Darmstadt Dribblers 2005: Humanoid Robot

Martin Friedmann, Jutta Kiener, Robert Kratz, Tobias Ludwig, Sebastian Petters, Maximilian Stelzer, Oskar von Stryk, and Dirk Thomas

Simulation and Systems Optimization Group
Department of Computer Science
Technische Universität Darmstadt, Germany
{friedmann, kiener, kratz, ludwig, petters, stelzer, stryk, thomas}@sim.tu-darmstadt.de,
WWW homepage: http://robocup.informatik.tu-darmstadt.de/humanoid

Abstract. This paper describes the hardware and software design of the two humanoid robot systems of the Darmstadt Dribblers. The robots are used as a vehicle for research in control of locomotion and behavior of humanoid robots with many degrees of freedom and many actuated joints. The Humanoid League of RoboCup provides an ideal testbed for such aspects of dynamics in motion and autonomous behavior as the problem of generating and maintaining statically or dynamically stable bipedal locomotion is predominant for all types of motions during a soccer game. A new modular software architecture has been developed for efficient and effective implementation and test of modules for sensing, planning, behavior, and actions.

1 RoboCup and aspects of robot motion and behavior

The RoboCup scenario of soccer playing legged robots represents an extraordinary challenge for the design, control and stability of bipedal and quadrupedal robots. In a game, fast motions must be planned autonomously and implemented online which preserve the robot's stability and can be adapted in real-time to the quickly changing environment. Existing design and control strategies for humanoid robots can only meet these challenges to some extent.

During the nineties, both trajectory planning methods relying on nonlinear robot dynamics and model-based control methods have evolved into the state-of-the-art for developing and implementing fast and accurate motions for industrial manipulators. Successful control of the nonlinear robot dynamics is also the key to fast and stable motions of bipedal and quadrupedal robots. Many subproblems remain unsolved in fulfilling this objective. One aim of the Darmstadt Dribblers is to contribute to this ambitious goal by discussing fundamental principles and recent methods in the modeling, simulation, optimization and control of legged robot dynamics. Further aspects on these topics are described in [4].

For developing and testing the different modules involved in an autonomous behavior as required in the RoboCup scenario a tailored software architecture is essential. The underlying software and robot control architecture has to accomplish the demands appearing in a highly nonlinear physical dynamical system as a humanoid robot.

A definition of the current RoboCup environment is described in the rules for Humanoid League [8].

2 Technical Data of the Humanoid Robots

Currently two different types of humanoid robots are used, which differ mainly in height: Mr. DD and Mr. DD junior, see Fig. 1. Mr. DD was first used in RoboCup 2004. It is based on a unique prototype of a biped walking machine which has been custom-made for our purposes by iXs Research Corporation (http://www.ixs.co.jp). Due to several hardware and software modifications the walking motion could be improved and some autonomous behavior was developed.. Mr. DD junior is based on the robot kit KHR-1 by Kondo (http://kondo-robot.com). Both of them have legs with six degrees of freedom (DOF) and are equipped additional with a camera system to sense the environment. They have nearly the same kinematic structure. In Fig. 1 the arrangement of joints is presented for Mr. DD. The small robot lacks the waist joints and one of the neck joints. For more information about hardware modification see section 4. The technical data are given in Tab. 1.

It is planned to add more platforms for humanoid robots in 2005 and 2006.

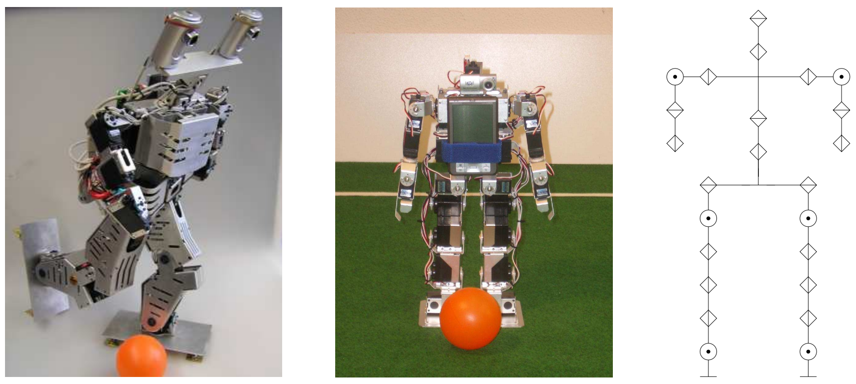

Fig. 1. Humanoid robot systems Mr. DD (left) and Mr. DD junior (middle) in real life and kinematic structure of Mr. DD (right).

