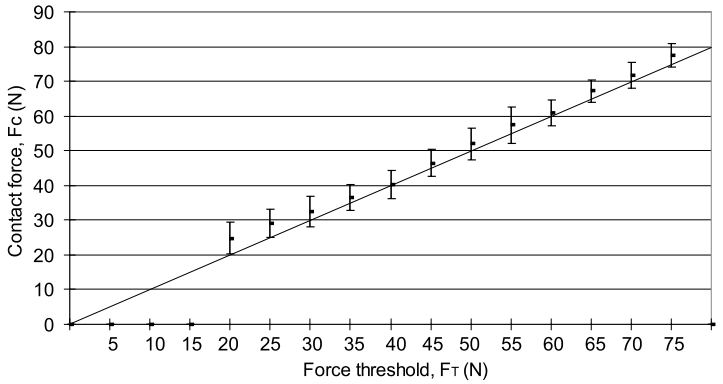


Fig. 8.10. Contact forces measured in testing of the virtual one-axis force sensor (test set C)

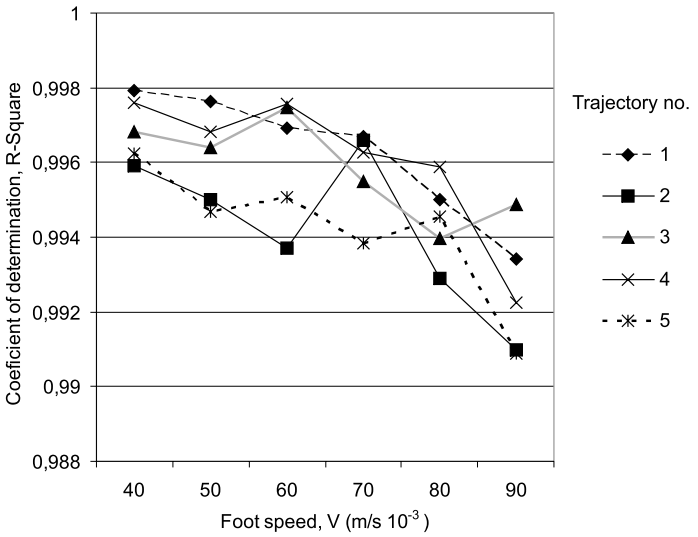


Fig. 8.11. Coefficient of determination of the one-axis force estimate as a function of foot speed for different trajectories

Virtual Three-axis Force-sensor Test

The accuracy of the estimate of the three foot force components has been characterized in the same way, with the statistics calculated for each of the three network outputs. Figure 8.12 shows how the network estimations mirror the calibration-sensor measurements in a typical example of foot/ground collision. The estimate quality is similar for the test examples included in sets A and C; the mean error is below 2%, and the standard error is below 5% of the total force range in all the test examples. Figure 8.13 shows these statistics for test set C.

Locomotion on Irregular Terrain

The immediate application of these three virtual sensors is to detect ground contact. Therefore, in order to compare their ability to accomplish this task, they have been evaluated with the method described above in this section. Table 8.1 summarizes the results obtained when the virtual sensors are evaluated using 12 different F_Ts. The results are characterized by the standard error of the measured contact forces about F_T in the case of test sets A and C, and about the mean contact force in the case of test set B.

The feasibility of this method to make possible the locomotion on irregular terrain has been tested experimentally. In the first experiment performed, the

Table 8.1. Standard error of the contact forces measured in ground detection experiments

	Switch	One-axis force sensor	Three-axis force sensor
Test A	6.5%	4.1%	5.3%
Test B	2.8%	2.9%	2.7%
Test C	8.3%	5.7%	6.9%

SILO4 robot walks over an unknown irregular terrain shown in Fig. 8.14 with the only help of the one-axis virtual sensor. Figure 8.15(a) plots the estimates of the one-axis virtual force sensor and the forces registered by a calibration sensor in a fore foot. The estimates are computed only during the final part of the transference, when they are required to detect ground contact. The leg downwards motion finishes when the estimated foot force reaches 50 N. In the second experiment the calibration sensor was used for ground detection in order to facilitate a comparison (see Fig. 8.15(b)).

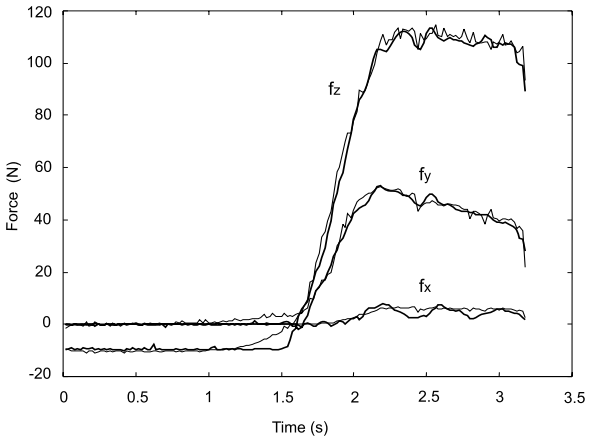


Fig. 8.12. Example of three-axis force estimation. Foot-force components measured by the calibration sensor (*thick solid lines*) and network estimate (*thin solid lines*).

8.5.5 Discussion

The results presented in the previous section demonstrate that the virtual-sensor design described here is not only a feasible but a functional method for detecting foot/ground contact. The experimental test confirmed that a feed-forward network with one hidden layer and five hidden neurons is enough to

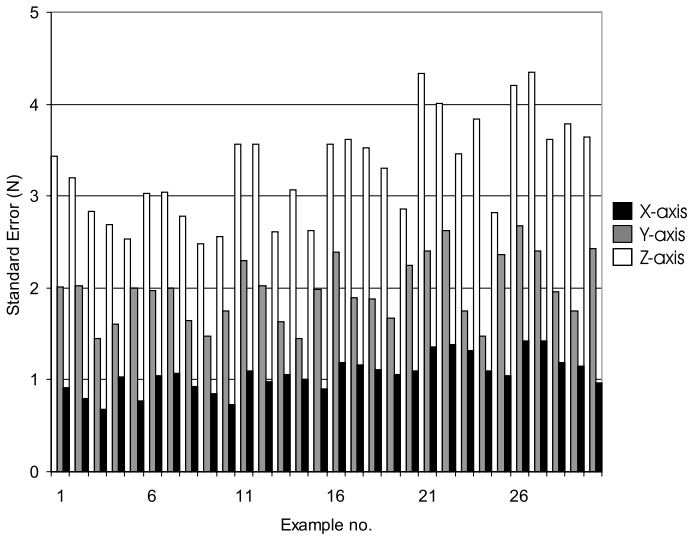


Fig. 8.13. Standard error of three-axis force estimation in the examples from test set C

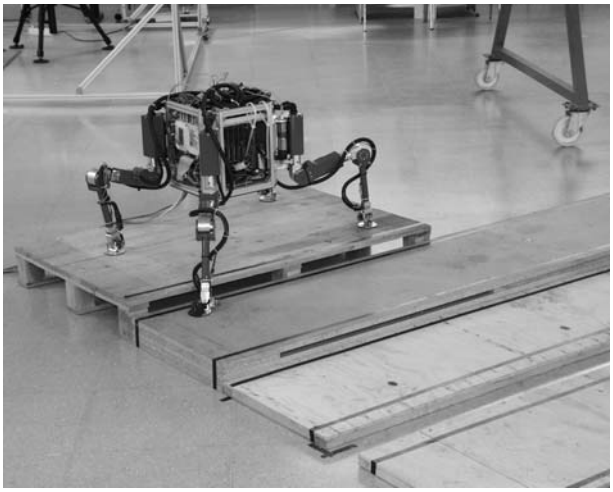


Fig. 8.14. The SILO4 experimental legged platform

model the system and gives satisfactory results in this application. The computational burden added by the use of this neural network is very low, since the complete time needed to compute the estimation represents less than 5%

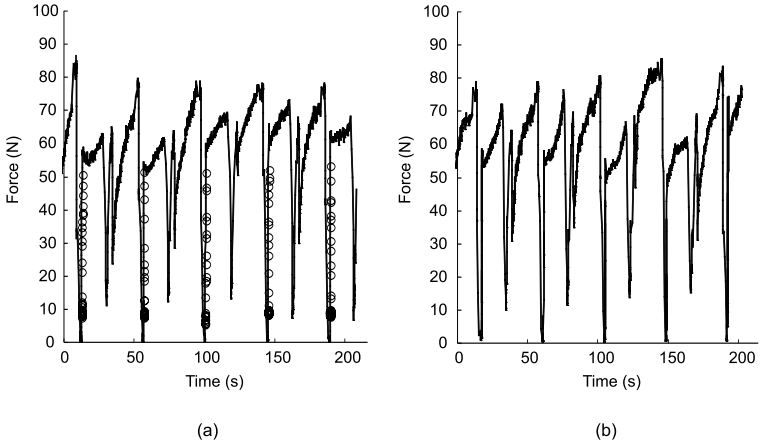


Fig. 8.15. Forces registered during the locomotion on irregular terrain: calibration sensor measurements (*solid line*) and virtual sensor estimates (*circles*). Terrain adaptation is performed employing: (**a**) virtual sensors estimates; (**b**) calibration sensor measurements.

of the control loop period. The elimination of sensors, cabling and electronic equipment is worth this slight increase in the computational burden, specially in systems where simplicity is mandatory, like legged robots; also in applications where price must be kept low. The set of input magnitudes employed, has made possible the modelling of various effects, like viscous and position dependent friction, gravity, *etc.* However, the correct modelling of effects appearing at very low joint speed, like static friction and stick-slip behavior, has not been confirmed. This is because the near-zero joint speed condition does not appear in the range of foot speed and trajectories considered for ground detection purposes, at least in this leg kinematic configuration. In the most general case of force estimation, a joint speed might descend to zero during the generation of a given foot trajectory; then, the calibration examples should contemplate this situation, and results should be carefully validated. The results obtained also permit the comparison between several kinds of response behaviors.

The accuracy of the virtual switch's estimate of the instant of collision proved to be very satisfactory in the operating conditions considered in the training set. The experiments show that there is no clear systematic dependence between contact forces and foot trajectory and speed, and the dispersion of the results due to differences between operating points is acceptably small (see Fig. 8.4). The repeatability of the virtual switch is also very satisfactory (see Fig. 8.6). However, the estimate's accuracy gets worse as the operating conditions differ more widely from the operating points included in the training set, a fact that reveals the poor generalization features of this virtual

sensor (see Fig. 8.5). The network's ability to infer the correlation between magnitudes is hampered by the limited information contained in the binary signal used to train the network. Our experiments also show that this virtual switch can be calibrated for different force thresholds inside an adequate force range. No clear systematic dependence between the force threshold and the precision of the estimate was found. Simplicity of experimental setup is the main advantage of this virtual sensor, as only a physical switch is needed as calibration sensor.

The one-axis virtual force sensor showed very good precision in the experiments mimicking the operating conditions considered in the training set (see Fig. 8.8). In addition this force estimation system showed excellent generalization features, obtaining a similar degree of precision for any trajectory and foot speed throughout the entire normal working range. This virtual force sensor is superior to the virtual switch in every aspect studied, especially generalization (see Fig. 8.9). While it is evident that the estimate's accuracy depends on the foot speed (the higher the speed, the lower the correlation), the dependence on foot trajectory has not been found to be systematic (see Fig. 8.11). The enhanced performance of this virtual force sensor is the consequence of the improved richness of the analogue information used to establish the relationship between magnitudes. The only disadvantage of this virtual sensor compared with the virtual switch is the higher complexity of the experimental additional hardware needed.

In the case of the virtual three-axis force sensor, our experiments show that the accuracy of the estimate is high (5%) and similar throughout the whole operating range (see Fig. 8.13). The precision of the estimate of each force component is also quite similar to the precision obtained with the one-axis force sensor. The use of this virtual sensor therefore brings further enhancement over the virtual one-axis virtual sensor while maintaining estimate precision. Its longer training time and the more complex experimental hardware it requires are its only drawbacks.

Considering these results, the number of operating points selected to make up the training example set was found to be sufficient to describe the full operating range for the virtual force sensors. Specifically, the virtual sensor is able to compute accurate estimates for any rectilinear foot trajectory contained in the leg workspace and for any foot speed suitable for use during locomotion (0.025 – 0.1 m/s). Using the Levenberg-Marquardt learning algorithm, the network can be trained with the training set described above in a short time with acceptable memory requirements. This fact makes the inclusion of a higher number of examples describing a wider variety of trajectories a suitable possibility. In all cases, the complete calibration process proved to be fast and straightforward.

The final comparison between virtual sensors in Table 8.1 shows that the virtual one-axis force sensor is the most effective option for detecting the ground contact. The experiments stated that a walking machine can traverse an unknown irregular terrain using this virtual sensor. In the experiments the

forces registered at the end of each leg transference exceeded the specified F_T (50 N) up to 10 N . However, this effect is also observed when the physic sensor is used to detect the ground (see Fig. 8.15). This is mainly due to the control loop period employed to check forces and stop the foot downwards motion. The errors observed in the second case are yet smaller, since this sensor was used to calibrate the virtual sensors. In any case, these errors do not affect the walking machine adaptation, and the final conclusion is that locomotion is not affected by the use of either physic or virtual sensors; because this method enhances the machine's robustness and the quality of the available sensorial information, machines equipped with virtual sensors can actually function better.

8.6 Conclusions

The development and testing of a ground-detection system for walking machines has been described. Joint-position error, joint speed and joint-position signals have been chosen as a feasible set of inputs for a virtual sensor to estimate the forces exerted by the robot's foot. These inputs can be drawn from the joint-position sensors available in most robotic systems, so the implementation of this virtual sensor does not result in an extra hardware burden.

Data-driven virtual sensors have been found to be a suitable choice for developing a common sensorial system for several robotic platforms. The use of feed-forward neural networks to process the input information and generate a convenient output has been proved experimentally to be an adequate solution in a real walking robot. A calibration procedure has also been established with the aim of facilitating the tuning of these virtual sensors. The calibration procedure allows the detection system to be calibrated with a minimal experimental setup in a short period of time, and the required experiments are quite similar to the normal locomotion process. The results obtained in this first approach have been found to be satisfactory and are currently being used in the SILO4 walking machine for ground detection. The good quality of the force estimates obtained with this virtual sensor makes it a feasible alternative to the use of physical sensors. Because this method is easy to adapt to other machines, it is a good choice for simplifying the hardware of a servo-controlled system, or for providing it with low-cost sensor redundancy.

9

Human-machine Interfaces

9.1 Introduction

Robots were devised to replace human operators in complex, fatiguing and hazardous tasks, and initial estimates posited that robots could operate fully autonomously in isolated environments. However, today industry and services are demanding robots that can operate together with humans in the same working scenario. This presents the problem of the two partners' communicating with one another through a dialogue system normally referred to as the Human-Machine Interface (HMI).

