Otto-von-Guericke University Magdeburg
Faculty of Electrical Engineering and Communicatory Technology
Institute of Electrical Power Systems

Contact Processing within Walking Robots' Design

by

M.Sc. Mykhaylo Konyev

A dissertation submitted in conformity with the requirements
for the degree of Doctor-Engineer (Dr.-Ing.)

Magdeburg, Germany
July 2011

I. Abstract

This work is to deal with virtual and real development of walking robots. The work is divided in two parts – development of virtual environment for designing and investigation of walking robots, and detailed presentation of two walking robots – six-legged robot "ANTON" and biped robot "ROTTO".

In the first part the contact processing, which is an important part of multi-domain simulation task is discussed in details. It includes collision detection and contact response and is one of the most difficult, but most important areas in the simulation of the multi-body systems. However, the most widespread multi-body simulators, like Matlab/SimMechanics, don't support the contact processing. Other multi-body simulators, like Vortex or ODE, support the contact processing, but are more limited in the rest of the functionality. This thesis presents the overview of the most popular techniques in the contact processing and the implementation of the chosen contact processing technique into Matlab/Simulink. The collision detection has been implemented using existing software tool Solid. The contact response for both contact phenomena - collision forces and friction forces – has been developed for the force-based approach. The functionality of the developed contact processing has been performed by the contact tasks of a six-legged and biped robots as well as of a number of different geometrical objects and has been compared with experiment results.

In the second part of this work a new improved six legged mobile robot "ANTON" and a new biped robot "ROTTO" are presented. Their mechanical structure, sensor system and control system are discussed in details. The hierarchically and modular build control algorithms are presented. The Software-in-the-Loop and Rapid Control Prototyping frameworks, which are used by robot development, are presented. An industrial Ethernet-based real-time communication protocol is introduced and the communication ability between the robot-side hardware and PC-side control system is investigated.

II. Acknowledgement

This research work has been carried out within my Ph.D. student fellowship at the Institute of Electrical Power Systems Otto-von-Guericke University Magdeburg in cooperation with Fraunhofer Institute for Factory Operation and Automation IFF Magdeburg.

I owe special thanks to Univ.-Prof. Dr.-Ing. habil. Dr.h.c. Frank Palis for support and help rendered to me on this work task organization. He showed me invaluable scientific and interpersonal aspects of research work. Also a note of thanks goes to Prof. Dr. sc. techn. Ulrich Schmucker for excellent support during the research work.

My deep gratitude goes to the research Dr.-Ing. habil. Anatoliy Schneider, Dr.Ing. Yuriy Zavgorodniy, Dr.Ing. Vadym Rusin and the whole team of RobotsLab group (M.Sc. A.Melnykov, M.Sc. A.Rudskyy and M.Sc. A.Telesh) for an excellent team-work and invaluable directions that helped me completing this work.

My appreciation and thanks to a group of qualified specialists of Institute of Electrical Power Systems and Fraunhofer Institute for Factory Operation and Automation IFF Magdeburg, for support and very good team work.

And finally I send my love and thanks to my wife and my parents, without whom this work and my studies were impossible.

Magdeburg, 18th June 2011

III. Contents

Chapter 1

Introduction

Walking robotics is concerned with the study of those machines that can replace human beings in the execution of a task, as regards both physical activity and decision making. The goal of the introductory chapter is to point out the problems related to the requests to walking robots, with reference to the general framework of robotics. A survey of the most known multiped walking robots as well as anthropomorphous is presented. Topics of virtual development and simulation of walking robots are introduced which will be examined in the following chapters.

Interest in developing walking machines as vehicles with increased possibility has picked up in the last sixty years. Walking machines have a number of principle advantages over wheeled and caterpillar machines [Sch06]:

- They can move over terrains with obstacles up to the size of a leg.
- The movement of their legs provides comfortable motion of the body with cargo and passengers over rough terrain.
- They can work in a complexly-structured environment (sloped, confined work and operation, etc.) and move on consolidating ground and unknown terrain with varying load capacity.
- They can use one or more legs as an auxiliary manipulator.

Although a large number of research works in the area of walking robots is already investigated, questions of development of walking robots, building of low level control systems as well as communication interfaces, design of control algorithms are still actual.

This work is to deal with virtual and real development of walking robots. The work is divided in two parts – development of virtual environment especially a contact processing library for designing and investigation of walking robots, and detailed presentation of two walking robots – six-legged robot "ANTON" and biped robot "ROTTO". This thesis is a result of collaboration work within the scope of RobotsLAB team (Otto-von-Guericke University Magdeburg, Faculty of Electrical Engineering and Communicatory Technology, Institute of Electrical Power Systems). A great number of results is already published in [Kon051][Kon05][Kon092][Kon101][Kon072][Kon061][Kon06][Kon10][Kon102][Kon09][Kon091][Kon08][Kon082][Kon081][Kon071][Kon07][Juh08][Zav08][Sch06][Rus07] and others. This thesis draws the final stroke on my work in RobotsLAB team on development, simulation and control of walking robots.

1.2 Multilegged Walking Robots

In the history of scientific research [Sch06] there are many examples of designing robotic devices like mechanical toys and anthropoid robots. For example, in the 4th century in China was built a mechanical crow. Design studies of Leonardo da Vinci and the development of mechanical dolls with human-like behavior in the middle age may be regarded as groundwork for legged locomotion.

The first mechanical constructions were based on a predetermined movement so that an adaptation to the ground was not possible. To obtain more flexibility several researches in fifties started to assign the motion control of the walking machine completely to a human operator by controlling the different degrees of freedom manually.

This section deals with an overview of successful multilegged walking robots of the last years. One of the first worthy of attention walking robot is a LAURON. LAURON [Wik1] is a six-legged walking robot, which is being developed at the Forschungszetrum Informatik Karlsruhe (FZI) in Germany [Gaß01]. The mechanics and the movements of the robot are biologically-inspired, mimicking the stick insect Carausius Morosus. The

development of the LAURON walking robot started with basic research in the field of six-legged locomotion in the early 1990s and led to the first robot, called LAURON. In the year 1994, this robot was presented to public at the CeBIT in Hanover. This first LAURON generation was, in contrast to the current generation, controlled by an artificial neural network, hence the robot's German name: LAUfROboter Neuronal gesteuert (English: Walking Robot, Neural Controlled). The current generation LARUON IV was finished in 2004.

The SCORPION [SCO] is an eight-legged walking robot for hazardous outdoor-terrain. It can go where other robots get into trouble. It uses a biomimetic control concept, which allows a very flexible, robust walking behavior. The walking gaits of the SCORPION robot are based on research on walking patterns of real scorpions. The SCORPION can be controlled in an intuitive way with a GUI and an optional voice control. The developed models of the biological motor systems enable the robot to adapt autonomously to a multitude of different terrains and obstacles. Possible future fields of application include exploration of hazardous environments, e.g. in SAR missions. Currently an amphibious version of the SCORPION is under development. A copy of the SCORPION is in use at the NASA Ames Research Center to evaluate the advantages of legged systems for extraterrestrial missions.

Researchers at the Leg Laboratory have built a series of legged robots [MIT], including one-legged hoppers, bipedal runners, bipedal walkers, a quadruped, and two kangaroo-like robots. Taken collectively, these machines traverse simple paths, run with several different gaits, run fast (13 mph), walk over flat ground and rolling terrain, jump over obstacles, climb a simplified stairway, and perform rudimentary gymnastic maneuvers. Although no one machine performs all these tasks, many of the machines use a common set of balance and control principles. The Leg Laboratory also uses physics-based computer simulations to study legged locomotion and to build automated computer characters.

Worldwide known firma Boston Dynamics [Bos] builds advanced robots with remarkable behavior: mobility, agility, dexterity and speed. BigDog is the alpha male of the Boston Dynamics robots. It is a rough-terrain robot that walks, runs, climbs and carries heavy loads. BigDog is powered by an engine that drives a hydraulic actuation system. BigDog

has four legs that are articulated like an animal’s, with compliant elements to absorb shock and recycle energy from one step to the next.

LittleDog is a quadruped robot designed for research on learning locomotion. LittleDog has four legs, each powered by three electric motors. The legs have a large range of motion. The robot is strong enough for climbing and dynamic locomotion gaits. The onboard PC-level computer does sensing, actuator control and communications. LittleDog's sensors measure joint angles, motor currents, body orientation and foot/ground contact.

Development and detailed information to multilegged robots which are developed in the Otto-von-Guericke-University Magdeburg will be discussed in the Chapter 3.

1.3 Anthropomorphous Walking Robots

First successful two legged walking robot which was able to walking relative stable was developed in 1972 in Waseda-University in Japan and called WL-5 [Kat72]. A number of other walking robots were built in this university. In 1982-1983 a WL-10R was introduced [Was], in which the rotaries type servo-actuator (RSA) was introduced and carbon fiber reinforced plastic (CFRP) was used in its structural parts. The WL-10R added one more degree of freedom at the yaw axis of the hip joint. Consequently the WL-10R acquired the function of walking laterally, turning and walking forward and also backward, which are called plane walking (4.4sec/step). And the first dynamic walking was offered in 1984 on walking robot WL-10D. Thereto, the torque sensors were attached to the ankle and the hip joint to allow flexible control of a change-over phase (transition-phase from standing on one leg to standing on the other leg) using torque feedback.

The second well known two legged robot is ASIMO from Honda Motor Co. ASIMO [Wik2] was created at Honda's Research & Development Wako Fundamental Technical Research Center in Japan. It is the current model in a line of twelve that began in 1986 with E0. ASIMO resembles a child in size and is the most human-like robot HONDA has made so far. The robot has 7 DOF in each arm – two joints of 3 DOF, shoulder and wrist, giving "Six degrees of freedom" and 1 DOF at the elbow; 6 DOF in each leg – 3 DOF at the crotch, 2 DOF at the ankle and 1 DOF at the knee; and 3 DOF in the neck

joint. The hands have 2 DOF – 1 DOF in each thumb and 1 in each finger. This gives a total of 34 DOF in all joints.

The research works at the Leg Laboratory was already discussed in the Section 1.2. This group has not only works on multilegged robots, but a lot of works in the area of development of two legged robots as well as their dynamical control.

Within the framework of the DFG "Priority Program Autonomous Walking", the JOHNNIE-project developed and builds an autonomous biped walking robot at TU Munich [Joh]. JOHNNIE's structure resembles the human locomotors apparatus and has a total of 17 actively driven joints. The overall weight is about 49kg, the height is 1,80m. Each leg incorporates six driven joints. Three of them are located in the hip, one actuates the knee and another two drive the ankle joint (pitch and roll). Furthermore, the upper body is equipped with a rotational degree of freedom about the body's vertical axis. Two arms with two DoFs each are employed to compensate the overall momentum about the body vertical axis. The joints are driven by brush DC-motors in combination with lightweight gears. Joint angles and velocities are measured by incremental encoders. Further, two six-axis force sensors in the feet measure the ground reaction forces. An inertial measurement unit consisting of a three-axis accelerometer and three gyroscope sensors determines the spacial orientation of the upper body. A PCI-I/O-board interfaces the main computer (Pentium IV 2,8GHz) with the sensors and motor drivers. The control algorithms run on the PC as RT-Linux kernel modules. Results from the JOHNNIE-project suggest that significant increases in walking speed, dexterity and autonomy are only possible if the robot's hardware is updated to the state of the art of mechatronic technology. Currently they are developing the humanoid robot LOLA [LOL] as part of the project-cluster "Natur und Technik intelligenten Laufens" (biological and technical aspects of intelligent locomotion) financed by the Deutsche Forschungsgemeinschaft (German Research Foundation).

Development and detailed information to biped robot which is developed in the Otto-von-Guericke-University Magdeburg will be discussed in the Chapter 4.

1.4 Virtual Development of Mechatronical Systems

Robot development process is complex and often very difficult to manage particularly when development processes are interconnected. „Virtual Development" is the keyword not only in industry, but also in robotic research and development.

Virtual development of mechatronical systems helps to carry out the most significant tasks such as:

- optimization of the existing or planning prototype construction;
- testing, optimization or/and updating of the proposal drive system;
- optimization of the control system properties;
- development and/or improvement of applied control algorithms with the purpose of expansion of capabilities of the robot (complex service operations, interaction with an unknown environment and/or a ground, complex locomotion and navigation);
- analysis of dynamic processes in the virtual environment, especially in absence of some physical sensor systems.

Development time for the most important tasks such as design, optimization and updating operations are significantly reduced in comparison with using physical prototypes.

1.5 Goals and Structure of Thesis

One of the scientific goals of this work is to develop a virtual environment for designing, simulation, control and investigation of walking robots. The developed virtual environment must satisfy the requirement of simulation of multi-body system such as:

- dynamical simulation of mechanical system, thereby be possible to solve the Langrage equation in linear as well as in non-linear form;
- be able to simulate the sensor system: analog devices (floating null point, measuring noise etc.) and digital devices (representation of measuring signal in fixed-point as well as in floating-point form, consideration of delays and samplings etc.);

- feature to simulate the actuators as well as possibility to develop the control systems for them (simulation of PWM-generators, DC/AC-motors, current-/velocity-/position-/force-control loops etc.);
- be able to detect and to simulate the interaction between some mechanical objects (contact recognition and processing).

Next scientific goal of this work is to develop a universal communication bridge/protocol to be able to realize communication between virtual environment and real mechatronical system. Such communication bridge must be able to connect the software and the hardware control systems.

On the basis of above mentioned tasks two real walking robots are developed. First step is the development of a virtual prototype of robots in virtual environment as well as the whole control system for them. The kinematical structure of mechanical construction of robots must be firstly investigated in virtual environment. The actuators as well as the supply power should be estimated after a number of experiments in virtual environment. Second step is production of real robots and development of communication bridge/protocol between real robots and the software control systems for them, which were already designed and investigated in virtual environment.

Therefore Chapter 2 deals with task of development of virtual environment. It presents the two simulation software tools for simulation of multi-body systems. Also a new contact processing library as well as the communication possibility between Matlab/Dymola and contact processing library will be introduced.

Requirements and goals to the walking robots will be discussed in Chapter 3 and Chapter 4. Here will be proposed hardware as well as a new software control system discussed. Two different communication bridges and communication philosophies will be introduced.

Chapter 2

Simulation of Multi-Body Systems

Simulation of multi-body system, including simulation of mechanical system, electrical part, contact processing and so on, is a necessary part of development and commissioning of the robotic system. Simulation results make it possible to accelerate development process, to find relevant errors and to simplify whole development process at the beginning, to obtain properties of the development system and so on.

Contact processing, including collision detection and contact response, is one of the most difficult, but most important areas in the simulation of the multi-body systems. However, the most widespread multi-body simulators like Matlab (with Simulink and SimMechanics) and Dymola (exploiting the Modelica language) don't support contact processing. Other multi-body simulators like Vortex or ODE support the contact processing, but are more limited in other functionalities.

After giving an overview of the both multi-body simulators, Matlab and Dymola, this chapter focuses on development and presentation of certain parts of multi-domain robotic systems on the example of the six-legged walking robot "ANTON" and the anthropomorphous robot "ROTTO". Advantages and disadvantages of both simulations software will be substantiated.

This chapter also presents an overview of the most popular techniques in contact processing by simulation of the multi-body systems and shows the implementation of the chosen contact processing technique into Matlab/Simulink as well as into Dymola. The collision detection has been implemented using existing software tool Solid. The contact response for both contact phenomena - collision forces and friction forces – has been developed for the force-based approach. The functionality of the developed contact processing has been performed by the contact tasks on different test objects, such as different simple geometrical objects, anthropomorphic manipulator, six-legged and anthropomorphous robots.

2.1 Introduction

A *multi-body system* [Wik01] is used to model the dynamic behavior of interconnected rigid or flexible bodies, each of which may undergo large translational and rotational displacements. The systematical treatment of the dynamic behavior of interconnected bodies has led to a large number of important multi-body formalisms in the field of mechanics. The simplest bodies or elements of a multi-body system were already treated by Newton (free particle) and Euler (rigid body). Euler already introduced reaction forces between bodies. Later on, a number of formalisms has been derived, only to mention Lagrange's formalisms based on minimal coordinates and a second formulation that introduces constraints. Basically, the motion of bodies is described by its *kinematics* behavior. The *dynamic* behavior results due to the equilibrium of applied forces and the rate of change in the momentum. Nowadays, the term multi-body system is related to a large number of engineering fields of research, especially in robotics and vehicle dynamics. As an important feature, multi-body system formalisms usually offer an algorithmic, computer-aided way to model, analyze, simulate and optimize the arbitrary motion of possibly thousands of interconnected bodies.

Dynamic simulation [Wik] is the use of a computer program to model the time varying behavior of a system. The systems are typically described by ordinary differential equations or partial differential equations. As mathematical models incorporate real-world constraints, like gear backlash (engineering) and rebound from a hard stop, equations become nonlinear. This requires numerical methods to solve the equations. A numerical simulation is done by stepping through a time interval and calculating the integral of the

derivatives by approximating the area under the derivative curves. Some methods use a fixed step through the interval, and others use an adaptive step that can shrink or grow automatically to maintain an acceptable error tolerance. Industrial uses of dynamic simulation are many and range from nuclear power, steam turbines, 6 degree of freedom vehicle modeling, electric motors, econometric models, biological systems, robot arms, mass spring dampers, hydraulic systems, and drug dose migration through the human body to name a few. These models can often be run in real time to give a virtual response close to the actual system. This is useful in process control and mechatronic systems for tuning the automatic control systems before they are connected to the real system or for human training before they control the real system.

In the mechanical systems certain machine elements usually interact with each other. When a mathematical model of such system is designed, the interactions between the parts can be divided in the two following categories:

- *Mechanical joints* are representing permanent constraints on relative motions between the connected parts.
- *Mechanical contacts* are observed while surfaces of bodies are touching each other. Two major phenomena – differentiated by the duration of the contact – can occur during the interaction:
    - *Resting* or *sliding contact* (where static or dynamic friction forces can arise).
    - *Impulsive collision forces* (almost instantaneous, typically short-time interactions caused by compression of contacting bodies).

The forces occur when the surfaces of bodies touch each other. Two major phenomena occur in the mechanical contacts: collision contacts (causing collision response forces) and friction contacts (causing static or dynamic friction forces).

Contact processing is a difficult task [Eng00][Ott04][Gal06]. The bodies can move in a complicated way, and they can have complex geometries. In the case of contacting bodies, the penetration of them must be prevented. There is a tradeoff between efficiency and accuracy. Accurate methods for computing contact forces are based on finite element methods. Such methods are based on subdivision of bodies into very small fragments. The surfaces of two colliding bodies are to be covered by a mesh and the rele-

vant forces in the contact are to be computed for each point on the mesh. The resulting forces can be defined by integration of all forces acting on the contact surface. These methods are implemented in software packages for FEM-analysis (ANSYS, Nastran etc) or in multi-body simulation (MSC Adams etc). Experiments [Fri97] show that these methods are accurate, but require tremendous computing resources and therefore are very slow. However, many simulation applications do not require extreme accuracy and additional assumptions are taken into account providing high simulation speed, but decreasing the accuracy. As a matter of fact, different assumptions lead to different computation methods but with the same (or nearly the same) computation results. In such cases, it's not important for the application what assumptions and methods were used.

2.2 Simulation of Multi-Body System

There exists a range of commercial computer packages that can be used to solve problems in multibody systems analysis. General-purpose MBS programs are able to address a large set of problems across a wide range of engineering industries and are not restricted to the applications in contact processing/response discussed here. The main use of contact processing library within the robotics is to simulate the interaction of robot and environment. The main analysis concept consists of combination of a number simulation tools that perform three-dimensional kinematic, static, quasi-static or dynamic analysis of mechanical systems. It must be also possible to include differential equations directly in the solution, which allows the modeling of a variety of control systems.

One of the aims of the work consisted in choosing the unified software tools for development and testing of the above mentioned subsystems by the criteria "convenience of use – availability – implemented capabilities". The optimum is the combination of Matlab/Simulink/xPC-Target, Dymola/Modelica, some external visualization tool and contact processing library.

2.2.1 Matlab

Matlab consists of a collection of integrated products for data analysis, visualization, application development, system modeling, simulation and code generation as well as embedded real-time operation system called xPC-Target. In combination with some visualization tool based on DirectX, Matlab provides an engine for real-time visualization

and simulation of rigid-body dynamics, collision detection, contact modeling, and collision response. Matlab is unsurpassed for any 3D interactive simulation requiring stable and accurate physics. Matlab/SimMechanics allows programmers to simulate natural behavior of objects in the physical world. By using the Matlab toolboxes it is possible to create interactive 3D simulations of the objects with constraints under any circumstances. The functionality provided by physics simulation software is particularly needed for training and prototyping purposes. 3D models of articulated machinery must also mimic the behavior of the actual machines.

2.2.2 Dymola

Modelica and Dymola have been chosen as target modeling language and multi-domain simulator. The development and promotion of the free Modelica language is organized by the non-profit Modelica Association. Modelica is a modern language supporting non-causal modelling with bidirectional mathematical equations and object-oriented paradigm to facilitate reusing of knowledge in many modeling domains at the same time. Besides process-oriented and control system components it provides modeling abilities in various engineering domains (including mechanical-, electrical-, hydraulic-, pneumatic- and thermal subsystems).

Modelica offers a general type system that unifies object-orientation, multiple inheritance and templates within a single class (model) construct. The components are based on standardized interface definitions, and can contain formalisms such as ordinary differential equation systems (ODEs), differential algebraic equations (DAEs), state machines or Petri nets.

Modelica programs are built from classes. Like in other object-oriented languages, a class contains variables, i.e. class attributes representing data. The main difference compared with traditional object-oriented languages is that instead of functions (methods) equations are used to specify behavior. Equations can be written explicitly, like *a=b*, or be inherited from other classes. Equations can also be specified by the connect statement. The statement *connect(v1,v2)* expresses general Kirchhoff coupling between the connector variables *v1* and *v2*: the potential values of those are meant to be equal, and the flow values (e.g.: electrical currents) sum to zero for each connected node. This gives

a flexible way of specifying topology of physical systems described in an object-oriented way using Modelica.

The Modelica language supports general multi-domain modeling, where arbitrary control algorithms can be specified for the mechatronic models. For a brief overview about the language please refer to the article [Elm99]. The complete reference of the 2.1 version of the language can be found in the book [Fri04].

Dymola – Dynamic Modelling Laboratory – is a commercial simulation environment for Modelica with unique multi-engineering capabilities. Using Dymola it is possible to simulate the dynamic modeling of mechanical, electrical, thermodynamic, hydraulic, pneumatic, thermal, power and control components described in Modelica language. The flexibility of Dymola depends on the powerful Modelica language and its aforementioned technologies.

The non-causal modelling possibility, with bidirectional data flow between components, as a major advantage of Dymola/Modelica (for example comparing against Matlab/Simulink) is emphasize in this work.

2.3 Contact Processing

Implementation of the contact processing is based on the following steps:

- Mechanical models, which describe physical bodies, should be extended to describe contacting physical bodies.
- There should be a routine that can *detect collisions* and can return detailed information regarding contact parameters, such as contact points, penetration depth and their velocities, crossover surfaces or volumes, etc.
- A special routine should calculate the *contact response* (reaction forces and torques) from contact parameters.
- Each of these components should have an interface that allows replacing its implementation without doing major redesign of the other components.

Figure 2.1 presents four basic components of contact processing, which are derived from discussion above and will be described more detailed later.

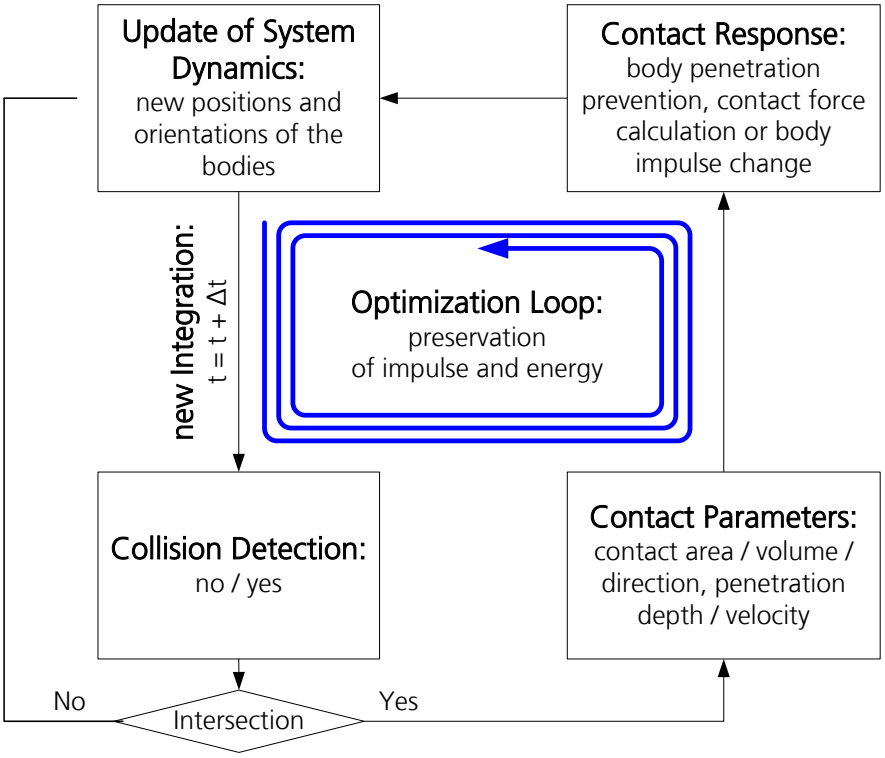

Figure 2.1 - General scheme of contact processing in the mechanical multi-body simulation [Kon071]

2.3.1 Collision Detection

The collision detection between bodies, which are indirectly described, for example using sets of points in the space, means that the contact points or trajectories between two bodies are to be defined.

In the first part (*collision culling*), the pairs of non-colliding objects should be excluded. These can be performed using the methods of space division like Quadtree/Octree, BSP-

Tree and Sweep-and-Prune. The whole space should be divided and the potentially non-colliding objects should be excluded. The same techniques are also used for object culling to accelerate 3D rendering. In a dynamic scene time coherence is utilized to refresh the data structures between consecutive calls for collision culling.

In the second part (*broad collision detection*), the possibility of two objects collision by means of so-called Bounding Volumes like AABB, sphere, OBB, k-DOP should be defined. These covers simplify the complex geometry objects and make the collision detection simpler and faster.

The third part (*exact collision detection*) defines the collision between the contacting objects. Spatial, hierarchical data structure, as AABB-tree, OBB-tree, sphere-tree or k-DOP-tree should be used for faster collision definition. In this part the closest features can be found, a contact manifold can be generated, contact points can be enumerated and contact parameters, such as contact plane, contact volume, contact normal and penetration depth can be obtained.

On the market there are a bunch of non-commercial software packages that support collision detection e.g. SWIFT, Bullet, ODE and others, offering usually C or C++ interfaces for external applications. Out of this set some tools (e.g.: V-Collide) – which can just indicate the existence of penetrating geometries and do not offer any useful contact parameters to be retrieved – must be excluded.

2.3.2 Contact Parameters

The intersection test follows the third part of collision detection and finds the geometrical contact parameters, as contact plane, contact volume, contact normal and penetration, with the aim of the following methods: intersection point of ray and sphere, intersection point of ray and plane, intersection point of ray and triangle, intersection line of two planes, intersection line (point) of two triangles. The presented on the market algorithms (Lin-Canny Closest Features Algorithm, I-/Q-COLLIDE, V-Clip, OBB-Tree, QuickCD, KDS, GJK, GJK-based EPA) combine "collision detection" and "contact parameters detection" tasks [Lin92][Chu96][Sch96][Eri99][Gva03] and are implemented in the leading software, e.g. SWIFT, SOLID, Bullet, ODE and others.

2.3.3 Contact Response

The contact response seems to be the most problematic and controversial part of the contact processing, since many computation approaches exist, which require different input information and may produce quite different numerical results. The following two methods are commonly used in the contact processing: the *impulse-based method* and the *force-based method*. Both assume that the bodies are rigid. The update of system dynamics is closely connected with calculation of contact response and therefore both parts should be explained together.

2.3.4 Impulse-based Approach

The impulse-based approach uses collision impulses between the bodies and changes the velocity vector of the bodies during the contact [Mir96][Zha96]. This method based on an impact law such as Poison's hypothesis. It considers the impulse conservation law and operates with the impulses of the colliding bodies before and after the collision as well as with the restitution coefficient of materials.

The main advantages of this method are that only a few constants are needed for description of the impact law and that the integrator step size is not influenced by the response calculation because it is performed during an infinitely small time instant. However, since the velocity is not continuous in the impulse-based model, the traditional ODE solvers can't be used. The continuous integration process in the solver should be stopped at the instant of collision and should be resumed with a new velocity. An alternative approach is based on writing a system of non-differentiable equations and applying a Newton [Pan90] method specially devised for such equations. The impulse-based approach can be easily used in MBS-based models if the collision impact on the other bodies in the system is negligible (i.e. in the system of free-flying bodies). In the other words, this approach can't be used in the cases of a static objects and structures consisting of several bodies connected by joints. Furthermore such idealized impact laws are only useful for stiff collisions. These properties restrict the applicability of the impulse-based method of the dynamical analysis.

2.3.5 Force-based Approach

An alternative approach of contact processing in multi-body mechanical systems is based on the force and torque model of collision. It is assumed that the contacting bodies penetrate each other and the separation forces are caused by this penetration. These forces try to prevent further penetration and to separate the contacting bodies.

The calculation of contact force magnitude is a difficult task and is sometimes not motivated by physics, but rather by numeric analysis. The overall result of collision should match physical laws (i.e. preservation of impulse, and preservation of energy). In addition it should be chosen so smooth that numerical methods used in simulation could

handle these functions. The many existing methods for the calculation of the force in the mechanical joints and contacts are divided into two following groups:

- *Force based methods with Lagrange multipliers formulation* models the mechanical constraints (contacts and joints) with the reactive forces, which are presented as Lagrange multipliers **λ**. The constraint forces perform no work on the environment and the physical meaning of the mechanical contact is lost. The mechanical interaction of the bodies caused by the contact is represented by these reactive forces **λ**, which should be optimized between the simulation steps in the additional optimization loop (see Figure 2.1) under consideration of the energy or/and impulse preservation laws.
- *Force based methods with penalty formulation* models the mechanical contact with the strong possibly nonlinear spring. The active contact/friction forces (see Figure 2.2) perform work on the environment and the physical background of the mechanical contact is not lost. The mechanical interaction of the bodies caused by the contact is represented by the active forces $F_{CONTACT}$ and $F_{FRICTION}$, without any additional optimization between the simulation steps.