3 Global software concept and modular software structure

To establish a communication between the different software modules of an autonomous robot, an object-oriented framework has been implemented. The main focus of the framework is to simplify the realization of a suitable robot-control-application for a programmer. The developer should not worry about the different operating systems present

Robot system:	Mr. DD	Mr. DD junior
Height:	68 cm	37.5 cm
Width:	31 cm	18.5 cm
Weight:	4.8 kg	1.5 kg
Degrees of freedom:	24 in total with 6 in each leg, 4 in each arm 2 in waist, 2 in neck	21 in total with 6 in each leg, 4 in each arm, 1 in neck
Sensors:	1 camera (Philips, ToUCam, resolution: 160 x 120) 24 joint angle encoders, 3 force sensors in each foot	1 camera (HewlettPackard, resolution: 160 x 120)
Control frequency:	20 ms	40 ms
Processor:	NEC Vr4181A 133 MHz	Intel PXA 255, 400 MHz
Operating system:	Linux	Windows CE
Network:	Wireless LAN, LAN	Wireless LAN, Bluetooth
Power supply:	on-board batteries	on-board batteries

Table 1. Technical data of both humanoid robots.

in our robots (Linux, Windows) just as little as about threads and communication, but concentrate very exclusively upon the features and actions which should be carried out.

Therefore one regards a highly simplified example of a standard robotic control as explained in Fig. 2. The rectangles mark here the functional components. The ellipses in contrast represent the exchanged information.

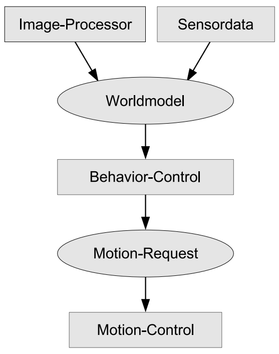

Fig. 2. Example of simplified robotic control

These functional components, which accomplish the data processing steps, are called modules in the further process. The goal of the framework is to handle the inter-module communication completely transparent for the modules. For developers it should be of no importance, whether another module, with which it exchanges data, runs in the same thread or on another computer. This gives the flexibility to relocate single modules, e.g. from the robot’s on-board computer to an external computer, without any relevance for the implementation of the modules.

Furthermore this approach allows for an easy exchange of data between different robots acting in a team.

3.1 Framework

To achieve this objective the framework must meet several requirements. On the one hand, during the programming-phase of the framework any unknown data structures for the exchange of information must be converted into a uniform format and backwards. In C++ this must be made available by hand, because it is not inherent possible as in other programming languages like Java. On the other hand, the framework must know about every module, which data are needed and are provided, and based on these information handle the communication between all modules automatically. Therefore the communication management is not hard-coded centrally, but is specified by the data-processing classes themselves and hence the necessary communication channels are generated during runtime. The central component of the framework is the so called *router*. With each started instance of the software exactly one routing class is responsible for the entire communication handling.

Each module (or group of modules running consecutively) is started by the router to run in it’s own thread of execution. The combination of one router, the threads started by this router and the modules running within these threads is called an *instance*. Besides starting modules the router is responsible for message-handling and -exchange between modules. Communication is possible between modules running in the same thread of execution, modules in different threads of the same instance as well as between modules running on different instances, no matter if they are on the same machine or on different machines connected by a network.

For clarification the previous example is shown in Fig. 3 in more detail. The modules ’image-processing’ and ’behavior-control’ were combined in one process. This is reasonable for example if the execution of behavior-control is necessary only after the generation of new images.

Further more these two modules run in another process than the module for actuating the joints, making it possible to trigger both processes with different frequencies, to fit the different requirements of the modules. In this example all modules needed for autonomous action of the robot run on the same instance, a second instance is shown which might execute modules necessary for a GUI (which might be executed on another machine).

For testing purposes a debug log system is implemented which offers debug messages of selected modules with user defined fineness of the warning content.

3.2 GUI

Besides executing the modules required for robot-control the framework must provide the possibility to visualize data structures from the robot in a graphical application and

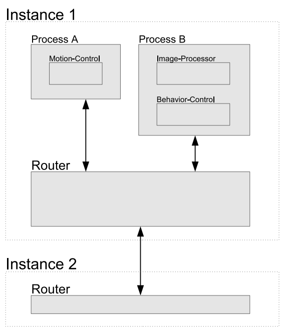

Fig. 3. Components of an instance

to be able to send back information to the robot. Like the main part of the framework the GUI, too, is intended to be platform independent. Hence, the graphical user interface is based on QT [9] – a multi-platform C++ GUI Toolkit.