The first HMIs were based on the supervisory control concept, in which the operator divided the overall task into small sub-tasks that could be successfully achieved by the robot. The operator was always in charge of the robot and was required to have a thorough knowledge of the robot's capabilities in order to perform effective and efficient control. This scheme (devised for tele-manipulators rather than tele-operated vehicles) is no good for robots designed to work as advanced tools for specialists in fields non-related with robotics (Blackmon and Stark, 1996). The latest attempt to improve human-machine interaction consists in inserting the human model into the control loop by considering both the mechanical model (limbs, muscles, *etc.*) (Prokopiou *et al.*, 1999) and the model of human behavior (Rosenblatt, 1997). This is referred to as Human-Robot control architecture, and it considers the operator as just another module that gives decision-making or human perception as additional modules that choose among a number of potential actions (Fong *et al.*, 1999).

Multi-modal operator interfaces and supervisory control are widely used in tele-operation and HMIs for mobile robots, and they appear to be adequate for walking machine tele-operation. Multi-modal interfaces offer the operator a variety of control modes and displays. These control modes include individual actuator control (joint control), co-ordinated control (leg control), task control (body motion, trajectory execution), *etc.*, while the displays provide numerical information (joint and foot positions, stability margins), geometric

information (robot pose), haptic information (which is a technique for touch and force-feedback information), *etc.*

Collaborative control is a step farther in the development of HMIs. This new concept introduced by Fong *et al.* (1995) considers that both the human and the robot controller are at the same hierarchical decision level, collaborating with each other to perform missions and accomplish objectives. The robot follows high-level tasks stated by the human operator, but it asks the human questions about how to achieve the task, how to use unclear sensorial information, *etc.* The robot controller decides how to use, modify or reject the human's indications as well. Thus, the robot assists the operator as a true expert and both of them work together to achieve the task (Fong *et al.*, 1995).

Walking machines are a special type of mobile robots. They are characterized by some advantages over wheeled robots and some disadvantages stemming from their complexity and inherent slowness (see Sects. 1.3 and 1.4). The omnidirectionality of legged robots and their ability to traverse very rough terrain are some of the most important features of this kind of machine. Consequently, these properties must be exploited to obtain clear advantages over conventional wheeled or tracked vehicles. However, the complexity of legged machines makes them difficult to operate for a robotics layman, a fact that reduces their applications in areas such as transport, construction, *etc.*

Although there is great activity in walking machine technology, especially in the areas of gait generation, leg design, force control, *etc.*, the activity in HMI for walking vehicles is very low indeed. Nevertheless, references (Bares and Wettergreen, 1999; Fong *et al.*, 1999; Takanobu *et al.*, 1999) furnish examples of the activity carried out. Operating a walking robot should be as easy as commanding a crab angle trajectory at a given speed. However, stable walking means generating adequate footholds and leg motion sequences, adapting the machine to terrain irregularities, avoiding stepping on forbidden areas, *etc.* All these tasks are very difficult for a human to accomplish and they must be in the charge of the robot's controller, which can take care of them by running specific continuous, discontinuous (see Chap. 3) or free gait algorithms (see Chap. 4). Nevertheless, when a machine performs free gait algorithms, which are especially adequate for walking on uneven terrain containing forbidden areas, leg deadlock can occur (see Chap. 4) and the attendance of a human becomes essential. Overcoming such a problem normally requires the operator to command the system at a very low level (joint level). Thus, a sort of multi-modal interface seems to be the best choice for a walking robot HMI. On the other hand, the information about the environment is crucial to perform terrain adaptation, to optimize locomotion, and to preserve the safety of the mission. However this information can be unavailable or incomplete due to a poor sensorial system. Therefore, the controller should be able to interchange information with the operator in order to complete its internal representation of the environment or to demand an appropriate command. Alternatively, commanding such a complex machine is a difficult and demanding task for the operator. Thus, the controller should be able to

reject, if necessary, erroneous or counterproductive operator commands, suggest possible solutions, and negotiate with the operator to get to an adequate instruction. This is a sort of collaborative control, which looks very interesting for the design of walking machine HMIs.

This chapter focuses on the design of an HMI for a walking robot. The HMI is based on a multi-modal interface that incorporates collaborative control features. Special attention has been paid to manoeuvrability and the HMI's human-friendliness. Operation has been improved by the use of a friendly graphic interface that assists the human operator in steering the robot through uneven terrain. Section 9.2 introduces the HMI, which is based on the system and gait characteristics as well as the concepts of both multi-modal and collaborative control. Sections 9.2.1 – 9.2.10 describe the different modules that compose the HMI. Section 9.2.11 depicts the operation of collaborative controller included in the HMI. Finally, some conclusions are reported in Sect. 9.3.

9.2 Human-machine Interface and Collaborative Controller

The graphic HMI developed in this chapter has two main features. First, it is useful to develop, simulate and optimize free gait algorithms for a quadruped robot (see Chap. 4 and (Estremera *et al.*, 2002)). Second, it is also useful as an operator interface to command and monitor the movements of a quadruped in real time while it is executing a free gait algorithm. These two functions are integrated in such a way that the Graphic User Interface (GUI) can receive data and generate commands for both the simulator and the real robot.

A collaborative controller is proposed to improve the HMI in order to achieve the following two main goals:

1. Make the steering of a legged robot as easy as possible, allowing the operator to guide the robot along complex paths composed of straight segments, arcs as well as pure rotations. This includes the following sub-goals:
   - Provide a method to combine different gaits (crab gait, turning gait, and spinning gait, (see Chaps. 3 and 4) and to modify its parameters (crab angle, turning radius, *etc.*).
   - Make it possible for the operator to ignore the particular characteristics of walking machines and gait generation allowing the robot's controller to disregard inadequate operator's commands or suggesting the appropriate ones.
2. Improve the terrain adaptability of the walking machine, without increasing excessively the operator's workload. The number of parameters that can be adjusted in a walking machine in order to optimize locomotion on irregular terrain is very high. To facilitate this tuning the collaborative controller should accomplish the following sub-objectives:

- Help complete the information about the environment needed to optimize the robot's locomotion. To achieve this, the collaborative controller can request information about general characteristics of the terrain (average slope, degree of irregularity and so on) which cannot be deduced from the limited and local information provided by the sensors on board. The collaborative controller can also ask the operator about some local aspects of the terrain when sensorial information is not clear or available. Finally, it can solicit a command from the operator when a potentially risky situation is detected.
- Conversely, the collaborative controller helps the operator to get to know some local aspects of the terrain, or the history of the traversed terrain, in order to make decisions. In addition, the collaborative controller can suggest the operator possible commands to execute, in order to lighten the operator's workload.

One example in which the collaborative controller permits the operator to ignore the characteristics of gait generation is the following. The gait generation module is the higher level of autonomous control implemented in the robot, and the operator is in charge of path planning. The free gaits implemented are performed autonomously and the operator only determines certain gait parameters to define the path, such as the crab angle, the turning center, *etc.* or certain requirements, such as the absolute stability margin. However, certain decisions of the operator can hamper gait planning and lead to a deadlocked situation. For instance, in certain situations the crab angle or the minimum stability margin commanded by the user can make difficult to find adequate footholds. As a result, the gait planner could infer that a deadlock situation is probable. In such cases, the collaborative controller can ask the human operator about relaxing these parameters or the controller can decide to change them on its own; for instance, if the human does not respond within a reasonable period.

One example in which the controller must collaborate with the human to improve the performance is described next. The robot can make an estimate of the terrain by using the discrete foot contact points. If all contact points are at the same height, the robot could be walking on flat terrain. In such a case, the robot should use a very short step height and so increase its overall speed. However, it is possible to obtain the same foot heights even when walking on uneven terrain, because the knowledge refers to only a few discrete points. The collaborative solution here is that the controller could either ask the human operator about the properties of the terrain. A human operator can easily deduce the properties of the terrain by analyzing simple TV images. Alternatively, the collaborative controller can propose a possible command (step height correction) and wait for the operator's approval. A similar case would be when the machine is walking on a slope. The controller can infer, based on foot positions, that it is going up or down a slope. In these cases, it is important to adjust the robot's height to the terrain to optimize

the gait, but it can be very difficult for the robot to assure these properties of the terrain. Again, the operator can help the controller to achieve the mission. These decisions could be made autonomously by the robot but, due to the type of sensors installed, they would be based on very local aspects of its environment, or on a history of the environment it has traversed. This is why an autonomous decision of the robot may be inaccurate or belated. Additionally, poor sensorial information can sometimes jeopardize the safety of the mission. In order to guarantee a safer operation, the controller can ask for the operator's collaboration (either ask for information or solicit approval for a proposed action) when a risky situation is deduced from sensorial data.

To conclude, if the operator is not able to collaborate, the robot would operate autonomously, with the consequent limitations and risks. The operator collaboration, if available, can help to complete the mission in a safer and more efficient way. On the other hand, a fully tele-operated solution would demand too much attention from the user to operate a very complex machine with poor sensorial feedback.

In the definition of the HMI attention has mainly been paid to high-level commands and magnitudes with the intention of lightening the operator's workload. Consequently, this is in a way a collaborative-gait-level oriented interface. However, different levels of control might be available for the operator in order to perform the variety of actions required for facing different situations along diverse stages of the development of a walking machine. Therefore, the HMI is also based on a multi-modal structure in which the operator can act at different control levels ranging from motion of an individual axis to autonomous walking.

The GUI is organized in multiple modules following the work done by Bares and Wettergreen (1999), so that appropriate commands or information for a particular type of function are grouped together in areas or environments in which a set of events occur. Information about the vehicle's status is provided to the operator through the graphic display, the numerical display and the sensorial context. The areas defined as control-simulation context, gait context, walk context, and actuator context correspond to different levels of vehicle control. The command line interface and the collaborative context represent alternative methods for getting information and generating control commands. In addition to these modules, a terrain modelling context and a graphic control context have been included in the GUI (see Fig. 9.1).

9.2.1 Graphic Display

This display is used to provide high-level information about the status of the robot and gait generation in an intuitive, friendly manner. The graphical representation is used mainly to facilitate the operator to get to know about the progress of gait generation (leg sequencing and foothold search). It is also used to get quickly some information about foot positions, stability

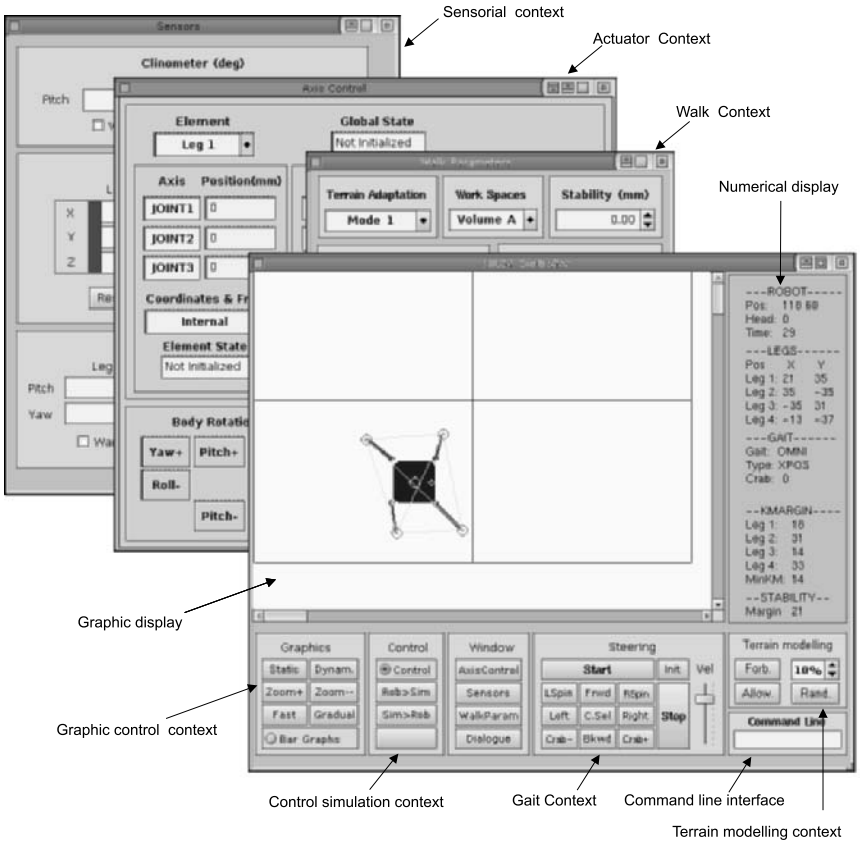


Fig. 9.1. Parts of the graphic user interface

margins, forbidden areas, *etc.* Based on this information, an experienced operator can change some gait parameters (crab angle, stability margin, *etc.*) to optimize the speed, or the safety of the locomotion. The graphic representation depends on the way the free gaits are generated. As mentioned in Sect. 4.2.1, only the horizontal projections of the robot are considered in gait planning, so accordingly a bi-dimensional graphic representation is enough to offer adequate information. The employment of discontinuous gaits along with the consideration of forbidden areas make possible the use of a bi-dimensional representation of both the robot and the terrain. Additionally, a bi-dimensional representation is a computationally low-cost solution for a walking robot controller. However, direct visual (or vision system) feedback is mandatory at this stage of development to control the machine effectively. Two different graphic representations, displayed at the top right area of the main window, are included in this graphic interface.

Static Representation

This representation shows a simple picture of the robot and some graphic information that is useful for testing and monitoring free gait algorithms. This representation consists of a top-view of the robot in which the observer is located above the body's reference frame. The following elements form this screen (see Fig. 9.2):

- **The robot:** a simple bi-dimensional scheme of the horizontal projection of the robot shows the body and the positions and status (transfer or support) of the legs. Legs in their support phase are indicated with a circle around their feet. The position and orientation of the robot's body remain (static) on the screen to facilitate observation. The horizontal projection of the leg workspaces, the support polygon and the horizontal projection of the center of gravity (surrounded by a circle with radius equal to the minimum desired stability margin, $S_{SM}^{\min}$) are also represented.
- **Foothold planning:** when an optimal foothold is being determined for a foot in transfer (see Chap. 4), the following elements (points and lines) can be seen on the screen (see Fig. 9.2):
  1. Several straight lines that show geometrical restrictions on the position of the future foothold (these lines define the foothold restricting areas described in Chap. 4).
  2. Some points that represent possible footholds that satisfy those restrictions.

