Bioloid based Humanoid Soccer Robot Design

Joerg Christian Wolf, Phil Hall, Paul Robinson, Phil Culverhouse

Centre for Robotics and Intelligent Systems, University of Plymouth
Drake Circus, Plymouth, PL4 8AA, United Kingdom
joerg.wolf@plymouth.ac.uk

***Abstract*— This paper presents an overview of a humanoid soccer robot, build out of readily available hardware. The system architecture is covered in detail thereby enabling the reader to build a starter system for entering bipedal, humanoid soccer competitions. The structure of motion-pages of a Bioloid robot is presented. A vision algorithm to detect goal posts, used for Self-Localisation of the robot, is described. The robot's performance in the FIRA WorldCup 2007 is analysed and some improvements suggested.**

I. INTRODUCTION

The FIRA (Federation of International Robot-Soccer Association) [1] World Cup was held in conjunction with RoboGames [2] in San Francisco CA, USA in June 2007. Besides the FIRA World Cup, there is another Robot Soccer World Cup called RoboCup [3]. The FIRA World Cup has a number of Robot Soccer leagues including the popular MiroSot league with wheeled robots. During the last few years the humanoid league HuroCup (formerly HuroSot) has gained popularity. In 2007 the number of teams participating in the humanoid league topped the number in MiroSot Middle League (12 teams) for the first time. 17 teams participated in the FIRA HuroCup in 2007. In 2006 only 8 teams participated in HuroCup and in 2005 only 5 teams. The University of Plymouth represented the UK in the HuroCup competition in 2007. This paper presents an overview of the bipedal robots used by the University of Plymouth in the FIRA 2007 competition. However, the RoboCup and FIRA humanoid leagues are related and the described robot's architecture can be used for both. Firstly an overview of the hardware components is given in Section II. Section III describes the motion control system. Section IV describes two vision algorithms for detection of goals and self-localisation. The performance of the robot is discussed in section V. Section VI concludes with suggestions for future work.

II. ROBOT ARCHITECTURE

A. Overview

The University of Plymouth bipedal, humanoid soccer player Mk1 (Mark 1) is based on a Robotis Bioloid Comprehensive Kit [4]. The choice of a Bioloid Kit rather than competing commercial humanoid kits such as KHR-2, Manoi or RoboNova was carefully considered. The advantages of the Bioloid over other robot kits on the market include greater servo torque (16.5 Kgcm), programmable servos and price. Furthermore, a Bioloild Kit can be used to build other types of bipedal robots. This provides a jumpstart to the team and the University for teaching purposes. Other teams such as Humanoid Team Humboldt [5] and NimBro [6] have also used Bioloid Robots combined with handheld computers.

The Bioloid is controlled by a CM-5 circuit board, designed by Robotis, with an ATMEL Mega 128 micro controller. In order to carry out vision processing, a more powerful CPU was required. The HP iPAQ hx2490b PDA (Personal Digital Assistant) with a weight of 164 gram was chosen and attached to the front of the robot. A light carbon fibre shell protects the PDA. The PDA and the ATMEL micro controller are interconnected by the RS-232 serial port. For Vision processing a Spectec SD Camera (1.3 MPixel) is connected to the SD-Card slot of the PDA, shown in figure 1.

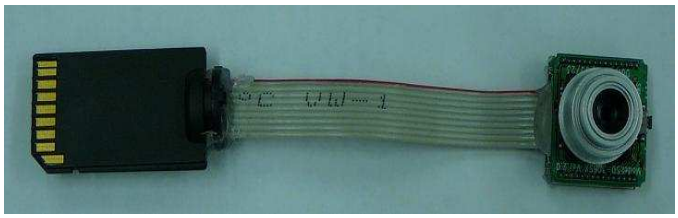

Figure 1. The Spectec SD-Slot camera. The cable to the SD-Slot in the PDA has been extended to allow the camera to be attached to a Pan-Tilt Servo mechanism in the robot's head.

Spectec Taiwan kindly provided us with the driver and API for Windows Mobile 2003 and Windows Mobile 5.0. Windows Mobile 5.0 applications can be programmed by using the Windows Mobile 5.0 Pocket PC SDK for Visual Studio 2005. The bipedal robot, including PDA and camera is shown in figure2.

Each Bioloid AX-12 Servo has its own Atmel MEGA8 microcontroller. The servos are connected to a RS-485-like half-duplex serial bus. The Atmel MEGA 128 acts as the host and sends servo commands to the AX-12 servos at a data rate of 1 Mbit/s.