Because some instances, e.g. on the robot, do not even need a graphical user interface this is realized as an extension based on the framework. Thus the flexible communication management is also an integral component of the GUI.

To be able to handle new data structures, the GUI can be extended with dialogues for new data-structures. One dialog can show several different data structures, as well as communicate at the same time with arbitrarily many modules (sending or receiving the data-structures) on any number of robots.

3.3 Current modules

At the moment both robots use mainly three interacting modules developed with the framework described above: vision, motion and behavior. In the near future further modules such as for world modeling and self localization will be developed and made available to enable participation in the two-by-two humanoid soccer games.

Vision. The communication with the camera and all image processing takes place in the vision module. The cameras of the two different robot types use different color spaces, Mr. DD uses YUV, Mr. DD junior RGB. Therefore different image-segmentation algorithms (and thus different modules) are used for both robots, nevertheless both modules generate the same kind of message describing a segmented image. Recognition of objects (ball, goal, . . .) is done by the same algorithm (and thus the same module) on both robots.

Motion. The current motion module is mainly used to calculate walking trajectories using an inverse kinematics model and to control the neck joint(s) with 1 or 2 DOF depending on the robot type. The control of the other joints in the arms is also possible, but not primary necessary for walking towards and kicking a ball.

For the walking the inverse kinematics engine takes three 6-dimensional reference points for pose (one in each foot, one in the hip) at a given time and calculates a 6-dimensional vector of joint angles for each leg. The engine can be fine tuned by several parameters (e.g. different length and time variations during one stride) which can be altered at runtime.

Behavior. The data provided by the vision module is used to plan a more complex behavior such as playing soccer. The main task is separated into subtasks until they can be described as a set of atomic actions. These are transfered to and interpreted by the motion module.

It is planned to add the capability of team member communications to the existing state machine similar to other leagues [7].

4 Hardware modifications

Both humanoid robots have and are still being modified in hardware to achieve better results in walking and environment recognition. The modifications include among others additional sensors, different order of joints and specially designed feet.

4.1 Mr. DD

The original robot platform is a prototype of iXs Research Corp., Japan, which delivered as an assembled hardware of servos including a control board, but without any humanoid robot abilities such as walking or vision.

Foot design. The interaction of a robot with the environment happens with its feet. Obviously unevenness in the ground influences the robot's balance. The goal of the research is a foot construction which adapts to small unevenness automatically. In Fig. 4 the foot consists of a base plate (a) with two springs (b) fixed to the base plate in the center. Each spring has two swiveling brass plates (c) to get a defined ground contact which is measured by one force sensor film (e) at each ground plate. To rise the friction of the brass plate, foam rubber (f) is fastened. To prevent the twist of the springs a spacer (h) is adjusted.

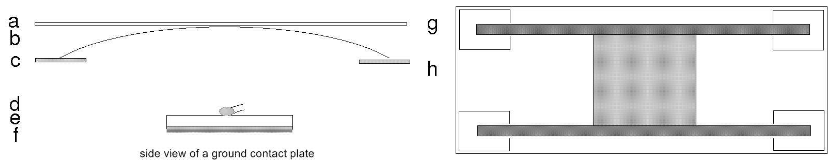

Fig. 4. Side and top view of the foot with base plate (a), spring (b), ground contact brass plate (c), joint (d), force sensor film (e), foam rubber (f), spring (g) and spacer (h).

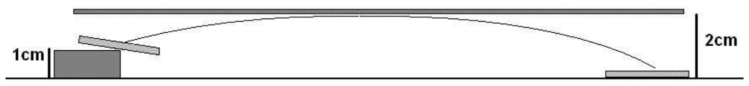

Fig. 5. Side view of the foot: uneven terrain may be compensated.

This foot design is able to compensate unevenness of about 10 mm without a tilt of the base plate, see Fig. 5. Because of the progressive elasticity coefficient only large obstacles cause a twist of the base plate. Therefore the robot maintains balance more easily in case of disturbances from the ground.