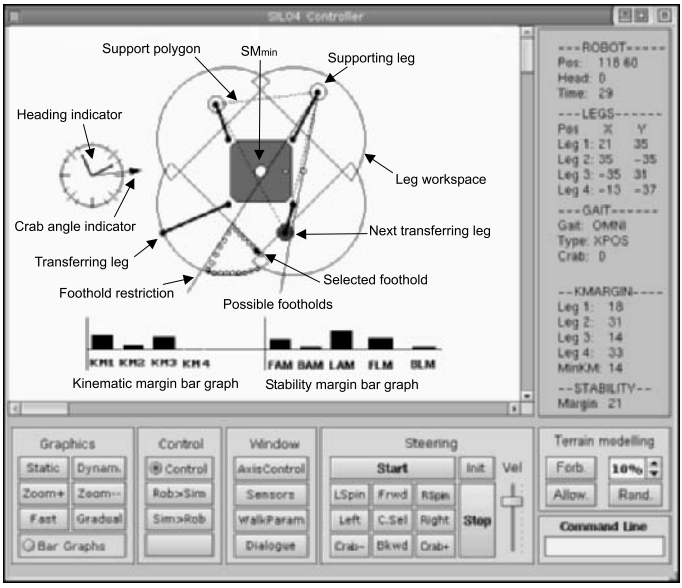


Fig. 9.2. Graphic representation: static representation

3. A point that indicates the position of the optimal foothold.

- **Sequence planning:** the next leg to be lifted is marked in a different color.
- **Heading indicator:** this indicator shows the orientation of the axes of the external reference frame. Since in this representation the robot's body remains static, the heading indicator acts as a compass helping the operator to know the orientation of the robot.
- **Crab angle indicator:** a vector that indicates the crab angle, showing clearly what direction the robot is moving in with respect to the external reference frame.
- **Bar graphs:** several graphic indicators show relevant magnitudes employed for gait planning: the kinematic margins (KM) of each leg, the absolute stability margin (S_{SM}), plus other stability measurements.

Dynamic Representation

The dynamic representation is an intuitive depiction of the robot in its environment. The main characteristics of this representation are described below (see Fig. 9.3):

- The observer is located in an external reference frame; therefore, there is a correspondence between the trajectories described by the robot on the

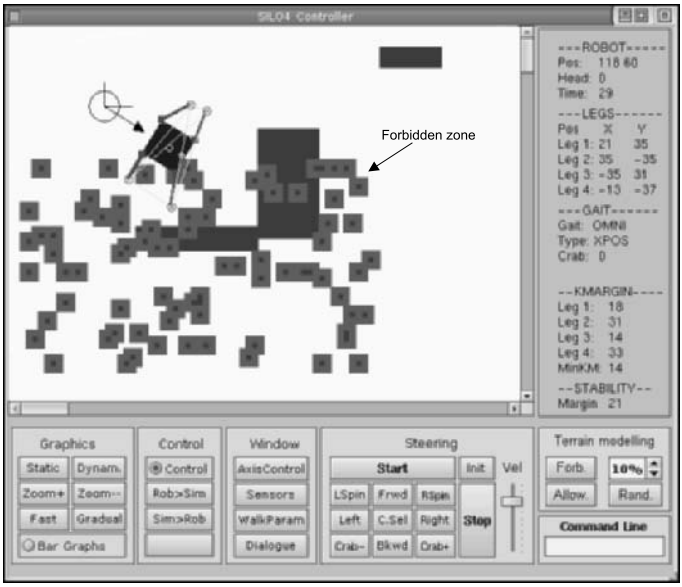


Fig. 9.3. Walking over forbidden cells: dynamic representation

terrain and the trajectories described by its graphic representation on the screen.
- A simulated environment with obstacles (impracticable terrain) can be created in order to determine the optimal trajectory for reaching a goal, as well as to evaluate whether the operator will be able to follow a particular trajectory, manoeuvring to avoid collisions.
- Forbidden areas may be included in the simulated terrain to determine if the robot will find the necessary footholds to follow a specific trajectory. The terrain is divided in square cells and the color of each cell represents the type of terrain contained in it (see Sect. 4.2.2).
- The crab angle indicator described in Sect. 9.2.1 helps to steer the vehicle.

9.2.2 Graphic Control Context

This area groups together some commands related with the graphic representation of the robot (see Fig. 9.1). It includes controls to zoom in and out, to change the position of the observer, and to change the drawing refresh rate to facilitate detailed observation or speed up the simulation. The operator can also switch between the static representation and the dynamic representation.

9.2.3 Numerical Display

This display gives accurate numerical information, although in a non-intuitive manner. It is especially useful for detailed analysis of gait generation and for commanding the robot at a low level. The following numeric information is printed out on the right side of the main window (see Fig. 9.3):

- Body co-ordinates and heading referring to the world reference frame, estimated by odometry.
- Leg co-ordinates referring to the robot's reference frame.
- Simulated time since the beginning of the motion.
- Type of gait, crab angle, turning center and turning radius.
- Kinematic margin of each leg and the smallest kinematic margin.
- Absolute stability margin, S_{SM}.

9.2.4 Sensorial Context

The displays grouped in this context are used to monitor sensorial information. Data incoming from the two-axis inclinometer, force sensors and foot potentiometers are shown as numerical indicators. This low-level information about the robot's status, useful in case of system failure and in the debugging phase, is presented in a separate window to avoid displaying an excess of data to the operator in normal conditions (see Fig. 9.4). In a collaborative system, this context could warn the user whether sensor data have entered a dangerous range. The controller could consider possible operator's suggestions or act on its own.

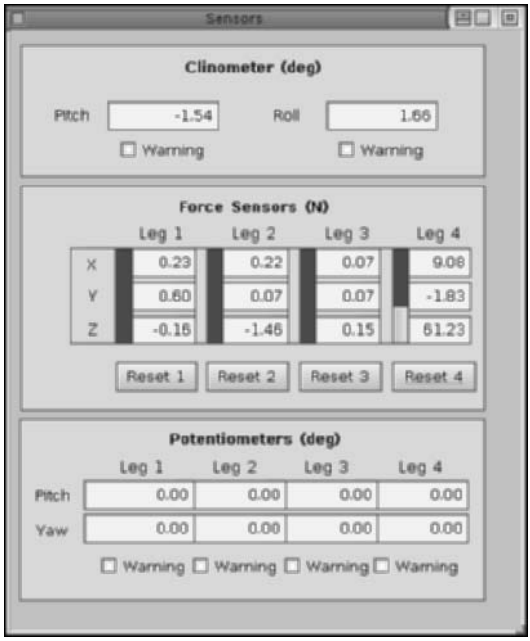


Fig. 9.4. Sensorial context

9.2.5 Terrain Modelling Context

In order to monitor the progress of the free gaits on terrain with forbidden areas, a simple mechanism for modelling the terrain was included in the controller (see Sect. 4.2.2). Commands placed in the terrain modelling context allow the operator to create a distribution of forbidden cells (see Fig. 9.1). This distribution may be done randomly, with the possibility of choosing the proportion of forbidden cells, or manually by the use of a cursor to simulate a particular terrain. The latter option is useful for simulating whether it is possible to steer the robot through a particular terrain with forbidden areas, helping to find the best way to reach a point before moving the real machine.

9.2.6 Control-simulation Context

Some commands have been added to facilitate the coexistence of the simulator and the controller. They make it possible to switch between these functions and select the simulator as a master or slave of the real robot (see Fig. 9.1). These commands have helped in the performance of some experiments and their previous simulations (Estremera *et al.*, 2002).

9.2.7 Gait Context

The objective of the commands included in the gait context (see Fig. 9.1) is to allow a human operator to steer the robot in real time. The operator just takes care of the machine's trajectory and speed, giving simple, intuitive orders as if the robot were a wheeled vehicle. Currently, this context is the vehicle's highest level of control. The crab gait, the turning gait and the spinning gait can be easily combined to follow a trajectory or to reach a point of the environment while avoiding obstacles. The operator can use the following controls in real time to drive the machine:

- **Crab angle:** the crab angle can be increased or decreased by a fixed amount by clicking on two buttons on the screen. This is useful for making small corrections in the trajectory of the robot or steering it along a smooth trajectory without losing its orientation. Four other buttons allow the operator to select four pre-determined crab angles so the robot will move along its x and y axes in both the positive and the negative directions.
- **Turning and spinning direction:** a clockwise or counterclockwise free spinning gait can be selected directly through two buttons. When executing a turning gait these buttons are used to select the turning direction.
- **Turning center:** the operator can select the center of rotation of the turning free gait by using a cursor; this is an intuitive way to get around obstacles. However, this procedure is not advantageous when the robot must follow a complex path composed of different arcs. Therefore, we have evaluated another method in which the position of the center of rotation can be moved along the transverse axis of the machine. This is equivalent to changing the turning radius. Thus, we can steer the robot as if it was a conventional wheeled vehicle.
- **Gait speed:** this control defines the average speed of the machine. The machine's average speed depends heavily on the terrain, the crab angle, *etc.*; therefore, the gait speed is only a kind of speed recommendation made by the operator.

Simulations and experiments have shown that with just these control commands it is possible to drive the robot to reach a given goal or to follow predefined paths using both rotations and changes in the crab angle.

9.2.8 Walk Context

The walk context is used to define the general specification of the robot's locomotion, generating commands in order to optimize the gait for a particular terrain or situation. Thus, the walk context represents a lower level of locomotion control, and consequently it has been implemented in a separate window to minimize the controls presented to the operator. The commands in this context are used to change parameters related mainly with terrain adaptation, stability and speed. These commands are (see Fig. 9.5):

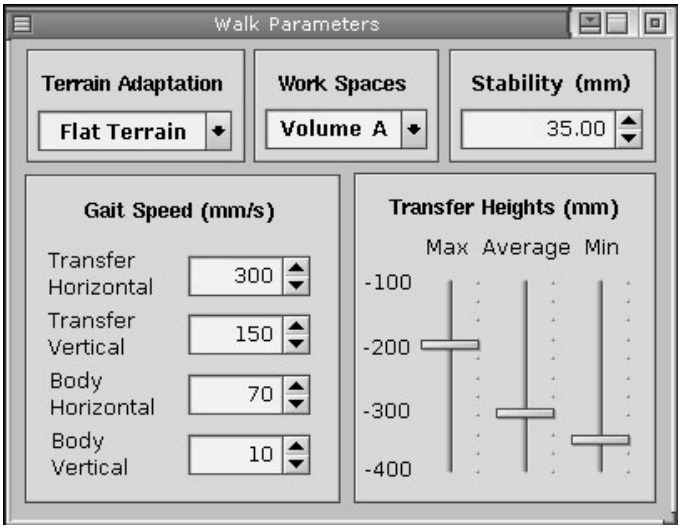


Fig. 9.5. Walk context

- **Terrain adaptation mode:** with this command different possibilities for ground detection and altitude/attitude control can be defined. The operator can choose a selection that goes from a flat terrain mode (where faster locomotion is achieved because it is not necessary to maintain body altitude and attitude nor to detect foot contact) to a rough terrain mode.
- **Leg workspaces:** different subsets of the total leg workspace can be selected in order to improve omnidirectional locomotion by the use of wide (although short) workspaces or to enhance adaptation to irregular terrain by the use of high (although narrow) workspaces.
- **Minimum stability margin:** the minimum absolute stability margin imposed to the gait planner, $S_{SM\min}$, can be varied in order to increase the stability of the machine or to diminish the probability of deadlock.
- **Body height and step height:** with these controls the operator can change the average body height and step height to optimize locomotion on slopes, even terrain, rough terrain, *etc.*
- **Velocity:** the user can adjust the velocity of the different elementary motions used during locomotion.

9.2.9 Actuator Context

Using the commands included in the actuator context, the operator can manually move any individual axis or groups of actuators (legs, for instance). As mentioned above, one of the main objectives of this interface is to simplify the safe steering of the robot (collaborative control). However, a lower level

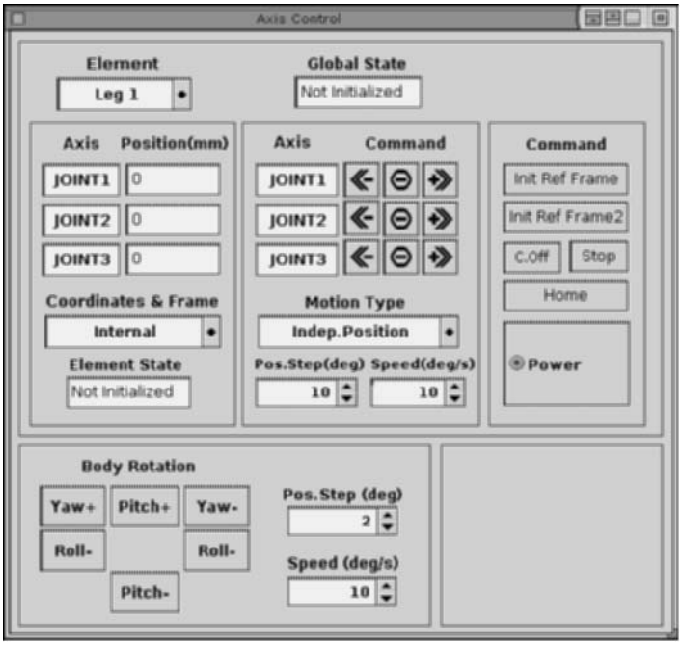


Fig. 9.6. Actuator context

control is required in the debugging phase or in the case of system errors. In order to abstract the user from these commands, the actuator context has been implemented in a separate window (Axis Control) as a multi-modal interface (see Fig. 9.6). The following are the main features of the actuator context:

- The operator can select the Element of the robot (leg or body) that will be affected by subsequent commands.
- It is possible to switch among four Motion Types: (1) independent-axis position motion, (2) independent-axis velocity control, (3) co-ordinated foot motion and (4) straight-line foot motion. If the body has been selected, only the straight-line motion is available.
- Both the position increment (Pos.Step) and the speed (Speed) of the motion command can be adjusted using two numeric controls.
- The motion controls, placed on the left of the actuator context, allow the operator to move the selected part of the walking robot. These commands are split into three arrays of buttons (increase, stop, decrease) used to move the feet depending on the selected operation mode: (1) along x, y or z axes, in the case of co-ordinated or straight-line motion; (2) around joints 1, 2 and 3 in the case of individual axis movement.

- The position of the selected leg is displayed beside the motion controls in internal or Cartesian co-ordinates referred to different frames.
- The rotation of the body around the x, y and z axes can be controlled by the use of the motion controls located on the left side of the actuator context (Pitch-Roll-Yaw).
- Several control commands are also included in the actuator context to facilitate the execution of certain useful or frequent manoeuvres, such as axis initialization.

9.2.10 Command Line Interface

Most of the control commands described above have been duplicated in a command line-based interface in order to provide a more accurate way of entering data (see Fig. 9.1). On the other hand, the command line interface is the way of defining complex trajectories for the robot. Some simple commands allow the operator to program consecutive simple trajectories, specifying the type of motion (crab, circular or spin), total distance to walk, the crab angle, *etc.* before locomotion begins. This provides an accurate way to follow prefixed trajectories, releasing the operator from exhaustive surveillance of the machine.

9.2.11 Collaborative Context

The collaborative dialogue is performed through a special dialogue box that opens automatically when the robot queries the operator. When the operator wants to query the robot, a similar dialogue box can be opened by clicking on a specific button (Dialogue button in Fig. 9.1). The dialogue consists in a few questions on robot status, robot motion and environmental conditions and a few answers of the "yes/no" or numeric type. Collaboration between the robot controller and the human operator is always active, even at very low levels. For example, while in the actuator context, supposedly the operator can move any joint or any leg, but if this motion alters the static stability of the robot, the operator will be warned.