The robots feet are made of wood. A rounded foot prevents the robot from catching the carpet. The FIRA rules allow a maximum cross section of 14 cm. For RoboCup, the foot size is depending on the robots height H, whereby the foot area A must not exceed $H^2/24$. A foot with larger cross section provides more stability. On the other hand, the mobility of the robot is limited with large feet.

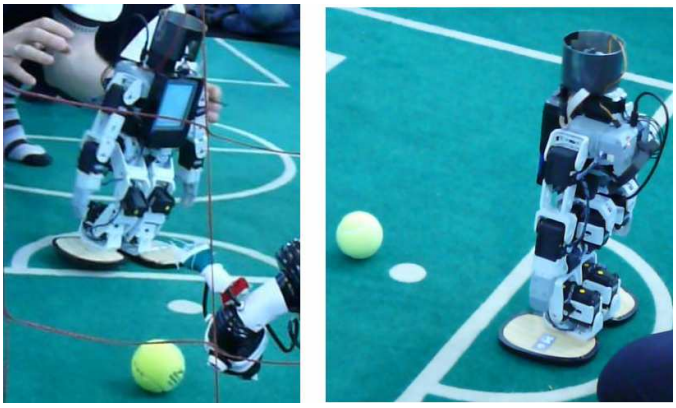

Figure 2. The University of Plymouth humanoid soccer robot gets ready to attempt a penalty. According to the FIRA HuroCup rules, the referee decides the position of the ball, which is not necessarily the penalty spot. Therefore a robot has first to identify the position of ball and goal before taking action.

B. The chain of Command

As argued by Stueckler and Behnke [7], a soccer game with a humanoid robot requires a hierarchy in the control software. Furthermore a communication system between these layers has to be defined. The communication interface protocol specification is a natural divider if the robot is developed, as is normal practice, by a team of engineers. Figure 3 below shows the architecture of the Plymouth Humanoid Mk1.

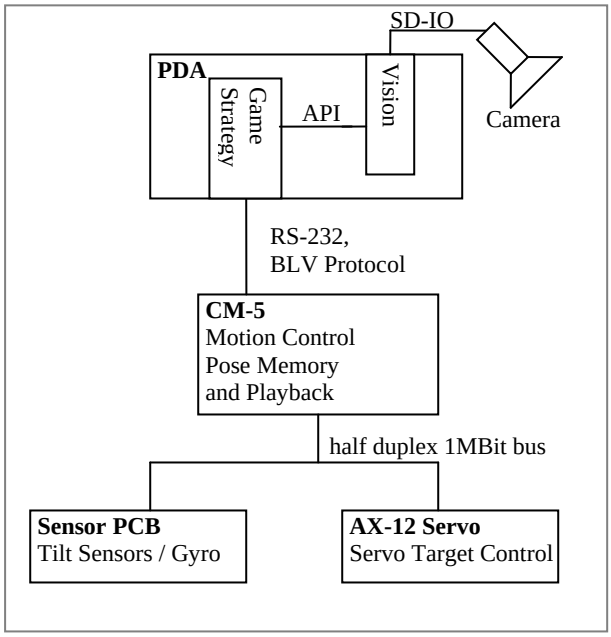

Figure 3. Overview of the Plymouth Humanoid Mk1 Control System

III. Motion Control

Robotis supplies motion-editor software which allows editing of the motion stored in the Atmel MEGA 128 flash memory. The flash memory is organised in pages, where each page can store up to 7 robot body postures. Playing back these postures quickly results in a very fluent robot motion. The strength of this approach includes ease of editing the robot's bipedal locomotion. The robot presented here has motion pages for motions such as kicking the ball, walking forward, walking backwards and walking sidewards, diving to save goals, and standing up. Creating the necessary motions is a labour intensive task and an efficient software editor is an essential tool.

A. Motion Control Software

Source code for the proprietary Robotis ATMEL onboard software is not publicly available. Robotis does however provide an example C program for simple servo control and encourages customers to write their own onboard software. In this paper we reveal the detailed structure of the flash memory motion. Knowing the detailed addresses and structure of the motion pages is the basis for writing custom motion playback software, while preserving the ability to use the existing motion editor. Custom motion playback software such as our BLV012-software [11] can incorporate feedback sensing for balancing. Section III A 1) and III A 2) give description of the motion pages that enable the reading/writing of custom motion playback software using poses stored by the Robotis Motion Editor.