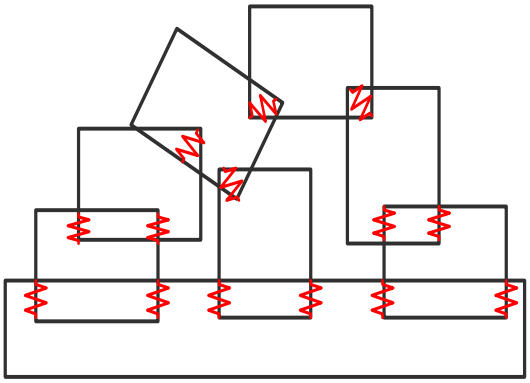

Figure 2.2 - Contact modelling with penalty-forces [Kon071][Kon07]

It is well known [Eng00] that during the collision a relatively large force occurs between the two colliding bodies for a very short period of time. The value and the direction of the force can be approximately computed for each simulation time step. The following properties [Eng00] of the collision force should be taken into account:

- The collision force and collision torque acting on an object is zero if the object does not collide.
- Between the start of collision and the end of collision a force is activated that prevents further penetration.

- If an ideal collision is modeled (collision of points masses), the resulting velocities after the collision are given by the law of preservation of linear momentum.
- A contact force acting on a body resting on a horizontal platform compensates for the gravitational force applied to the body. Therefore such an object does not move (its vertical acceleration and velocity is zero).

In order to balance the required accuracy and available computational power, the following rules [Eng00] must be applied for collision force computation:

- The collision force acting on a body is zero if the body does not penetrate with any other body.
- If body A penetrates body B, collision forces are created to act on the objects A and B and are applied at the point of contact on each body. The force is directed so that A and B are pushed away from each other due to this force. The direction of this force corresponds to the shortest displacement that can separate the bodies.
- The magnitude of the force is proportional to the depth of penetration of A and B. This depth is the length of the shortest displacement that can separate the bodies. This corresponds to the model of spring and damper, inserted between the bodies. The bodies are rigid, but the spring and the damper are not rigid. This also corresponds to the physics of collisions between elastic and homogeneous (isotropic) bodies.

The main advantages of the force-based approach are the simplicity and the possibility of using it for stiff and soft contacts. This approach works reasonably well if several contact points are also present at the same instant time. The disadvantage of this approach is that the integrator step size should be reduced in the contact phase in order to catch the rapidly changing contact forces and torques. And similar to the impulse-based method there is necessary to choose the contact parameters (spring, restitution) because the contact force is proportional not only to the penetration depth/velocity but also to the contact area and the contact volume.

2.3.6 Definition of contact points and normal direction

The point of contact can be defined in many different ways. The naive definition states that this is the point where two bodies touch each other the very first time (at the first instant of collision). Such a definition is only good for very short-duration contacts. If the bodies have longer contacts (i.e. the separation force should be computed during several time steps), then the point of the very first contact can differ from the point of contact a few steps later. An example of such a behavior is two bodies colliding and then keeping sliding contact. In this case the point of the first contact cannot be used for evaluation of contact force for the next steps.

Collision detection software packages determine a point which belongs to the intersection of the objects A and B if they collide. If the objects do not collide, the closest pair of their points can be determined.

The normal direction of the collision force can be determined in several different ways too. For a shot contact it can be natural to define the direction of the collision force as opposite to the relative velocity of the contact points, i.e. the points of the first touch between the bodies.

For more accurate determination of force direction, a mesh based on colliding surfaces has to be constructed, and a normal vector to the surface in each mesh point is used as local force direction. The resulting force direction is then found by integrating the vectors of all forces acting on the contact surface.

2.3.7 Constraints on the contact force

- Conditions at the start of the collision (constraint C_1):
  - $F(0) = 0$: The force should be zero at the start of collision.
  - $x(0) = 0$: The penetration depth should be zero at the start of collision.
  - $v(0) = v_0 > 0$: The actual speed is known at the start of collision.
- During the collision (C_2):
  - $F(t) < 0$: The force should always push the object away from the obstacle.
  - $x(t) > 0$: The penetration depth is positive (the object penetrates the obstacle).

- $v(t)$: The penetration speed is reduced to zero and, probably to negative values.

- If no external forces act on the colliding body (or this force is negligible), it behaves exactly according to the impulse law (C_3):
  - The time when collision ends is τ (it is an unknown duration parameter).
  - $F(\tau) = 0$: The force should be zero at the end of the collision and after that.
  - $x(\tau) = 0$: The penetration depth should be zero at the end of the collision.
  - $v(\tau) = v' = v_1 = -\varepsilon \cdot v_0$: The speed at the end of the collision can be predicted using the restitution coefficient $0 < \varepsilon < 1$.
- If a constant external force F_{ext} acts on the colliding body, it either behaves as above or, in case F_{ext} is large enough, the body rests on the obstacle, and the collision never ends ($\tau \to \infty;\ C_4$):
  - $\lim_{t\to\infty} F(t) = -F_{ext}$: The collision force should compensate the external force.
  - $\lim_{t\to\infty} x(t) = x_{rest} > 0$: The penetration depth should stabilize at some value.
  - $\lim_{t\to\infty} v(t) = 0$: The body rests, i.e. does not move anymore.

There can be many definitions for F satisfying these equations and relations. However, in most cases, the difference between these definitions (i.e. difference between the overall effects they cause) is negligible in comparison with the effect caused by the constraints. In practice, the definition for F is sometimes not motivated by physics, but rather by numeric analysis. It is chosen such way that overall result of collision matches physical laws (i.e. preservation of impulse, and preservation of energy). In addition it should be chosen so smooth that numerical methods used in simulation are able to handle with such function.

2.3.8 Collision Force Model (without friction)

In [Eng00] a number of collision force models are introduced, discussed and proved. One of simplest collision force model is a *first order linear collision force model*. This is a traditional penalty method, and is based on a linear dependency between the collision

force and the penetration depth. The collision force $F(t)$ depends on penetration $x(t)$ as follows:

$$F(t) = \begin{cases} -k \cdot x(t), & if\ x(t) > 0 \\ 0, & if\ x(t) \leq 0 \end{cases} \tag{2.1}$$

where k is a positive constant, called penalty coefficient. Physically this corresponds to a stiff spring, temporarily placed between the objects during the collision. The expression $-k \cdot x(t)$ corresponds to an ideal spring. The expression contains $x(t)$ in the first power only, therefore this method is a first order linear model.

For brevity of the solution, k might be replaced by another positive constant, $q^2 m$. In this case, taking into account the constraints C_1 and C_3, the system of equations appears as follows:

$$\begin{cases} F(t) = -q^2 m x(t), \\ F(t) = m\ddot{x}(t), \\ \dot{x}(0) = v_0, \\ x(0) = 0. \end{cases} \tag{2.2}$$

This results in the equation:

$$\ddot{x}(t) + q^2 x(t) = 0, \tag{2.3}$$

which has a solution:

$$x(t) = \frac{v_0 \sin qt}{q}, \tag{2.4}$$

and velocity is:

$$\dot{x}(t) = \frac{v_0 q \cos qt}{q}. \tag{2.5}$$

The properties of simple first order linear collision force model don't satisfy the above listed constraints on contact force, notably constrain C_3, while at the end of collision the velocity $v(\tau) = -v_0 \neq \varepsilon \cdot v_0$. This also means that the resulting velocity doesn't depend on the mass of objects, and doesn't depend on the penalty coefficient. Moreover, below will be shown that the collision force model, which based on position only, is purely elastic.

Force depending on position is $F(t) = -p\big(x(t)\big)$. In the same time $F(t) = m\ddot{x}(t)$. Therefore:

$$\ddot{x}(t) + p\big(x(t)\big) = 0, \tag{2.6}$$

where p is positive during the collision. Substituting $\dot{x}(t) = y\big(x(t)\big)$ into (2.6) one get:

$$\ddot{x}(t) = \frac{dy}{dt} = \frac{dy}{dx}\frac{dx}{dt} = \dot{x}\frac{dy}{dx} = y\frac{dy}{dx}. \tag{2.7}$$

The initial equation can be rewritten as:

$$y\frac{dy}{dx} + p(x) = 0. \tag{2.8}$$

If all parts of (2.8) are integrated, the result is:

$$\int ydy + \int_0^x p(u)du = const. \tag{2.9}$$

The first component can be transformed:

$$\int ydy = \frac{y^2}{2} = \frac{\dot{x}^2}{2}. \tag{2.10}$$

Using the initial condition in (2.2), one gets:

$$\frac{v_0^2}{2} + \int_0^0 p(u)du = const, \quad \xRightarrow[simplify]{} \quad const = \frac{v_0^2}{2}\ . \tag{2.11}$$

At the end of the collision process $t = \tau$ and $x(\tau) = 0$. Substitution in (2.2) gives:

$$\frac{\dot{x}^2(\tau)}{2} + \int_0^{x(\tau)} p(u)du = \frac{\dot{x}^2(\tau)}{2} + \int_0^0 p(u)du = \frac{v_0^2}{2}, \quad \xRightarrow[simplify]{} \quad x^2(\tau) = v_0^2. \tag{2.12}$$

Therefore, according to (2.6) $v' = -v_0$, which means that the collision is purely elastic ($\varepsilon = 1$) and doesn't satisfy above mentioned constraints.

It has been shown that besides the penetration depth a new factor is essential to calculate the collision force, as a magnitude of velocity at the end of collision must be reduced by the actual factor ε.

Because the penetration velocity changes gradually during the collision (actually it is decreasing monotonously), it was an obvious idea to use it as a new factor in the old (2.1) equation. However, this factor cannot be an additive one, because the collision force must be zero right after collision process and also at the beginning (right at the first moment of impact), while the penetration velocity is usually nonzero at these times.

According to these criterions, the new modified equation (2.1) looks like:

$$F(t) = \begin{cases} -H\big(\dot{x}(t)\big)k \cdot x(t), & if\ x(t) > 0 \\ 0, & if\ x(t) \le 0 \end{cases} \tag{2.13}$$

If $\varepsilon = 1$, the $H\big(\dot{x}(t)\big)$ should be equal to constant 1, thus for convenience:

$$H\big(\dot{x}(t)\big) = 1 + G\big(\dot{x}(t)\big) \tag{2.14}$$

Simplest function for $G\big(\dot{x}(t)\big)$ is the linear one: $G(v) = d \cdot v$, which is called the damping factor. After all substitutions, system of equations (2.2) can be rewritten as follows:

$$\begin{cases} F(t) = -k \cdot x(t) \cdot \big(1 + d \cdot \dot{x}(t)\big), \\ F(t) = m\ddot{x}(t), \\ \dot{x}(0) = v_0, \\ x(0) = 0. \end{cases} \tag{2.15}$$

The ε restitution factor shows a dependency between the first moment collision velocity v_0 and the constant d. The damping coefficient d is unknown: it has to be determined from the material parameters of the colliding bodies. In the works [Eng00] and [Juh08] is proposed to use:

$$\varepsilon(d \cdot v_0, 1) \cong \frac{1}{1 + d \cdot v_0}, \tag{2.16}$$

therefore the damping factor d becomes the following forms:

$$k = \frac{1 - \varepsilon}{\varepsilon \cdot v_o}, \;\; 0 < \varepsilon < 1 \tag{2.17}$$

It can be seen that ideally $\varepsilon = 1$ leads to $d = 0$, which is non-damped case. Thereby the new introduced collision force model satisfies the listed above constraints to the contact model.

2.3.9 Collision Force Model (with friction)

The collision force model with friction introduced in this section is based on the above discussed model (2.13). The friction force is defined as a function of relative velocity in the tangential direction and is assumed to be the sum of Stribeck, Coulomb and viscous components. As mentioned in [Arm95] the Stribeck friction is characterized by a characteristic with negative slope taking place at low velocities, Coulomb friction is a constant force at any velocity and viscous friction is proportional to the relative velocity. The physical effect of breakaway friction is defined as a sum of Stribeck and Coulomb frictions. Such relations between friction force and relative velocity in tangential direction is shown schematically on Figure 2.3. Due to the discontinuous characteristic of the friction model at $v_{tan} = 0$ (Figure 2.3), and resulting computation problems, another approximated friction model is proposed. It has been proven [Arm95] that the discontinuous model is a nonphysical simplification in the sense that the mechanical contact with

distributed mass and compliance cannot exhibit an instantaneous change in force. The new friction model is shown on Figure 2.4. Both figures show relations between whole friction force ($F_{friction}$) and its components: Stribeck friction F_S; Coulomb friction F_C; viscous friction F_V; and the sum of Stribeck and Coulomb frictions – breakaway friction F_{brk}. Additionally a new parameter of threshold velocity is introduced for the approximated friction model. According to these relations, a friction force equation can look like:

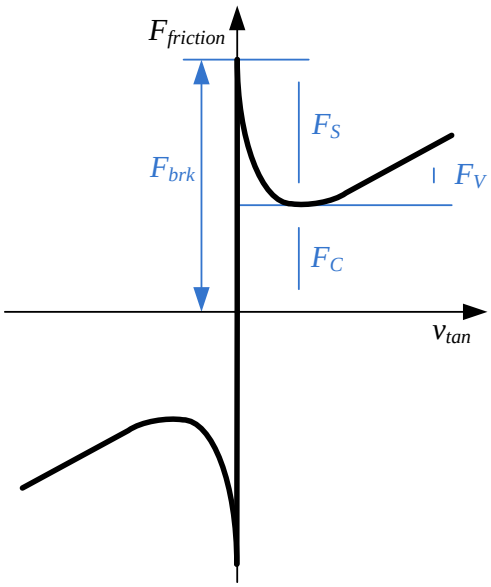

Figure 2.3 – Friction force model

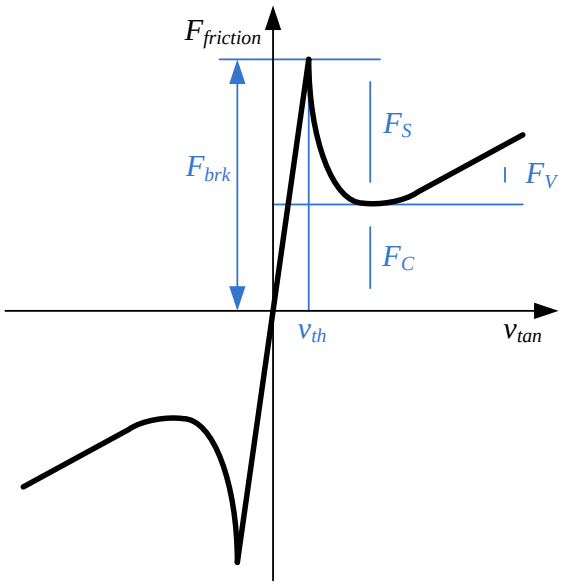

Figure 2.4 – Approximated friction force model

$$F_{friction} = (F_C + (F_{brk} - F_C) \cdot \exp(const \cdot |v_{tan}|)) \cdot sign(v_{tan}) + F_V, \quad (2.18)$$

where the value of $const$ can be free-selective.

A new proposed approximated friction model equation looks like:

$$\begin{cases} F_{friction} = const_{brk} \cdot const_{damping} \cdot v_{tan}, & if\ v_{tan} < v_{th} \\ F_{friction} = const_{damping} \cdot v_{th} \cdot (const_{brk} - 1) \cdot \exp\left(const \cdot \dfrac{v_{tan} - v_{th}}{v_{min}}\right) + \\ \quad + const_{damping} \cdot v_{tan}, & if\ v_{tan} \geq v_{th} \end{cases} \quad (2.19)$$

where the value of $const_{brk}$ represents relations between breakaway friction and Coulomb friction; $const_{damping}$ represents viscous friction coefficient; v_{min} is a small velocity value, after which the viscous friction acts.

Pure implementation of proposed friction model (2.19) and normal contact force model (2.13) require continuous solutions methods from the simulation environment. Such

requirement leads to difficulties in the realization of proposed algorithms in the scope of distributed MBS ideology. Solution of proposed models using discrete solver requires from the solver availability of variable value of sample time and can lead to a very small sample times and a long simulation process. On the other hand, big or constant sample time by discrete solver can lead to instability of simulation environment. Due to above discussed restrictions by using of discrete solver it is proposed to use the collision force model with friction which is shown on Figure 2.5.

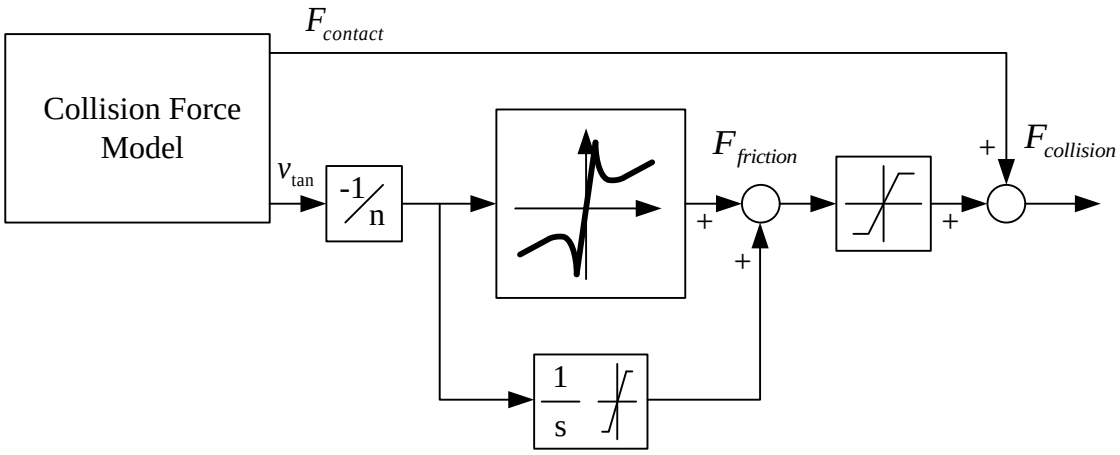

Figure 2.5 – Proposed collision force model with friction

Whole collision force algorithm consists of three parts:

- Detecting collision; calculation of normal components of penetration and velocity of penetration; calculation of normal component of contact force; calculation of tangential component of sliding velocity. The term of sliding velocity is shown on Figure 2.6. At the first moment of impact a contact point is anchored in space and a virtual connection is given to it. During the collision the distance between the anchor and the second contact point is chanced. Thereby the derivative of this distance is not another as a sliding velocity.
- Calculation of non-linear friction force; calculation of additional friction force for prevention of sliding velocity; limitation of friction force.
- Final summary of normal component of collision force and tangential one, which is represented as friction force.

The collision force model is based on (2.13) and can be shortly rewritten as:

$$F_{contact} = \begin{cases} \left(1 + \frac{1-\varepsilon}{\varepsilon \cdot v_{impact}} \cdot v_{norm}\right) \cdot const_{contact} \cdot p, & if\ p > 0 \\ 0, & if\ p \leq 0 \end{cases} \tag{2.20}$$

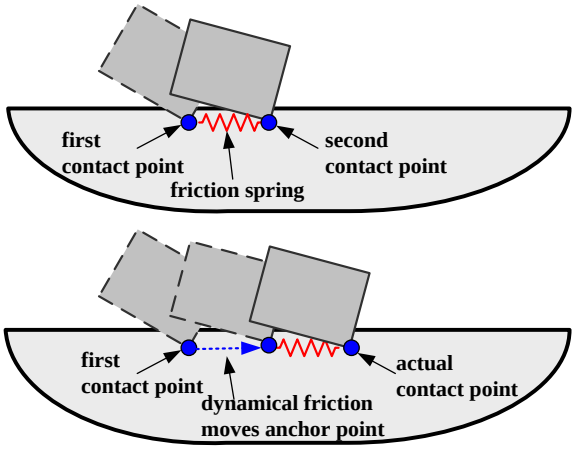

Figure 2.6 – Proposed friction model [Kon071][Kon07]

where ε is some restitution factor conditioned on the losses by the impact, can be $0 < \varepsilon \leq 1$; v_{impact} represents the relative velocity by impact moment; v_{norm} is a penetration velocity during the contact time, theoretical must be $|v_{norm}| \leq v_{impact}$; p is a penetration depth; and $const_{contact}$ is a spring stiffness factor due to material properties of bodies.

Tangential component of sliding velocity, v_{tan}, is obtained using the vector of penetration velocity and the relative vector of whole velocity of bodies.

The non-linear friction force component is based on (2.19) but has some scale down factor $1/n$ for the tangential component of sliding velocity. Such assumption is assumed due to the instability of the friction model by using the discrete solver. The scale down component must be selected in such way, that the friction model doesn't lead to any instabilities of the whole simulation model. To improve accuracy of the friction model a second loop is proposed, which has integral properties. The integration properties of the second loop should be selected in a way, that the friction model satisfies real requirements and the whole simulation model doesn't lead to any instabilities during work. The friction force equations can be presented as:

$$F_{friction} = \begin{cases} non\_const_{friction} \cdot v_{tan}/n, & if\ F_{friction} < \mu \cdot F_{contact} \\ \mu \cdot F_{conact}, & if\ F_{friction} \geq \mu \cdot F_{contact} \end{cases} \quad (2.21)$$

where $non\_const_{friction}$ is a non-linear coefficient and is based on Figure 2.4; μ represents the relations between the normal collision force and the friction force, it is specified by the properties of materials of bodies.

The main advantages of the proposed approach are the simplicity and the possibility of using it for stiff and soft contacts. This approach works reasonably well if several contact points are also present at the same instant time. The disadvantage of this approach is that the integrator step size should be reduced in the contact phase in order to catch the rapidly changing contact forces and torques. And similar to the impulse-based method it is necessary to choose the contact parameters (spring, restitution) because the contact force is proportional not only to the penetration depth/velocity but also to the contact area and the contact volume.

2.3.10 Update of System Dynamics

Update of system dynamics takes place each integration step depending on the chosen contact response method.

The update of system dynamics by impulse-based approach occurs according to the follow two methods: *Propagation* und *Simultaneous* update.

- *Propagation Update* says that the impulse is calculated and applied only for one contact per time, i.e. the collision events are treated individually.
- *Simultaneous Update* says that the impulses are calculated and applied for all contacts at the same time, i.e. the collision events are treated after an integration step together.

These two models are identical for the contact situations with only one contact point. However they produce different results for the contact situations with several contact points. Figure 2.7 shows the results for the simulation of three identical billiards balls without loss of kinetic energy.

The update of system dynamics by force-based approach takes place at every integration step depending on the active contact forces. The calculation of the contact forces is often determined by an optimization algorithm. Besides, its aim is to minimize the function

$$func(F_{conact}) = \sum_{i=1}^{N} F_{conact}^{i} \tag{2.22}$$

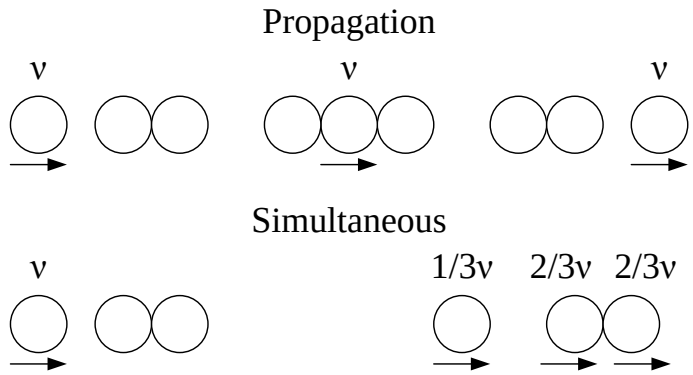

Figure 2.7 – „Propagation" vs. „simultaneous" update of the system dynamics [Kon071][Kon07]

This classical optimisation task from the area of the linear programming is called Linear Complementarity Problem (LCP) and is solved with the help of different algorithms, e.g. Lemke's algorithm or Unique Sinking Orientations. If this optimisation task is to be solved in a multi-body simulator, the appropriate optimisation methods are called several times between two integration steps (Figure 2.1).

2.4 Contact Processing and MBS-Simulation

Section 2.2 deals with possibilities of simulation of multi-body systems. Such tasks as simulation of electrical components (electrical motors), simulation of mechanical components (solution of system of dynamical Lagrange equations), simulation of embedded system (control algorithms, control of electrical drives), simulation of sensor system (sampling of the signals, signal noise) and etc. have been discussed. In section 2.3 some aspects of contact processing are proposed. This section deals with the integration of contact processing algorithms into the MBS simulation. A number of different interfaces for such implementation will be discussed. Figure 2.8 shows principle flow schema of MBS simulation with contact processing. At same time it presents some different interfaces to communicate between separate parts of the whole simulation.

Interface 1 allows single transfer of dynamical and kinematical parameters of construction from CAD. This makes the task of assignment of mechanical system in the MBS-simulation software (Matlab, Dymola etc.) more easy. This helps to define the mechanical properties of bodies and materials such as rotational and translational moments of inertia, parameters of soft materials (stiffness, restitutions factors etc.), number of degrees of freedom and their location etc.

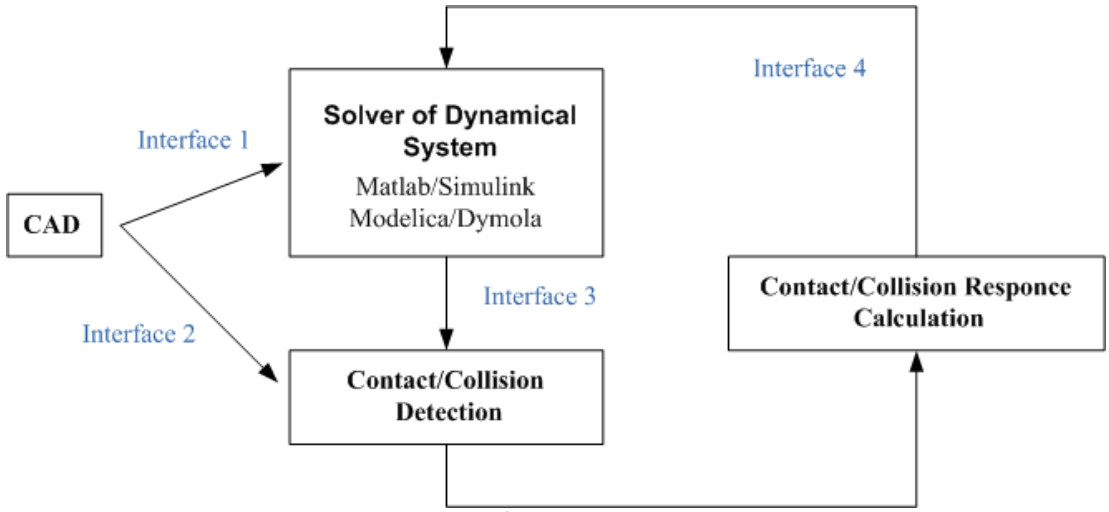

Figure 2.8 – Principle schema of MBS-simulation with contact processing

Interface 2 allows single transfer of contact properties (stiffness, restitutions factors, location of center of mass of single bodies) of possible contacted bodies from CAD to contact processing, definition of geometrical primitives from complex mechanical constructions to allow fast collision detection.

Interface 3 plays the role of communication interface between MBS simulation and contact processing. It transfers positions, velocities of single bodies in the simulation world of MBS simulator to contact processing.

Interface 4 transfers calculated values of contact forces and torques back to the MBS simulator.

2.5 Experiments

This section deals with the experimental part of complete MBS simulator with integrated contact processing. Two different experimental scenes will be presented. The first one shows the interaction between different geometrical objects with each other. The second one compares simulation results during walking of the six-legged robot "ANTON" and real force sensor information.

2.5.1 Experiments: Different Geometrical Objects

Functionality and stability of proposed MBS simulation environment will be proved firstly on the example of interaction between different geometrical objects in dynamical world. The first experimental scene consists of such objects:

- Surfaces, defined as triangles network.
- Sphere in analytical form.
- Sphere, defined as triangles network.
- Box, defined as triangles network.
- Capsule in analytical form.
- Capsule, defined as triangles network.

Experimental scene is shown on Figure 2.9. Initial conditions of objects can be different from each other, this means, that different objects can have different initial location, initial velocity and acceleration as well. The sample time of discrete solver in MBS simulator is equal 0.1ms. Results of contact processing are shown on Figure 2.10 – Figure 2.14.