The collaborative dialogue is also useful during the locomotion in order to prevent failures. To do that, the controller informs about a possibly hazardous situation and asks for a user command, as depicted in the following examples. Data coming from the inclinometers can indicate an anomalous body attitude, caused by a foot slippage, for instance. In this case, the robot can ask the operator for a manual (tele-operated) recuperation of the attitude, as a preventive measure. If this action is not provided by the user, the robot will try to recover the attitude using its own attitude controller. In a similar way, based on data provided by both force sensors and foot potentiometers, the controller can also estimate if a foot is correctly placed on the terrain or if a foot tip over is probable due to an incorrect placement. In those cases the collaborative controller can ask for the operator's approval for the foothold or

try a new one. As mentioned above, the collaborative dialogue also helps to determine certain gait parameters to improve ground adaptation and optimize locomotion on irregular terrain. For example, when the robot is walking on a slope, it may be useful to change the attitude of the robot to adjust it to the average slope of the terrain. In doing so, the terrain surface contained in the legs' workspaces can be maximized, and consequently, the number of reachable (potential) footholds is increased, and deadlock likelihood is decreased. In this case, the robot makes an estimation of the slope based on the terrain's history, and proposes a new body attitude through the dialog window, which the operator can accept or reject. Finally, the collaborative dialogue can inform about the state of the machine or about the evolution of the locomotion algorithms when queried by the user. Table 9.1 summarizes queries and answers in the current HMI version.

The degree of availability of human attention (currently determined as a function of the expiration of a fixed delay between a user query and a response from user) should be adjustable explicitly by the operator (or determined somehow by the system) to avoid unnecessary delays. In this way, the performance obtained when operator attention is not available could be similar to the performance obtained by an autonomous controller.

9.3 Conclusions

Research on walking machine technology has focused mainly on producing autonomous walking machines. However, human intervention at different levels is going to be mandatory in many applications for industrial and service walking robots. This chapter has presented the HMI developed to steer walking robots featuring collaborative and multi-modal properties, although it has been particularized for the SILO4 robot.

The operator can command the controller at different levels, from individual joints to high-level missions (multi-modal). The graphic user interface is organized by its function into groups of commands and displays called contexts. The design of the GUI is oriented to facilitate high level control of the robot (path planning), and also to monitor the free gait generation. Lower level controls are also available for experienced operators.

Also, the robot is allowed to ask the human questions about how to achieve the mission, and to decide how to use, modify or reject such instructions (collaborative). Thus, the robot assists the operator and the two of them collaborate in achieving the mission. The collaborative control contributes to optimize the locomotion on irregular terrain and to improve ground adaptation as well as to prevent failures of a complex machine with poor sensorial inputs, requiring human dedication in a flexible manner. A closer attention of the operator yields a better adaptation and safety, and no attention yields an autonomous behavior.

Table 9.1. Queries and answers in the collaborative dialogue

Operator to robot	Response from robot
How are you? (Status)	• Stability plots using foot positions, inclinometer data and force sensor data • Terrain adaptation: – OK – Blocked: a foot cannot touch the ground or it cannot be lifted enough • Deadlock likelihood: numerical data (%)
How is the environment?	• Estimation of the terrain slope using inclinometer data and foot position data • Percentage of forbidden areas

Robot to operator	Response from operator	Comments
Can I change my body attitude to adapt it to the slope?	Yes, No	It can decrease deadlock likelihood
Can I rotate my body to align it with the trajectory?	Yes, No	It can decrease deadlock likelihood and increase speed
Should I lift my body to negotiate such high obstacles?	Yes, No	In doing so stability decreases
Should I lower my body to go on down this slope?	Yes, No	It can increase stability and speed but speed will decrease if the robot starts to go up a slope
Can I decrease the step height?	Yes, No	If the terrain is smooth, the robot's speed may increase
Is foot N adequately supported on the ground?	Yes, No	The potentiometer readings indicate a strange foot pose
Leg N has crashed: • Can I try it again? • Can I try a different foothold? • Can you give me the foothold co-ordinates?	Yes, No, Data	

This HMI also proved to be a useful tool for gait analysis and design. The HMI for the SILO4 walking robot has been assessed positively after many simulations and experiments on natural terrain (see Chap. 4 and (Estremera, 2003)).

A

The SILO4 Walking Robot

A.1 Generalities

The SILO4[1] walking machine is a medium-sized quadruped mechanism built for basic research and development as well as educational purposes (see Fig. 1.11). The SILO4 is a compact, modular, robust machine capable of negotiating irregular terrain, surmounting obstacles up to 0.25 m tall and carrying about 15 kg in payload at a maximum velocity of about 1.5 m/min, depending on the gait it is using.

The robot's total weight was considered an important feature in view of its intended uses in education and basic research. Walking machines have error recovery difficulties during debugging sessions, and it is necessary to handle them manually from time to time. Therefore a lightweight machine that an adult can carry is an important achievement. Also, obviously, lightweight devices are cheaper than big machines, and thrift is a basic requirement for educational robots. The SILO4 was configured as a four-legged robot because gait generation and stability are more difficult for quadrupeds than for hexapods, for instance, and, therefore, research in quadrupeds seems to be more attractive.

The SILO4 was conceived as an indoor walking robot, but it can work in an outdoor environment under non-extreme conditions. That means, for instance, the robot can work on highly irregular terrain but not under rainy conditions.

The SILO4 walking robot features the following main characteristics:

- Four legs.
- Small size and low weight for easy handling.
- Mechanical robustness.
- Slenderness, so as to avoid motor positions that give big leg volumes.
- Compactness, with all motors and electrical cables conveniently housed.

[1] Spanish acronym that means four-legged locomotion system.

- Agility in changing trajectories for good omnidirectionality.
- Control provided by a true real-time multitasking operating system supporting network communications.

A detailed description of the SILO4 robot mechanism can be found in Galvez *et al.* (2000) and SILO4 (2005).

A.2 Mechanical Structure

The SILO4 mechanical structure consists of four similar legs placed around a body. The foot is the leg end-effector or device that contacts with the ground and can adopt different structures. The following paragraphs present the robot configuration, the body structure, the leg configuration and different foot designs.

A.2.1 Robot Configuration

The SILO4's legs are placed around the body in a circular configuration. In a statically stable walking robot, this configuration enhances omnidirectionality and maneuverability because of the symmetrical distribution about the longitudinal and transversal body axes. However, the maximum achievable velocity is lower in comparison with that of other configurations. For instance, when the legs are placed along the sides parallel to the longitudinal axis of the body, as in a mammal configuration, average robot speed can be higher because better use is made of the leg stroke. Nevertheless, from the research point of view, omnidirectionality and maneuverability open up many possibilities in developing new algorithms, and in this sense the SILO4 configuration has been assessed positively.

A.2.2 Body Structure

The body of the SILO4 is similar to a parallelepiped measuring about $0.31 \times 0.30 \times 0.30$ m. It contains all the drivers and electronic cards as well as the force sensor amplifiers and a two-axis inclinometer that provides pitch and roll body angles. The body structure is made of aluminum, and the body's weight including electronics is about 14 kg.

The upper part of the body is a flat plate where auxiliary equipment and exteroceptive sensors such as TV cameras and laser range-finders can be installed. The four side walls can also be used for the same purposes.

A.2.3 Leg Configuration

The legs of the robot are based on an insect-type configuration, *i.e.* the second and third joints' axes lie parallel to each other and perpendicular to the axis of

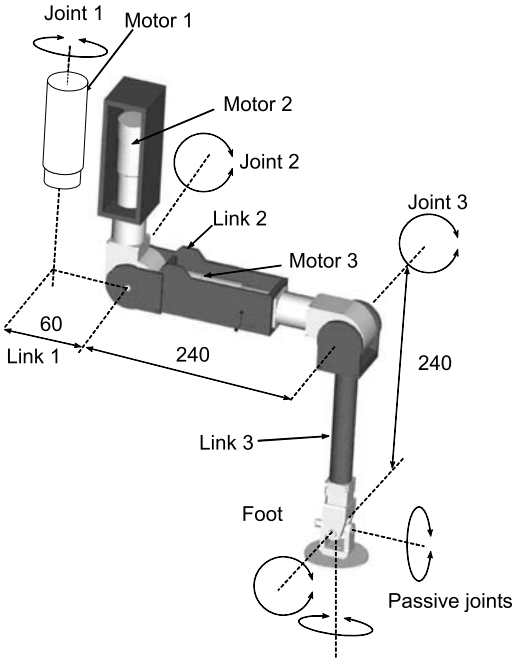
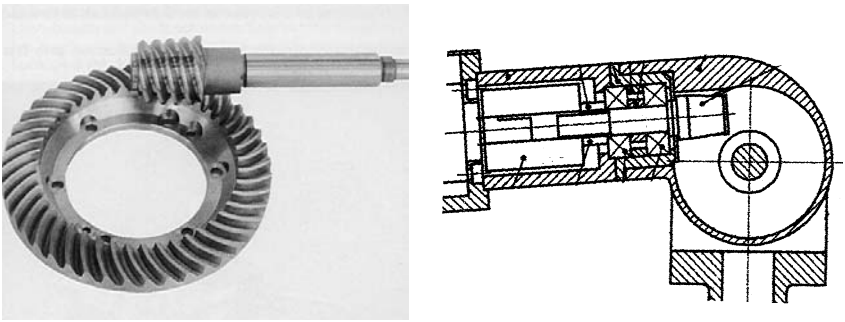


Fig. A.1. The SILO4 leg

the first joint (see Figs. 1.11 and A.1). The first link is about 0.06 m long, and the second and third joints are about 0.24 m long apiece. Each joint is actuated by a DC servomotor with a permanent-magnet stator and a coreless self-supporting coil-wound rotor. The motors are embedded in the leg structure, which results in only a very slight encumbrance. This makes for slender legs, thus helping to avoid crashing against obstacles. The motors are provided with planetary gears. The output shaft of a planetary gear directly drives the first joint. The second and third joints have an additional reducer based on a skew-axis spiroid mechanism.

Spiroid gears consist of a tapered pinion, which resembles a worm, and a face gear with teeth curved in a lengthwise direction. Spiroid gears have a position between spiral bevel, worm and face gear design. They are more robust than any of others and allow for higher reduction ratios with less weight and size. Figure A.2(a) shows a spiroid gear, and Fig. A.2(b) shows how the gear is mounted in joints 2 and 3.

The leg parts are mainly manufactured in aluminum, but some specific parts that must support heavy stress are made of aluminum 7075T6. The resulting leg weighs about 4 k including the foot. Table A.1 summarizes the



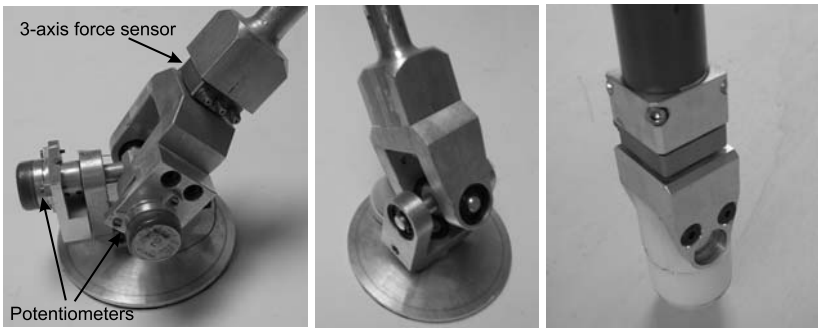
(a) (b)

Fig. A.2. (**a**) Spiroid gear; (**b**) A spiroid gear mounted in the second and third joints of the legs

main characteristics of the leg joints. Additional mechanical features can be found in Tables (6.2) – (6.7).

A.2.4 Foot Design

The normal SILO4 foot consists of a passive universal joint that connects the third link with a circular sole through yet another passive joint (see Fig. A.3(a). These three passive joints enable adequate contact between the circular planar sole and the ground's surface. The angle between the sole and the third link can be measured by two potentiometer (see Fig. A.3(a)). A three-axis piezoelectric force sensor placed in the third link above the passive joints measures the contact forces between the foot and the ground (see Fig. A.3(a)). Data acquired from these sensors can be used to run force-control



(a) (b) (c)

Fig. A.3. The SILO4 foot configurations

algorithms to improve locomotion on soft and irregular terrain. A simpler articulated foot without a force sensor (see Fig. A.3(b)) and a half-sphere foot with a passive joint, especially indicated for hard terrain (see Fig. A.3(c)), can also be used. There exist three SILO4 prototypes featuring (a), (b) and (c) foot designs at the Industrial Automation Institute-CSIC (Spain), the Ecole Nationale Suprieure d'Ingnieurs de Bourges-ENSI (France) and the University of Murcia (Spain). Table A.2 summarizes the main mechanical features of the SILO4 walking robot.

Table A.1. Leg-joint features of the SILO4 walking robot

Joint	1	2	3
Motor type	MINIMOTOR 3557K024C	MINIMOTOR 3557K024CR	MINIMOTOR 3557K024CS
Motor power (W)	14	72	26
Motor no-load speed (rpm)	4800	5300	5500
Motor stall torque (mNm)	105	510	177
Gearhead reduction	246:1	14:1	14:1
Spiroid reduction	—	20.5:1	20.5:1
Encoder	HEDS5540A14	HEDS5540A14	HEDS5540A14
Joint angle (degrees)	±80	+45 to -90	+10 to -135

Table A.2. Main features of the SILO4

Legs	
Number of legs	4
Leg type	Insect (three rotary joints)
Leg arrangements	Circular configuration
Leg dimension	0.24 m each link
Foot speed	0.20 m/s
Material	Aluminum 7075T6
Foot	
Foot base	Circular
Foot joints	1 or 3 passive joints (see the Foot Design section)
Material	Aluminum 7075T6 and steel
Body	
Dimensions	$0.31 \times 0.31 \times 0.30$ m
Material	Aluminum
Total Weight	≈ 30 kg
Payload	≈ 15 kg
Color	Red and aluminum color

A.2.5 Kinematics

Forward kinematics

The forward kinematic equations of the SILO4 robot, derived in Sect. 6.2, are

$$x = C_1(a_3C_{23} + a_2C_2 + a_1) \tag{A.1}$$

$$y = S_1(a_3C_{23} + a_2C_2 + a_1) \tag{A.2}$$

$$z = a_3S_{23} + a_2S_2. \tag{A.3}$$

where link parameters, a_i, and joint variable, θ_i, are defined in Fig. 6.2 and their values are given in Table (6.1). Let us remember that $C_i = \cos(\theta_i)$, $S_i = \sin(\theta_i)$, $C_{ij} = \cos(\theta_i + \theta_j)$ and $S_{ij} = \sin(\theta_i + \theta_j)$.