1) Organisation of Flash Memory

The base address of the Motion data is 0xE000. Each page in the motion editor is 512 Bytes (0x200) and contains up to 7 poses. Each pose has a size of 64 Bytes. The first 64 Bytes of a page are reserved for page settings and the rest of the page consists of 7x64 Byte blocks of poses.

TABLE I
ADDRESSES FOR MOTION PAGE SETTINGS

Address	Size	Description
0x0F	1	POSE_PAGE_PLAYCOUNT
0x14	1	POSE_PAGE_NUM_OF_MOTIONS
0x16	1	POSE_PAGE_MOTION_SPEED
0x18	1	POSE_PAGE_ACCEL_TIME
0x19	1	POSE_PAGE_NEXT_PAGE
0x1A	1	POSE_PAGE_EXIT_PAGE

(at the beginning of a motion page)

TABLE II
FONT SIZES FOR PAPERS

Address	Size	Description
0x3E	1	POSE_PAUSE_ADR (pause in 7.8msec steps)
0x3F	1	POSE_SPEED_ADR
ServoNo x 2	2	Servo Target Angle (LSB,MSB)

(within a pose offset, wherey the first pose starts at 0x40)

2) Timing in Motion Playback

All poses of a page are played back and if "Next Page" is not 0 then the program will move on to play back the page given by POSE_PAGE_NEXT_PAGE.

There are two values to influence the speed of playback:
1. POSE_PAGE_MOTION_SPEED
2. POSE_SPEED_ADR.

These will be referred to as "PageSpeed" and "PoseSpeed" respectively. By experimentation it was found that if both are set two 32 (default value) the speed of a servo is 90°/sec. From this the default speed was derived.

$$\omega_{PPN} = \frac{PageSpeed}{32} * \frac{PoseSpeed}{32} \quad (1)$$

So if PageSpeed=32 and PoseSpeed=32 then ω_{PPN} =1 therefore it is called normalised page-pose (PPN) speed.

To play back a motion editor page, all servos should move simultaneously and the servos must all arrive at their target position at the same time. Therefore the arrival time has to be estimated. The servo that has to move furthest will move at 100% of the allowed speed, the other servos have to move proportionally slower in order to finish their move simultaneously. In practice this is dependant upon gearing and loading but as a first approximation these factors are ignored here.

With a simple search through all servos, which compares the current servo position to the target position of the pose, the servo that has to move furthest can be found.

$$\Delta\theta_{Max} = abs(\theta_{Target} - \theta_{Now}) \quad (2)$$

The servo position θ is stored in values ranging from 0 to 0x3ff (1023). The default range of a AX-12 servo is 300°. Therefore 1° ≡ 3.41 units.

Using the relationship $\theta = \omega t$, the time to complete the move can be found:

$$t = \frac{\Delta\theta_{Max} * 1/3.41}{\omega_{PPN} * 90°/\text{sec}} \text{ seconds} \quad (3)$$

One timestep in the motion editor is 7.8msec (128Hz) therefore t in time-steps is:

$$t_{steps} = \frac{\Delta\theta_{Max} * 1/3.41}{\omega_{PPN} * 90°/\text{sec}} * 128 = \frac{\Delta\theta_{Max}}{\omega_{PPN}} * 0.4170 \text{ steps} \quad (4)$$

Each servo is given a target position and servo speed. Maximum Servo Speed depends on ω_{PPN} . The maximum Dynamixel servo speed command is 0x3ff = 1023 ≡ 114 rpm

$$114 \text{ rpm} = 1.0 \text{ rps} = 684 °/\text{sec} \quad (5)$$

Empirical evidence indicates that if ω_{PPN} =1 the servo speed should be 90 °/sec

$$(90 / 684) * 1023 = 135 \text{ servo speed units} \quad (6)$$

Therefore the conversion factor from ω_{PPN} to servo Speed = 135

Each servo is programmed with a fraction of ω_{PPN} , depending on how far it has to move in comparison to the servo with the furthest way to move:

For a given servo n:

$$\omega_n = \omega_{PPN} * 135 * \frac{\Delta\theta_n}{\Delta\theta_{Max}} \text{ servo speed units} \quad (7)$$

Each pose can have a pause time (POSE_PAUSE_ADR). The pause happens after the pose-move is complete and before the next pose-move starts. The pause is scaled with the PageSpeed. Therefore the total time of a pose is

$$t_{pause} = t_{pause-flashmem} * \frac{32}{PageSpeed} \quad (8)$$

$$t_{pose-total} = t_{steps} + t_{pause} \quad (9)$$

3) Issues with motion replay

Acceleration has been neglected in this model. However, the Bioloid motion editor actually considers acceleration. Ideally servo speed and motion time should be estimated using s-curves. For acceleration time POSE_PAGE_ACCEL_TIME < 32, which is commonly used, the time seems to be negligible.