Figure 2.9 – First experimental scene with different geometrical objects

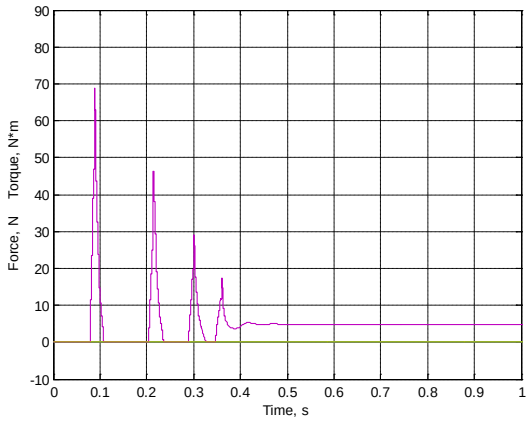

Figure 2.10 – Contact forces and torques between surface and sphere in analytical form (magenta is the vertical reaction, opposite to gravity vector)

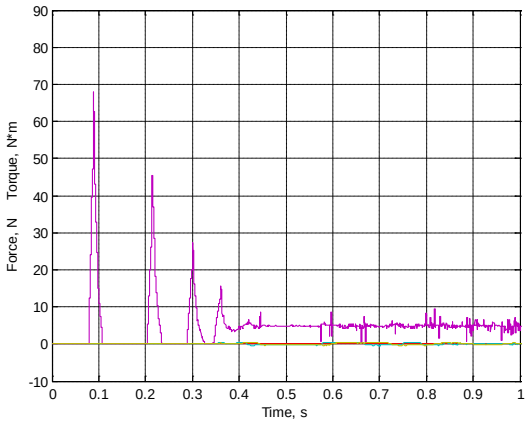

Figure 2.11 – Contact forces and torques between surface and sphere defined as triangles network (magenta is the vertical reaction, opposite to gravity vector)

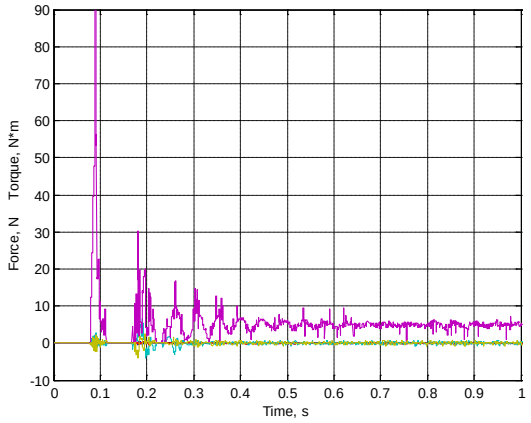

Figure 2.12 – Contact forces and torques between surface and box (magenta is the vertical reaction, opposite to gravity vector)

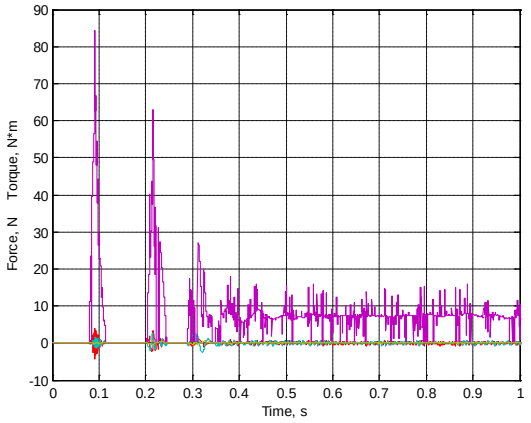

Figure 2.13 – Contact forces and torques between surface and capsule defined as triangles network (magenta is the vertical reaction, opposite to gravity vector)

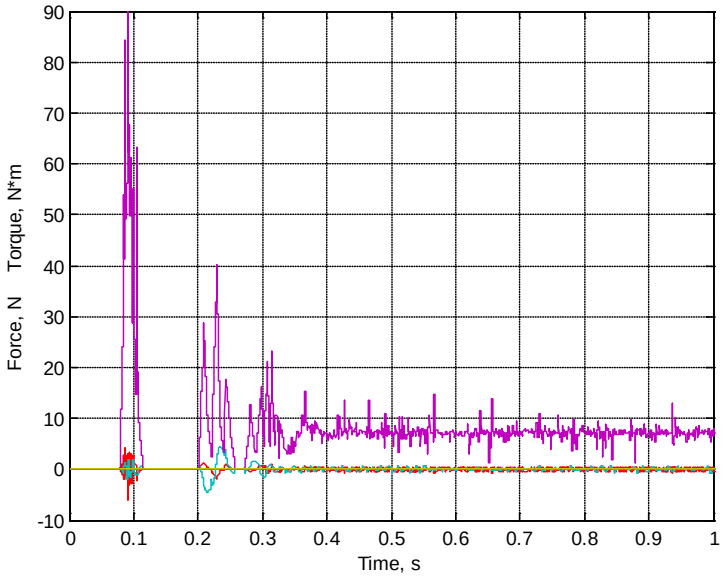

Figure 2.14 – Contact forces and torques between surface and capsule defined in analytical form (magenta is the vertical reaction, opposite to gravity vector)

Simulation of interaction between surface and sphere defined in analytical form shows better results as other ones. In this case one can find the contact process more stable due to constant steady-state process. In other cases the resulting contact forces have some oscillations, but these oscillations don't bring any instability into the simulation process. In relation to the simulation time, the first one shows better results too. These two aspects mean that it is better to present different complex mechanical bodies as a number of simple primitives, notably analytical one.

2.5.2 Experiments: Walking Robot "ANTON"

The construction and properties of the six-legged walking robot "ANTON" are already presented. This section shows comparison between simulation results during some walking and real one obtained from the force sensors placed in the robot feet. The robot had to move forward on an even horizontal plane in tripod gait mode (3 legs support the body in a triangle arrangement, while 3 others could swing freely in the air) with a constant reference longitudinal translation speed of 0.1 m/s. The physical data from the force sensors placed in the front left, middle right and rear left feet are collected. Although the data of the onboard sensors have to be interpreted natively in the coordinate system of their respective feet, the control algorithm of "ANTON" assures that the legs are always vertical in global world space, thus the sensor data can be directly interpreted in the Cartesian world coordinate system.

The simulation results and the real force sensor information are shown on Figure 2.15.

It can be observed that the weight of robot is balanced in a way that the front and middle legs carry a little bit more load that the rear ones, because the head of "ANTON" has two stereo cameras. By the first step one can detect a different force distribution in respect to the following ones. This can be explained due to the different gait pattern by the beginning of walking and during the steady-state walking. Comparing the simulation results with the real ones, it can be noticed a very good match to the reality of proposed MBS simulator with contact processing.

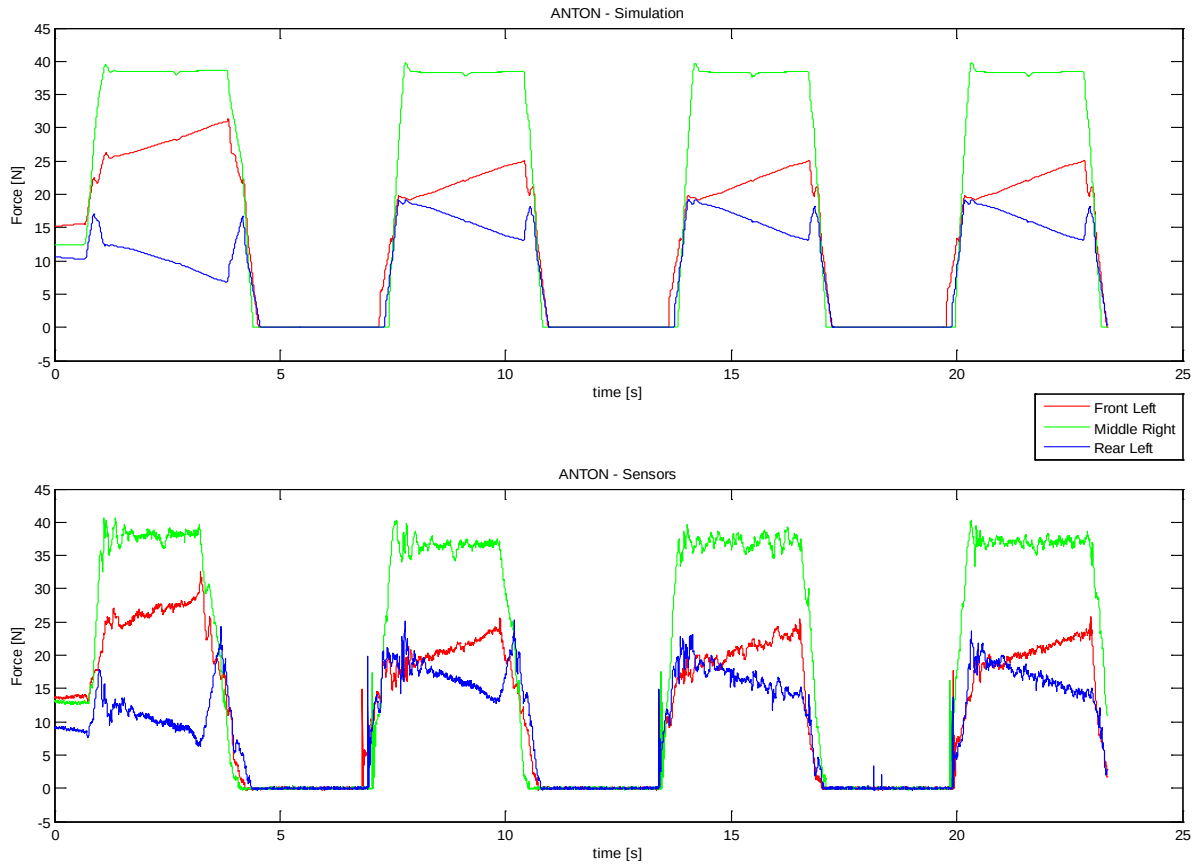

Figure 2.15 – Simulated and real contact forces during the walking of robot "ANTON"

2.6 Conclusion

The force based contact processing method with penalty formulation, represented in Chapter 2, has been implemented according to Figure 2.1 into the multi-domain simulation environments Matlab/Simulink and Modelica/Dymola by means of Solid as collision detection software and has been proved by stability criteria on different experiments, as well as has been compared with the real experimental results. The collision forces are calculated using virtual spring and damper elements acting during a contact both in normal and in tangential directions, considering also the restitution of the colliding materials. The restitution property of the material is more tangible than the impact velocity-dependent damping factor that was used in other works. The six-legged robot "ANTON" has been used as a real experimental object. Two modes, staying on the ground and walking on the surface, have been investigated and compared in the mentioned simulation environments (see Figure 2.15).

The developed contact processing environments in Matlab/Simulink and Modelica/Dymola are free from aforementioned disadvantages. It can use integration methods of higher order, and is sufficiently precise by calculation of the contact forces, because (2.20) and (2.21) have been laid out under consideration of the energy/impulse preservation law.

Chapter 3

Six-Legged Walking Robot "ANTON"

Today, many questions about multilegged walking robots (the choice of construction, the design of control system, as well as locomotion organization) are known and well investigated. These investigations consider usually multilegged robot locomotion on smooth or easy rough terrain, overcoming simple obstacles, movement on soft ground, body maneuvering etc. From the algorithmically point of view this class of operations can be realized on robot kinematical level just in automatically mode by means of periodic gaits and contact information in discrete form (yes/no). From the mechatronical point of view this class of operations can realize robots with rigid bodies and simple contact sensors.

Recently investigations of improved robot tasks are concerned with the multilegged robot locomotion over an impassable road or a strongly complex terrain such as earthquake affected area, mountain regions, high ledges, ditches, trenches. The key performance for such tasks is the *additional body maneuvering*, the *measurement and control of support reactions* as well as the *control and forecasting of the robot motion stability* [Son89][Okh84][Gor90]. From the mechatronical point of view it is connected with enhancements in the robot construction and using the interaction force sensors. From the algorithmically point of view it is connected with the software development for the

climbing tasks (with a possibility of using both legs and body for support during the separate stages of movement).

A number of the problems devoted to climbing tasks have been shown in publications [Pfe00][Gol03] considering robot movement inside or outside a pipeline. For this task the robot has to press itself against the inner or outer surface of the pipe and to move, so that support reactions were inside cones of friction. In [Son89] are considered in details the problems of climbing up/down a large obstacle with the walking robot with rigid body. However, the functional capabilities of the walking robot on overcoming the obstacles can be essentially expanded by change of the body design. The problems of overcoming of obstacles equal to the geometrical size of the robot are not covered in the literature except the works [Ale98], which simulate the climbing of six-legged robot with an articulated body on a high ledge by means of a change in the body configuration during the separate phases of climbing. In [Bra99] the design of such walking robot is considered and its mathematical modeling is presented. It is shown, that size of the obstacles, which the robot can overcome, increases in case of the body having controlled segments.

3.1 Introduction

This section deals with development aspects and control of six-legged walking robot "ANTON". In the last 7 years the numbers of legged robots (see Figure 3.1) were developed in the RobotsLAB Team. Their mechanical structure and control system were improved. These improvements touch such question as the mechanical construction and the locomotion control system as well as the embedded control system.

Recently investigations of improved robot tasks are concerned with multilegged robot locomotion over an impassable road or a strongly complex terrain such as earthquake affected area, mountain regions, high ledges, ditches, trenches. The key performance for such tasks is the high body maneuvering, the measurement and control of support reactions, the control and forecasting of the robot motion stability as well as the availability of information about obstacles. From the mechatronical point of view it is connected with enhancements in the robot construction, with using sensors of "outer information" (interaction force, near navigation, cameras). From the algorithmical point of view it is connected with software development for the adaptive interaction with envi-

ronment, for the robot stability control and for the free gait (not mentioned in this article).

Figure 3.1 – Developed robots constructions in the period 2003-2007 year ("SLAIR1", "SLAIR2" and "ANTON")

The control system of the full or part of autonomous legged robot is almost controlled by embedded system. Commonly the embedded systems are designed to control complex plants such as engines, satellites, vehicles, spacecrafts, and of course CLAWAR. They generally require a high level of complexity within the embedded system to manage the complexity of the plant under control.

Development and test of complex real-time embedded systems consists of many steps from modeling and simulation of the plant till the implementation of the source code in the real hardware. Hybrid techniques, that are used increasingly, are the Software-in-the-Loop (SiL) and the Rapid Control Prototyping (RCP).

Thus the technical task is development of a robot with additional controlled DOF (degrees of freedom) in the body as well as with the possibility of the measurement and control of support reactions, as well as development of such control system that allows simple and rapid development of the complex control algorithms. The algorithmical purpose of this work is the development of control algorithms for the desired mechanical interaction with environment. Moreover the flexible communication bridge for real-time communication between the virtual/real control system and the virtual/real robot is presented as well.

3.2 Design Philosophy of Walking Robot "ANTON"

In accordance with the requirements discussed above a multilegged walking robot with articulated body "ANTON" (see Figure 3.2) has been developed. The robot mechanics, sensor system and control system guaranty an additional flexibility in the body, to meas-

ure and control the support reactions as well as to control and forecast the robot motion stability. The real-time communication bridge based on industrial Ethernet protocol EtherCAT was developed. It guarantees the communication between robot onboard electronic and robot control system with sample time 1ms.

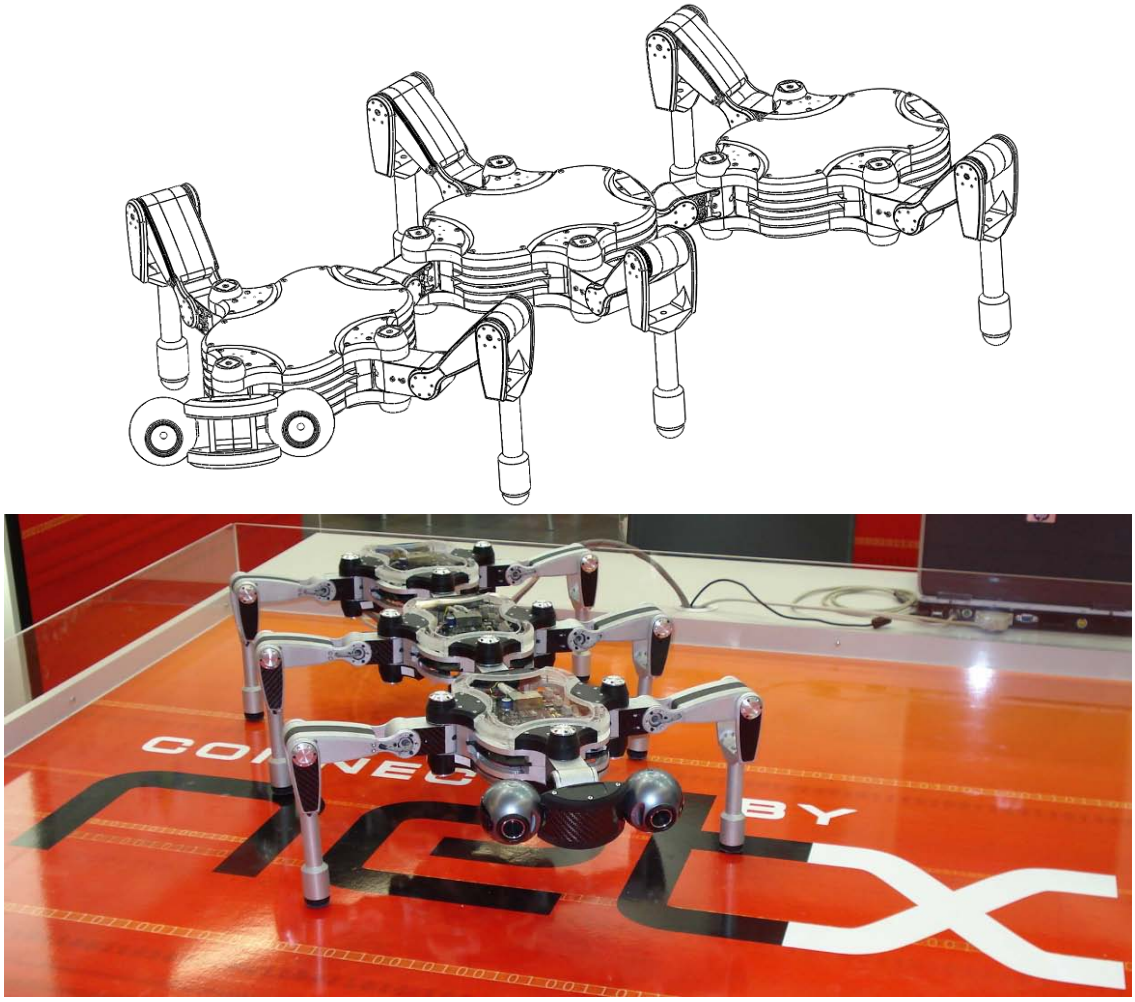

Figure 3.2 – Six-legged walking robot "ANOT" (CAD and real photo)

The robot construction has three modular segments linked to each other through two DOF joints and 6 legs. Each shoulder includes one articulated body segment linked with two 3-DOF-insectomorphic legs. The robot drives are servomotors, with maximal power 2.8W in knee and 8.68W in other joints. The gear number is 390 in knee and 251 in other joints. The sensor system of the robot consists of components that are standard for mobile robots and that make it possible to achieve autonomous robot functions in an environment. It includes:

- 24 potentiometers and IGRs (installed in each robot joint),

- 6 three-component force sensors (mounted in each leg's shank),
- 2-axis gyroscopic sensor (located in body) and
- 2 mono cameras in the head of robot.

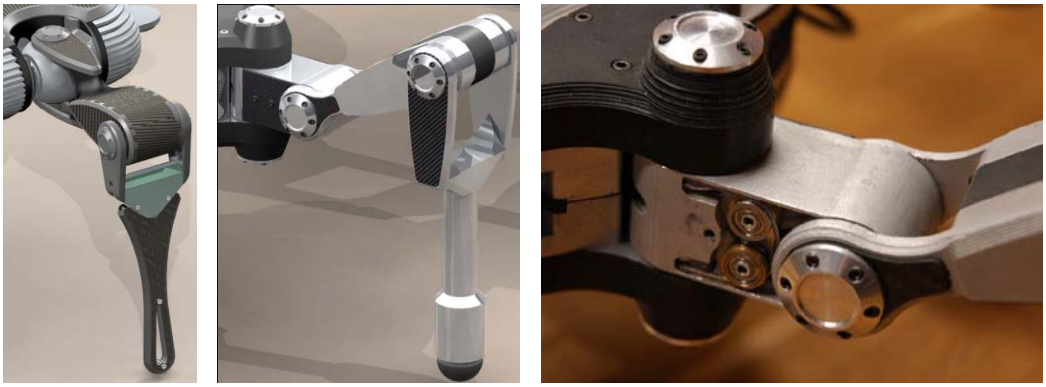

Figure 3.3 – Constructions of legs and the shoulder's differential joints (SLAIR2, ANTON, real differential joint on robot "ANTON")

It should be pointed out, that the construction of the shoulder's differential joint (Figure 3.3) brings the double rotation moment on the two rotation axis and realizes the required two degrees of freedom in the shoulder. The workspace of new differential joint (Figure 3.3, center, right) was improved due to the vertical placement of the driving motors and the separation of the rotation axis.

In accordance with requirement on measurement and control of support reactions the developed force sensor (Figure 3.4) consists of two parts of the core measuring lateral components of support reaction, and the elastic parallelogram module for measurement of the longitudinal component.

In the compact sensor unit the force-measuring elastic plates are located in plane parallel to the action of the longitudinal force. The module is mechanically selective to the action of twisting moment; the strains caused by lateral components of a supporting reaction in the plates have identical sings.

Figure 3.4 – Three-component force sensor

The sensing units are made of metal (duralumin), are simple to manufacture, and joined together with a threaded coupling. Three-component sensor with the amplifier is mounted in the shank of a leg, and the bottom end of the sensor is connected to the foot. The sensor is designed for loadings up to 50N; interference between channels does not exceed 1%.

Additionally the foot design should provide good friction with a support along all directions of action of support reaction. Therefore for increase of friction it's expedient to make a foot from rubber. The spherical form of foot is more preferable at absence the gimbal-lock between a leg and foot.

3.3 Hardware Control System

Control system with real-time decentralized data gathering and processing builds the kernel of robot system and makes possible development of control algorithms with help of hybrid simulation. The first version of control system has a simple architecture based on DSP and RS485 communication protocol. The next one has been improved and based on WindowsPC-host system with Ethernet communication [Pal07]. The last control system has been already reached the industry level and based on the industry real-time communication EtherCAT. The developed hardware (see Figure 3.5) consists of NetX communication processor from Hilscher and FPGA.

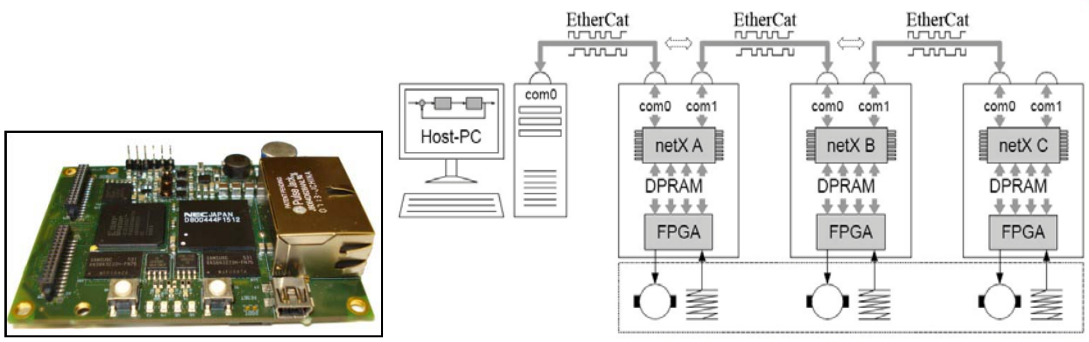

Figure 3.5 – Flexible communication system based on netX processors [Kon08]

The communication bridge emulates a router, which connects the control system with all the components of the control object (i.e. legged robot). These components can be connected to the bridge via diverse bus systems (Ethernet, CAN, UART, SPI etc). The robot components in such system can also be represented in virtual environment. Through a common physical interface is the virtual or real control system isolated from the object, and therefore the development effort of communication interface to the legged robot is reduced. The output signals from the control system (reference signals) are demultiplexed and sent to corresponding actuators with the help of the communication bridge. On the other side the outputs from the robot (actual sensor signals) are multiplexed by bridge and sent to the control system. These routing tasks are entirely performed by the communication bridges. The communication is implemented in full duplex mode, whereby simultaneous sending and receiving of the packets are possible. The task of the host PC is in the periodic sending of request and the collecting of all response packages in the same communication cycle with all netX units (see Figure 3.7).

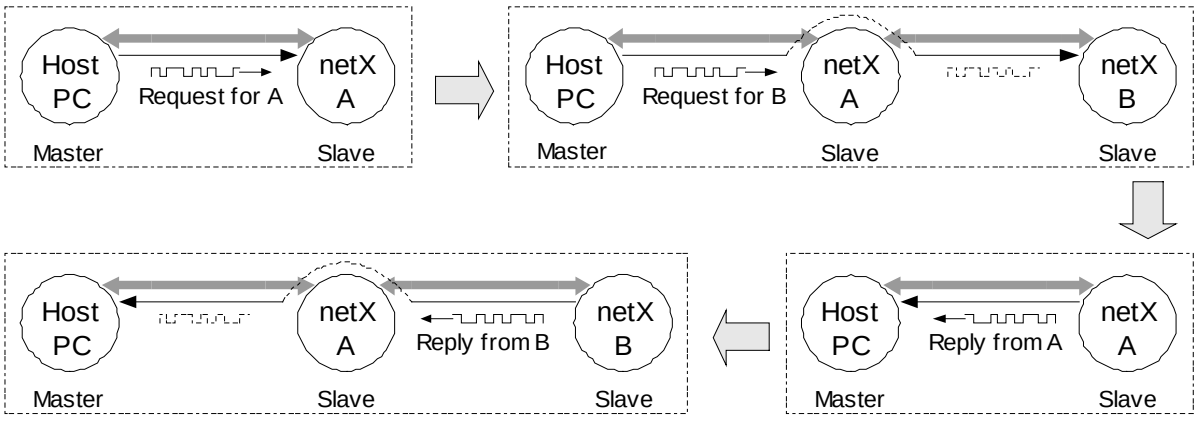

Figure 3.6 – Data flow within communication cycle between Host-PC and netX chain [Pal07]

Figure 3.7 shows the time diagram of communication process. The control system (Control) on Host-PC (xPC Target) is computed each sample time (Tsample) by the infor-

mation received from the sensors (RD – Receive Data). Computed reference signals of PWM (SD – Send Data) are copied to the EtherCAT packet and the EtherCAT master (CIFx-50) to initialize a telegram (Packet n) transfer. The received data by the EtherCAT slaves (NetX 500) are copied through the DPM (Dual Port Memory) to the FPGA and then to the PWM module.

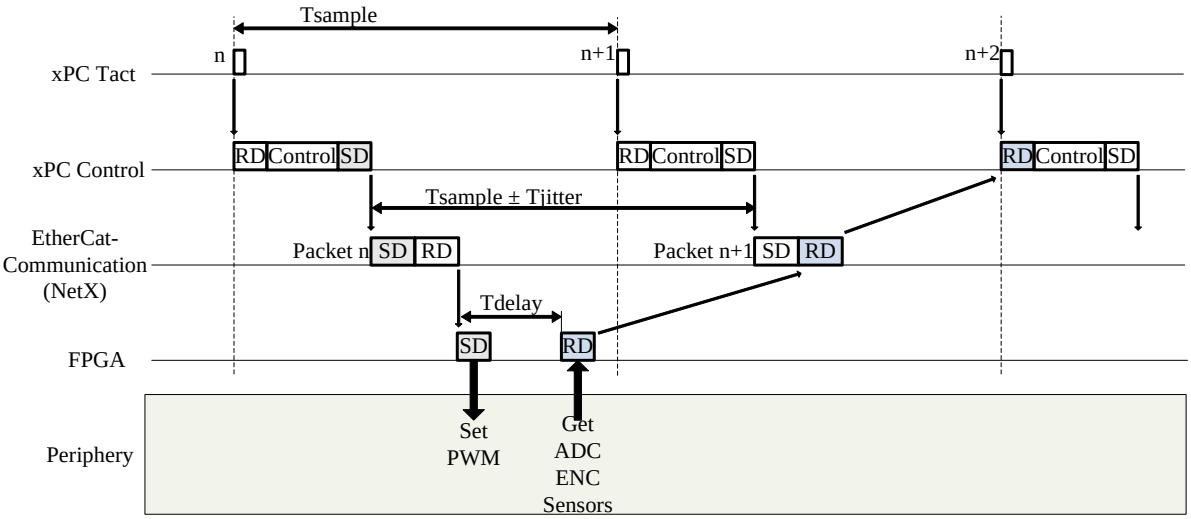

Figure 3.7 – Time diagram of communication process [Kon081]

The feature of the EtherCAT communication protocol is the exchange of the data in the loop Master-Slave-Master with one solid telegram. Therefore sending of the response information by the EtherCAT slave is required preliminary preparation of the transfer buffer. The delay $T_{delay} = T_{sample}/2$ between the receipt of the telegram by EtherCAT slave and the monitoring of the sensors is realized. This was done for the minimization of the time difference between monitoring of actual sensors information (ADC, ENC and etc.) and the instant of usage them in the control calculation (n+2 xPC Tact). Received information from sensors (RD) is saved to the transfer buffer of the EtherCAT slave. After that it will be sent in the next closest transfer packet (Packet n+1) and will be actual on the n+2 work period.

The standard PC is used as a control computer (Host PC). Therefore the jitter time (Tjitter) in the communication system due to large number of hardware peripheries is occurred. For the minimization of jitter time the practically all of hardware peripheries are tuned off in the BIOS of PC. The maximal jitter time in our communication system is less than $15\mu s$.

Such communication system allows using of important technique in the robotic area as hybrid simulation technique. Starting with the SiL simulation, where the developed control system in Matlab/Simulink has the possibility to communicate with the real legged robot, the control system has acquired a new level so-called Rapid Control Prototyping (see Figure 3.8).