Inverse kinematics

The inverse kinematic equations of the SILO4 robot, also derived in Sect. 6.2, are

$$\theta_1 = \arctan 2\,(y, x) \tag{A.4}$$

$$\theta_2 = -\arctan 2\,(B, A) + \arctan 2\,(D, \pm\sqrt{A^2 + B^2 - D^2}) \tag{A.5}$$

$$\theta_3 = \arctan 2\,(z - a_2S_2, xC_1 + yS_1 - a_2C_2 - a_1) - \theta_2. \tag{A.6}$$

Jacobian matrix

The Jacobian matrix of the SILO4 leg used in Chap. 5 is

$$\mathbf{J} = \begin{pmatrix} -S_1(a_3C_{23} + a_2C_2 + a_1) & -C_1(a_3S_{23} + a_2S_2) & -a_3C_1S_{23} \\ C_1(a_3C_{23} + a_2C_2 + a_1) & -S_1(a_3S_{23} + a_2S_2) & -a_3S_1S_{23} \\ 0 & a_3C_{23} + a_2C_2 & a_3C_{23} \end{pmatrix}. \tag{A.7}$$

The derivation of the Jacobian matrix is beyond the scope of this book.

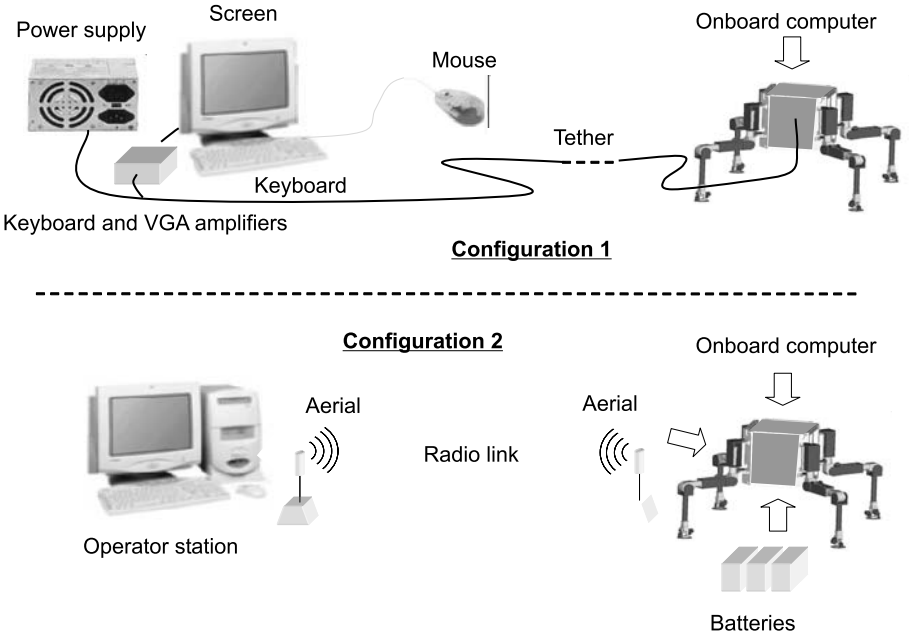


Fig. A.4. SILO4 system configurations

A.3 Control System Configuration

The SILO4 has been envisaged as an autonomous walker supervised by a remote operator who is in charge of defining the main features of the robot's motion such as speed and direction of motion. The supervisor station is remotely located, and communications are performed mainly through a tether that also carries the power supply. Radio communications and batteries could also easily be installed on board the robot. Thus, the overall SILO4 system admits two different configurations. In the first configuration there is a unique computer on board the robot. Computer and robot motors are powered externally through a tether that also carries an extension of the cables for the computer screen and keyboard (see Configuration 1 in Fig. A.4). In the second configuration, there are two computers, the Onboard Computer and the Operator Station. The robot controller runs on the Onboard Computer, and its power is supplied along with that of the robot motors by onboard batteries. Communications between the Onboard Computer and the Operator Station are run via serial radio link (see Configuration 2 in Fig. A.4).

A.3.1 Computing System

The control system, entirely installed on board the robot, is a distributed hierarchical system composed of a PC-based computer, a data-acquisition board

and four three-axis control boards based on the LM629 microcontrollers, interconnected through an ISA bus. The LM629 microcontrollers include digital PID filters provided with a trajectory generator used to execute closed-loop control for position and velocity in each joint. Every microcontroller commands a DC motor-joint driver based on the PWM technique. An analogue data-acquisition board is used to acquire sensorial data from the different proprioceptive sensors. Additional components could be added depending on the sensors used in the system. For instance, if piezoelectric force sensors were used, as shown in Fig. A.3(a), then some charge amplifiers would be included.

A general diagram of the SILO4 hardware architecture is shown in Fig. A.5. This is the hardware configuration used in the three SILO4 robots built so far. Nevertheless, researchers are encouraged to test other configurations so the much-needed comparisons can be drawn.

A.3.2 Sensors and Sensor System

The SILO4 walking robot does not possess exteroceptive sensors. Exteroceptive sensor integration is a task that potential users are expected to accomplish. As internal sensors, this machine uses an encoder on each joint as a position sensor, which is employed by the axis control boards to close the position loop. A couple of orthogonal inclinometers are also used to maintain the body at a given attitude. Depending on the foot type used, the machine can include different sensors. If the machine has articulated feet with force sensors, then the sensor system could possess a three-axis force sensor and two potentiometers per foot. With these sensors, the system can detect foot/ground interaction or implement force distribution techniques. If the foot is articulated with no force sensors, then the sensor system could incorporate an ON/OFF switch on each foot sole for ground detection. For the fixed foot configuration, the system does not include any sensors, and ground detection can be performed through a virtual sensor based on a neural network that uses the encoder data as network inputs (see Chap. 8). This last strategy can be indeed applied to any foot type.

No absolute sensors have been installed in the robot to fix the origin of the encoders. This task is performed by checking position errors while a given joint moves toward a mechanical limit. If position error grow suddenly, a mechanical limit is assumed to be reached. Thus no absolute sensors (switch, inductive or proximity sensor, *etc.*) are needed, and the cumbersome duty of running more electrical cables from the controller to each joint is also avoided.

A.3.3 Control Algorithms

The Onboard Computer is in charge of gait generation, trajectory generation, kinematics, signal processing and user interface, as well as coordination of the micro-controllers. These tasks are distributed in a software architecture that

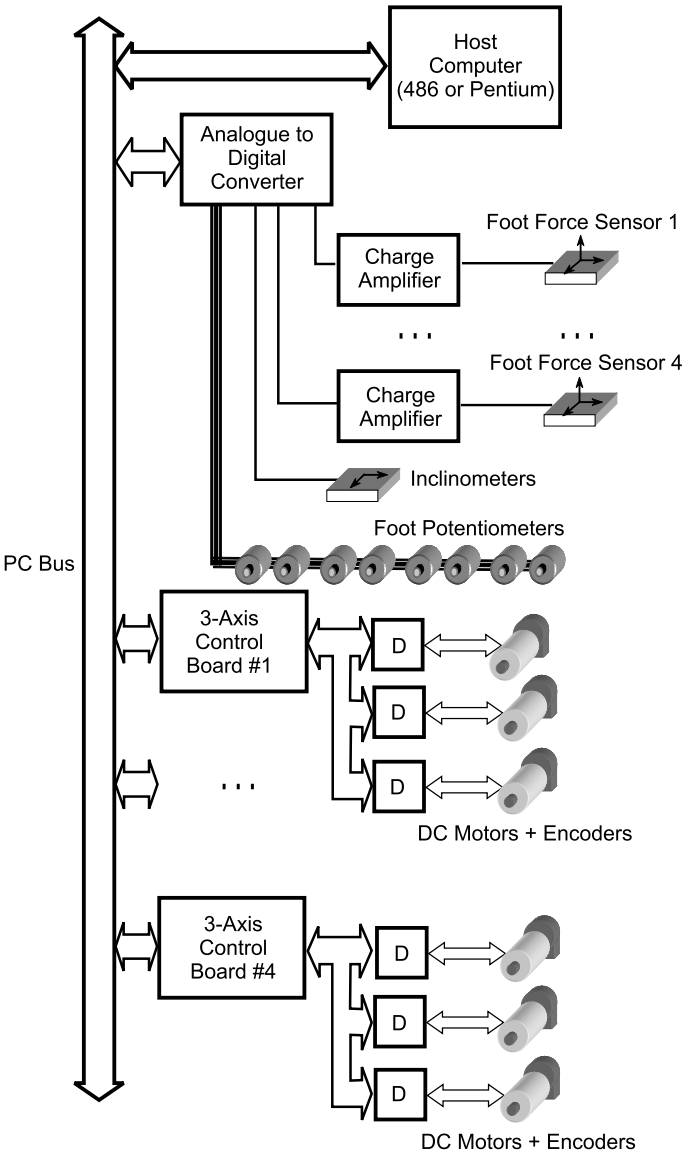


Fig. A.5. SILO4 hardware architecture

consists of layers developed on a bottom-up basis. These layers can be mainly divided into (see Fig. A.6):

- Hardware interface: this layer contains the software drivers.
- Axis control: this layer performs the control of individual robot joints, which is based on a PID controller.

- Leg control: this layer is in charge of coordinating all three joints in a leg to perform coordinated motions.
- Leg kinematics: this layer contains the direct and inverse kinematic functions of a leg.
- Trajectory control and robot kinematics: this module is in charge of coordinating the simultaneous motion of all four legs to perform straight-line or circular motions.
- Motion processes: this module executes the Terrain Adaptation, Attitude Control and Altitude Control algorithms.
- Stability module: this layer determines whether a given foot position configuration is stable or not.
- Gait generator: this layer generates the sequence of leg lifting and foot placement to move the robot in a stable manner. The stability module guarantees static stability. The SILO4 gait generator is based on four gaits: a discontinuous gait, a free crab gait, a spinning gait and a turning gait (see Chaps. 3 and 4).
- Sensor module: takes care of the data acquisition of the different robot sensors.
- Graphic and user interfaces: this layer contains the functions for plotting on the computer screen a simple graphic representation of the robot's pose and the dialogue boxes and bottoms of the HMI.

This software runs under QNX, a multitasking, real–time operating system that provides networking facilities. This last characteristic propitiates communication with other systems via serial line or Ethernet in order to render the construction of distributed systems and remote robot operation feasible. The lower levels of the control system are implemented in C++ language libraries that can be compiled under QNX or MS-Windows operating systems. Photon Application Builder for the QNX operating system has been used to develop the HMI.

Fig. A.6 illustrates the different software modules and their interconnections. The C++ code for the libraries involved in the software architecture as well as the HMI can be obtained from SILO4 (2005).

A.4 Simulation Tools

A simulation tool for the SILO4 has been built for preliminary studies of the robot's features. This simulation tool is based on a commercial simulation construction set (Yobotics, 2002) and takes into consideration the kinematics and dynamics of the robot as well as the terrain model and the foot/ground contact model. The control law for each joint can be defined and output data can be obtained through variable panels, graphics and a 3D graphics window. All control variables as well as all system variables are available at the variable panels. The variables can be selected for plotting in the graphics area as

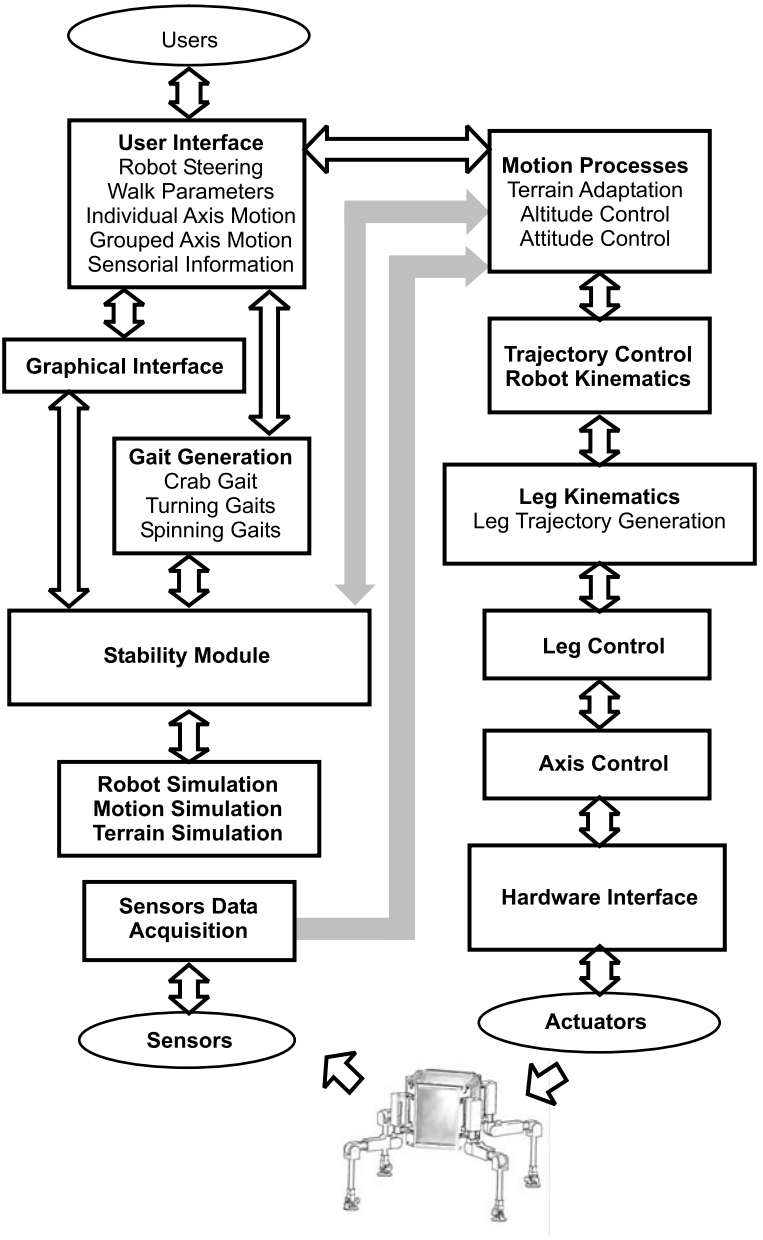


Fig. A.6. SILO4 software architecture

functions of time. The simulation tool can export all these data for plotting using standard software packages. Finally, the 3D graphics window can show a 3D animation of the robot on the defined ground model. This is a very

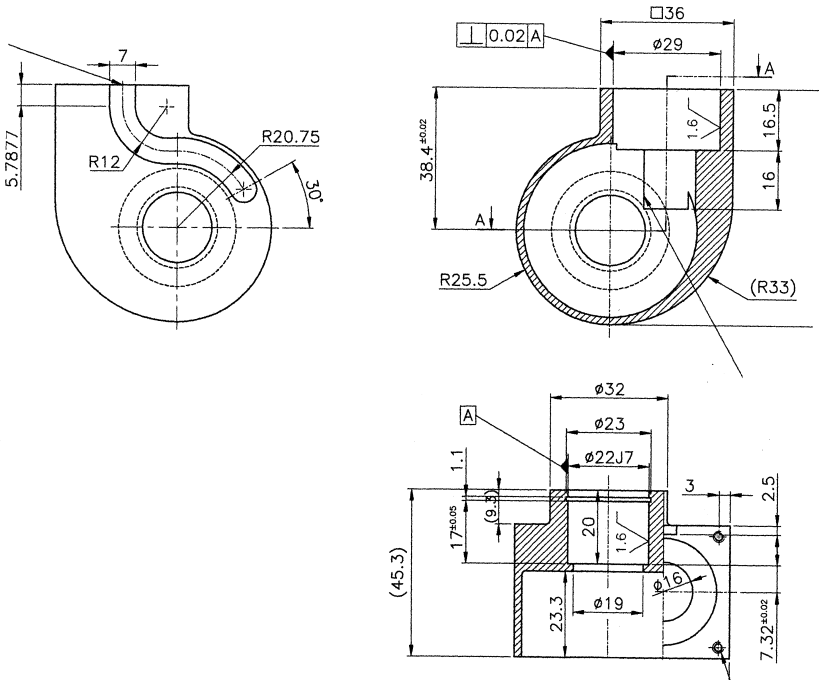


Fig. A.7. Example of a manufacturing drawing of the SILO4 walking robot

powerful tool for tasks such as designing and debugging new gaits and control algorithms before testing in the actual machine or checking the robot's behavior when its load distribution is changed. The simulation construction set is a collection of functions written in Java language and can be obtained from Yobotics (2002). SILO4 robot classes for building the SILO4 simulator can be obtained from SILO4 (2005). This simulation tool is detailed in Appendix B.