Another issue is the non-linear behaviour if movements are very small, i.e. $\Delta\theta_{Max} < 20$. In this case the motion editor plays back the motion much faster than it should do when considering ω_{PPN} . We speculate that Robotis might have a dead-band statement in the software that cuts down t_{steps} for small moves.

IV. Vision and Self-Localisation

A. Goal Detection

One of the primary tasks for the vision system is to locate a particular goal and calculate the robot's position in relation to it. This not only allows the robot to attempt to shoot a ball

towards the goal, but is also be useful for robot self localisation.

Over the years the University of Plymouth team has found, mainly from MiroSot competitions, that vision systems based upon region growing algorithms are particularly robust. Region growing was therefore investigated as a method of finding the goal. The figure below shows the colour calibration interface which affects the region growing algorithm.

Region growing algorithms are also used to locate the ball. Colour blobs that have been identified by region growing are a useful and robust source for further image processing, as demonstrated by Maggi et. al. [8] where an Erosion-kernel can be applied beforehand to make the method even more robust.

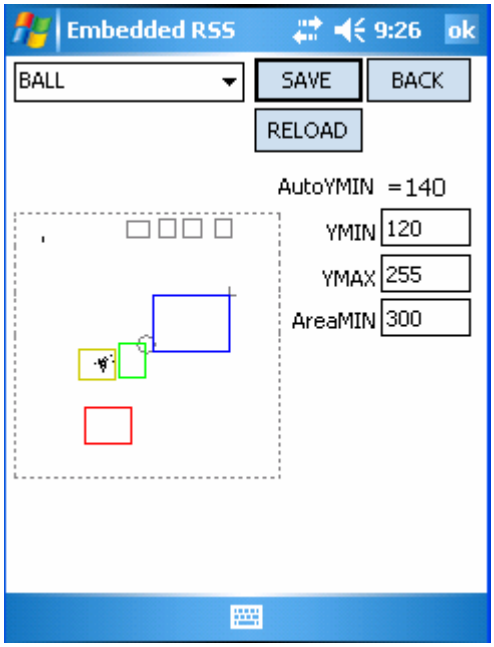

Figure 4. Colour Configuration Menu of the embedded robot soccer software in the PDA. The dashed box represents the UV-colour-plane. The dots are a histogram of the region on the tennis ball, previously selected by the user from an image. In this menu, the user selects areas of the UV plane for calibrating the colour segmentation and region growing. It is important that such configuration menus are easy to access and to use, since there is not much time for setting up the robot in a competition.

For self-localisation it is considered important to locate both of the vertical goal posts separately. Where the crossbar was in view the region grow algorithm located the two posts and crossbar all as one region. This caused the loss of important information such as the lower extent of each post (to allow distance estimation to the posts) and the vertical centre line for each post (to allow an estimation of the bearing to the post). Because of these limitations the algorithm employed for goal detection was therefore not based on region growing. Our goal detection algorithm carries out the following steps:

1. For every column of pixels in the image the number of 'goal coloured' pixels are counted. To be considered goal coloured a pixel must have a UV-colour within a predetermined range and minimum saturation, both of which are calibrated at runtime.

2. The algorithm then selects the image column with the most goal coloured pixels. If this has more than a predetermined number of pixels, which was set at compile time, then the column is considered to be a candidate for a goal post.

3. The algorithm then looks to the columns left of the initial column until one is found with less than 25% of goal coloured pixels when compared to the initial column. This determines the left most extreme of the post. The same process is applied to columns to the right, thereby determining the right most extreme of the post.

4. At this stage the algorithm checks the entire area of the post by summing the pixels in columns that have been identified as within the post on the previous steps. This must be greater than a predetermined threshold that is set at compile time.

5. The X coordinate for the post is considered to be half way between the left and right most columns identified in the previous steps. This, combined with knowledge of the orientation of the camera with respect to the robot chassis, allows a bearing to the goal post to be estimated.

6. The algorithm scans up the image between the left and right most columns until at least half of the pixels are goal coloured. This defines the lowest extent of the post. Since the camera is mounted high on the chassis pointing downwards it is possible to make a crude estimate of the distance from the robot to the post based upon the actual camera orientation and the lowest extent of the post in the image.