Distance sensors. To optimize the walking of the robot it is necessary to get information about the orientation of the upper part of the body. A simple and fast evaluation of the orientation of the upper body can be obtained by installation of four infrared sensors outwardly attached to the shoulders (Fig. 6).. With this method there is only a need of

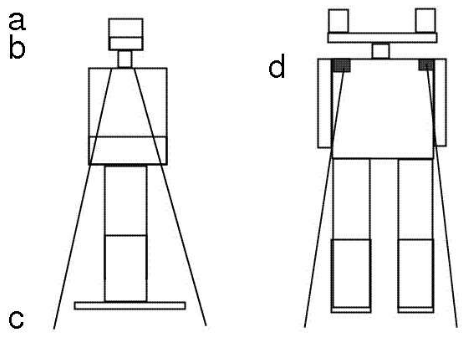

Fig. 6. Side (left) and front (right) view of the robot with distance sensors. Head (a), infrared sensors (b), ground (c), and infrared sensors (d).

few data to get informations about parts of the position and orientation of the upper body. If both front sensors measure a shortening the chest is tending to the ground. If the sensors on the right or left measure a reduction of the distance the upper body is tending to the side. With an exact adjustment of the sensor data all possible positions are measurable exactly. The problem of the method is the relatively data to the ground. This can be compensated by considering an inclination sensor at the feet.

4.2 Mr. DD junior

Leg design. The base kit of Kondo KHR-1 consists of servos for a robot with 5 DOF in each leg. One additional joint has been added in the hip to gain a six DOF leg to utilize the three dimensional space in position as well as in orientation for leg movements. The modification with the joint in hip considered the aspect of a more human like leg design (see Fig. 1).

On-board PC and camera. Originally the robot hardware is provided to store given joint angle trajectories on the servo boards via a serial connection since the robot's base version has no external sensors at all. To accomplish autonomous behavior a pocket PC was mounted to the robots chest. As the only external sensor a camera is mounted on the neck joint and connected to its SD card slot. The pocket PC uses the serial port of the robot to set its joint angles almost continuously in the frequency of 40 ms.

5 Generating single walking steps

We applied two different methods for generating gaits for humanoid robots: for the first one, we prescribe the movement of the hip and the feet by several parameters and solve for the joint angles using inverse kinematics; for the second one a full three dimensional model of the robot is involved into an optimal control problem which is solved by efficient numerical optimization methods. The two methods and their advantages and disadvantages shall be described in this chapter.

5.1 Generating gaits using kinematics model and inverse kinematics of the legs

With prescribed movements of hip and feet, the joint angles may be calculated using the inverse kinematics of the robot's feet. The trajectories of the hip and the feet respectively are parameterized by several parameters. Some of them influence both hip and feet trajectories. Inverse kinematics solutions to the gait generation problem are very fast but have the drawback of not taking into account the system's dynamics and stability. Stability and dynamics restrictions must be checked separately for each set of parameters.

5.2 Generating gaits using dynamics model and optimal control techniques

The problem of generating stable gaits involving the robots' dynamics may be formulated as an optimal control problem. It is much more laborious to get a solution but this is accepted for the sake of full dynamics and stability taken into account.

A short overview of the techniques used shall be given in this section. More details may be found in [6, 2, 5, 1].

Efficient dynamics modeling. Walking robots are characterized by a large number of actuated joints and subsequent changes in the kinematic structure due to the different contact situations that result in kinematic loops. With special treatment of the contacts the system may be solved with methods for tree structured systems. Articulated Body Algorithm, a recursive linear-order algorithm is used.

Optimal control techniques. An efficient direct collocation method is used for solving the optimal control problem of finding a gait that minimizes some merit function (time, energy) subject to the system of differential equations and linear and nonlinear boundary and inequality constraints.

Reduced dynamics. The multibody systems of differential equations must be extended to a system of differential algebraic equations when treating contact algebraic conditions explicitly, e.g. in the descriptor form. By not solving for the states related to legs in contact with the ground but calculating this states by known contact foot points and hip position using inverse kinematics of the legs a reduced system of ordinary differential equations may be treated in the optimal control problem.

Problem formulation of generating stable gaits. Besides the merit function and the systems of differential equations several conditions can be imposed for stability (COG, ZMP, FRI), foot placement and swing height, motor characteristics and proportion of single and double support phase.