A.5 Manufacturing Drawings

The whole mechanical design of the SILO4 walking robot is available from the Internet (SILO4, 2005). The drawings include materials, dimensions and required mechanical accuracy, and they are ready for manufacturing. A list of commercial parts is also included. Figure A.7(a) shows just an example of a manufacturing drawing, and Fig. A.8 shows how to connect all the parts. The numbers indicating different parts are the numbers of the drawings in the collection (SILO4, 2005). For instance, Fig. A.7 plots drawing 14, which is the part indicated as 14 in Fig. A.8.

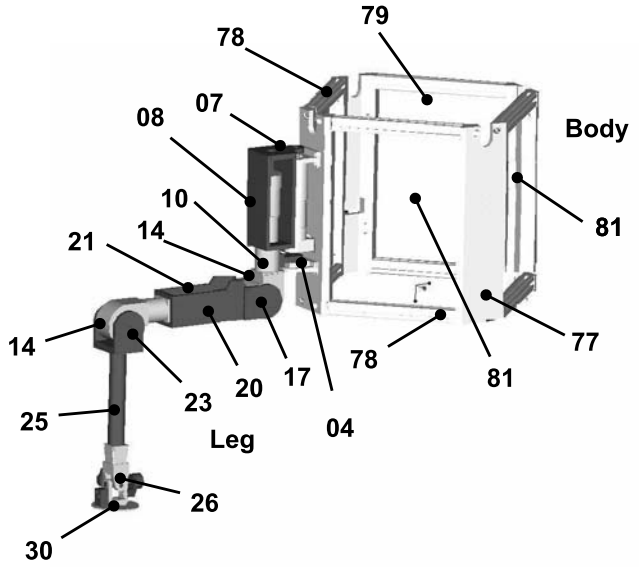


Fig. A.8. Enumeration of the parts in the SILO4 walking robot

A.6 Conclusions

Walking machines exhibit many advantages over traditional vehicles; however, they still have a long way to go before they are used in industry and services. One of the problems is that researchers do not have common testbeds available for true comparison of new developments and algorithms. Some commercial legged platforms can be purchased on the market for those purposes, but they lack important features such as maintainability, terrain adaptability, part replacement, and so on. Comparison is imperative for any real improvement of walking machine techniques, but efficient, effective comparison can only be accomplished by using similar machines.

To overcome these problems the IAI-CSIC, based on its experience in the construction of three previous walking machines, has developed the SILO4 walking robot and offers its complete design on the Internet to other research teams willing to manufacture a replica by themselves (SILO4, 2005). The main aim of the design is to configure and develop a small, easy-to-handle, reliable walking robot with both great terrain adaptability and omnidirectionality.

These book authors encourage other researchers to share the SILO4 design as a comparative walking robot and hopes to contribute to the development of duly assessed new techniques.

B

Simulation Software for Walking Robots

B.1 Introduction

The Yobotics! Simulation Construction Set (SCS) (Yobotics, 2002) is a software package developed by Yobotics Inc. for easily and quickly creating graphical and numerical simulations of mechanical devices, biomechanical systems, and robots. The simulation created with the SCS involves kinematic and dynamic models of the robotic system, a geometric model of the terrain, and a ground-contact model. The resulting simulation is a tool such as the one shown in Fig. B.1 to follow the simulation of the robotic system simultaneously in three ways:

- Numerically: every system variable is accesible through a variable panel (see Fig. B.1). Variables such as control gains, stability margins or simulation parameters can be monitored and modified along the simulation process using the numeric-entry boxes.
- Graphically: the time evolution of system variables is also monitored graphically in the graphic panel (see Fig B.1).
- Visually: the evolution of variables in the simulated robotic system is also reflected in the 3D-animation that runs in the 3D-graphics window (see Fig B.1). Thus, the simulated robot motion is there at a glance, providing meaningful feedback of the tested control algorithm.

The simulation GUI includes a menu and toolbar (see Fig. B.1) to manage the simulation, such as recording data, importing and exporting from/to MATLAB and other file formats. It also allows image capturing and movie generating.

The next sections detail how to use the SCS to create the robot simulations that have been used to obtain simulation results along this book. The SILO4 quadruped robot has been used as a model for the created simulation.

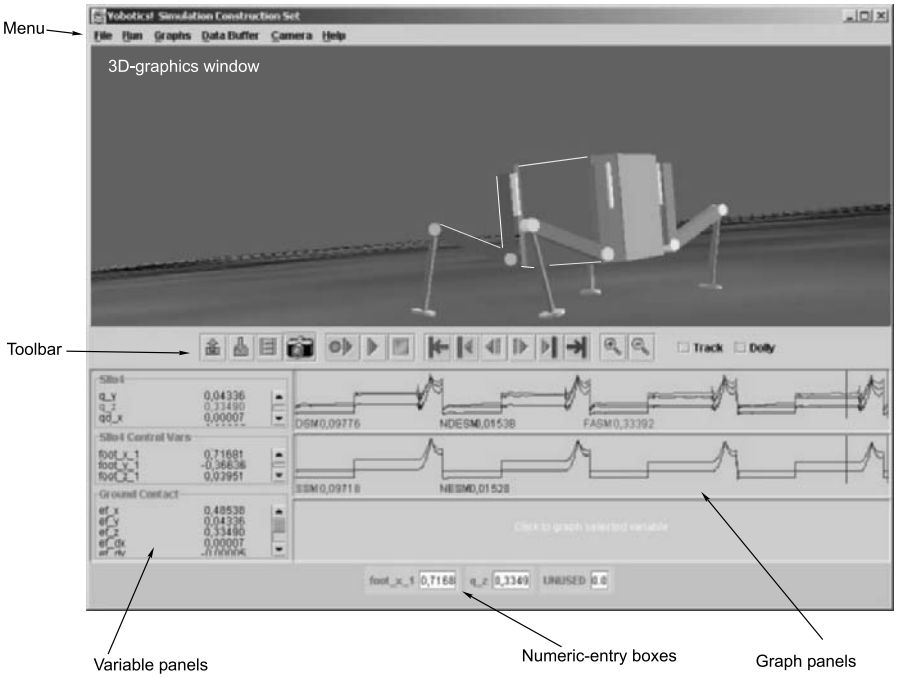


Fig. B.1. Screen snapshot of SILO4 simulation built using the *Yobotics! Simulation Construction Set*

B.2 Simulation Parameters

The integrator used for the simulation is based on the Runge-Kutta fourth-order method with an integration period of 0.4 ms. However, the data is collected for graphic comparison at a sampling time of 0.02 s.

B.3 Programming a Simulation

The simulation is programmed in Java language and it is organized into the following classes (see Fig. B.2):

- **Simulation class:** contains functions for setting the parameters of the simulation, such as the integration time step, camera position, variables initially plotted in the graphs, *etc.* It also relates the simulation with the SILO4 robot.
- **Robot class:** defines the geometry and dynamics of the robot. The robot consists of a tree of joints and links with shape, colour, mass and inertia. Some ground contact points are defined in the robot and calls to the ground contact model, ground profile and controller are performed.

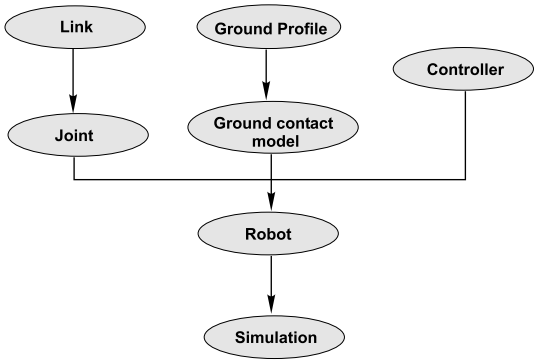


Fig. B.2. Yobotics! Simulation Construction Set Class diagram

- **Controller class:** here, the control algorithm for the motion of the robot is programmed. The SILO4 walks in a two-phase discontinuous gait, which is controlled by a state machine, where swing and support phases of the robot legs are selected. Leg trajectory generation is also programmed, where the foot follows a straight line, and joint trajectories are PD controlled.
- **Ground Profile class:** it defines the shape of the ground, *i.e.* planar, inclined, uneven, *etc.*
- **Ground contact model class:** here the dynamics of the robot/terrain interaction are modelled. The model used for the SILO4 simulation is a spring-dumper in the x_0, y_0 and z_0 directions.

B.4 Creating the SILO4 Robot

The Robot class defines the geometry and aspect of the robot, and the kinematic and dynamic model as well. Robot kinematics are defined as a tree of joints. Therefore, the body is the root joint, and four branches of joints define the four legs. Each leg consists of three pin joints, and each joint has a link associated to it, which defines the shape, size and colour of the structure. Therefore, the Robot class utilizes elements from two other classes:

- **Link class:** allows links of several shapes, colours and textures to be generated. The mass, center of mass, and inertia of each link are also defined so that the robot's dynamics can be computed from the defined parameters using the Featherstone algorithm (Featherstone, 1987).
- **Joint class:** it is used to insert different types of joints into the kinematic structure (*i.e.* rotary, cylindrical, spherical, *etc.*). Each link is jointed with a given joint so that the Denavit-Hartenberg formulation fits. Joint variables will be controlled through the Controller class and can be monitored in the variable and graphic panels. Ground-contact points are defined once robot kinematics is complete.

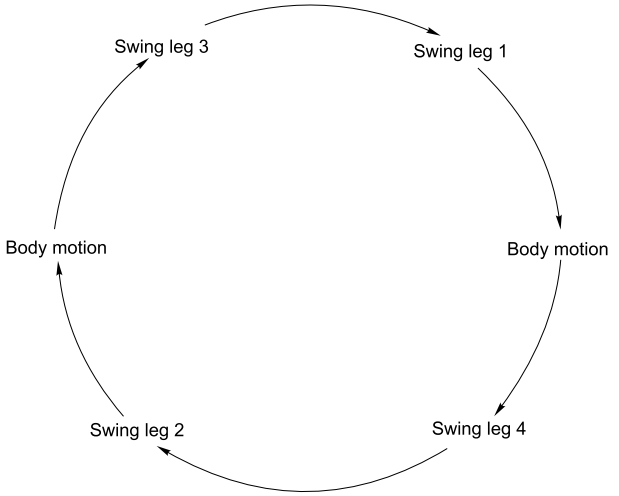


Fig. B.3. Finite state machine for a two-phase discontinuous gait

The Robot class also defines the ground profile and the ground-contact model by means of calling the Ground Profile and Ground-contact Model classes.

B.5 Gait Control

The SILO4 robot walks using a two-phase discontinuous gait (see Chap. 3), which is programmed by means of a finite state machine, shown in Figure B.3. Two leg swing states precede each body motion, where the body is propelled forward. The swing of a leg consists of three straight-line trajectories at the foot (lift, forward motion and landing), which are generated on-line. The body motion consists of the straight backward motion of the four legs simultaneously. Every trajectory generation process determines desired joint trajectories, which are PD-controlled at the joint level, that is

$$\tau_i = K_p(\theta_i^{des} - \theta_i) + K_v(\dot{\theta}_i^{des} - \dot{\theta}_i) \tag{B.1}$$

where subscript i denotes the joint number, θ and $\dot{\theta}$ are joint position and velocity, respectively, θ^{des} and $\dot{\theta}^{des}$ are the reference joint position and velocity, respectively, and K_p and K_v are the elastic and dumping constants. The output of the PD controller in (B.1) is the torque τ_i required at the joint i.

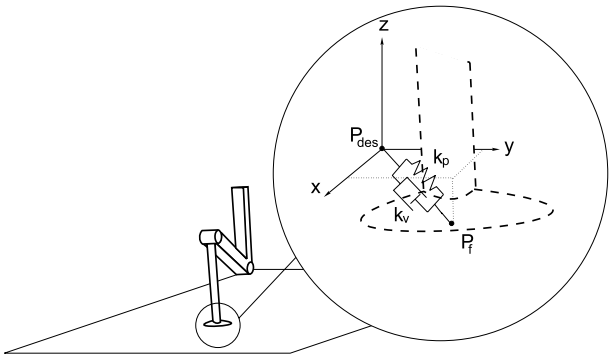


Fig. B.4. Ground contact model

B.6 Ground Profile

The shape of the terrain is programmed as an elevation function relative to a fixed reference frame $x_0\, y_0\, x_0$. Therefore a sloped terrain in the x_0 direction is described by the following function:

$$z_0 = \alpha\, x_0 \tag{B.2}$$

where z_0 is the height of the terrain surface, and α represents the slope.

A random uneven terrain can be modeled by

$$z_0 = A_1 \sin(\omega_1 x_0 + \varphi_1)\, A_2 \sin(\omega_2 y_0 + \varphi_2) \tag{B.3}$$

where A_1 and A_2 are the amplitude of roughness, ω_1 and ω_2 are their frequencies, and φ_1 and φ_2 denote their phases.

B.7 Ground Contact Model

The dynamic model of the robot/terrain interaction consists of three orthogonal spring-dumper systems along the simulated x, y and z spatial directions, respectively, attached to the feet (see Figure B.4). Each time the z coordinate of a foot enters the ground profile, a ground-reaction force is applied against it, whose Cartesian coordinates are given by

$$F_x = k_p(x_{des} - x_f) - k_v \dot{x}_f \tag{B.4}$$

$$F_y = k_p(y_{des} - y_f) - k_v \dot{y}_f \tag{B.5}$$

$$F_z = k_p(z_{des} - z_f) - k_v \dot{z}_f \tag{B.6}$$

where $(x_{des}, y_{des}, z_{des})$ are the Cartesian coordinates of the point P_{des} at the initial foot/terrain contact, and (x_f, y_f, z_f) are the coordinates of point P_f, which represents the foot position at any later instant.