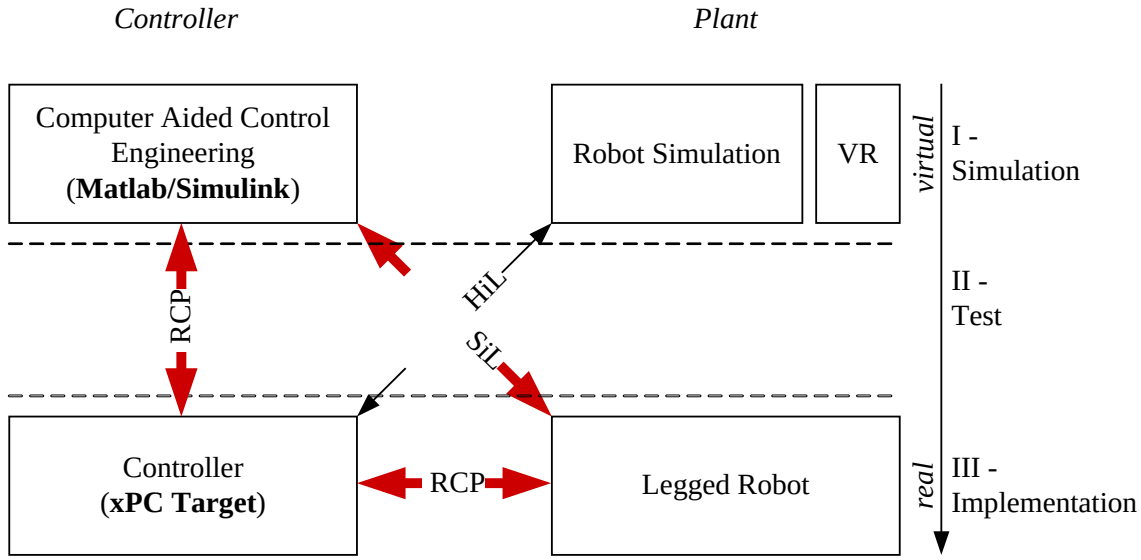

Figure 3.8 – Common scheme of the SiL/RCP structure of the mechatronical system [Kon08]

3.4 Software Control System

The hierarchically organized modular control structure (Figure 3.9 – Figure 3.11) is completely located on PC-side that implements additionally the interaction with user and produces the control signal for robot drives as well as monitors all actuators and sensors of the robot. The robot-side is implemented by fast and flexible FPGA, includes the hardware abstraction layer (HAL) for drive and sensors. The real-time connection between two parts is made via proposed netX® based communication system, described below. All this provides flexibility and simplifies the development of control algorithms. The control system can also be extended for the additional shoulders in exactly the same manner as the mechanical structure.

The PC-side hierarchical control system has a three control levels: Action, Primitive and Servo levels. Each of them is for own task responsible. The action level represents the level of references: references for locomotion tasks and references for manipulation tasks. The primitive level ensures a regular walking pattern in accordance with parameters which comes from the action level. This level can be divided in four parts: step cycle

generation, compliance control, COM stabilization and force and position feedback. The last level, servo level, is for the realization of the position control of servo DC drives responsible. The whole control process is sampled with the sample time of T_{sample}=1 ms.

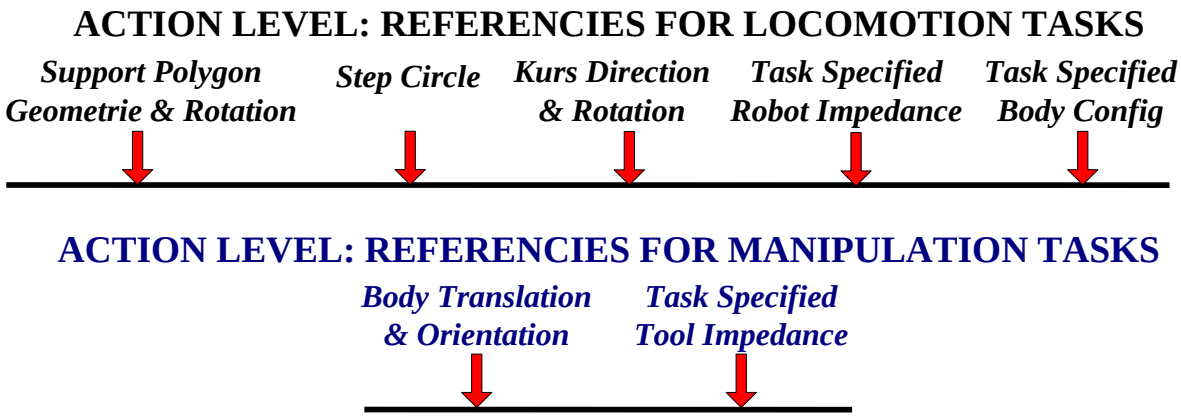

Figure 3.9 – The hierarchically organized robot control system: action level [Kon072]

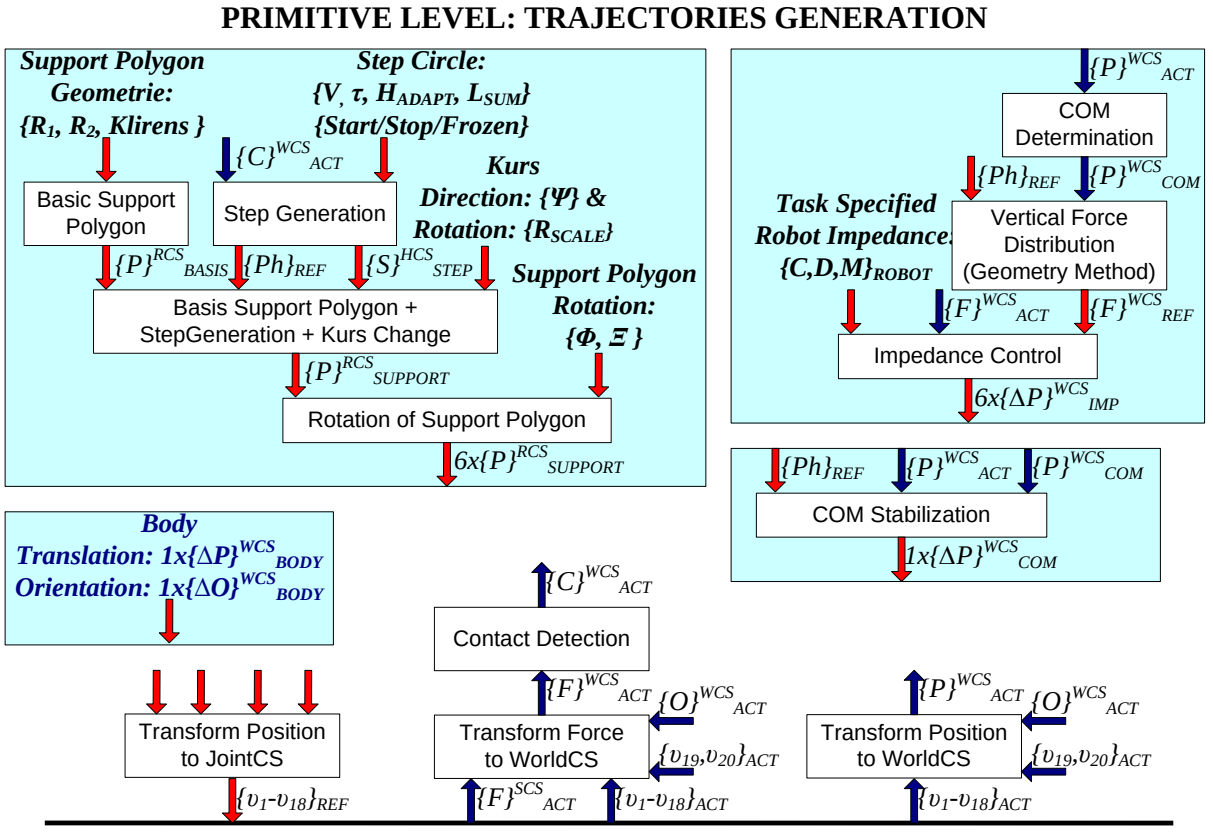

Figure 3.10 – The hierarchically organized robot control system: primitive level [Kon072]

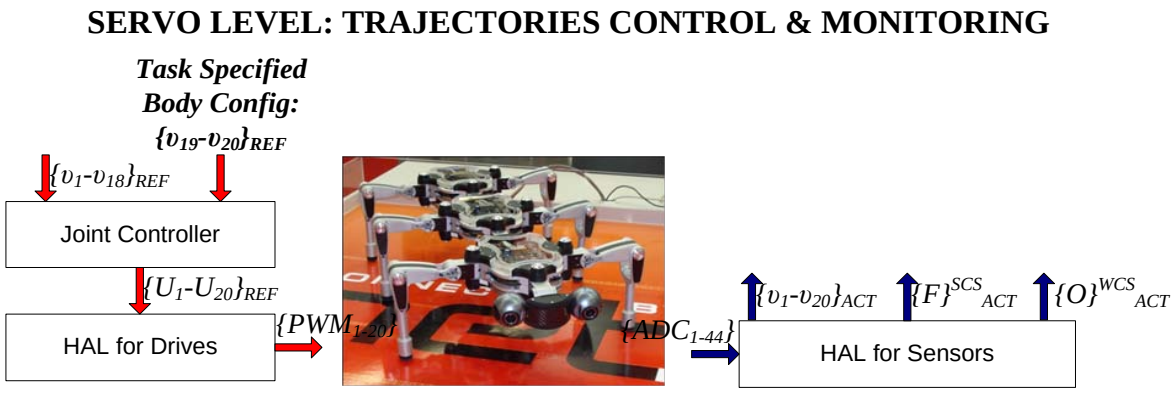

Figure 3.11 – The hierarchically organized robot control system: servo level [Kon072]

3.4.1 Locomotion: Reactive Gaits Algorithms

Legs play two fundamental roles in a walking robot: sustaining the body and propelling the robot. For statically stable walking, sustaining the body implies having, at any time, at least three legs on ground forming a triangle containing the vertical projection of the centre of gravity of the robot. Propelling of the robot is achieved by coordinately moving each supporting leg with respect to the body between two positions denoted the anterior extreme position (AEP), and the posterior extreme position (PEP).

The standard periodic wave gaits (Figure 3.12), such as wave, gallop, and tripod, are used for the task "sustaining the body" and "propelling of the robot". Numeration of the robot's legs is shown on the Figure 3.13.

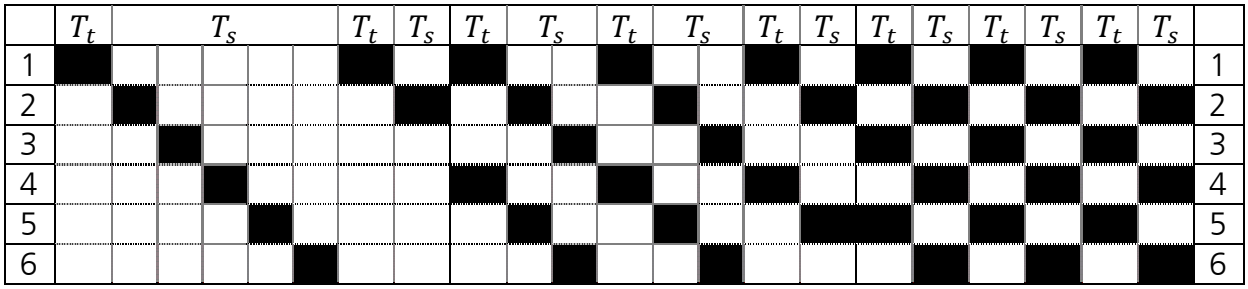

Figure 3.12 – The three main periodic wave gaits [Rus07]

A $\tau = {}^{T_t}/_{T_p}$ is a so-called duty-factor of the gait art, where T_t is a time of transfer phase,

T_s s – a time of stand phase and $T_p = T_t + T_s$ is a whole gait period time.

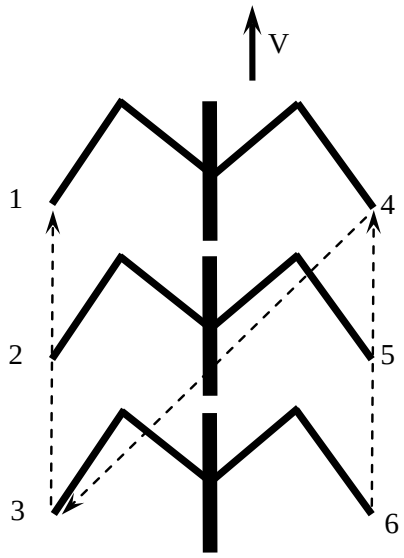

Figure 3.13 – Numeration of the robot's legs [Rus07]

Gait generation system must be able to adapt to different kinds of perturbations in the walking process. Thus the wave gaits can be modified by reactive controller in accordance to the following stability and sequencing rules.

Stability Rule: A leg must remain in the support phase whenever one (or both) of its neighbors is in the air.

Sequencing Rule: A leg must remain in the support phase whenever one (or both) of its neighbors is close to reach, or has reached before, it's PEP than itself.

The combination of the standard wave gaits with the reactive controller gives the developer the simplicity with the gait generation system, and gives the robot good stability during possible disturbances in the walking process.

A number of experimental results are shown on Figure 3.14 - Figure 3.17. Figure 3.14 shows the change of transfer and stand phases for each leg of robot during the start of walking, then continuous walking (after about 2 second) and stop (after about three steps) of it. All three types of wave gaits, notably wave gait, gallop gait and tripod one, are shown on Figure 3.14 too. Figure 3.15 - Figure 3.17 show change of transfer and stand phases of all six legs during changing of periodical gate from one to another, for example Figure 3.15 shows change of periodical gate from wave to gallop one and backwards.

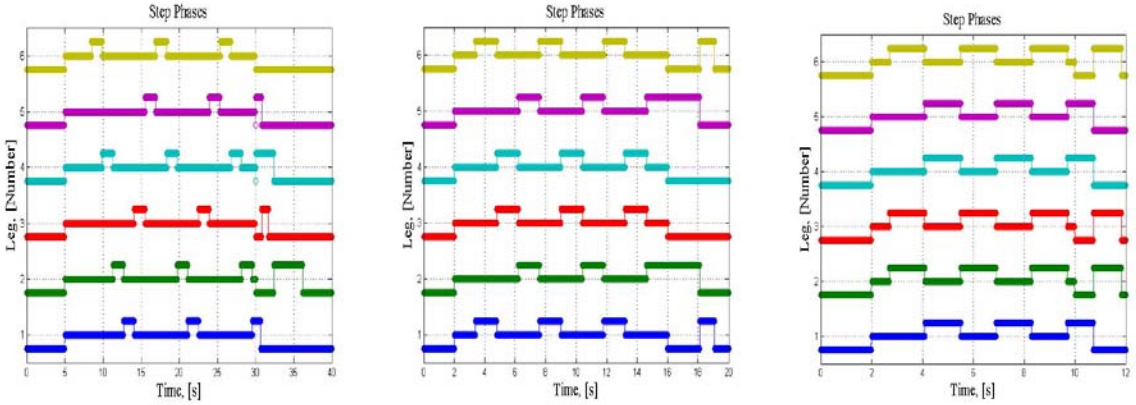

Figure 3.14 – Change of transfer/stand phases of all six legs during the start of walking, walking and stop of walking by the wave, gallop and tripod periodical gates of the robot "ANTON"

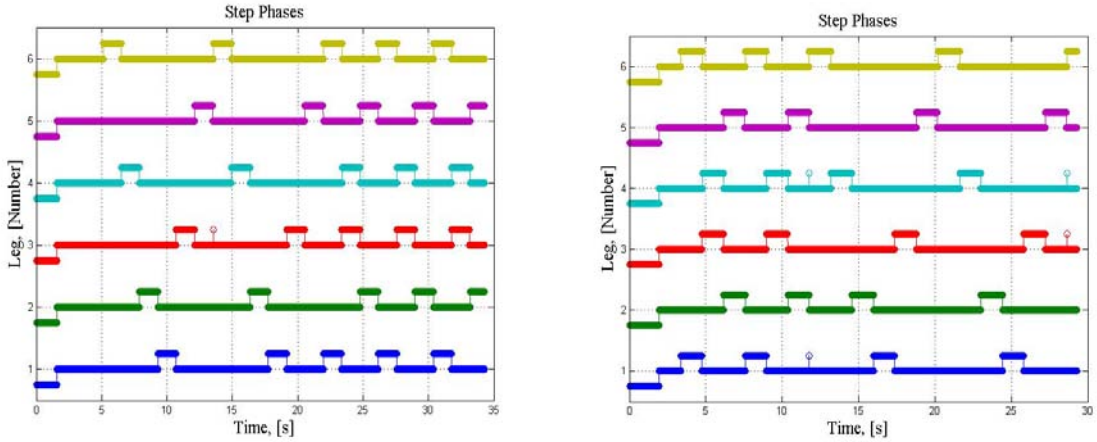

Figure 3.15 – Change of transfer/stand phases of all six legs during changing of periodical gate from wave gate to gallop one and backwards

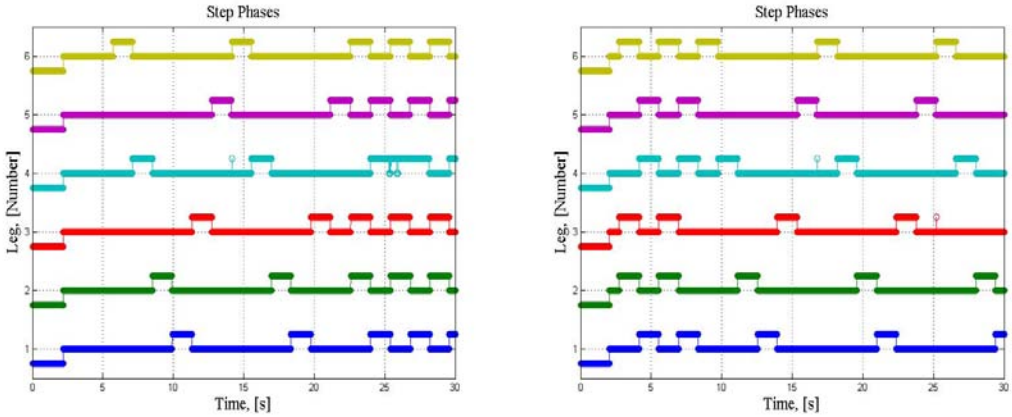

Figure 3.16 – Change of transfer/stand phases of all six legs during changing of periodical gate from wave gate to tripod one and backwards

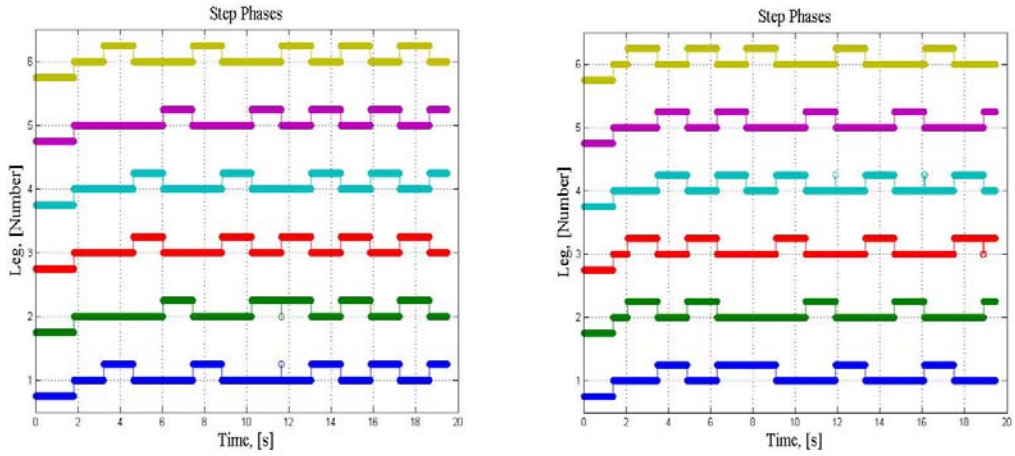

Figure 3.17 – Change of transfer/stand phases of all six legs during changing of periodical gate from gallop gate to tripod one and backwards

Figure 3.18 shows change of feet coordinates in direction of robot walking during changing of periodical gate from wave gate to tripod one. It is obvious that the stability rule hold some legs in the stand phase a little longer that another one to provide the stability. The length of the step in this experiment was 7 cm.

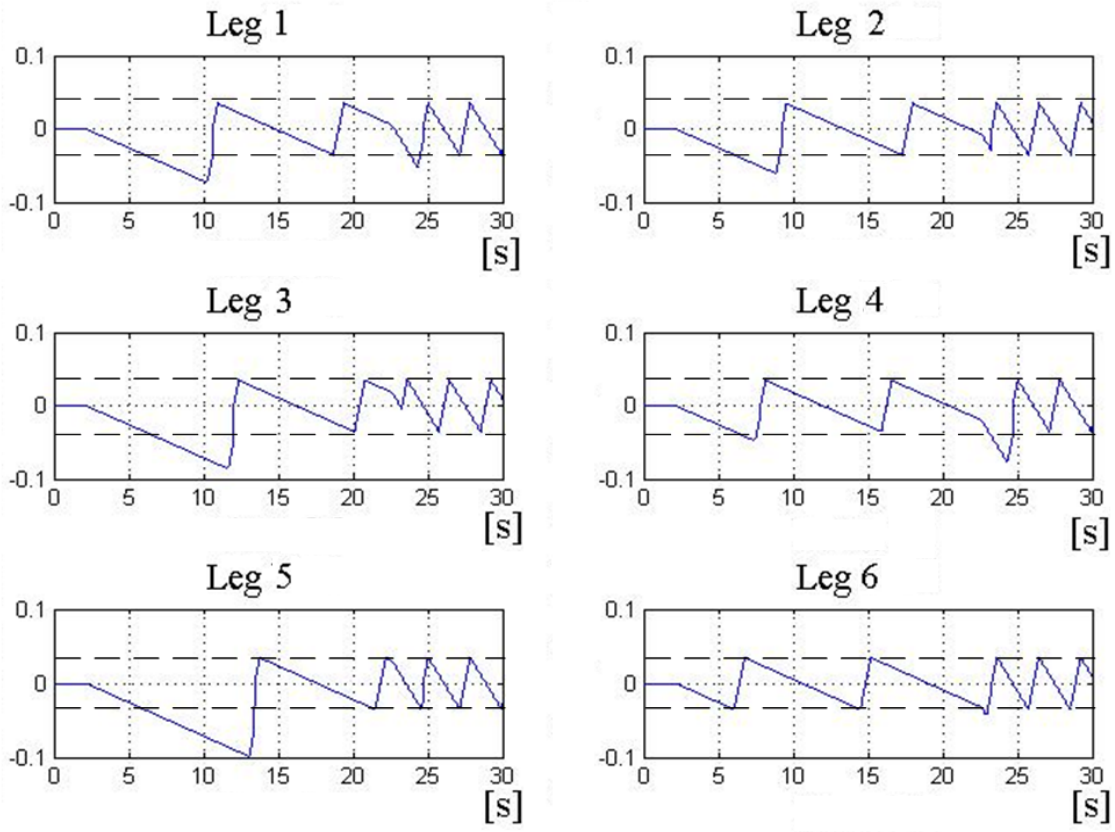

Figure 3.18 – Change of feet coordinates in direction of robot walking during changing of periodical gate from wave gate to tripod one

3.4.2 Locomotion: Step Cycle Geometry

Object of a geometrical generation of step cycle:

- Generate the step cycle trajectory for each foot in a Gait Coordinate System (GaitCS).
- Realize the adaptive algorithm of motion.
- Synchronize all foots.

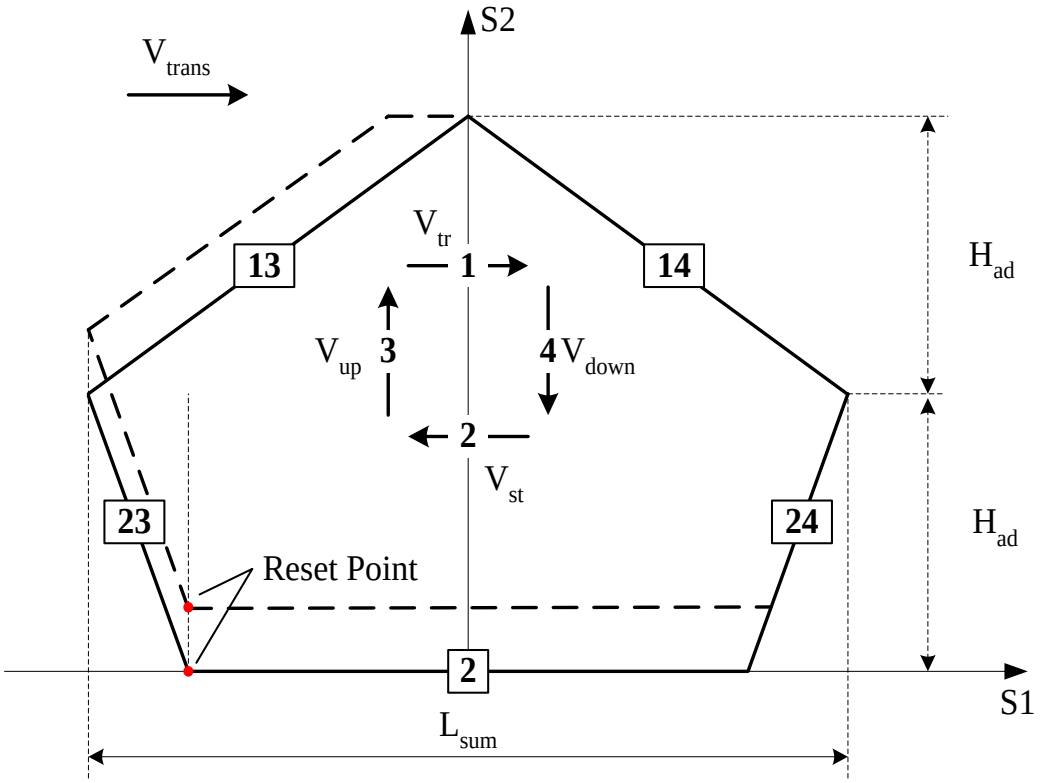

Figure 3.19 – Geometry structure of the step cycle

The step cycle is shown on Figure 3.19 and its disposition relative to robot coordinate system is shown on Figure 3.21. Step cycle has four different motion's phases:

1 transfer phase (horizontal motion's phase);
2 stand phase (horizontal motion's phase);
3 up phase (vertical motion's phase);
4 down phase (vertical motion's phase).

Also there are four combinations of these motion's phases: 23, 13, 14 and 24 phases. L_{sum} is a whole length of the step cycle and H_{ad} is a height of the adaptation zone of the step cycle. Additionally a V_{trans} – horizontal speed of the foot in the transfer phase

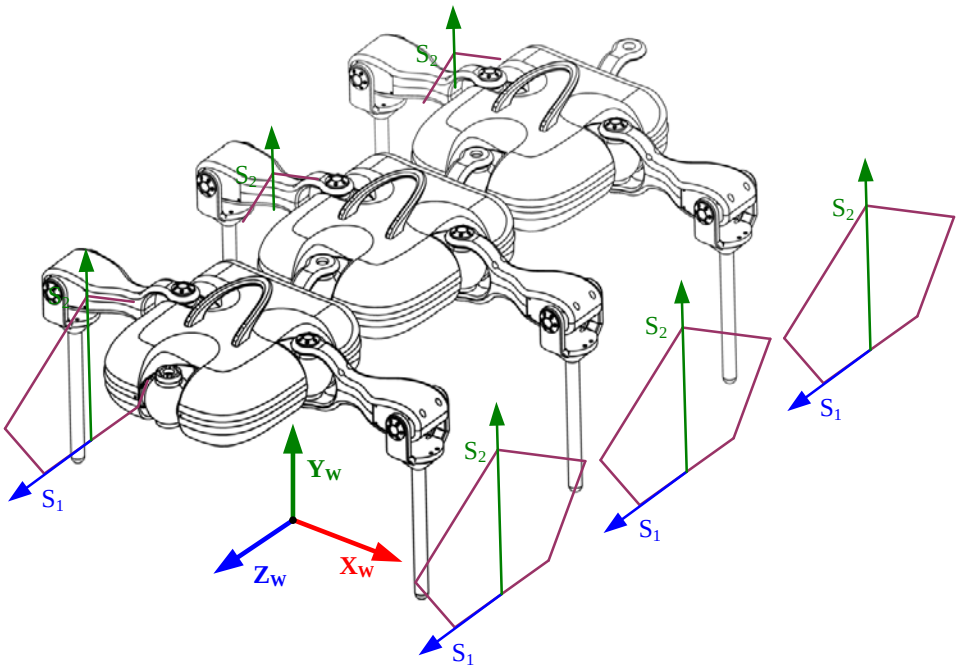

Figure 3.20 – Step cycle coordinate systems relative to robot world coordinate system

(limited by the maximal speed of the motor) and a τ – parameter of the art of the gait – relation between the transfer phase time and whole step cycle time (this parameter can be chosen between 0 and ½, see section 3.4.1) are introduced. Other missing variable can be estimated as follow:

$V_{tr} = V_{trans}$ – speed in the transfer phase (by the initial conditions); (3.1)

$V_{up} = V_{down} = 2 \cdot H_{ad} \cdot \frac{V_{tr}}{L_{sum}}$ – speed in the up (down) phase (the motion time in the combination of phases 1 and 3, hence 13 or 1 and 4, hence 14 is $\frac{L_{sum}/2}{V_{tr}}$); (3.2)

$T_{tr} = \frac{2 \cdot H_{ad}}{V_{up}} + \frac{2 \cdot H_{ad}}{V_{down}}$ – time of the transfer phase (height of the transfer phase is $2 \cdot H_{ad} + 2 \cdot H_{ad}$); (3.3)

$Period = \frac{T_{tr}}{\tau}$ – period of the step cycle (by mean of τ); (3.4)

$T_{st} = Period - T_{tr}$ – time of the stand phase; (3.5)

$V_{st} = \frac{L_{sum}}{Period - \frac{L_{sum}}{V_{tr}}}$ – speed in the stand phase. (3.6)

The main feature of the developed step cycle generator is assuming the velocities in the transfer, support, rising, and sinking phase of the step cycle to be constant despite the contact point. The step cycle generation unit provides the following features:

- permanency of full gait cycle time without dependence upon the contact point;
- vertical placement of leg on support and vertical detachment from support;
- automatic adaptation to small surface irregularities;
- able to modify of gait cycle (height, length, orientation in space).

Principle of Adaptive Step Generation: Main idea of adaptability is to generate two independent step cycles – ideal step cycle and reference one. Ideal step cycle generator always build ideal motion trajectory for each leg using only information about parameters of step cycle and information about contact detection with ground. Reference one tries to follow the ideal trajectory, but in the case of instability it doesn't follow the ideal trajectory and tries to carry out the stability rule. When the instability is over the reference generator tries to "catch up" the ideal one.

Principle of Adaptation to a Bearing Ground: Main idea of an adaptation to a bearing ground is to control the height of the clearance by displacement the gaits cycles in a vertical position. For this, the value of adaptation must be determinated – in our case it is the minimal height of the ground obstacles. The example of an adaptation is shown on Figure 3.21. The art of gait is "Tripod". Black line – ideal gait cycle, blue one – reference one and the lilac – the real adapted trajectory. An experimental result of adaptation to bearing ground with a height of 2.5 cm is shown on Figure 3.22.