7. Finally the process is repeated in an effort to find a second post.

The goal detector is called up to twice for a given camera image with the expected goal colour passed as a parameter. The function returns a count of the number of posts found; 0, 1 or 2.

B. Self Localisation

Usually mobile robot odometry errors are expected to accumulate over time so that the position of a robot becomes less certain the further it travels. This problem is expected to be substantially worse for bipedal robots with limited balance capabilities. Slipping on the match carpet, as used in competitions, is a main contributory factor to bipedal odometry errors. Indeed, following a fall, the position of the robot would probably need to be recalculated from scratch with no consideration to odometry measurements due to the

erratic nature of a fall. For these reasons a self-localisation system must be considered early in the development process.

Probably the simplest localisation method would be to place beacons around the pitch and have the robot take bearings from each. However, it is stated in the FIRA robot football rules [9] that such systems are likely to be outlawed as the standard of play improves. The unknown timeframe for obsolescence caused the University of Plymouth team to discard this approach.

Self-localisation based upon the white pitch markings via the onboard vision system was also considered. However, the symmetry of the pitch would cause ambiguity in the localisation, to resolve this Monte Carlo Localisation (MCL) [12] could be applied.

For the 2007 competition in San Francisco the team chose to use the goals for self-localisation, due to the simplicity compared to MCL. The FIRA rules [9] state that there is a red goal and a blue goal; hence it should be possible to yield an unambiguous position on the pitch by establishing the robot's location in relation to one or both goals. The goal bars are 5 cm wide strips and there is no solid coloured back wall like in RoboCup.

An algorithm to locate the goal for the situation where the robot may be awarded a penalty kick. could be employed for both penalty kicks and general self localisation. .

The goal detector algorithm is applied twice, once for the red goal and once for the blue goal, to all camera frames in an opportunistic attempt at self localisation. If the goal detector algorithm indicates that two posts have been found then the localisation algorithm is invoked. The localisation algorithm follows these steps:

1. The goal detector specifies the positions X1 and X2 of the located goal posts within the pixel space. These values are converted into a bearing of the posts from the robot point of view. Angle C then defines the difference between the bearing to the left post and the bearing to the right.

2. The goal detector also specifies the lowest extent of the goal posts as Y coordinates in pixel space. This is converted into an estimate of the distance of the posts from the robot.

3. The "Law of sines" states that for the triangle shown,

$$\sin(\frac{\alpha}{A}) = \sin(\frac{\beta}{B}) = \sin(\frac{\gamma}{C}) \qquad (10)$$

Since we know the angular distance between the goal posts C, we can calculate the bearing of the robot from each goal post's point of view. This is translated into an x and y position on the pitch. (See figure 5)

4. Finally the orientation of the robot can be calculated since we know the bearing of each post as viewed by the robot, the orientation of the camera within the robot chassis and the location of the robot on the pitch.

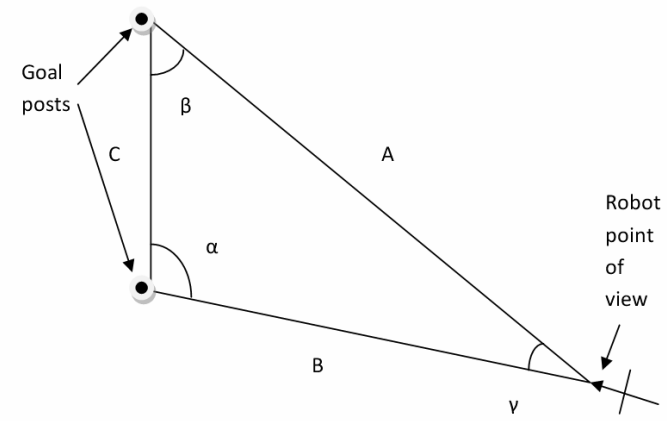

Figure 5 Self-Localisation of the robot based on detection of goal posts

V. Discussion of Performance

Humanoid soccer is still in its infancy. According to Stuecker and Behnke [7], only a fraction of the teams in RoboCup are able to play a decent bipedal soccer game. In the FIRA WorldCup 2007, none of the teams were able to play a decent soccer game. When comparing FIRA and RoboCup humanoid soccer, it appears that FIRA rules are an even greater challenge. The goals in FIRA are marked by only a 5 cm wide strip thereby making localisation more difficult. In addition a FIRA game consists of two teams of 3 players.