Joint elasticity can be also modeled using the ground contact model. Assuming a Cartesian spring-dumper model of joint compliance (Shih *et al.*, 1987), the composition of the elastic and dumping effects of the three joints of a leg at the instant that the foot contacts the ground can be considered an equivalent spring-dumper system at the foot. Therefore, the addition of the equivalent elastic and dumping constants to the ground contact model reflects the additional effect of joint elasticity during walking.

The simulation of the SILO4 robot described in this chapter can be obtained from SILO4 (2005).

References

Ablameyko, S., Goras, L., Gori, M., and Piuri, V. (2003). *Neural networks for instrumentation mesurement and related industrial applications*, volume 185. IOS Press.

Akizono, M., Iwasaki, M., Nemoto, T., and Asakura, O. (1989). Development on walking robot for underwater inspection. In *International Conference on Advanced Robotics*, pages 652–663. Springer-Verlag.

Alexander, R. N. (1977). *Terrestrial Locomotion, Mechanics and Energetics of Animal Locomotion*. Alexander, R.N. and Goldspink, G., editors. Chapman and Hall, London.

Angle, C. M. and Brooks, R. A. (1990). Small planetary rovers. In *IEEE/RSJ Int. Workshop Intelligent Robots and Systems*, pages 383–388. Ikabara, Japan.

Armstrong, B. (1989). On finding excitation trajectories for identification experiments involving systems with nonlinear dynamics. *The International Journal of Robotic Research*, **8**(6), 28–48.

Artobolevsky, I. I. (1964). *Mechanism for the generation of plane curve*. Pergamon Press. Oxford.

Bai, S., Low, J., and Zielinska, T. (1999). Quadruped free gait generation based on the primary/secondary gait. *Robotica*, **17**, 405–412.

Bares, J. and Wettergreen, D. (1999). Dante II: Technical description, results, and lessons learned. *The International Journal of Robotic Research*, **18**(7), 621–649.

Bares, J. and Whittaker, W. L. (1989). Configuration of an autonomous robot for mars exploration. In *World Conference on Robotics Research: The Next Five Years and Beyond*, volume 1, pages 37–52. Gaithersburg.

Bares, J. E. and Whittaker, W. L. (1993). Configuration of autonomous walkers for extreme terrain. *The International Journal of Robotic Research*, **12**(6), 535–559.

Baudoin, Y., Acheroy, M., Piette, M., and Salmon, J. P. (1999). Humanitarian demining and robotics. *Mine Action Information Center Journal*, **3**.

Bekker, M. G. (1960). *Off-the-road locomotion*. Ann Arbor: University of Michigan Press.

Bennani, M. and Giri, F. (1996). Dynamic modelling of a four-legged robot. *Journal of Intelligent and Robotic Systems*, **17**, 419–428.

Berns, K. (2005). *The Walking Machine Catalogue*. Available: `http://agrosy.informatik.uni-kl.de/wmc/start.php/`.

Bessonov, A. and Umnov, N. (1973). The analysis of gaits in six-legged vehicles according to their static stability. In *Proceedings Of CISM-IFToMM Symposium on Theory and Practice of Robots and Manipulators*, pages 117–123. Udine, Italy.

Big-Muskie (2005). *The Big Muskie Web Page*. Available: `http://www.little-mountain.com/bigmuskie/`.

Bihari, T. E., Wallister, T. M., and Patterson, M. R. (1989). Controlling the adaptive suspension vehicle. *IEEE Computer*, **22**(6), 59–65.

Blackmon, T. and Stark, L. (1996). Model-based supervisory control in telerobotics. *Presence*, **5**(2), 205–223.

Bobrow, J., Dubowsky, S., and Gibson, J. (1985). Time-optimal control of robotic manipulators along specified paths. *The International Journal of Robotic Research*, **4**(3), 3–17.

Brooks, R. (1989). A robot that walks: emergent behaviors from a carefully evolved network. *Neural computation*, **1**, 253–262.

Buehler, M., Battaglia, R., Cocosco, A., Hawker, G., Sarkis, J., and Yamazaki, K. (1998). Scout: A simple quadruped that walks, climbs and runs. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1701–1712. Leuven, Belgium.

Butch (2005). *Dino Butch*. Available: `http://hebb.mit.edu/people/russt/robots/`.

Byrd, J. S. and DeVries, K. R. (1990). A six-legged telerobot for nuclear applications development. *The International Journal of Robotic Research*, **9**(2), 43–52.

Chen, C., Kumar, V., and Luo, Y. (1999a). Motion planning of walking robots in environments with uncertainty. *Journal of Robotic Systems*, **16**(10), 527–545.

Chen, W., Low, K., and Yeo, S. (1999b). Adaptive gait planning for multilegged robots with an adjustment of center-of-gravity. *Robotica*, **17**, 391–403.

Craig, J. J. (1989). *Introduction to Robotics*. Addison-Wesley, 2nd edition.

Cruse, H., Kindermann, T., Schumm, M., Dean, J., and Schmitz, J. (1998). Walnet-a biologically inspired network to control six-legged walking. *Neural Networks*, **11**, 1435–1447.

Dean, J., Kindermann, T., Schmitz, J., Schumm, M., and Cruse, H. (1999). Control of walking in the stick insect: from behavior and physiology to modeling. *Autonomous Robots*, **7**, 271–288.

Dettman, J. (1988). *Mathematical methods in physics and engineering*. Dover Publications Inc., New York.

Eldershaw, C. and Yim, M. (2001). Motion planning of legged vehicles in an unstructured environment. In *Proceedings of the International Conference on Robotics and Automation*, pages 3383–3389. Seoul, Korea.

Estremera, J. (2003). *Free gaits and virtual sensors for walking robots*. Ph.D. thesis, Universidad Complutense de Madrid.

Estremera, J. and Gonzalez de Santos, P. (2002). Free gaits for quadruped robots over irregular terrain. *The International Journal of Robotic Research*, **21**(2), 115–130.

Estremera, J., Garcia, E., and Gonzalez de Santos, P. (2002). A multi-modal and collaborative human-machine interface for a walking robot. *Journal of Intelligent and Robotic Systems*, (35), 397–425.

Estremera, J., Gonzalez de Santos, P., and Lopez-Orozco, J. A. (2005). Neural virtual sensor for terrain adaptation of walking machines. *Journal of Robotic Systems*, **22**(6), 299–311.

Featherstone, R. (1987). *Robot Dynamics Algorithms*. Kluwer Academic Publishers, Boston-Dordrecht-Lancaster.

Fong, T., Pangels, H., and Wettergreen, D. (1995). Operator interfaces and network-based participation for Dante II. In *SAE 25th International Conference on Environmental Systems*, pages 131–137. San Diego, CA,.

Fong, T., Thorpe, C., and Baur, C. (1999). Collaborative control: A robot-centric model for vehicle teleoperation. In *AAAI Spring Symposium: Agents with Adjustable Autonomy*, pages 210–219. Stanford, CA.

Frik, M., Guddat, M., Karatas, M., and Losch, D. C. (1999). A novel approach to autonomous control of walking machines. In *Second International Conference on Climbing and Walking Robots (CLAWAR'99)*, pages 333–342. Portsmouth, UK.

Fu, K. S., Gonzalez, R. C., and Lee, C. S. G. (1987). *Robotics: Control, Sensing, Vision, and Intelligence*. McGraw Hill.

Fujita, M. (2001). AIBO: Toward the era of digital creatures. *The International Journal of Robotics Research*, **20**(10), 781–794.

Fukuoka, Y., Kimura, H., and Cohen, A. (2003). Adaptive dynamic walking of a quadrupeed robot on irregular terrain based on biological concepts. *The International Journal of Robotic Research*, **22**(3-4), 187–202.

Galvez, J. A., Estremera, J., and Gonzalez de Santos, P. (2000). SILO4: A versatile quadruped robot for research in force distribution. In *Proceedings of the International Conference on Climbing and Walking Robots*, pages 371–384. Madrid, Spain.

Garcia, E. and Gonzalez de Santos, P. (2001). Using soft computing techniques for improving foot trajectories in walking machines. *Journal of Robotic Systems*, **18**(7), 343–356.

Garcia, E. and Gonzalez de Santos, P. (2005). An improved energy stability margin for walking machines subject to dynamic effects. *Robotica*, **23**(1), 13–20.

Garcia, E., Gonzalez de Santos, P., and Canudas de Wit, C. (2002). Velocity dependence in the cyclic friction arising with gears. *The International Journal of Robotic Research*, **21**(9), 761–771.

Garcia, E., Galvez, J., and Gonzalez de Santos, P. (2003). On finding the relevant dynamics for model based controlling walking robots. *Journal of Intelligent and Robotic Systems*, **37**(4), 375–398.

Ghasempoor, A. and Sepehri, N. (1998). A measure of stability for mobile manipulators with application to heavy-duty hydraulic machines. *ASME Journal of Dynamic Systems, Measurement and Control*, **120**, 360–370.

Gonzalez de Santos, P. and Jimenez, M. (1995). Generation of discontinuous gaits for quadruped walking machines. *Journal of Robotic Systems*, **12**(9), 599–611.

Gonzalez de Santos, P., Jimenez, M., and Armada, M. (1998). Dynamic effects in statically stable walking machines. *Journal of Intelligent and Robotic Systems*, **23**(1), 71–85.

Gonzalez de Santos, P., Armada, M., and Jimenez, M. (2000). Ship building with ROWER. *IEEE Robotics and Automation Magazine*, **7**(4), 35–43.

Gonzalez de Santos, P., Galvez, J., Estremera, J., and Garcia, E. (2003). SILO4 - a true walking robot for the comparative study of walking machine techniques. *IEEE Robotics and Automation Magazine*, **10**(4), 23–32.

Gonzalez de Santos, P., Estremera, J., Garcia, E., and Armada, M. (2005). Including joint torques and power consumption in the stability margin of walking robots. *Autonomous Robots*, **18**, 43–57.

Goswami, A. (1999). Postural stability of biped robots and the foot-rotation indicator (FRI) point. *The International Journal of Robotic Research*, **18**(6), 523–533.

Grieco, J., Prieto, M., Armada, M., and Gonzalez de Santos, P. (1998). A six-legged climbing robot for high payloads. In *IEEE International Conference on Control Applications*, pages 446–450. Trieste, Italy.

Gurfinkel, V. S., Gurfinkel, E. V., Schneider, A. Y., Devjanin, E. A., Lensky, A. V., and Shitilman, L. G. (1981). Walking robot with supervisory control. *Mechanism and Machine Theory*, **16**, 31–36.

Habumuremyi, J. C. (1998). Rational designing of an electropneumatic robot for mine detection. In *First International Conference on Climbing and Walking Robots (CLAWAR'98)*, pages 267–273. Brussels, Belgium.

Hagan, M., Demunth, H., and Beale, M. (1996). *Neural network design*. PWS Publishing Company.

Hanzevack, E., Long, T., Atkinson, C., and Traver, M. (1997). Virtual sensors for spark ignited engines using neural networks. In *Proceedings of the American Controls Conference*. Albuquerque, New Mexico.

Hildebrand, M. (1965). Symmetrical gaits of horses. *Science*, (150), 701–708.

Hines, J., Uhrig, R., and Wrest, J. (1998). Use of autoassociative neural networks for signal validation. *Journal of Intelligent and Robotic Systems*, **21**, 143–154.

Hirose, S. (1984). A study of design and control of a quadruped walking robot. *The International Journal of Robotic Research*, **10**(2), 113–133.

Hirose, S. and Kato, K. (1998). Quadruped walking robot to perform mine detection and removal task. In *Proceedings of the 1st International Conference on Climbing and Walking Robots*, pages 261–266. Brussels, Belgium.

Hirose, S., Kikuchi, H., and Umetani, Y. (1986). The standard circular gait of a quadruped walking vehicle. *Advanced Robotics*, **1**(2), 143–164.

Hirose, S., Inoue, S., and Yoneda, K. (1990). The whisker sensor and the transmission of multiple sensor signals. *Advanced Robotics*, **4**(2), 105–117.

Hirose, S., Yoneda, K., and Tsukagoshi, H. (1997). TITAN VII: Quadruped walking and manipulating robot on a steep slope. In *International Conference on Robotics and Automation (ICRA'97)*, pages 494–500. Albuquerque, NM.

Hirose, S., Tsukagoshi, H., and Yoneda, K. (1998). Normalized energy stability margin: Generalized stability criterion for walking vehicles. In *Proceedings of the International Conference on Climbing and Walking Robots*, pages 71–76. Brussels, Belgium.

Ilon, B. E. (1975). *Wheels for a course stable self-propelling vehicle movable in any desired direction on the ground or some other base*. U.S. Patent No. 3 876 255.

Ishino, Y., Naruse, T., Sawano, T., and Honma, N. (1983). Walking robot for underwater construction. In *International Conference on Advanced Robotics*, pages 107–114.

Jimenez, M. and Gonzalez de Santos, P. (1997). Terrain adaptive gait for walking machines. *The International Journal of Robotic Research*, **16**(3), 320–339.

Jimenez, M., Gonzalez de Santos, P., and Armada, M. (1993). A four legged walking testbed. In *IFAC Workshop on Intelligent Autonomous Vehicles*, pages 8–13. Hampshire, United Kingdom.

Kaneko, M., Abe, M., and Tanie, K. (1985). A hexapod walking machine with decoupled freedoms. *IEEE Journal of Robotics and Automation*, **RA-1**(4), 183–190.

Kang, D., Lee, Y., Lee, S., Hong, Y., and Bien, Z. (1997). A study on an adaptive gait for a quadruped walking robot under external forces. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2777–2782. Albuquerque, New Mexico.

Kato, K. and Hirose, S. (2001). Development of the quadruped walking robot, TITAN-IX - mechanical design concept and application for the humanitarian de-mining robot. *Advanced Robotics*, **15**(2), 191–204.

Kepplin, V. and Berns, K. (1999). A concept for walking behaviour in rough terrain. In *Proceedings of the International Conference on Climbing and Walking Robots*, pages 509–515. Portsmouth, UK.

Kimura, H., Shimoyama, I., and Miura, H. (1990). Dynamics in the dynamic walk of a quadruped robot. *Advanced Robotics*, **4**(3), 283–301.

Klein, C. A. and Chung, T. S. (1987). Force interaction and allocation for the legs of a walking vehicle. *IEEE Journal of Robotics and Automation*, **RA-3**(6), 546–555.

Kugushev, E. I. and Jaroshevskij, V. S. (1975). Problems of selecting a gait for a locomotion robot. In *Proceedings of the 4th International Joint Conference on Artificial Intelligence*, pages 789–793. Tilisi, Georgia, USSR.

Kumar, V. and Waldron, K. J. (1988). Gait analysis for walking machines for omnidirectional locomotion on uneven terrain. In *Proceedings of the 7th CISM-IFTOMM Symposium on Theory and Practice of Robots and Manipulators*. Udine, Italy.