Algorithm of step generation can be divided in two cases: start and stop logic.

Start Logic:

1. Update the step cycle time

   Must be updated each time step individually for each leg and reset at begin of combination of phases 2 and 3, hence 23. Used to:

   – change phase 2 to phase 23;

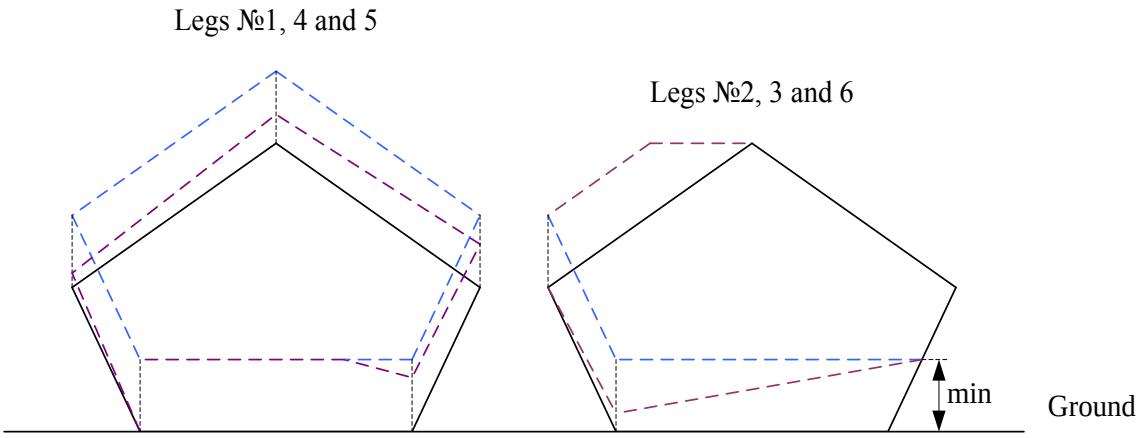

Figure 3.21 – Step cycle adaptation to a bearing ground

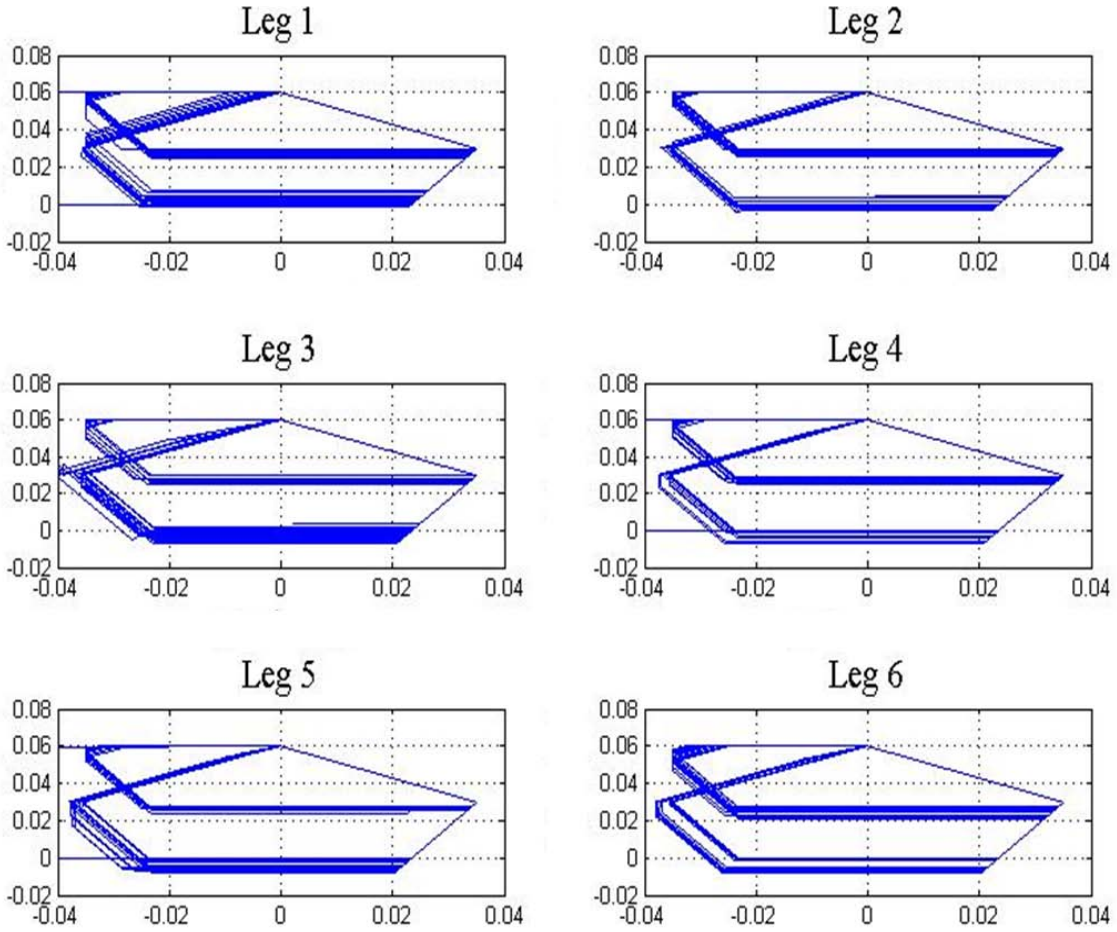

Figure 3.22 – Experimental results of step cycle adaptation to a bearing ground

– determinate the top of the step cycle trajectory (end of the combination of phases 3 and 1, hence 13, and begin of combination of phases 1 and 4, hence 14);

– determinate the initial position of the foots by the changing of art of the gait;

2. Update the ideal step cycle trajectory

    a) if all phases are 0 (start phases)

    – calculate all parameters of the step cycle;

    b) if the last foot is in the start point (begin of combination of phases 2 and 3, hence 23) and phases of all legs are 2 or the initial parameters of step cycle are changed

    – calculate all parameters of the step cycle;
    – determinate initial position of foots.

3. Update the reference step cycle trajectory

    In this case, the stability rule is checked. If the process is stable, then the reference step cycle trajectory generator follows the ideal one, otherwise carries out the stability rule.

4. Update the ideal and reference integrators

    In this case, the new coordinates of foots are calculated.

5. Adaptation of the clearance to a bearing ground

    determinate the value of adaptation, if all legs stay higher than 0-level horizon;

    apply adaptation to each leg.

Stop Logic:

1. Update the ideal step cycle trajectory

    In this case, the ideal phase and ideal coordinates of foots are equated with 0.

2. Update the reference step cycle trajectory

    In this case, the stability rule is checked. If the process is stable, then the reference step cycle trajectory generator follows the ideal one, otherwise carries out the stability rule.

3. Update the ideal and reference integrators

In this case, the new coordinates of foots are calculated.

4. Adaptation of the clearance to a bearing ground

determinate the value of adaptation, if all legs stay higher than 0-level horizon;

apply adaptation to each leg.

3.4.3 Locomotion: Walking

To realize a walking of the six legged robot above mentioned two rules (see Section 3.4.1) and step cycle geometry (see Section 3.4.2) must be applied to whole robot. For this purpose some new coordinate systems are introduced.

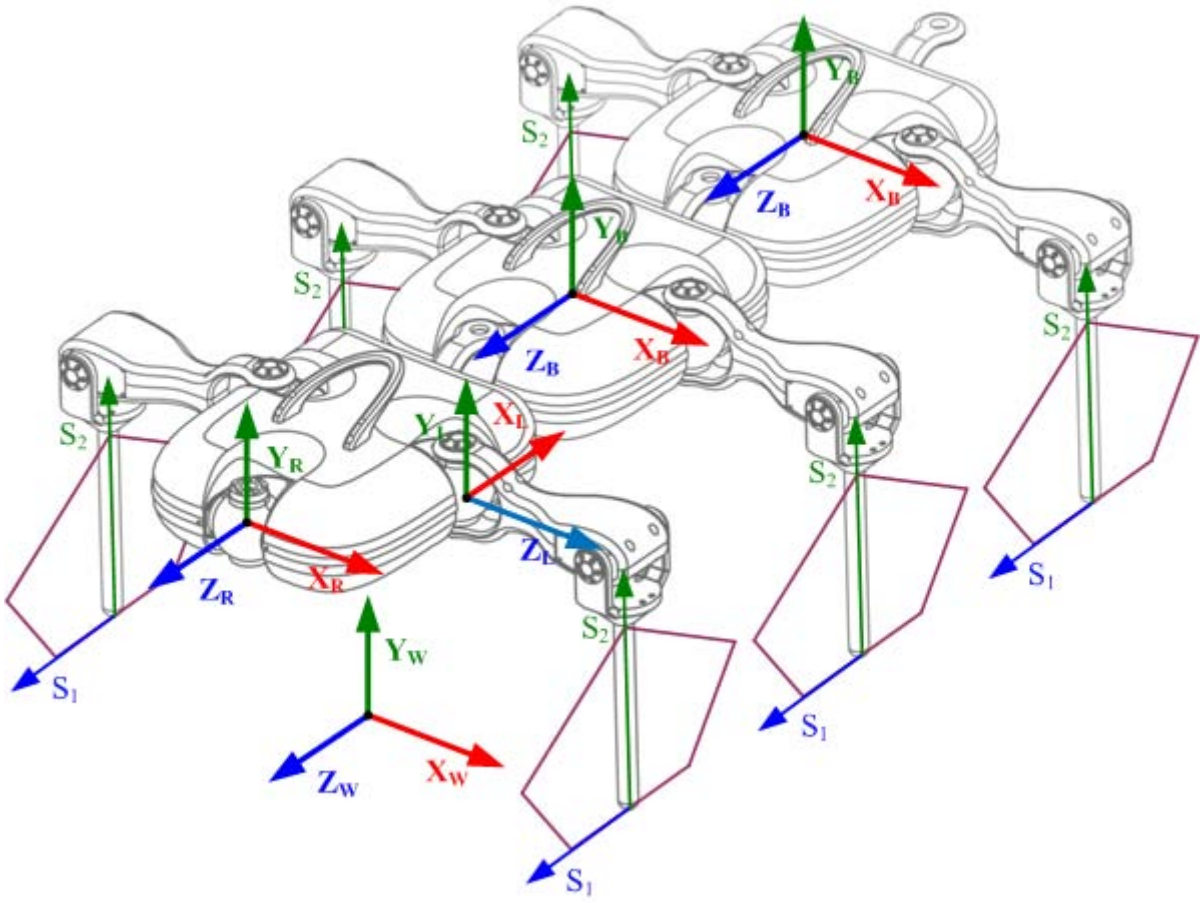

Figure 3.23 – Coordinate systems in the control algorithms

Figure 3.23 shows the disposition of different coordinate systems which are used in the control algorithms for the six legged robot "ANTON". Coordinate system $\{X_w; Y_w; Z_w\}$ describes the world coordinate system (WorldCS) which is a main coordinate system for all algorithms. Axis Y_w is placed along the gravitation vector and the axis Z_w is placed

along the straight walking of the robot. Second coordinate system is a robot coordinate system (RobotCS). This one is connected with a head of the robot and describes the clearance (translation along axis Y between WorldCS and RobotCS) and inclinations (rotation relative axes Z und X between WorldCS and RobotCS) of the robot. Third coordinate system consists of three different coordinate systems which are placed in the middle of each robot's bodies and is called body coordinate system (BodyCS). The task of it is an interaction between each body and the whole RobotCS due to additional degree of freedoms between the robot's bodies. Next coordinate system is a leg one (LegCS). It plays the role of detachment of two legs from one body. In other worlds it presents the coordinate system of each leg of the robot and is connected with first joint in each leg. Axis Z_L is always perpendicular to the straight motion of the robot.

Algorithm of walking can be divided in some important parts (some basic theory of coordinate transformation is described in [Sci00][Sch00]):

1. First of all initial position of the legs must be defined in the WorldCS. It is proposed to place the legs of robot around the ellipse or around rectangle (see Figure 3.24). These positions describe at the same time the positions of center of GaitCS of each leg.

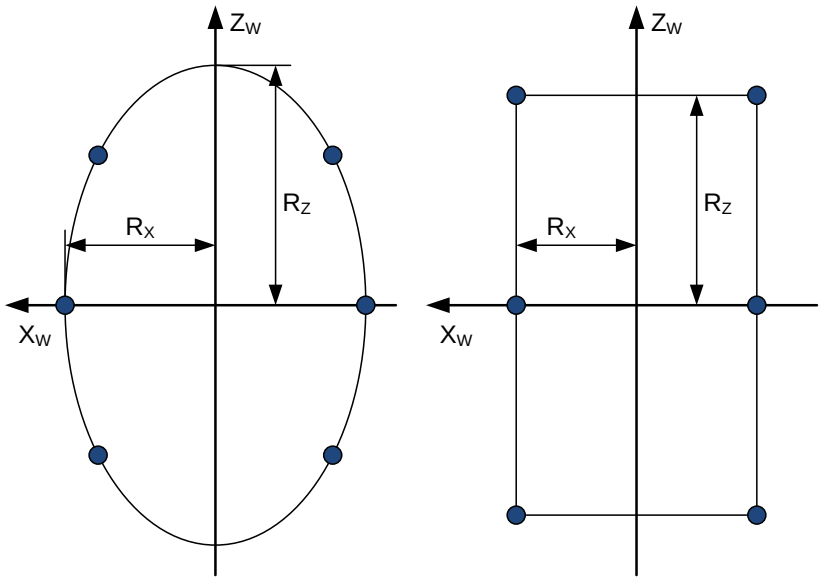

Figure 3.24 – Initial position of the legs in WorldCS

2. Second step is a recalculation of generated in GaitCS trajectories into WorldCS. Additionally, the generated gait trajectories in GaitCS can be scaled and rotated

independently for each leg along the axes of WorldCS. The additional scale along the axis Y_W can be used to walk without translation speed – this used to realize the turn of the robot on the same place. The additional rotation of the gait trajectories around axis Y_W applies side walking of the robot. Some another combination of scale-rotation of gait trajectories is used to adapt the robot to walk with disturbances or over complex terrains.

3. After the coordinates of each leg of the robot is obtained in WorldCS, the corresponding coordinates of each leg must be recalculated into RobotCS. In this case the clearance and the inclination of the robot are considered. Additionally, the whole robot can be rotated around the axes of RobotCS. For example, the rotation around axes Z_R and X_R are used to align the robot's body parallel to horizon.
4. Next step – is a recalculation of legs' coordinates from RobotCS to BodyCS. Body coordinate system allows some operation with the robot's body. First of all, it allows using additional degree of freedom between robot's bodies. Additionally the rotation and translation motions of the bodies can be presented in BodyCS. For example, the correction of position of center of mass (see Section 3.4.4) can be realized using the translation of robot's bodies.
5. Last stage – is a representation of leg's coordinates in LegCS. After this representation the inverse kinematic algorithm will be calculated and the corresponding joint's angles will be obtained.

3.4.4 Locomotion: Stabile Motion of Robot Without Disturbances

If the walking robot is not subject to external disturbances and the mass distribution in the robot is a-priori known, then the definition of the robot COM (centre of mass) needs only additional determination of the robot orientation relative to bearing surface and can be found according the following formulas:

$$\begin{aligned} COM_X &= rot\left(\frac{\sum X_{BODY}\cdot M_{BODY}\cdot g+\sum X_{LEG}\cdot M_{LEG}\cdot g}{\sum M_{BODY}\cdot g+\sum M_{LEG}\cdot g}\right) \\ COM_Z &= rot\left(\frac{\sum Z_{BODY}\cdot M_{BODY}\cdot g+\sum Z_{LEG}\cdot M_{LEG}\cdot g}{\sum M_{BODY}\cdot g+\sum M_{LEG}\cdot g}\right) \end{aligned} \tag{3.7}$$

In order to guaranty stabile robot motion, the COM must be forecasted for the next step cycle phases. The most critical is the transition between the step cycle phase (especially from support to rising phase) by the symmetrical gallop gait. The vector correction

$\Delta corr$ (Fig.6) scaled in accordance to the desired criteria is used to control the robot stability. Figure 3.26 shows the correction of the position of COM during the robot walking.

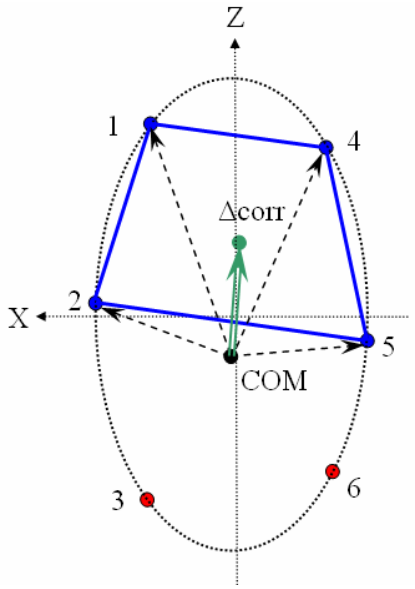

Figure 3.25 – Control of COM of robot [Kon072]

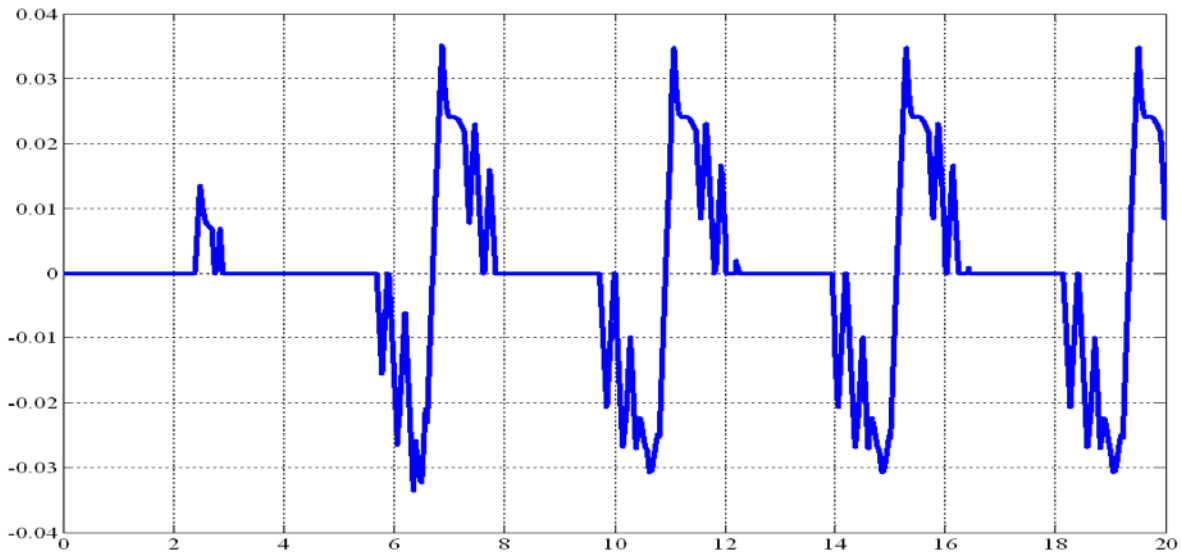

Figure 3.26 – Correction of the position of COM

3.4.5 Extended Locomotion

All above-mentioned properties of mechanical structure, sensor system and control system make possible from mechatronical point of view the standard locomotion tasks of multilegged robots such as:

- locomotion over rough terrain,

- foot force distribution,
- estimation or identification of the ground mechanical properties,
- maintenance of statically stability in a complex terrain,
- control of linear and angular movements of the body during operation,

as well as advanced or extended locomotion tasks such as:

- motion over complex terrain with different mechanical ground properties: foot force distribution;
- climbing the obstacle with smooth and vertical slope.

In contract to basic locomotion tasks which can be achieved through modification of control parameters on "primitive level" extended locomotion tasks are more complex and must modify additional parameters on the "action level" of the control system using the sensors of "outer information" (interaction force and near navigation).

3.4.6 Extended Locomotion: Stabile Motion of Robot with Disturbances

The motion over complex terrain with various mechanical properties of the bearing surface (especially soft ground) differs strongly from the motion on the terrain with hard ground.

Firstly, in the absence of force control, the inaccuracies in the robot kinematics, the uncertainties in sensor or control system can lead to significant mechanical loading upon separate legs. Therefore the reactive forces under the feet must be actively distributed and controlled.

As the system is statically indeterminate with respect to forces acting on the legs (support can be made with more than three legs), so support reactions change in a random way. The commanded vertical force components are computed from the body orientation relative to the gravity vector, and from the leg configuration. They must satisfy the static equilibrium equations:

$$\sum_{leg} F_{VERT}^{leg} = P\,,\ \sum_{leg} F_{VERT}^{leg} x^{leg} = PX_{COM}\,,\ \sum_{leg} F_{VERT}^{leg} x^{leg} = PZ_{COM} \tag{3.8}$$

where P is the robot weight, x^{leg}, z^{leg} are coordinates of the foot. It is clear that if n>3, the solution may be chosen ambiguously and the additional criteria must be added.

There are different ways of eliminating the indeterminacy. Let us require that the vertical force components should satisfy:

$$\sum_{leg=1}^{SUPPORT\_LEGS} \left(F_{VERTICAL}^{leg}\right)^2 \rightarrow \min \tag{3.9}$$

This condition has the sense of energy optimization. Solution of equation (3.9) is discussed in [Rus07]. Thus, the active distribution of support reactions allows for the reduction of loads on the robot structure and of the energy consumption of leg drives (Figure 3.27 - Figure 3.29).

3.4.7 Extended Locomotion: Climbing

First example of climbing – climbing over smooth obstacles (Figure 3.30) will be explained. The main idea is that the robot body must follow the big irregularities of the ground (bigger than standard adaptation zone in the generation of the step cycle) concerning the algorithm pictured here:

- firstly, the sensor system for near navigation (or an operator in our case) detects the imaginary radius of corner and determinate normal to the calculated surface taking into consideration the robot clearance;

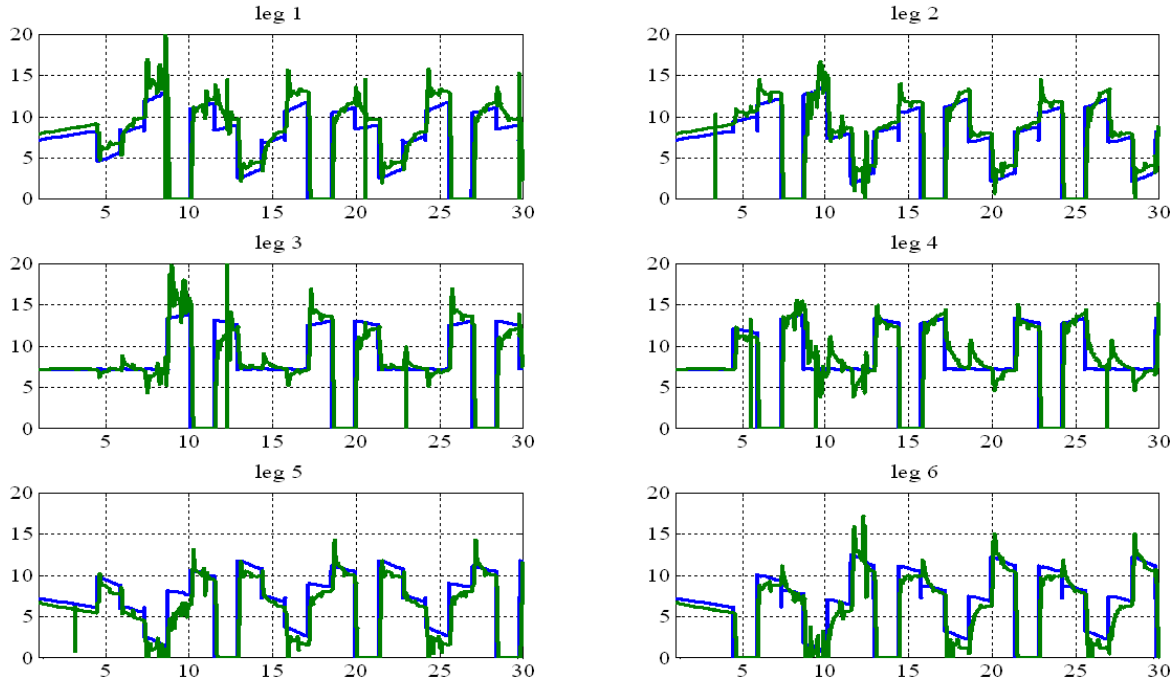

Figure 3.27 – Experimental results of vertical components of food forces in locomotion over rigid surface during the walking using wave gate (blue – calculated reference signal, green – real vertical force component)

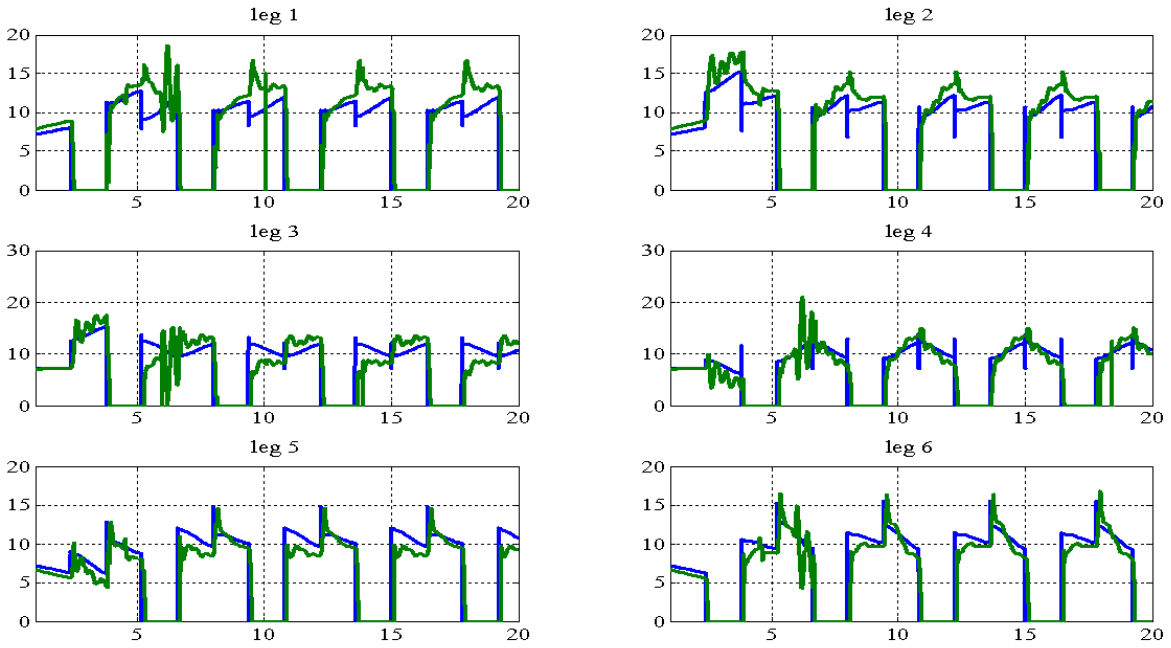

Figure 3.28 – Experimental results of vertical components of food forces in locomotion over rigid surface during the walking using galop gate (blue – calculated reference signal, green – real vertical force component)

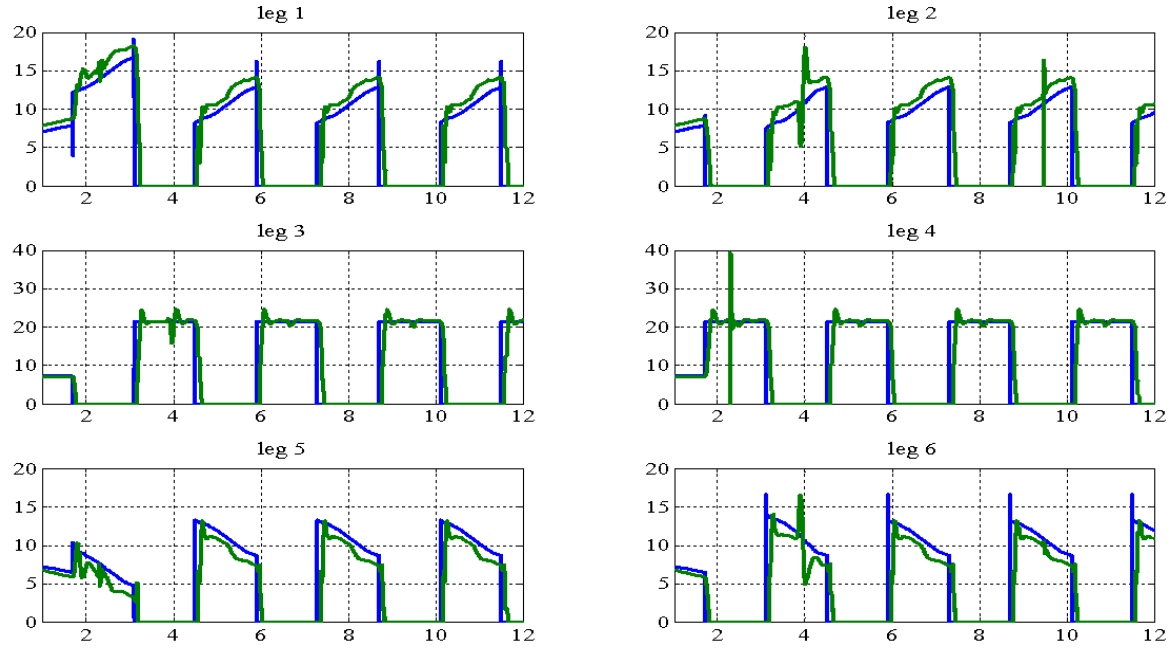

Figure 3.29 – Experimental results of vertical components of food forces in locomotion over rigid surface during the walking using tripod gate (blue – calculated reference signal, green – real vertical force component)

- than concerning the determined normal the robot body segments must be controlled according to the given formula:

$$\varphi_{REF}^{BODY} = \{\varphi_{NORMAL}^{NEXT\_SEGMENT} - \varphi_{NORMAL}^{PREVIOUS\_SEGMENT}\} \quad (3.10)$$

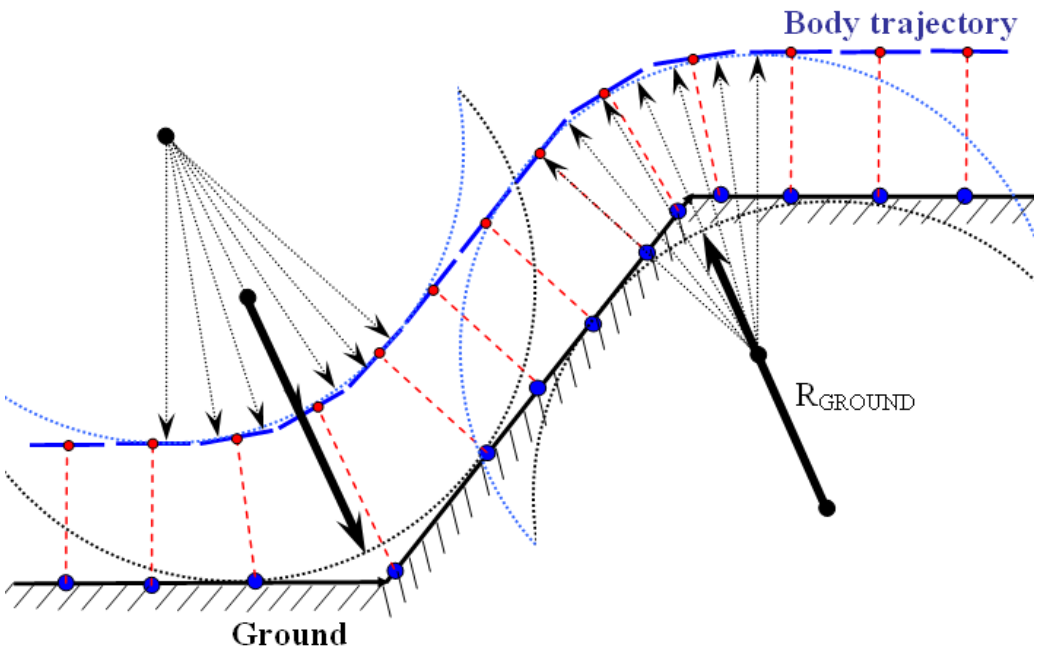

Figure 3.30 – Climbing over smooth obstacles [Kon06]

The more and smaller body segments the robot has, the smoother it can follow the bearing surface and the smaller deviations from the calculated support surface (cycle) are. Real deviations will be eliminated by adaptive step cycle generation unit (adaptation zone) and reactive gait controller (transition to support phase only with contact).