The developed localisation algorithm was found to work adequately on a small sample of tests. As may be expected, accuracy decreased with distance from the goal. However, it is possible that this would not cause too many problems during game play. To be declared fit for purpose it would be necessary to subject the algorithm to far more rigorous tests. In practice the localisation algorithm is entirely dependent on data provided by the goal post detector and it would therefore seem sensible to initially spend more time developing this part of the system.

The algorithm was tested by drawing a live camera image on the screen of the PDA and then overlaying a bright, goal coloured, vertical line where the posts were estimated to be. In a brightly lit but fairly cluttered laboratory environment the algorithm produced good results after careful calibration. Before the algorithm can be declared a success it would be necessary to undertake far more formal testing, perhaps using a large set of test images taken in varying light intensities and with various occlusions or false targets, such as spectators in goal coloured clothes. These tests are high on the priority list for future work.

The University of Plymouth robots performed well in the 2007 FIRA robot-dash challenge. This success is mainly due to the brave decision to program the robots to run as fast as they could without consideration to either balance or direction. Surprisingly robust when subject to such indiscriminate

behaviour, these severe tests seem to validate our decision to choose the Bioloid kits over rival commercial systems. However, only regular use and more competition will enable their full capabilities, and weaknesses, to be identified.

VI. Conclusions and Future Work

An autonomous bipedal robot football team, based upon the Bioloid robot kit and running in-house developed vision and control software, was demonstrated at the FIRA 2007 World Cup in San Francisco. One of the main short comings of the current robot, amply demonstrated during competition, is its open-loop control systems. Open-loop gait, lacking any active balancing, severely limits dynamic behaviours to simple, slow, predetermined movements. Unexpected events, such as collisions and tripping on the carpet, are difficult to recover from.

Dynamic balancing is currently being implemented by incorporating 3 axis accelerometers and two axis gyros. We are also investigating dynamic walking methods such as creating oscillations rather than playback of poses. It remains an open question if a generation of robot gait will prove to perform better, since there are many different moves in soccer, such as getting up and kicking, which cannot be easily generated by oscillations.

A PDA is a heavy load for a competition robot, and we are investigating the possibility of replacing the PDA with a Toradex Colibri [10] board. This board has the same CPU as a HP PDA thereby easing the migration of software.

Finally, the work on bipedal robot football at the University of Plymouth has encouraged research and development in related fields such as intelligent, articulated, interactive toys.

References

[1] FIRA website (2007), [Online]. Available: http://www.fira.net

[2] David Calkins, (2007) Robogames website. [Online]. Available: http://robogames.net

[3] RoboCup website (2007), [Online]. Available: http://www.robocup.org

[4] Robotis website (2007), [Online]. Available: http://www.robotis.com

[5] Manfred Hild, Robin Meissner, Michael Spranger, (2007) *Humanoid Team Humboldt Team Description 2007* for RoboCup 2007, Atlanta U.S.A..

[6] S. Behnke, J. Mueller and M. Schreiber, *Using Handheld Computers to Control Humanoid Robots*, In Proccedings of 1st International Conference on Dextrous Autonomous Robots and Humanoids (darh2005), Yverdon-les-Bains, Switzerland, paper no. 3.2, May 2005.

[7] J Stuecker and S. Behnke, *Soccer Behaviors for Humanoid Robots*, Workshop on Humanoid Soccer Robots of the IEEE-RAS International Conference on Humanoid Robots (Humanoids'06), Genoa, Italy, pp. 62-70, December 2006.

[8] A. Maggi, T. Guseo, F. Wegher, E. Pagello, E. Menegatti, *A light software architecture for a Humanoid Soccer Robot*, Workshop on Humanoid Soccer Robots of the IEEE-RAS International Conference on Humanoid Robots (Humanoids'06), Genoa, Italy, Dec 2006

[9] (2007) Jacky Baltes, FIRA HuroCup rules. [Online]. Available: http://www.fira.net/soccer/hurosot/overview.html

[10] Toradex AG, (2007) website. [Online] Available: http://www.toradex.com

[11] Joerg Wolf, SWRTEC website [Online] Available: http://www.swrtec.de/swrtec/hurosot/

[12] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, *Monte Carlo Localization: Efficient Position Estimation for Mobile Robots*, Proc. of the Sixteenth National Conference on Artificial Intelligence (AAAI'99), Orlando, U..SA., 1999