Kumar, V. and Waldron, K. J. (1989). *A review of research on walking vehicles*, pages 243–266. In O. Khatib, J.J. Craig and T. Lozano-Perez, editors. *The robotics review*. The MIT Press, Cambridge, Massachusetts.

Lamit, L. G. (2001). *Pro/ENGINEER 2000i2*. BROOKS/COLE, Thomson Learning.

Leal, R., Butler, P., Lane, P., and Payne, P. (1997). Data fusion and artificial neural networks for biomass estimation. In *IEEE Proceedings: Science, Measurement and Technology*, volume 144, pages 69–72. Gatlimburg, Tennessee.

Lee, J. and Song, S. (1990). Path planning and gait of walking machine in an obstacle-strewn environment. *Journal of Robotic Systems*, **8**(6), 801–827.

Lewis, M. and Bekey (2002). Gait adaptation in a quadruped robot. *Autonomous Robots*, **12**(3), 301–312.

Lin, B. and Song, S. (1993). Dynamic modeling, stability and energy efficiency of a quadrupedal walking machine. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 367–373. Atlanta, Georgia.

Logsdon, T. (1984). *The Robot Revolution*. Simon and Schuster.

Luo, R., Su, K., and Phang, S. (2001). The development of intelligent control system for animal robot using multisensor fusion. In *Sensor Fusion Challenges and Applications in Emerging Technologies*. Baden-Baden, Germany.

Maes, P. and Brooks, R. (1990). Learning to coordinate behaviours. In *Proceedings of the 8th National Conference on Artificial Intelligence-AAAI'90*, pages 796–802. Boston, MA.

Mahalingham, S., Whittaker, W., and Gaithersburg, M. (1989). Terrain adaptive gaits for walkers with completely overlapping work spaces. In *Robots 13*, pages 1–14.

Mamdani, E. (1981). *Fuzzy Reasoning and Its Applications*. Academic Press.

Manko, D. (1992). *A general model of legged locomotion on natural terrain*. Kluwer Academic Publishers, Boston-London-Dordrecht.

Marques, L., Rachkov, M., and Almeida, A. T. (2002). Control system of a demining robot. In *Tenth Mediterranean Conference on Control and Automation*. Lisbon, Portugal.

Masson, M., Canu, S., and Grandvalet, Y. (1999). Software sensor design based on empirical data. *Ecological Modelling*, **120**, 131–139.

Matia, F. and Jimenez, A. (1996). On optimal implementation of fuzzy controllers. *International Journal of Intelligent Control and Systems*, **1**(3), 407–415.

Matia, F., Jimenez, A., Galan, R., and Sanz, R. (1992). Fuzzy controllers: Lifting the linear-nonlinear frontier. *Fuzzy Sets and Systems*, **52**(2), 113–128.

MATLAB (1992). *Neural Network Toolbox User's Manual*. The Math Works Inc., Natick, Mass., USA.

Mayeda, H., Osuka, K., and Kangawa, A. (1984). A new identification method for serial manipulator arms. In *IFAC 9th World Congress*, volume 6, pages 74–79. Budapest.

McGhee, R. B. (1968). Some finite state aspect of legged locomotion. *Mathematical Bioscience*, **2**, 67–84.

McGhee, R. B. and Frank, A. A. (1968). On the stability properties of quadruped creeping gaits. *Mathematical Bioscience*, **3**, 331–351.

McGhee, R. B. and Iswandhi, G. I. (1979). Adaptive locomotion for a multilegged robot over rough terrain. *IEEE Trans. on Systems, Man, and Cybernetics*, **SMC-9**(4), 176–182.

McKerrow, P. J. (1991). *Introduction to robotics*. Alexander, R.N. and Goldspink, G., editors. Addison-Wesley Publishing Co.

McMahon, T. A. (1984). *Muscles, Reflexes, and Locomotion*. Princeton, New Jersey.

Messuri, D. (1985). *Optimization of the locomotion of a legged vehicle with respect to maneuverability*. Ph.D. thesis, The Ohio State University.

Messuri, D. and Klein, C. (1985). Automatic body regulation for maintaining stability of a legged vehicle during rough-terrain locomotion. *IEEE Journal of Robotics and Automation*, **RA-1**(3), 132–141.

Mocci, U., Petternella, N., and Salinari, S. (1972). Experiments with six-legged walking machines with fixed gait. Technical Report 2.12, Institute of Automation, Rome University. Rome, Italy.

Monagan, M., Geddes, K., Labahn, G., and Vorkoetter, S. (1998). *MapleV Programming Guide*. Maple Waterloo Software.

Moravec, H. (1988). *MIND CHILDREN: The Future of Robot and Human Intelligence*. Harvard University Press.

Morrison, R. A. (1968). Iron mule train. In *Cornell Aeronautical Lab. ISTVS Off-Road Mobility Research Symposium*. Washington DC, June.

Muybridge, E. (1957). *Animals in motion*. Dover Publications, Inc., New York. (First published in 1899).

Nabulsi, S. and Armada, M. (2004). Climbing strategies for remote maneuverability of roboclimber. In *35th International Symposium on Robotics (ISR'04)*, pages 121–126. Paris, France.

Nagy, P. (1991). *An investigation of walker/terrain interaction*. Ph.D. thesis, Carnegie Mellon University.

Nonami, K., Huang, Q. J., Komizo, D., Shimoi, N., and Uchida, H. (2000). Humanitarian mine detection six-legged walking robot. In *Third International Conference on Climbing and Walking Robots (CLAWAR'00)*, pages 861–868. Madrid, Spain.

Ogata, K. (1996). *Modern Control Engineering*. Prentice Hall, 3rd edition.

Orin, D. (1976). *Interactive control of a six-legged vehicle with optimization of both stability and energy*. Ph.D. thesis, The Ohio State University.

Orin, D. (1982). Supervisory control of a multilegged robot. *The International Journal of Robotic Research*, **1**(1), 79–91.

Pack, D. and Kang, H. (1999). Free gait control for a quadruped walking robot. *International Journal of Laboratory Robotics and Automation*, **11**(2), 71–81.

Pal, P. and Jayarajan, K. (1991). Generation of free gaits–a graph search approach. *IEEE Transaction on Robotics and Automation*, **RA-7**(3), 299–305.

Papadopoulos, E. and Rey, D. (1996). A new measure of tipover stability margin for mobile manipulators. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3111–3116. Minneapolis, Minnesota.

Paul, R. (1981). *Robot Manipulators*. The MIT Press, Cambridge, Massachusetts.

Pfeiffer, F. and Rossmann, T. (2000). About friction in walking machines. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2165–2172. San Francisco, CA.

Pfeiffer, F. and Weidemann, H. J. (1991). Dynamics of the walking stick insect. *IEEE Control Systems Magazine*, **11**(2), 9–13.

Plustech-Oy (2005). *The walking forest machine concept*. Available: `http://www.plustech.fi/`.

Prokopiou, P., Tzafestas, S., and Harwin, W. (1999). A novel scheme for human-friendly and time-delays robust neuropredictive teleoperation. *Journal of Intelligent and Robotic Systems*, **25**(4), 311–340.

Raibert, M. (1986). *Legged robots that balance*. The MIT Press, Cambridge, Massachusetts.

Raibert, M., Chepponis, M., and Brown Jr., H. (1986). Running on four legs as though they were one. *IEEE Journal of Robotics and Automation*, **RA2**(2), 70–82.

Riddestrom, C., Ingvast, J., Hardarson, F., Gudmundsson, M., Hellgreen, M., Wikander, J., Wadden, T., and Rehbinder, H. (2000). The basic design of the quadruped robot WARP1. In *Proceedings of the International Conference on Climbing and Walking Robots*, pages 87–94. Madrid, Spain.

Rosenblatt, J. (1997). DAMN: A distributed architecture for mobile navigation. *Journal of Experimental and Theoretical Artificial Intelligence*, **9**(2), 339–360.

Rossmann, T. and Pfeiffer, F. (1998). Control of a pipe crawling robot. In *Conference Proceedings Biology and Technology of Walking, Euromech 375*, pages 133–140. Munich, Germany.

Russell, M. (1983). ODEX I: The first functionoid. *Robotics Age*, **5**(5), 12–18.

Salmi, S. and Halme, A. (1996). Implementing and testing a reasoning-based free gait algorithm in the six-legged walking machine MECANT. *Control Engineering Practice*, **4**(4), 487–492.

Schneider, A. and Schmucker, U. (2000). Adaptive six-legged platform for mounting and service operations. In *Proceedings of the International Conference on Climbing and Walking Robots*, pages 193–200. Madrid, Spain.

Sciavicco, L. and Siciliano, B. (2000). *Modelling and control of robot manipulators*. Springer-Verlag, London, 2nd edition.

Shih, C. and Klein, C. (1993). An adaptive gait for legged walking machines over rough terrain. *IEEE Transactions on Systems, Man and Cybernetics*, **23**(4), 1150–1155.

Shih, L., Frank, A., and Ravani, B. (1987). Dynamic simulation of legged machines using a compliant joint model. *The International Journal of Robotic Research*, **6**(4), 33–46.

Shimizu, M., Tawara, T., Okumura, Y., Furuta, T., and Kitano, H. (2002). A sensor fusion algorithm and a sensor management mechanism for the compact humanoid Morph with numerous sensors. In *Proceedings of the International Conference on Climbing and Walking Robots*, pages 279–285. Paris, France.

Shin, K. and McKay, N. (1985). Minimum-time control of robotic manipulators with geometric constraints. *IEEE Transactions on Automatic Control*, **AC-30**(6), 531–541.

Shing, T. K. (1994). *Dynamics and Control of Geared Servomechanisms with Backlash and Friction Consideration*. Ph.D. thesis, The University of Maryland.

SILO4 (2005). *The SILO4 Walking Robot*. Available: `http://www.iai.csic.es/users/silo4`.

Song, S. and Waldron, K. (1989). *Machines that walk: The adaptive suspension vehicle*. The MIT Press, Cambridge, Massachusetts.

Spong, M. W. (1987). Modeling and control of elastic joint robots. *ASME Journal of Dynamic Systems, Measurement, and Control*, **109**, 310–319.

Spong, M. W. and Vidyasagar, M. (1989). *Robot Dynamic and Control*. John Wiley and Sons, Inc.

Sun, S. S. (1974). *A theoretical study of gaits for legged locomotion systems*. Ph.D. thesis, The Ohio State University, Columbus, Ohio.

Swevers, J., Ganseman, C., Chenut, X., and Samin, J. (2000). Experimental identification of robot dynamics for control. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 241–246. San Francisco, CA.

Takanobu, H., Tabayashi, H., Narita, S., Takanishi, A., Guglielmelli, E., and Dario, P. (1999). Remote interaction between human and humanoid robot. *Journal of Intelligent and Robotic Systems*, **25**(4), 371–385.

Tirmant, H., Baloh, M., Vermeiren, L., Guerra, T. M., and Parent, M. (2002). B2: An alternative two wheeled vehicle for an automated urban transportation system. In *IEEE Intelligent Vehicle Symposium*. Versailles, France.

Todd, D. J. (1985). *Walking machines: An introduction to legged robots*. Kogan Page, Ltd.

Todd, D. J. (1991). An evaluation of mechanically co-ordinated legged locomotion (the iron mule train revisited). *Robotica*, **9**, 417–420.

Tomovic, R. (1961). A general theoretical model of creeping displacements. *Cybernetica*, **4**.

Valentin, N. and Denoeux, T. (2001). A neural network-based software sensor for coagulation control in a water treatment plant. *Intelligent Data Analysis*, **5**, 23–39.

Voth, D. (2002). Nature's guide to robot design. *IEEE Intelligent Systems*, pages 4–7.

Vukobratovic, M. and Juricic, D. (1969). Contribution to the synthesis of biped gait. *IEEE Transactions on Biomedical Engineering*, **BME-16**(1), 1–6.

Waldron, K. J. and McGhee, R. B. (1986). The adaptive suspension vehicle. *IEEE Control Systems*, pages 7–12.

Waldron, K. J., Vohnout, V. J., Pery, A., and McGhe, R. B. (1984). The adaptive suspension vehicle. *The International Journal of Robotics Research*, **3**(2), 37–42.

Wettergreen, D. and Thorpe, C. (1992). Gait generation for legged robots. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 1413–1420.

Wettergreen, D., Thorpe, C., and Whittaker, W. L. (1993). Exploring mount erebus by walking robots. *Robotics and Autonomous Systems*, **11**, 171–185.

Wickstrom, N., Taveniku, M., Linde, A., Larsson, M., and Svensson, B. (1997). Estimating pressure peak position and air-fuel ratio using the ionization current and artificial neural networks. In *Proceedings of IEEE Conference on Intelligent Transportation Systems (ITSC'97)*. Boston, USA.

Williams, D. M. (2005). *Book of Insect Records*. University of Florida, Available: `http://ufbir.ifas.ufl.edu/chap30.htm/`.

Wilson, D. M. (1966). Insect walking. *Annual Review of Entomology*, **11**.

Wong, H. and Orin, D. (1993). Dynamic control of a quadruped standing jump. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 346–351. Atlanta, Georgia.

Yang, H. and Slotine, J. J. (1994). Fast algorithms for near-minimum-time control of robot manipulators. *The International Journal of Robotic Research*, **13**(6), 521–532.

Yi, K. Y. and Zheng, Y. F. (1997). Biped locomotion by reduced ankle power. *Autonomous Robots*, **4**(3), 307–314.

Yobotics (2002). *Yobotics! Simulation Construction Set: Users Guide*. Yobotics Inc., Boston, MA. Available: `http://www.yobotics.com/`.

Yoneda, K. and Hirose, S. (1997). Three-dimensional stability criterion of integrated locomotion and manipulation. *Journal of Robotics and Mechatronics*, **9**(4), 267–274.

Yoneda, K., Iiyama, H., and Hirose, S. (1996). Intermitent trot gait of a quadruped walking machine dynamic stability control of an omnidirectional walk. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3002–3007. Atlanta, Georgia.

Yuan, J. and Vanrollehghem, P. (1998). One step ahead product predictor for profit optimisation of penicillin fermentation. In *Proceedings FAC Conference on Computer Applications in Biotechnology*, pages 183–188. Osaka, Japan.

Zhang, C. and Song, S. (1989). Gaits and geometry of a walking chair for the disabled. *Journal of Terramechanics*, **26**(3/4), 211–233.

Zhang, C. and Song, S. (1990). Stability analysis of wave-crab gaits of a quadruped. *Journal of Robotic Systems*, **7**(2), 243–276.

Zhou, D., Low, K., and Zielinska, T. (2000). A stability analysis of walking robots based on leg-end supporting moments. In *Proceedings of the IEEE International*

Conference on Robotics and Automation, pages 2834–2839. San Francisco, California.

Zykov, V., Bongard, J., and Lipson, H. (2004). Evolving dynamic gaits on a physical robot. In *Genetic and Evolutionary Computation Conference GECCO'04*. Late Breaking Paper.

Index