Second example, the possibility of using the robot for the climbing on the obstacles equal to robots body (Figure 3.31) will be examined. The key words for construction of the "climbing tasks" are: the "symmetrical gallop" gait mode, the definition of the foot support points on the "action level", the check and the prediction of motion stability and the movement in cones of friction.

The following algorithm [Pal04] is used for completing the climbing task in the simulation:

1. robot motion to the ledge as well as determination and processing of information about the obstacle dimensions (as a result the calculation of the foot support points);
2. switch to "symmetrical gallop" gait mode (always in following sequence "rear-middle-front" shoulder) and transfer the legs possibly near to the ledge concerning the stability margin (Centre of Mass COM within support pattern);

3. lift up the front shoulder legs with the forward movement of the body at the same time and setting them on the horizontal obstacle part (the first gallop wave after that is complete);
4. transfer the body to the obstacle (COM must be near as middle foots);
5. transfer the rear shoulder legs to the obstacle, than lift up the middle shoulder legs on the obstacle surface (on the edge), and then transfer forward the front shoulder legs as far as possible (the second gallop wave after that is complete);
6. transfer the body after the obstacle (COM between middle and front shoulder) ;
7. support on the middle and front shoulders as well as body and lift up the back shoulder legs, what is connected with the strict control of the support reaction and of the center of mass location inside the supporting area at each moment of time;
8. transfer than middle and front shoulder legs (after that the third gallop wave and that climbing task is complete).

As can be seen in algorithm considered above and in the pictures (Figure 3.31) the usage of the articulated body allows closely approaching the COM to the ledge for the robot with flexible body rather than for robot with rigid body. Articulated body makes also possible to adjust steeper an angle between the body segment and horizon and therefore to overcome bigger obstacles. The usage of articulated body brings significant advantages concerning to calculation of the required maximal motor torque. Because in the most critical for a drive case (step 7 in the algorithm, Figure 3.31, picture 4) the maximal motor torque is inversely proportional to cosine of angle between the body segment and horizon.

The maximal overcoming obstacle high can be estimated as following:

$$h_{MAX} = l_{LEG} + l_{BODY\_SEGMENT} \cdot \sin(\alpha) \tag{3.11}$$

If the body segment length is smaller than the leg length, the usage of articulated body doesn't yield significant advantages concerning to obstacle height. In our case, the usage of articulated body brings significant advantages concerning to calculation of the required maximal motor torque. In the most critical case for a drive (support on the middle and front shoulders and lift up the back shoulder legs) the maximal motor torque $\tau_{MAX} \sim l_{BS} \cdot \cos(\alpha)$ is inversely proportional to cosine of angle α .

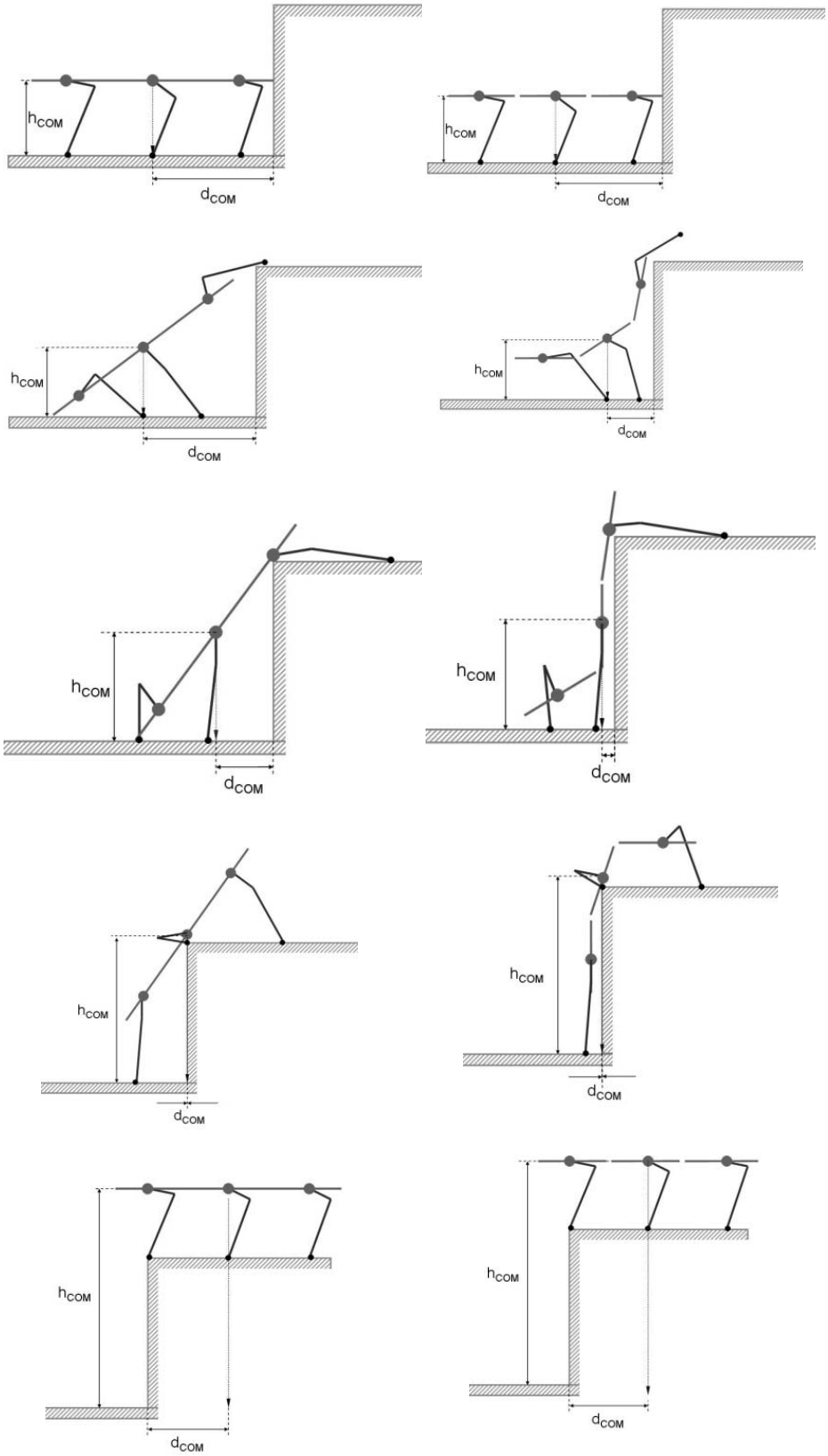

Figure 3.31 – Comparison of climbing between robots with rigid (left) and flexi (right) body [Kon06]

3.5 Conclusion

The multilegged robot with articulated body has been developed. The modular design of the robot and of the control system allows easily to extend and to upgrade the robot with additional capabilities.

The sensor system makes possible the completion of the standard and advanced autonomous tasks in complex environment.

The robot is specially intended for development of control algorithms, for the investigation of the robot motion over extremely complex terrain, as well as for climbing obstacles that are equal to robot's body dimensions.

A control system based on force information from an environment can solve the problem of legged robot movement or body displacement along a constraint (most applications require this kind of control).

Chapter 4

Two-Legged Walking Robot "ROTTO"

During the last years many researchers are engaged in tasks how to develop the efficient robot prototypes and their control system. This section deals with designing and developing a biped robot based on experience after development of multilegged robots. Moreover, the control system of fully or partly autonomous biped robot is almost always controlled by embedded system. Commonly the embedded systems are designed to control complex plants such as engines, satellites, vehicles, spacecrafts, and of course CLAWAR. They generally require a high level of complexity within the embedded system to manage the complexity of the controlled plant.

Development and test of complex real-time embedded systems require many steps from modeling and simulation of the plant till the implementation of the source code in the real hardware. Nowadays hybrid techniques like Software-in-the-Loop (SiL) and the Rapid Control Prototyping (RCP) are used more and more often. We use these methods while developing and testing the biped robot.

Thus the next technical purpose of this work is to develop a biped robot with the possibility of dynamical walking, as well as to develop a control system that allows simple and

rapid development of the complex control algorithms. The purpose of this work is the development of control algorithms for desired mechanical interaction with environment. Moreover, flexible communication bridge for real-time communication between the real control system and real robot is presented.

4.1 Design Philosophy of Biped Robot "ROTTO"

The main purpose of biped robot constructions is to develop mobile robot able to solve following research tasks:

- research and investigation of low-cost energy gaits (ballistic walking is one of the possible gaits),
- development and research of the methods of force/impedance control of robot foots during contact with the ground,
- investigation of dynamical walking and methods of robot's body stabilization during walking.

To perform above mentioned requirements a biped robot "ROTTO" (see Figure 4.1 and Figure 4.2) has been designed and constructed. The kinematical construction of the robot is shown in Figure 4.2. The robot has been constructed to provide more than 25 Degree Of Freedom (DOF) similar to what human being provides. Suggested robot's construction is characterized by modular structure using linear drive systems in each joint. Bearing structures of the robot are fabricated from carbon material and connected to each other with optimized milled-out aluminum constructions. The aluminum-carbon construction reduces the weight of robot and, at the same time provides sufficient robustness.

The construction of the "hip's" and the "ankle's" joints (Figure 4.3) are implemented in parallel kinematic. Two linear drives actuate the corresponding 2-DOF using the connecting rods and the ball joint. The synchronous and the asynchronous motions of the drives produce the corresponding motion relative to one or another DOF in the joint. It should be pointed out, that such construction ensures the double rotation moment on the two rotation axis.

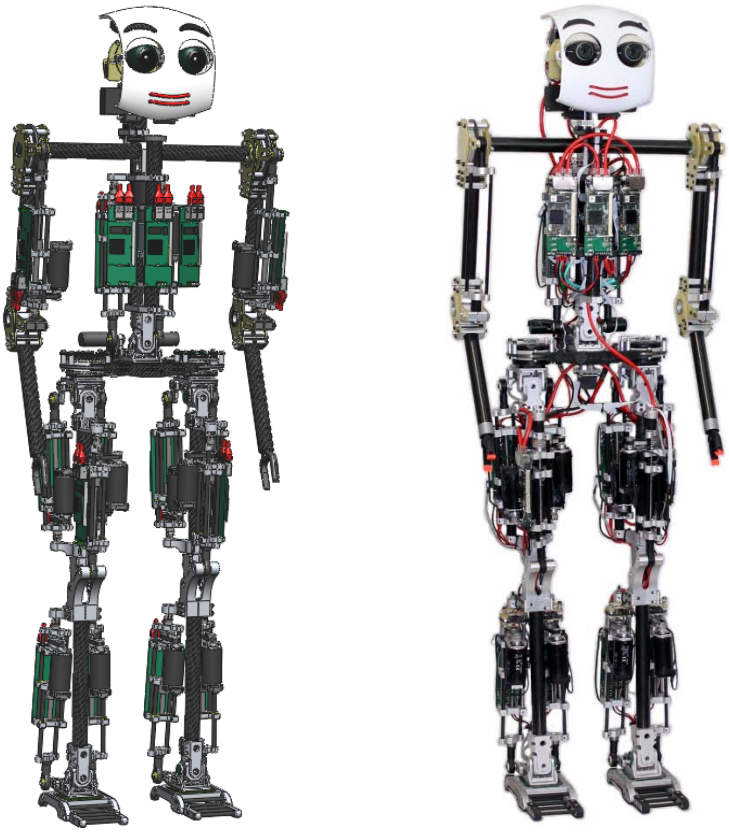

Figure 4.1 – CAD construction (left); real biped robot "ROTTO" (right)

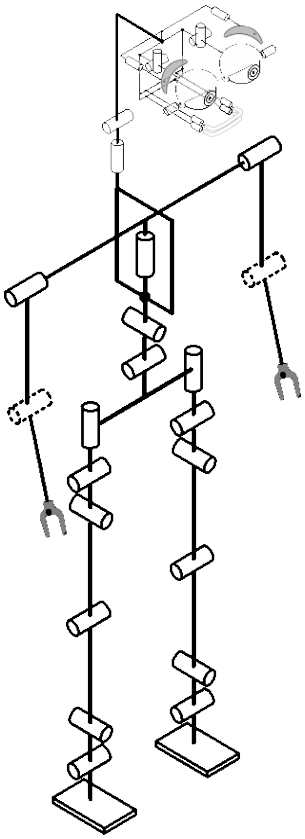

Figure 4.2 – Kinematic structure of biped robot "ROTTO"

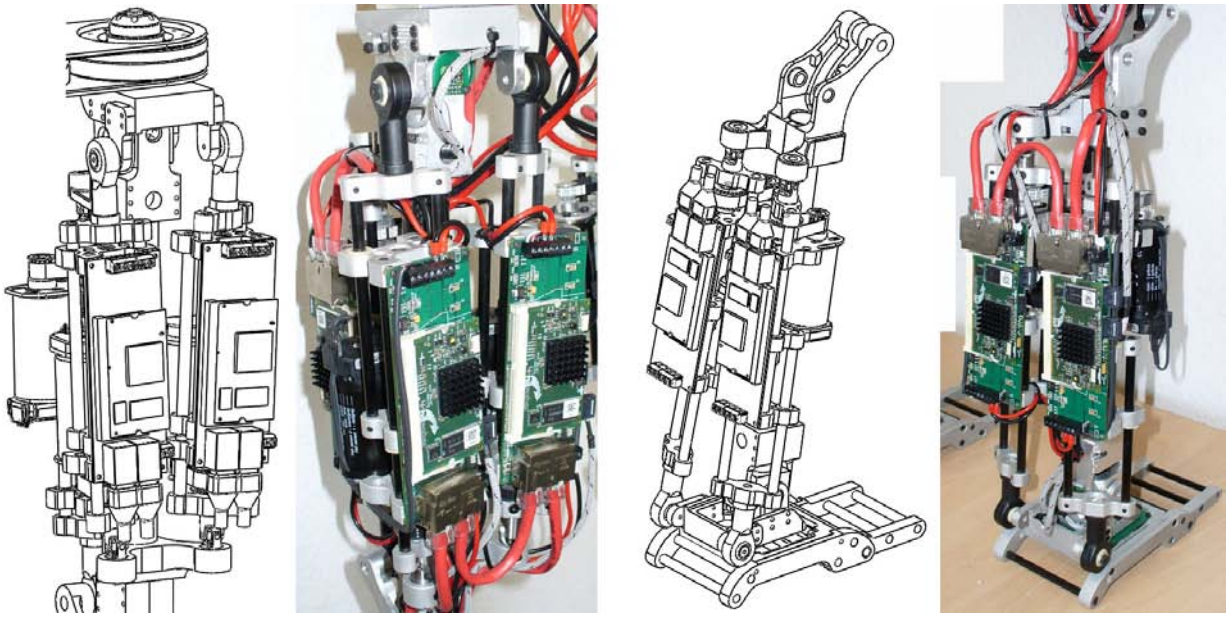

Figure 4.3 – Construction of the "hip's" joint (left); construction of the "ankle's" joint (right) [Kon091]

Opposite to the construction of the "hip's" and the "ankle's" joints the construction of the "knee's" joint (Figure 4.4) is implemented in serial kinematic. One linear drive actuates the corresponding 1-DOF using the connecting rods and the ball joint.

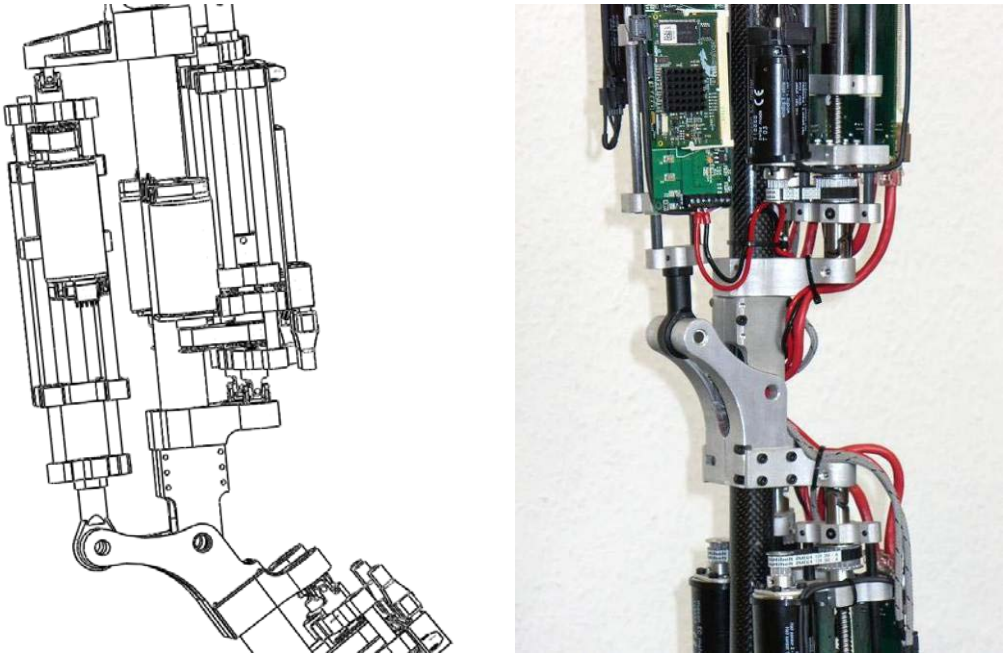

Figure 4.4 – Construction of the "knee's" joint [Kon091]

Construction of pelvis is shown on Figure 4.5. Two linear actuators drives realize the verticals degree of freedom for each leg accordingly.

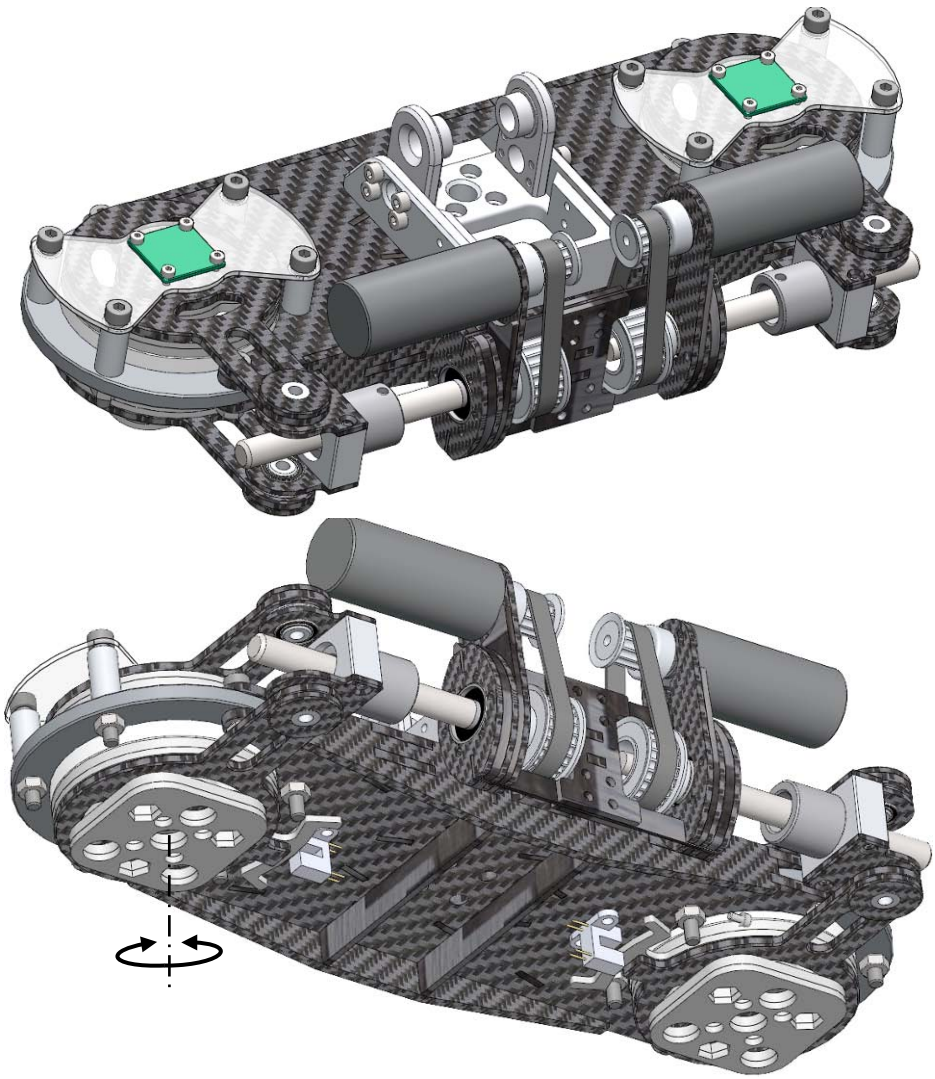

Figure 4.5 – Construction of the pelvis (CAD)

4.1.1 Actuators of Biped Robot "ROTTO"

The actuators of the biped robot "ROTTO" are implemented as linear drives (Figure 4.6). The brushless EC-motor drives the ball screw using belt transmission with a reduction ratio of two. The linear motion of the ball screw's nut is realized by carbon sticks which are at the same time the direction guide. There is an incremental position sensor on the motor axis. For safety purpose, two limit switches are installed. The on-board electronics collect all the data from sensors and accomplish current control of the EC-motor. The nominal force of this actuator is 128 N with a nominal speed of 0,64 m/s, but the maximal force is 400 N and the maximal velocity is 0,72 m/s. It should be pointed out, that the maximal values of the actuator force and speed can't be reached at the same time.

The complete integration in one module of the motor, the gear, the sensors and the electronic offers the following advantages:

- reduced volume of actuator;
- better dynamic properties;
- higher power density;
- better efficiency;

the EC-motor and the modular structure of drive provides better reliability as well as integrated diagnose and observation functionality.

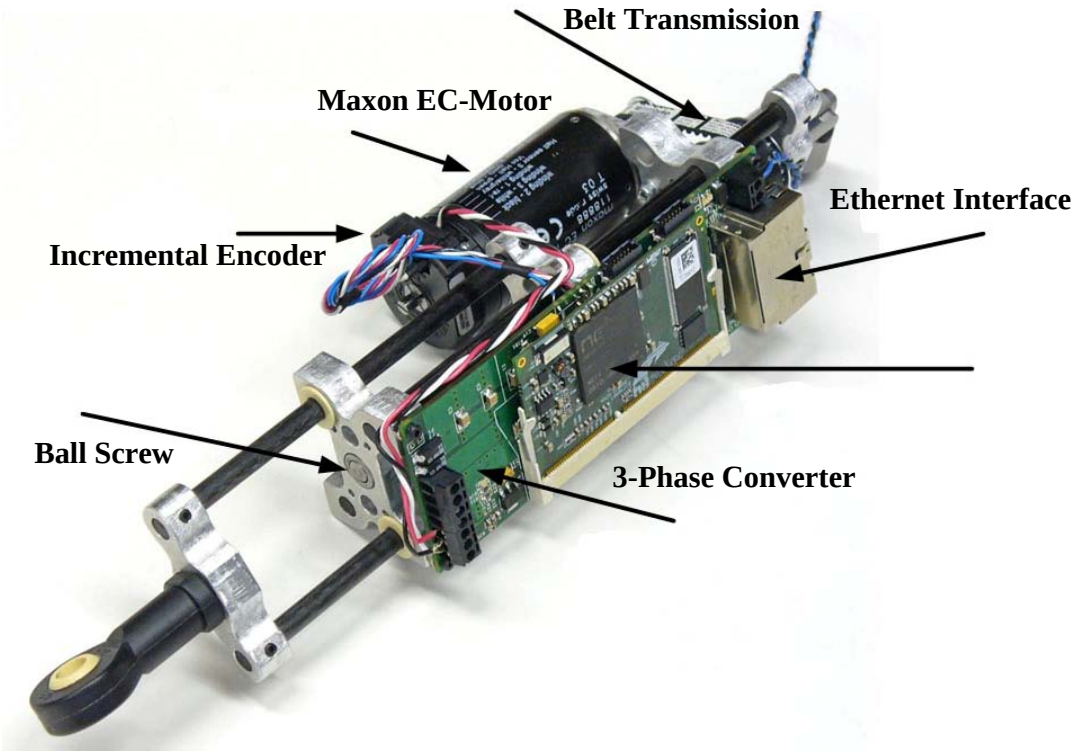

Figure 4.6 – Linear actuator of biped robot "ROTTO" [Kon091][Kon102]

4.1.2 Modified Actuators of Biped Robot "ROTTO"

The developed design of the elastic element, combined with the force sensor is shown in Figure 4.7. Modern composite materials allow a higher energy storage density in the elastic element by smaller masses. The main operating element is a flat coal-plastic spring 5. The deformation of the spring is measured by the Hall-sensor 4 placed in the magnetic field of the two neodymium magnets 3. The glass fibre plate 2 and the screw 1 serve to fix the actor.

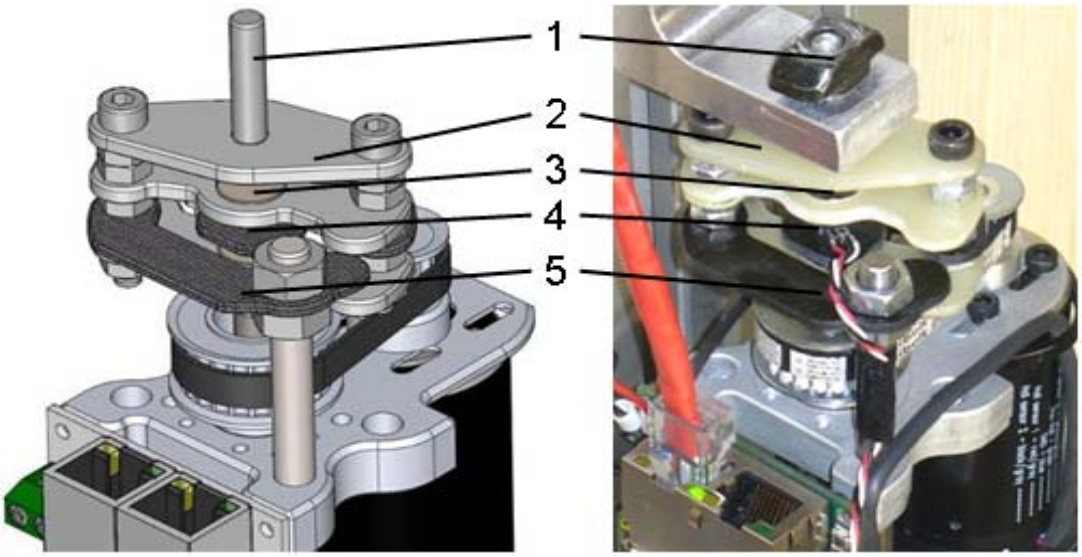

Figure 4.7 – The CAD Model of the elastic actuator (left) and actuator of the robot (right) [Kon092][Kon102]

4.1.3 Sensors of Biped Robot "ROTTO"

The sensor system of the robot consists of components that are standard for mobile robots and that makes it possible to achieve autonomous robot functions in an environment. It includes:

- absolute magnetic encoder installed in each robot joint;
- incremental position sensor, two current sensors and two limit switches installed in each of the actuator module;
- six-component force sensor mounted in each leg's shank;
- high precision tri-axis inertial sensor installed on the robot body.

Designed force sensor provides measurement of forces in wider range in order to comply with the requirements on measurement and control of the support reactions during the dynamical walking. The maximal allowed vertical force is 400N. The maximal values of tangential and lateral forces are chosen up to 300N. The developed sensor is shown on Figure 4.8. The force sensor (Figure 4.8) is of "cross" form and consists of four radial beams (1) which connect the robust central flange (2) with the outer flange (3). The positions of eight strain gauges that are used for the calculation of the reaction forces and torques are determined using the FEM analyze of the force sensor. The results of the FEM analyze is shown on the Figure 4.9. Usage of the strain gauges is the most widely-known method of the experimental measurement of material tension. The active resistance of the strain gauge is changed with the changing of the material tension.