6 Autonomy of the Robot

So far autonomous capabilities of the most successful walking, humanoid robots (like Honda Asimo or Johnnie of TU München) are restricted mainly to locomotion, e.g., walking without loosing balance and vision-based navigation which represent quite difficult tasks on their own. However, locomotion and navigation are the most fundamental, autonomous capabilities required by a mobile, autonomous robotic system. For realizing further, autonomous capabilities enabling the humanoid robot to solve complex tasks the development of a complex, deliberative (for solving complex tasks through planning), reactive (for quickly reacting on new sensor data) or hybrid, reactive-deliberative control architecture is needed. For this reason our developed framework for a modular, object oriented control architecture will be enhanced in aspects of the predominant influence of motion dynamics of the humanoid robot. Facets of planning of deliberative or reactive robot controls will be considered.

As a starting point we use the framework with different software modules for a efficient inter-process-communication on one robot. Further steps will be to establish an inter-robot-communication between several PCs and two robots.

Currently our robots' hardware enables autonomy with respect to motions based on joint angle or the camera. The computations required for image processing and object recognition can be performed by the on-board processor. For all online computations as well as for all actuated joints the on-board batteries provide enough power for autonomous operation during competitions.

Autonomous kicking and penalty shoot-out demonstrations at RoboCup German Open 2005 with Mr. DD junior are available as video on the project website.

7 Conclusions

At the moment in our group are two different humanoid robot systems. They agree in six DOF in leg and a camera as main environment sensor, but differ mainly in height (small: 37 cm, large: 68 cm). So aspects of movement in varying large humanoid walking systems can be investigated.

Both robot systems are equipped with different on-board PCs. A newly developed general software framework is implemented on both PCs with the capability of fast communications of software modules on one robot and PC, between several PCs and between several robots.

Future plans are to develop and implement autonomous team actions towards participation in the two-by-two games similar to other leagues in RoboCup, however with the challenge of dynamically stable bipedal locomotion.

Further informations are presented on our homepage (http://robocup.informatik.tu-darmstadt.de/humanoid).

References

1. M. Buss, M. Hardt, J. Kiener, M. Sobotka, M. Stelzer, O. von Stryk, D. Wollherr: ”Towards an Autonomous, Humanoid, and Dynamically Walking Robot: Modeling, Optimal Trajectory Planning, Hardware Architecture, and Experiments”, In: Proc. IEEE/RAS Humanoids 2003, Karlsruhe – München, Sep. 30 - Oct. 3, to appear.
2. M. Hardt, O. von Stryk: “Dynamic modeling in the simulation, optimization, and control of legged robots.” *Z. Angew. Math. Mech.*, vol. 83 (2003) no. 10, pp. 648-662.
3. T. Röfer, H.-D. Burkhard, U. Düffert, J. Hoffmann, D. Göhring, M. Jüngel, M. Lötzsch, O. von Stryk, R. Brunn, M. Kallnik, M. Kunz, S. Petters, M. Risler, M. Stelzer, I. Dahm, M. Wachter, K. Engel, A. Osterhues, C. Schumann, J. Ziegler: ”GermanTeam 2004.” Team Report RoboCup 2004, preprint available from http://www.sim.informatik.tu-darmstadt.de/publ
4. J. Kiener, M. Stelzer, O. von Stryk: “Darmstadt Dribblers 2004: Humanoid Robot. Team Description Paper.” RoboCup 2004, Lisbon, Portugal. preprint available from http://www.sim.informatik.tu-darmstadt.de/publ
5. M. Stelzer, M. Hardt, O. von Stryk: ”Efficient dynamic modeling, numerical optimal control and experimental results for various gaits of a quadruped robot.” In: CLAWAR: International Conference on Climbing and Walking Robots, Catania, Italy, September 17- 19, 601-608.
6. O. von Stryk: ”User’s Guide for DIRCOL Version 2.1”, Simulation and Systems Optimization Group, Technische Universität Darmstadt (2002) WWW: http://www.sim.informatik.tu-darmstadt.de/sw/dircol
7. M. Lötzsch, J. Bach, H.-D. Burkhard, and M. Jüngel: “Designing Agent Behavior with the Extensible Agent Behavior Specification Language XABSL”, In 7th International RoboCup Symposium 2003 (Robot World Cup Soccer Games and Conferences), Lecture Notes in Artificial Intelligence, Padova, Italy, 2004. Springer. to appear.
8. “The Rules & Setup for Osaka 2005. Humanoid Kid Size League, Humanoid Medium Size League”, http://er04.ams.eng.osaka-u.ac.jp/humanoid_webpage/humanoid.pdf
9. Homepage of QT, http://www.trolltech.com