Changing of active resistance of the strain gauge can be easy measured by the Wheatstone bridge. The whole mechanical construction of the force sensor is also optimized according to FEM results.

During the motion the radial beams are being deformed by the external forces as well as the external torques applied to the central flange of the sensor (Figure 4.9). These deformations are measured by the strain gauges on all radial beams. The recalculation measured data from the strain gauges or, to be more correct from the Wheatstone bridges, is described in [Gor00]. The recalculation consists of two stages. First stage is calculation of the sensor (material) stiffness and the second one – sensitivity of the sensor [Gor00].

After large number of experiments on the sensor, the characteristic of the force sensor is defined. The experiments are carried out on trial stand. The force sensor is loaded with the precise weights, so to say with external forces and torques. Obtained results are used for evaluation of accuracy, resolution and other properties of the sensor. The characteristic of the force sensor is shown in the Tabelle 4.1.

Tabelle 4.1 - Characteristic of the force sensor

	F_x, F_z	F_y	M_x, M_z	M_y
Resolution, [bit]	12	12	12	12
Accuracy, [N], [Nm]	0,29	0,39	0,098	0,1
Measurement range, [N], [Nm]	± 300	± 400	± 10	± 20
Resettability, [%]	< 80	< 80	< 80	< 80

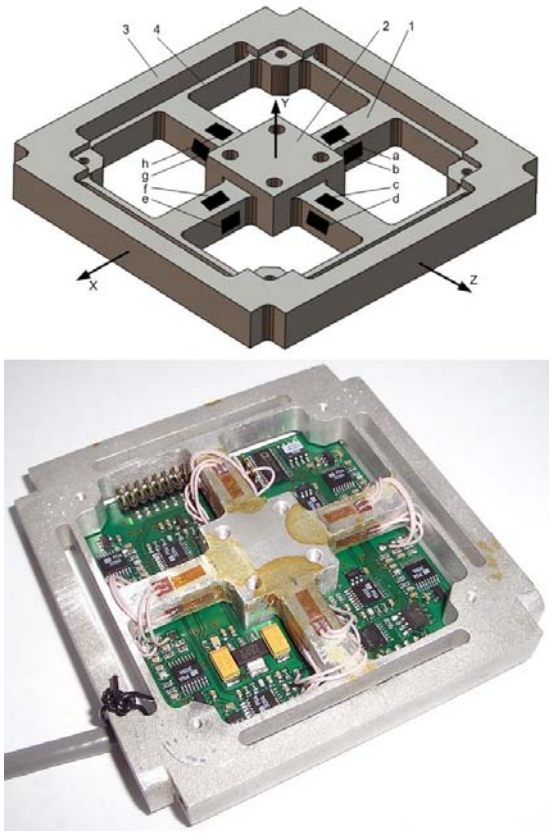

Figure 4.8 – Six-component force sensor of biped robot "ROTTO": CAD model (top); real force sensor (bottom)

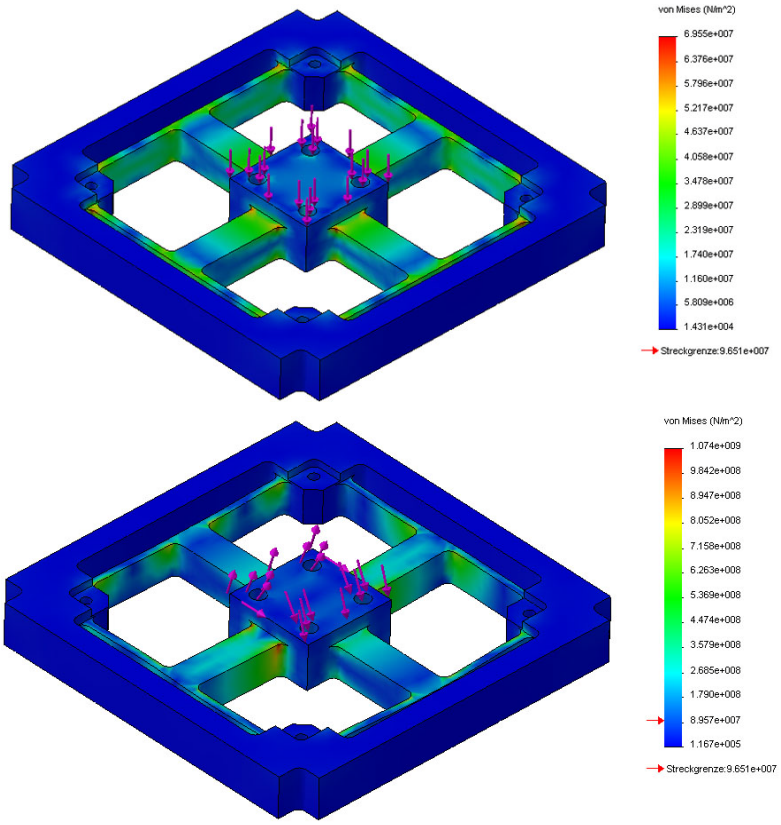

Figure 4.9 – Six-component force sensor of biped robot "ROTTO": CAD model (top); real force sensor (bottom)

Experimental results obtained during the force sensor test are shown on the Figure 4.10. The top picture shows the raw information received from all measurement channels of the force sensor. The left bottom picture shows the recalculated resulting values of the torques applied on the sensor. The right one shows the recalculated resulting values of the forces. Obtained results prove quite sufficient accuracy and good resettability of sensor.

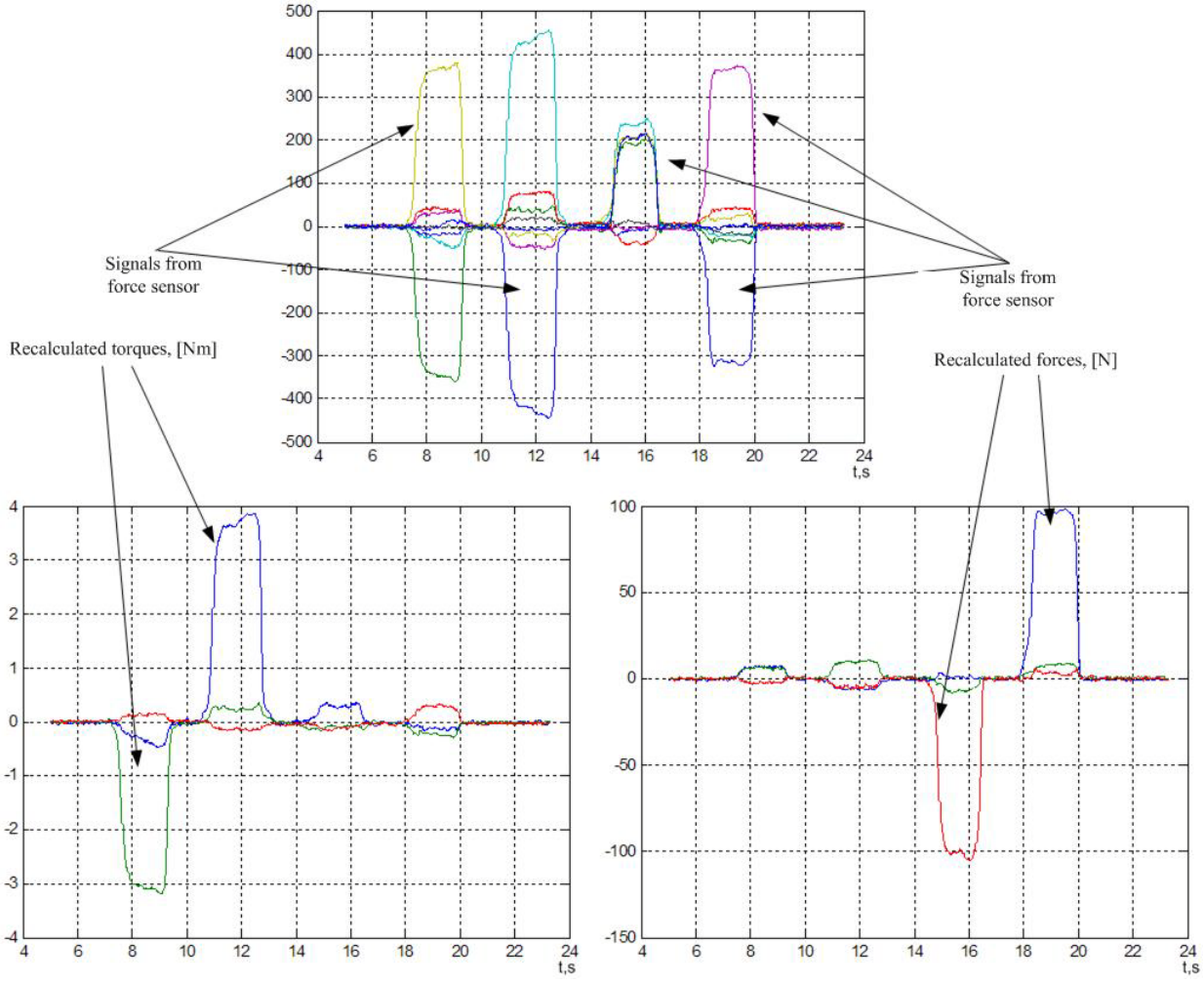

Figure 4.10 – Experimental results of the force sensor under the different loads

4.2 Hardware Control System

The control electronics (Figure 4.11) as well as the all control architecture and algorithms for the biped robot "ROTTO" have been developed and showed satisfactory results. The control electronic consists of two boards: the small one netX SODIMM module, and the second one with power electronics.

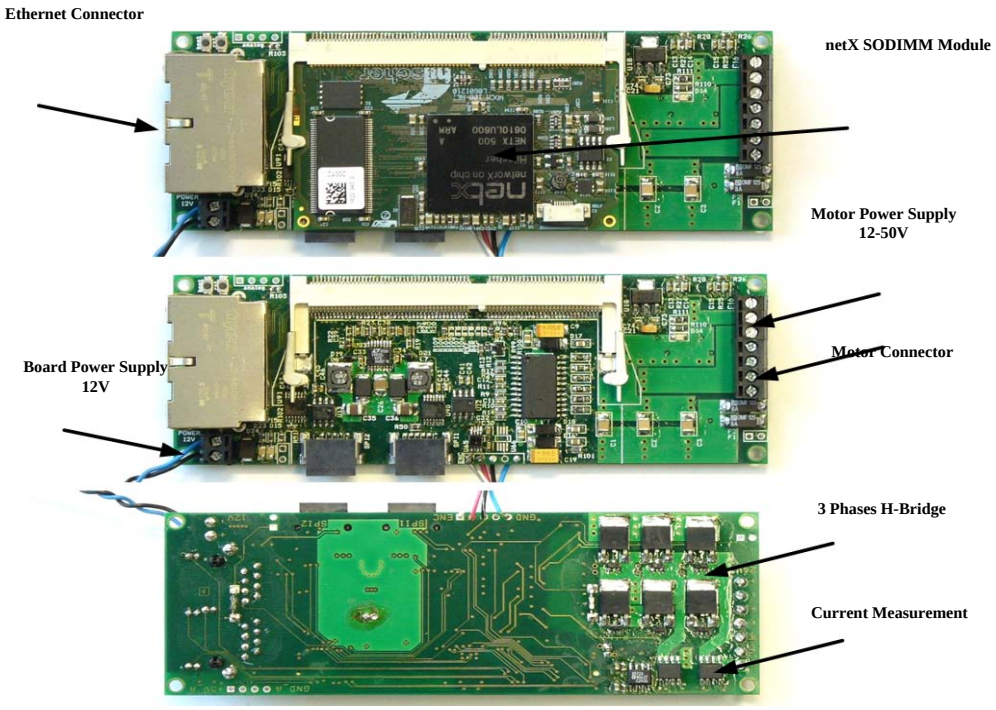

Figure 4.11 – Decentralize control electronics of the biped robot "ROTTO"

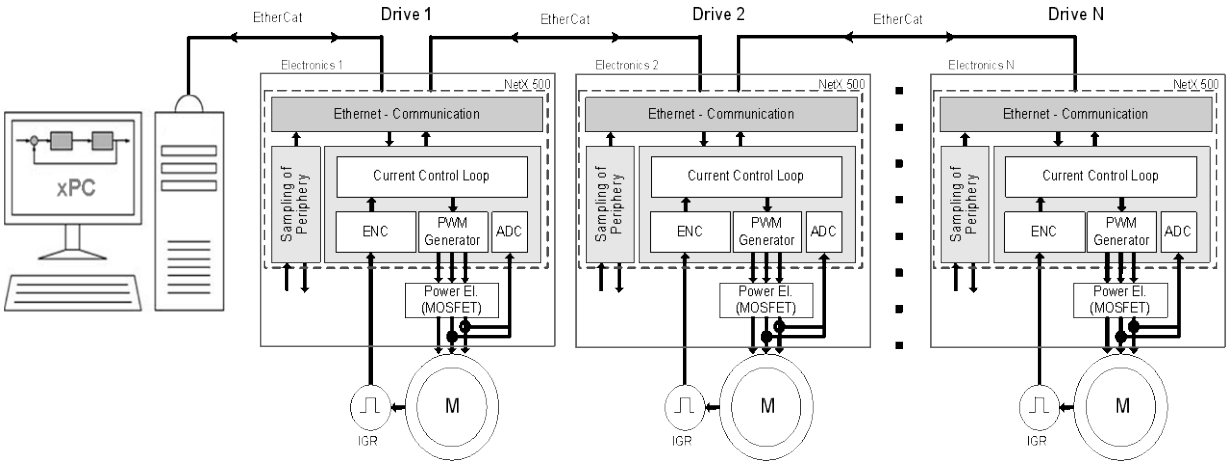

Figure 4.12 – Flexible communication system based on netX processors [Kon09]

Control system with real-time decentralized data gathering and processing builds the kernel of robot system and allows developing of control algorithms with help of hybrid simulation. The implemented control system is already at industry level and is based on typical real-time communication EtherCAT. The developed hardware (see Figure 4.12) consists of netX communication processor (Hilscher GmbH). The time diagram of communication process is explained in Figure 4.13. The control system (Control) on Host-PC

(xPC Target) is computed each sample time (Tsample) by the information received from the sensors (RD – Receive Data). Computed reference signals of PWM (SD – Send Data) are copied to the EtherCAT packet and the EtherCAT master (CIFx-50) to initialize a telegram (Packet n) transfer. The received data by the EtherCAT slaves (NetX 500) are copied through the DPM (Dual Port Memory) to the xPEC and then to the PWM module. Detailed presentation of decentralize Single-Chip controller based on the netX processor is shown in Figure 4.14.

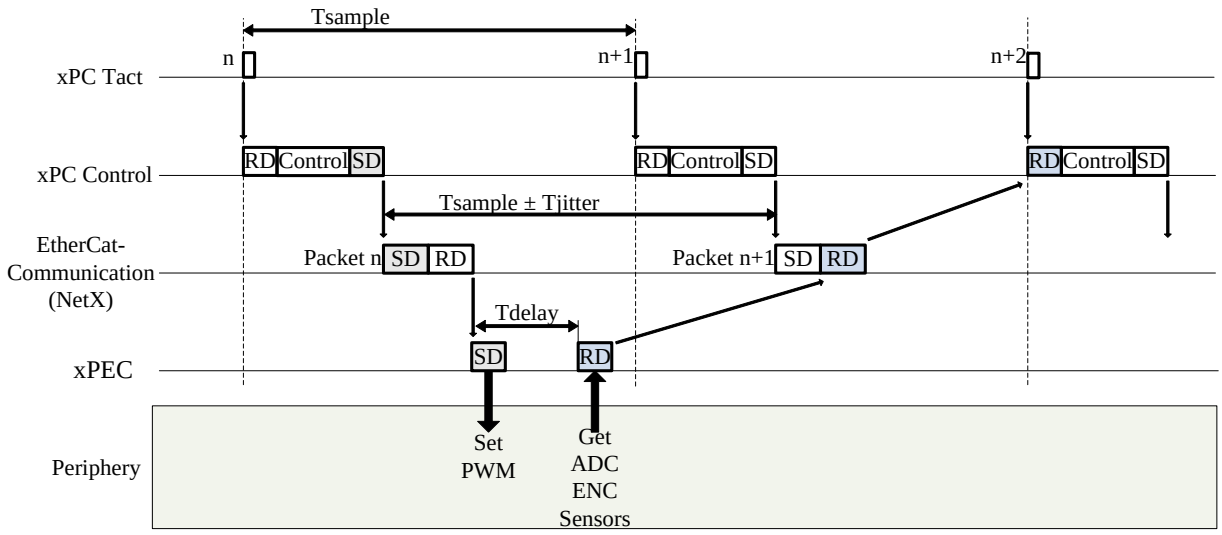

Figure 4.13 – Time diagram of communication process [Kon09]

The main feature of the EtherCAT communication protocol [Pal07] is the exchange of the data in the loop Master-Slave-Master with one solid telegram. Therefore sending of the response information by the EtherCAT slave is required preliminary preparation of the transfer buffer. The delay $T_{delay} = T_{sample}/2$ between the receipt of the telegram by EtherCAT slave and the monitoring of the sensors is realized. This was done for the minimization of the time difference between monitoring of actual sensors information (ADC, ENC and etc.) and the instant of usage them in the control calculation (n+2 xPC Tact). Received information from sensors (RD) is saved to the transfer buffer of the EtherCAT slave. After that it will be sent in the next closest transfer packet (Packet n+1) and will be actual on the n+2 work period.

The standard PC is used as a control computer (Host PC). Therefore the jitter time in the communication system due to large number of hardware peripheries is occurred. For the minimization of jitter time the practically all of hardware peripheries are tuned off in the BIOS of PC. The maximal jitter time in our communication system is less than $15\mu s$.

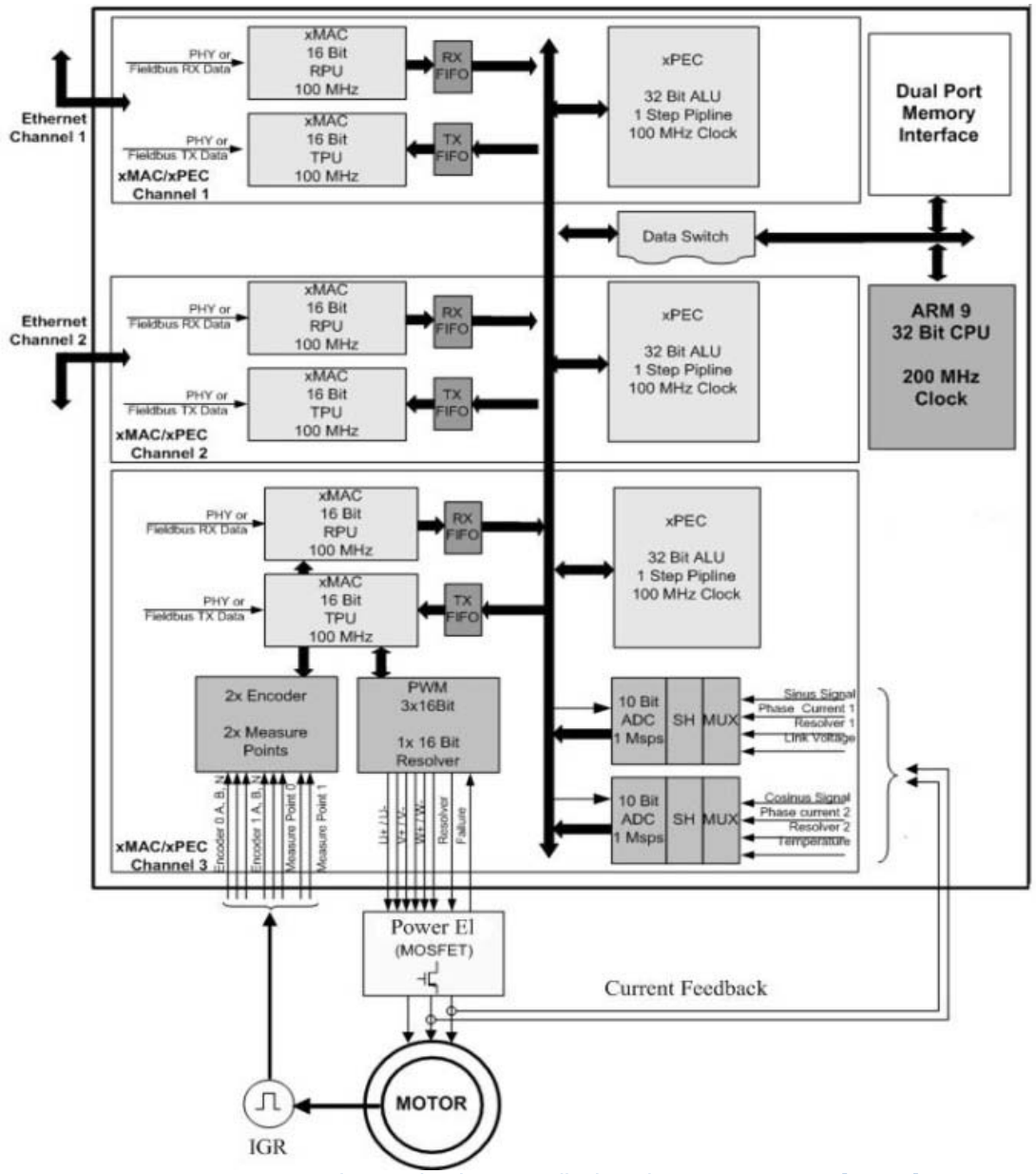

Figure 4.14 – Decentralize Single-Chip controller based on netX processor [Kon09]

Such communication system allows using of important technique in the robotic area as hybrid simulation technique. Starting with the SiL simulation, where the developed control system in Matlab/Simulink has the possibility to communicate with the real legged robot, the control system has acquired a new level so-called Rapid Control Prototyping.

4.2.1 Current Control Loop

The current control loop is shown in Figure 4.15. Current control loop is implemented in the netX/xPEC. The current control is calculated synchronously with the PWM module with 28 kHz cycle. The control signal (reference current value) and the sensors signals (motor position, velocity and so on) are exchanged between netX processor and external

control PC (with xPC-Target real-time OS) using an EtherCAT communication protocol. Sensors signals are used as the feedback signals for the position/velocity control loops. The sample time of the position/velocity control loops depends on the power of the control PC. In our case (3GHz processor from Intel) the sample time is equal 1ms and can be decrease to 0,5ms.

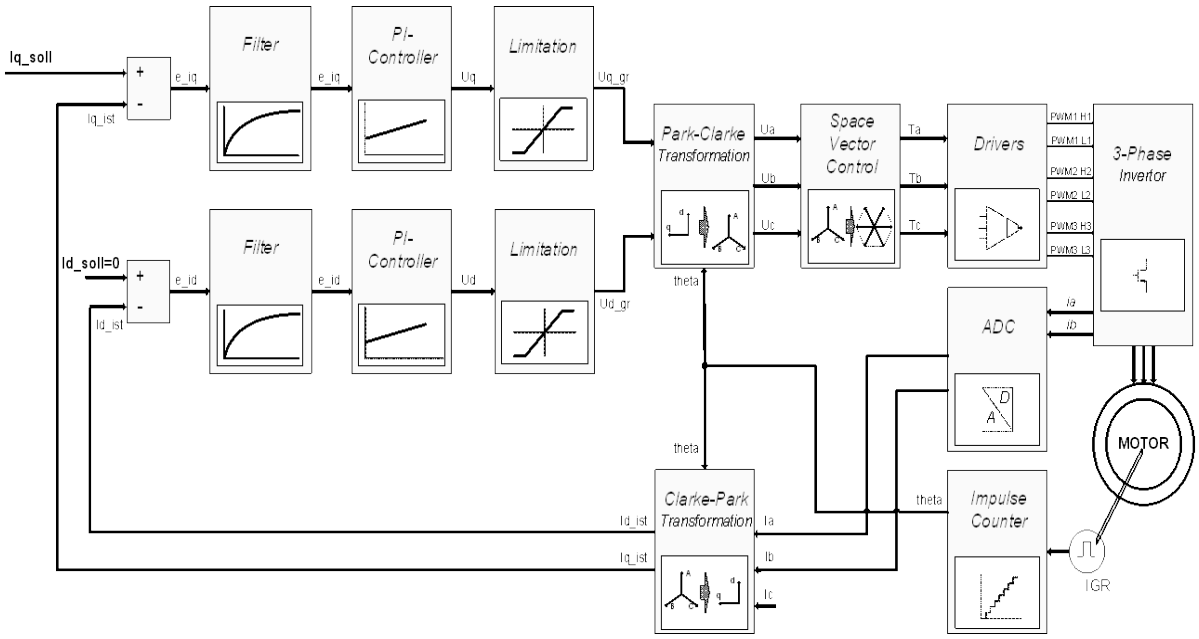

Figure 4.15 – Current control loop in the netX/xPEC [Kon09]

Phase currents (ia_ist, ib_ist) are measured in the netX by the way of sampling of intern ADC module and are recalculated from coordinate system ABC in coordinate system DQ using Park-Clarke transformation:

$$i_D = \frac{3}{2}\left[i_A \sin(\varphi) + i_B \sin\left(\varphi - \frac{2\pi}{3}\right) + i_C \sin\left(\varphi + \frac{2\pi}{3}\right)\right]$$
$$i_Q = \frac{3}{2}\left[i_A \cos(\varphi) + i_B \cos\left(\varphi - \frac{2\pi}{3}\right) + i_C \cos\left(\varphi + \frac{2\pi}{3}\right)\right] \quad (4.1)$$

The error between reference and real current is controlled by the PI controller:

$$\frac{U}{e_i} = P_{Cont} + I_{Cont} \cdot \frac{1}{1 - z^{-1}} \quad (4.2)$$

The voltage is transformed from the coordinate system DQ in coordinate system ABC using the inverse Clarke-Parke transformation:

$$
\begin{aligned}
U_A &= i_D \sin(\varphi) + i_Q \cos(\varphi) \\
U_B &= i_D \sin\left(\varphi - \frac{2\pi}{3}\right) + i_Q \cos\left(\varphi - \frac{2\pi}{3}\right) \\
i_C &= i_D \sin\left(\varphi + \frac{2\pi}{3}\right) + i_Q \cos\left(\varphi + \frac{2\pi}{3}\right)
\end{aligned}
\tag{4.3}
$$

4.2.2 Velocity/Position Control Loops

The velocity and position control loops are developed in Matlab/Simulink and are compiled to the control PC using Real-Time-Workshop-Tools [math]. The velocity and the position control loops are shown in the Figure 4.16.

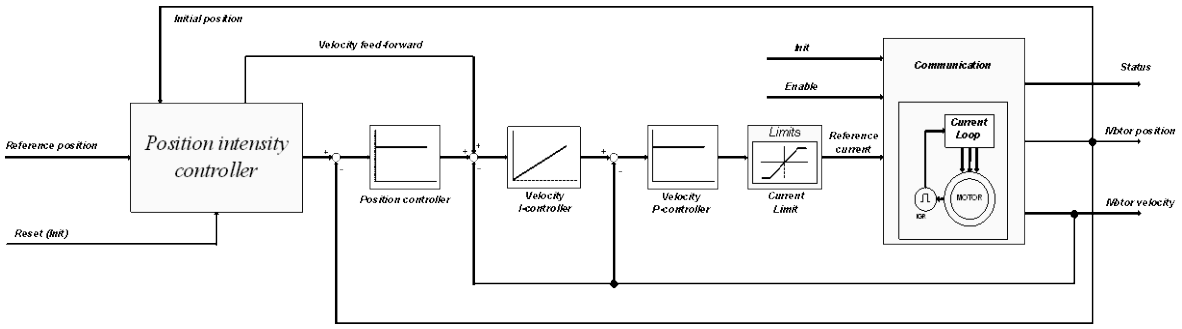

Figure 4.16 – Position/velocity control loops [Kon09]

The accuracy and the correctness of the Permanent Magnet Synchronous Motor (PMSM) control depend on the correct determination of the rotor position. Therefore following algorithm has been developed to determine the exact position of the rotor:

1. reference currents are id_ref = 5A, iq_ref = 0A for rotor positioning along the flux axis D;
2. one second pause for the exact positioning;
3. reset of the IGR's counter;
4. start the current loop, in other words wait for the reference value of iq_ref, the reference value of id_ref is equal 0A.

To ensure correct working of the position control loop when a step reference signals is applied a second order position intensity controller (Figure 4.17) has been developed.

The experimental results of position/velocity control loops as well as current control loop are shown in Figure 4.18. Experiments have been carried out on the linear drive without any loads. The position reference signal has been applied as the step signal.

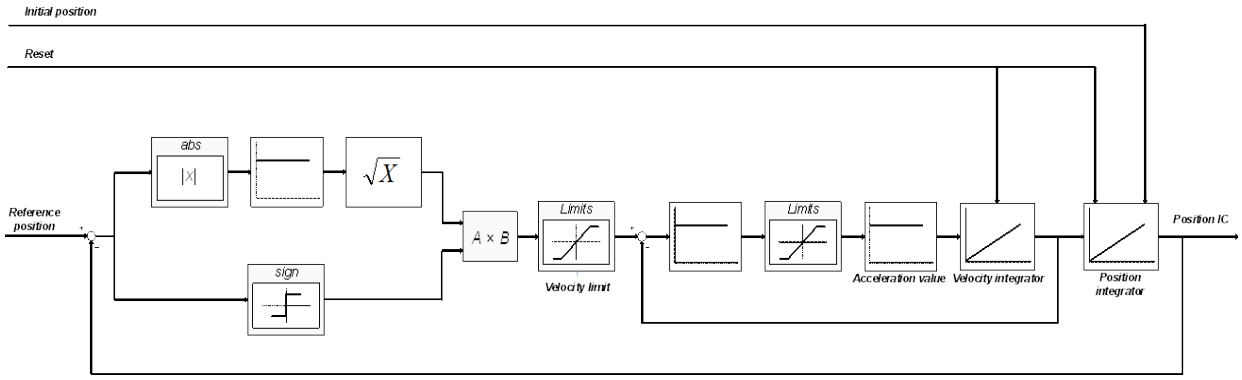

Figure 4.17 – Position intensity controller [Kon09]

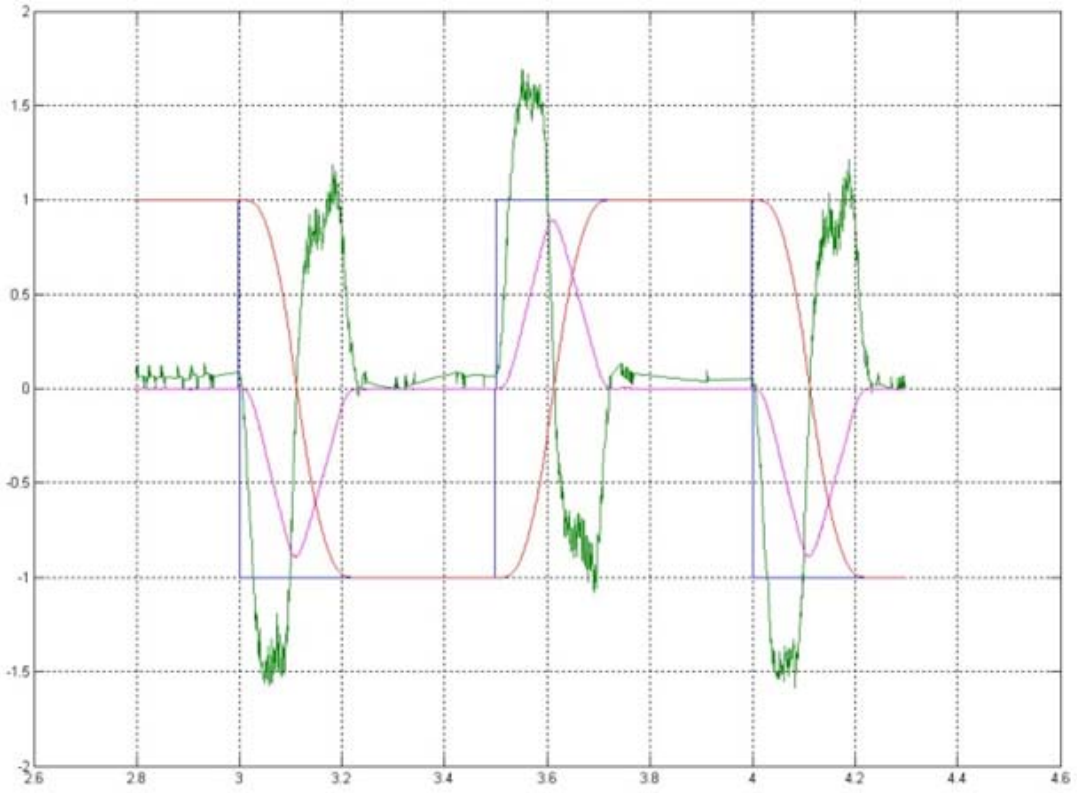

Figure 4.18 – Experimental results: blue – reference position (±10 revolutions); red – actual position (±10 revolutions); magenta – velocity (±13000 rpm); green – current (±3,6A) [Kon09]

4.2.3 Force/Position Control Loops of Modified Actor

The structure of the force control system is shown in Figure 4.19.

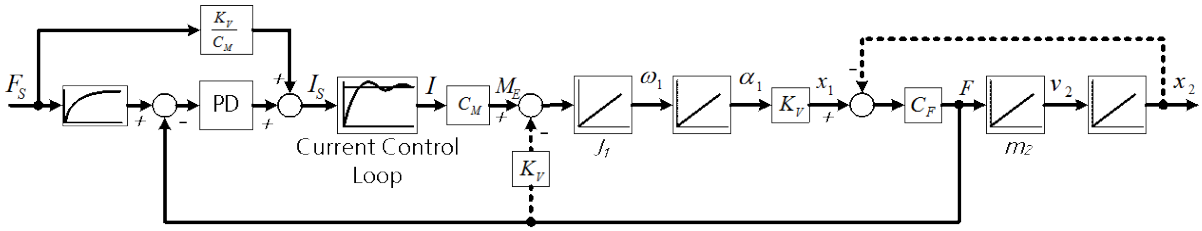

Figure 4.19 – Structural scheme of the force control system, where CF – the coefficient of the spring rigidity (spring constant), KV – coefficient of the conversion of the rotational motion into the translational motion, CM – coefficient of the conversion from current to mechanical torque [Kon102]

The force control system has a subordinate structure. There is only one inner control loop – the current control loop. Such approach allows reaching the highest dynamical characteristics of the force control loop. Some successful attempts of using such structure are shown in the works of MIT-University [MIT].

The force control loop is controlled by the PD-regulator. The dotted feedbacks in Figure 4.19 are not taken into account when designing the PD-controller. This neglect can be made because of the high dynamics of the force control circuit. These feedbacks are considered as a disturbances and don't cause any instabilities in the system. The step response of the force control loop is shown in Figure 4.21. The first sequence is 15 ms by the bandwidth of the force control loop of 50 Hz.

The position control system is shown in Figure 4.20. The above considered force control loop is the inner loop of the position control system.

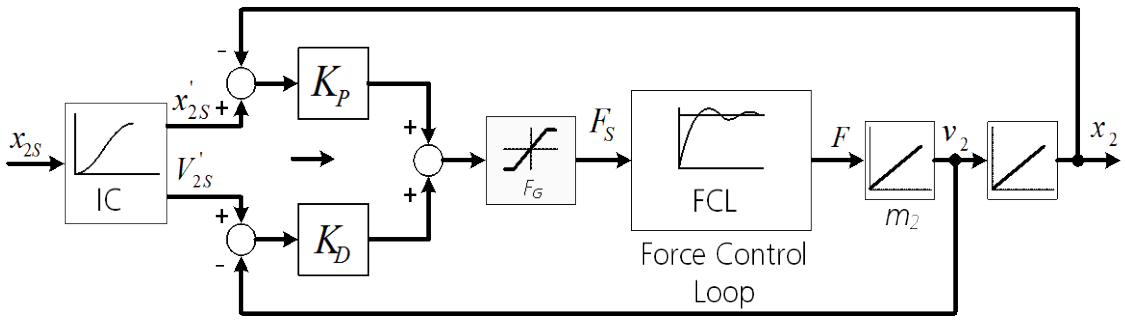

Figure 4.20 – The position control loop [Kon102]

The coefficient KD determines the system damping. The coefficient KP and the value FG are the most interesting because their variation allows the achieving the specific properties of the position control system when getting into contact with environment. For example, by setting a small value of coefficient KP the behaviour of the system in contact is elastic (low impedance, artificial spring). Another task – a precise position control with force saturation in the contact point – can be achieved by setting of higher value of coefficient KP and a given value of FG. Such case is illustrated in Figure 4.22, where the end-position is not reached as a result of an impact with an obstacle.

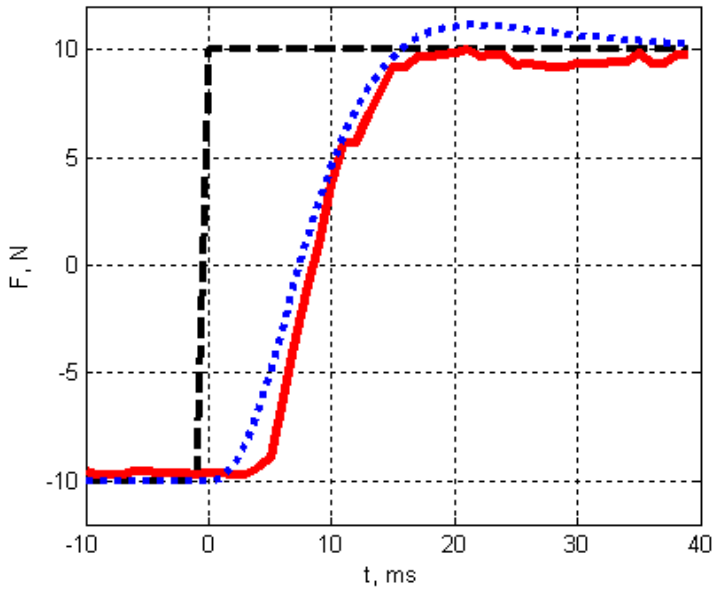

Figure 4.21 – Step response of the force control loop: dotted line – modelling; solid line – experiment [Kon102]

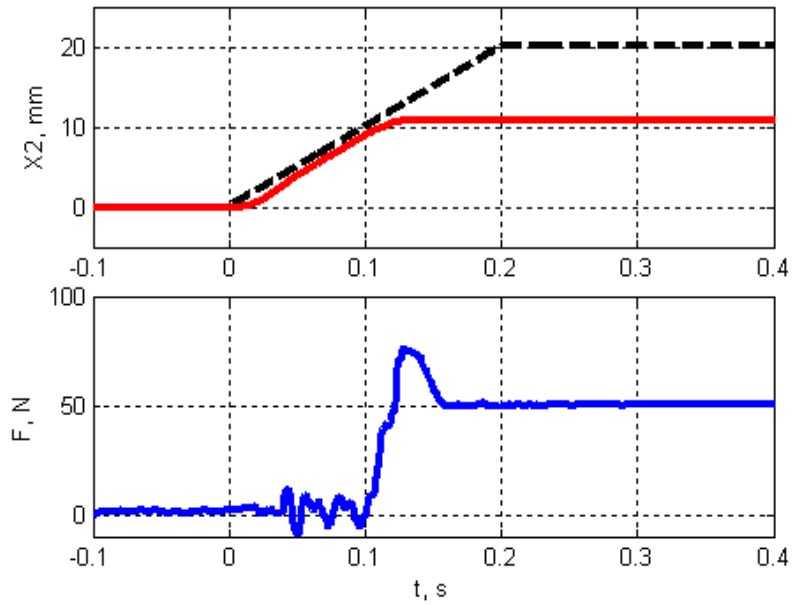

Figure 4.22 – Impact with an obstacle by position control: dotted line – desired position; red line – actual position; blue line – force in the contact [Kon102]

4.3 Software Control System

The software control system of biped robot "ROTTO" is based on the same philosophy as well as on the same hierarchically organized control system as by six-legged robot "ANTON" (Section 3.4). Figure 4.23 shows the disposition of different coordinate systems which are used in the control algorithms for the biped legged robot "ROTTO".

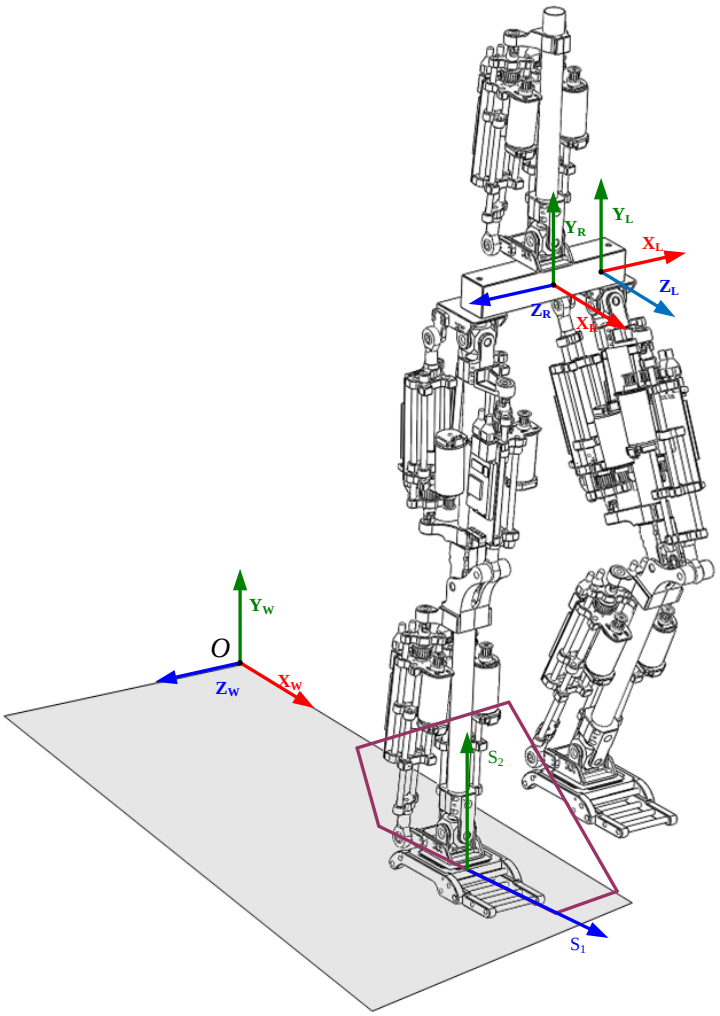

Figure 4.23 – Coordinate systems in the control algorithms

Coordinate system $\{X_w; Y_w; Z_w\}$ describes the world coordinate system (WorldCS) which is a main coordinate system for all algorithms. Axis Y_w is placed along the gravitation vector and the axis Z_w is placed along the straight walking of the robot. Second coordinate system is a robot coordinate system (RobotCS). This one is connected with a pelvis of the robot and describes the clearance (translation along axis Y between WorldCS and RobotCS) and inclinations (rotation relative axes Z und X between WorldCS and

RobotCS) of the robot. Next coordinate system is a leg one (LegCS). It plays the role of detachment of two legs from one body. In other worlds it presents the coordinate system of each leg of the robot and is connected with first joint in each leg. Axis Z_L is always perpendicular to the straight motion of the robot. The step cycle geometry has same structure as well as adaptation possibilities as by six-legged robot "ROTTO" (see Section 3.4.2).

The most important part in software control system of biped robot is a stabilization of COM or ZMP (Zero-Moment-Point) inside the supporting polygon. Algorithm of computation of position of COM has the same ideology as in Sections 3.4.4 and 3.4.6. Control of position of COM is shown on Figure 4.24 and is explained in Section 3.4.4. Obtaining of ZMP is based on information from force sensors and is detailed discussed in [Zav08].

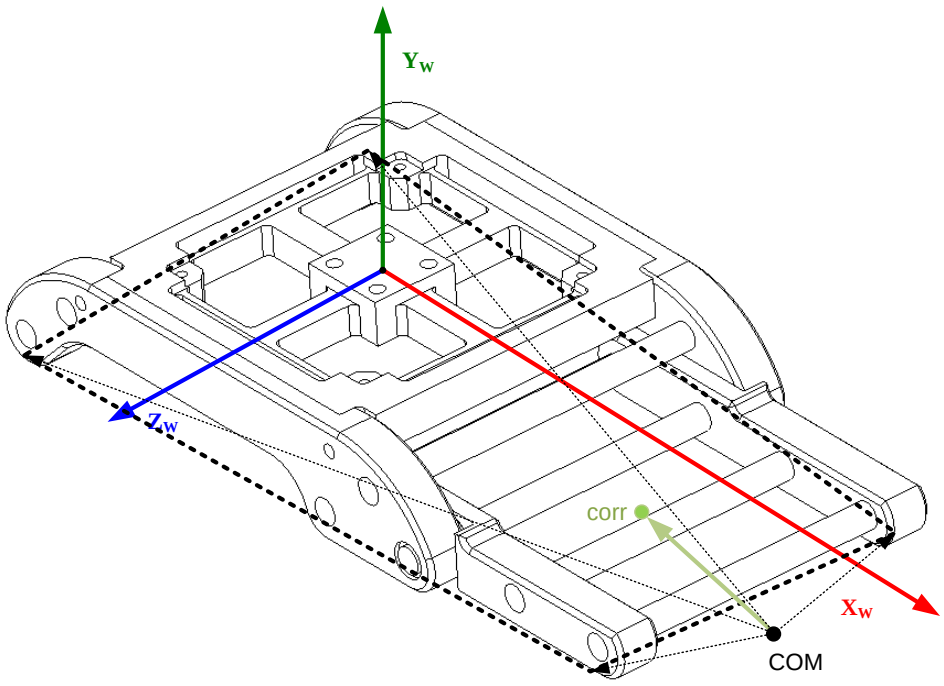

Figure 4.24 – Control of COM of robot

4.4 Conclusion

New biped robot "ROTTO"'s construction as well as the sensor system have been successfully developed. The numbers of requirements such as construction's modularity, robot's expansibility, easy manufacturing as well as the numbers of control requirements have been performed. The robot has enough sensors on board, provides fast and safe

communication between the robot electronics and the control computer and is able to solve various locomotion as well as the intelligent tasks.

The intelligent decentralized linear drive system was developed and introduced. It consists of velocity and position controlled brushless EC-motor with ball screw and belt transmission. Force sensors allow the implementation of a force/position control loop. Due to this feature mechanical impedance can be controlled in a wide range.

The control electronics as well as the all control architecture and algorithms for the biped robot "ROTTO" have been developed and showed satisfactory results. The control electronic consists of two boards: the small one netX SODIMM module, and the second one with power electronics. The control algorithms as well as communication between on-board electronics and external control PC have been successfully tested on a large number of experiments as well as by the static walking of biped robot "ROTTO".

Chapter 5

Conclusions (German Version)

Die vorliegende Arbeit wurde der Konstruktion, Simulation und Steuerung von mobilen Schreitrobotern gewidmet.

Für die Untersuchung des statischen/dynamischen Verhaltens von Robotern wurden Simulationsmodelle unterschiedlicher Komplexität entwickelt. Dafür ist die Entwicklung von virtueller Simulationsumgebung notwendig. Das Ziel des Vorhabens bestand in der Erarbeitung eines neuen Konzepts zur Einbindung von mathematischen Modellen für Kontaktaufgaben in Tools der Multi-Körper-Simulation (MKS) als Teil eines domänenübergreifendes Simulationstools. Die wichtigste Teilaufgabe bestand in der Konzipierung einer durchgängigen Lösung zur einheitlichen Kontaktbehandlung in MKS-Tools (Ableitung der kinematischen und dynamischen Systemparameter, Simulation kontaktierender Objekte, Kopplung an CAE-Tools, Anwendungsmöglichkeiten in Soft-/ Hardware-in-the-Loop Tests, online Visualisierung der Ergebnisse), welche eine breite Palette von Kontaktaufgaben von einfachsten bis zu komplizierten Modellen berücksichtigt. In der vorliegenden Arbeit wurden Algorithmen der Kollisionserkennung und der Kontaktkraftberechnung betrachtet, die auf Tiefeindringung zwischen zwei oder mehrere Objekten basieren. Im Ergebnis wurde eine ContactProcessing-Bibliothek entwickelt und auf Basis

der Simulation eines sechsbeinigen mobilen Roboters als auch von eines zweibeinigen mobilen Roboters bewiesen.

Ein hierarchisches Steuerungssystem für einen sechsbeinigen mobilen Roboter als auch für einen zweibeinigen mobilen Roboter wurde auf Basis entwickelter Simulationsumgebung vorgestellt. Weiterhin wurde eine technische Lösung zum Aufbau des Steuerungssystems, das auf den entwickelten vernetzten Antriebseinheiten basiert, präsentiert. Zu den wichtigsten Eigenschaften des entwickelten Konzepts gehören hohe Flexibilität und Erweiterbarkeit sowie leistungsfähiges Kommunikationssystem. Die Verwendung des leistungsfähigen Kommunikationssystems ermöglicht die Realisierung neuer Regelungskonzepte, wobei alle Teile des Steuerungs- bzw. Regelungssystems auf dem Zentralrechner platziert ist und die Regelungsschleifen komplett über den Bus mit der Zykluszeit von bis zu 100us geschlossen werden können.

Die in vorliegender Arbeit dargestellten Konzepte und Verfahren wurden simulativ und praktisch untersucht. Die entwickelten Prototypen des sechsbeinigen bzw. zweibeinigen mobilen Roboters stellen somit eine solide Testplattform für verschiedene Steuerungs- und Regelungsstrategien dar. Aus funktioneller Sicht ist zu erwähnen, dass die entworfenen Roboterkonstruktionen sich durch einfache aber dennoch für Lokomotionsaufgaben ausreichende Kinematiken mit optimalen Verhältnissen und größeren Arbeitsbereichen auszeichnen.

Chapter 6

Bibliography

[Ale98] Alexandre, P., Doroftei, I. and Preumont, A. 1998. An autonomous micro walking ma-chine with articulated body. *CLAWAR'98, Brussels.* 1998.

[Arm95] Armstrong, B. and Wit, C.C. de. 1995. Friction Modeling and Compensation. *The Control Handbook, CRC Press.* 1995.

[Gva03] Bergen, G. van den. 2003. Collision Detection in Interaction 3D Environments. 2003.

[Bra99] Brazevic, P., et al. 1999. Development of multi-legged walking robot with articulated body. *CLAWAR'99, Portsmouth, UK.* 1999.

[Chu96] Chung, K. and Wang, W. 1996. Quick Collision Detection of Polytopes in Virtual Environments. *ACM Symposium on Virtual Reality Software and Technology 1996, 1-4 July, University of Hong Kong.* 1996.

[Bos] Dynamics, Boston. http://www.bostondynamics.com/bd_index.html. [Online]

[Elm99] Elmqvicst, H., Mattsson, S.E. and Otter, M. 1999. Modelica - A Language for Physical System Modelling, Visualization and Iteraction. *Proc. of the 1999 IEEE Symposium on Computer-Aided Control System Design.* 1999.

[Eng00] Engelson, V. 2000. Integration of Collision Detection with the Multibody System Library in Modelica. *PhD Thesis, PELAB, IDA, Linkoping University.* 2000.

[Eri99] Erickson, J., et al. 1999. Separation-sensitive collision detection for convex objects. *Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms.* 1999.

[Fri04] Fritzon, P. 2004. Principles of Object-Oriented Modeling and Simulation with Modelica 2.1. *John Wiley and Sons.* 2004.

[Fri97] Fritzson, D., et al. 1997. Rolling Bearing Simulation on MIMD Computers. *Int. Journal of Supercomp. Appl. and High Performance Computing.* 1997, 11.

[Gal06] Galoppo, N., et al. 2006. Fast Simulation of Deformable Models in Contact Using Dynamic Deformation Testures. *ACM SIGGRAPH Symposium on Computer Animation.* 2006.

[Gaß01] Gaßmann, B., Scholl, K.-U., Berns, K. 2001. Behavior Control of LAURON III for Walking in Unstructured Terr. *In Proc. of Int. Conference on Climbing and Walking Robots CLAWAR01.* 2001.

[Gol03] Golubev, Y.F. and Korianov, V.V. 2003. Motion design for six-legged robot overcom-ing the vertical column by means of friction forces. *CLAWAR'03, Catania, Italy.* 2003.

[Gor00] Gorinevsky, D., Formalsky, A. and Schneider, A. 2000. Force Control of robotics Systems. *CRC Press.* 2000.

[Gor90] Gorinevsky, D.M. and Schneider, A.Yu. 1990. Force control in locomotion of legged vehicle over rigid and soft surfaces. *Int. J. Robot. Res., 9(2).* 1990.

[math] http://www.mathworks.com/products/rtwembedded. [Online]

[Joh] Johnnie. http://www.amm.mw.tum.de/index.php?id=182&L=1. [Online]

[Juh08] Juhasz, T. 2008. Advanced Solutions in Object-Oriented Mechatronic Silulation. *Ph.D. Thesis, Budapest University of Technology and Economics.* 2008.

[Kat72] Kato, I., Tsuiki, H. 1972. The hydraulically powered biped walking machine with a high carrying capacity. *In Proc. of the 4th Int. Symposium on External Control.* 1972.

[Kon051] Konyev, M. and Palis, F. 2005. Using FIS as a Function Approximator of a State-Action Table. *Summer School, Sozopol.* 2005, s.49-60.

[Kon081] Konyev, M., Palis, F., et al. 2008. Development and Test of the Walking Robot "ANTON". *Vestnik NTU "KPI".* 2008, s.57-58.

[Kon05] Konyev, M., Palis, F., Guilou, X. and Glorennec, P. 2005. Fuzzy Logic and Reinforcement Learning. *Vestnik NTU "KPI".* 2005.

[Kon092] Konyev, M., Palis, F., Melnikov, A. and Rudskiy, A. 2009. Linear Drive with Serial Elastics for Anthropomorphous Robot "ROTTO" (on russian). *Vestnik NTU "KPI".* 2009.

[Kon101] Konyev, M., Palis, F., Melnykov, A. and Rudskiy, A. 2010. Biped Robot "ROTTO": Stiff and Compliant. *RAAD.* 2010, s.261-266.

[Kon072] Konyev, M., Palis, F., Rusin, V. and Zavgorodniy, Y. 2007. Control System of Six Legged Autonomous Intelligent Robot. *ROBOTICA.* 2007.

[Kon061] Konyev, M., Palis, F., Rusin, V. and Zavgorodniy, Y. 2006. Realization of Improved Locomotion Tasks for Multi-Legged Robot. *Thematic Issue ONPU.* 2006.

[Kon06] Konyev, M., Palis, F., Rusin, V., Zavgorodniy, Y. 2006. SLAIR 2 – Six Legged Autonomous Intelligent Robot. *Thematic Issue ONPU.* 2006.

[Kon10] Konyev, M., Palis, F., Schmucker, U. and Melnykov, A. 2010. Biped Robot "ROTTO": Design, Simulation, Experiments. *ISR-ROBOTIC.* 2010, s.780-785.

[Kon09] Konyev, M., Palis, F., Schmucker, U., et al. 2009. Low-Level Control System of a New Biped Robot "ROTTO". *CLAWAR.* 2009, s.559-566.

[Kon091] Konyev, M., Palis, F., Schmucker, U., et al. 2009. Presentation of a New Biped Robot "ROTTO". *CLAWAR.* 2009, s.551-558.

[Kon102] Konyev, M., Palis, F., Schmucker, U., Melnykov, A. and Rudskiy, A. 2010. Linear Elastic Actuator of a Biped Robot "ROTTO". *CLAWAR.* 2010, s.588-595.

[Kon082] Konyev, M., Palis, F., Zavgorodniy, Y., et al. 2008. Linear Drive for Anthropomorphous Robot "ROTTO" (on russian). *Vestnik NTU "KPI".* 2008, s.55-56.

[Kon08] Konyev, M., Palis, F., Zavgorodniy, Y., Melnikov, A., Rudskiy, A. and Telesh, A. 2008. Walking Robot "ANTON": Design, Simulation, Experiments. *CLAWAR.* 2008, s.922-929.

[Kon071] Konyev, M., Schmucker, U. and Rusin, V. 2007. Contact Processing in the Simulation of the Multi-Body Systems. *EUROSIM.* 2007.

[Kon07] Konyev, M., Schmucker, U., Juhasz, T. and Rusin, V. 2007. Contact Processing in the Simulation of CLAWAR. *CLAWAR.* 2007.

[Lin92] Lin, M.C. 1992. Efficient Collision Detection for Animation and Robotics. *Ph.D. thesis, University of California, Berkeley.* 1992.

[LOL] LOLA. http://www.amm.mw.tum.de/index.php?id=183&L=1. [Online]

[Mir96] Mirtich, B. 1996. Impulse-based Dynamic Simulation of Rigid Body Systems. *PhD Thesis, University of California, Berkeley.* 1996.

[MIT] MIT. http://www.ai.mit.edu/projects/leglab/robots/robots.html. [Online]

[Okh84] Okhotsimsky, D.E. and Golubev, Yu.F. 1984. Mechanics and motion control of an au-tomated walking vehicle. *Published House „Nauka", Moscow.* 1984.

[Ott04] Otter, M., Elmqvist, H. and Lopez, J.Diaz. 2004. Collision Handling for the Modelica MultiBody Library. *4th International Modelica Conference.* 2004.

[Pal07] Palis, Dzhantimirov, Schmucker, Zavgorodniy, Telesh. 2007. HIL/SIL by development of six-legged robot SLAIR 2. *10th Int. Conference on CLAWAR.* 2007.

[Pal04] Palis, F., et al. 2004. Legged Robot with Articulated Body in Locomotion over Complex Terrain. *CLAWAR.* 2004.

[Pan90] Pang, J.S. 1990. Newton's method for B-differentiable equations. *Mathematics of Operations Research, Vol. 15/2, May.* 1990.

[Pfe00] Pfeiffer, F. 2000. Refined control for the tube crawling robot- Moritz. *CLAWAR'02, Madrid, Spain.* 2000.

[Rus07] Rusin, V. 2007. *Adaptive Regelung von Robotersystemen in Kontaktaufgaben.* Magdeburg : s.n., 2007.

[Sch00] Schilling, R. 2000. *Fundamentals of Robotics.* 2000.

[Sch96] Schmalstieg, D. and Tobler, R.F. 1996. Real-time Bounding Box Area Computation. *Institute of Computer Graphics, Vienna University of Technology.* 1996.

[Sch06] Schneider, A. and Schmucker, U. 2006. Force Sensing for Multi-legged Walking Robots: Theory and Experiments – Part 1: Overview and Force Sensing. *Mobile Robots.* 2006.

[Sci00] Sciavicco, L. and Siciliano, B. 2000. *Modelling and Control of Robot Manipulatiors.* 2000.

[SCO] SCORPION. http://robotik.dfki-bremen.de/en/research/projects/space-robotics/scorpion.html. [Online]

[Son89] Song, Shin-Min and Waldron, K.J. 1989. Machines that walk. *MIT Press Cambridge, Massachusetts.* 1989.

[Was] Waseda. http://www.humanoid.waseda.ac.jp/booklet/kato_4.html. [Online]

[Wik] Wikipedia. [Online] http://en.wikipedia.org/wiki/Dynamic_simulation.

[Wik01] WikipediA. [Online] http://en.wikipedia.org/wiki/Multibody_system.

[Wik2] Wikipedia. http://en.wikipedia.org/wiki/ASIMO. [Online]

[Wik1] —. http://en.wikipedia.org/wiki/LAURON. [Online]

[Zav08] Zavgorodniy, Y. 2008. Konstruktion und Steuerung von Schreitrobotern mit ballistischem Laufverhalten. 2008.

[Zha96] Zhang, P. 1996. Physically Realistic Simulation of Rigid Bodies. *Thesis, Department of Computer Science, Tulane University.* 1996.