FURGBOL Team Description Paper

Thiago Santos, Emanuel Estrada, Eder Mateus Gonçalves, Vinicius Oliveira, and Silvia Botelho

Universidade Federal do Rio Grande - FURG
Av. Itália Km 8, Rio Grande, RS, Brazil,
{thiagoss, maninho, eder, vinicius, silviacb}@ee.furg.br
http://www.ee.furg.br/~furgbol/

Abstract. The present paper describes the FURGBOL robot F-180 team. The FURGBOL RoboCup team uses inexpensive and easily extendible hardware components and a standard linux software environment. We propose a centralized vision system and a modular architecture, having three main stages: *i.* a Deliberative Stage (associated with strategy and path planning issues), *ii.* a Communication Stage and, *iii.* a Embedded Reactive Control. We describes the relevant aspects of our architecture like software, hardware and design issues.

1 Introduction

The field of multi-robot systems has become enlarged [1]. RoboCup is a long-term effort of the academic and industrial research community to develop teams of robotic football/soccer players. Issues associated with accurated motion, team coordination, computer vision, communication, embbeded systems must be treated, regarding real time restrictions [2–6]. The non controlled robocup scenario is being used as an excellent test-bed for research in several areas associated with the Computer Engineering and Science. Several domains must be treated, for instance:

- properties of single players (vision, real-time sensor fusion and control, autonomous agents, robotics);
- teamwork (multiagent cooperation, context recognition);
- understanding the competition (cognitive modeling) the ability to develop and execute plays and strategies in real time (strategy acquisition, real-time reasoning and planning and reactive behavior).

To treat these issues, several important approaches propose to build sophisticated multi-robot teams through the combination of expensive and complex hardware and mechanical devices [2–5].

From an educational perspective, the RoboCup Competitions is also a great motivation for exposing students to design, build, manage, and maintain complex robotic systems. However, nowadays, how to participate of a RoboCup Competition with a very limited budget, bringing together recent state-of-art robotic

concepts? Is it possible to implement good solutions and sophisticated design methodologies with low cost robotic and sensors platforms? The leap from theory to robotic implementation is often difficult to do, and to do well or efficiently, even more difficult.

The FURGBOL F-180 Team is an effort of the Center for Computational Science of the Universidade Federal do Rio Grande, Brazil. Our goal is to stimulate research, teaching, and applications in the fields of artificial intelligence and collaborative robotics. FURGBOL group is composed by undergraduated students. Our team use inexpensive and easily extendible hardware components and a standard linux software environment. Besides, the FURGBOL platform is entirely based on open source software. Even a very limited budget (U$ 1500,00), FURGBOL has show to be a relatively successful approach; since it started, in 2001, we are five times champion of Brazilian Robocup, vice-champion of Latin American Robocup twice, and last year FURGBOL wins the Latin American Robocup.

This paper describes a set of issues associated with our F-180 Robocup Team. In section 2, we introduce our architecture compose by three main stages: Embedded Reactive Control, Communication and Deliberative Stages. Next sections detail each one of these stages. Finally, we present our implemented system which illustrates the principal aspects of our contribution.

2 An Overview of Our Team

The idea is to have an omnidirectional team to play soccer. Our robots uses omnidirectional wheels, and each wheel has its own motor. In this way each motor needs an independent control and imposes a force in one from the two possible directions. The resulting force composed by the forces (from each wheel) moves the robot towards the desired direction.

The chassis consists of one laser-cut aluminium plate, having a diameter of 176mm and a height of 145mm, see figure 1.

Starting from the **Plan-Merging** Paradigm for coordinated resource utilization - and the **M+ Negotiation for Task Allocation - M+NTA** for distributed task allocation, we have developed a generic architecture for multi-robot cooperation [7]. This architecture is based on a combination of a local individual reactive control and a central coordinated decision for incremental plan adaptation to the multi-robot context. In this paper we present an adaptation of this system to use in a RoboCup Team, see Figure 2.

A Centralized Deliberative System is in charge of the global perception of the field and teams, path planning and the robot behavior. The communication system exchanges informations between robots and Central Station (CS). Finally, we have a reactive embedded control. This stage receives the high level global information from CS, reacting to local environmental changes. Next Sections detail each one of the architecture stages.

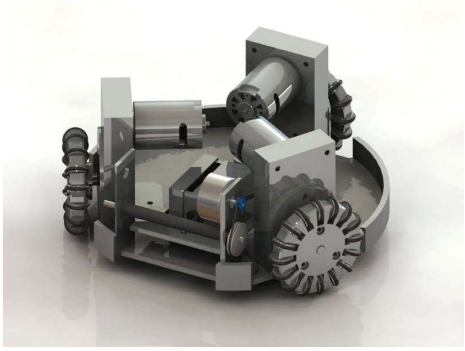

Fig. 1. The CAD chassis of FURGBOL robot.

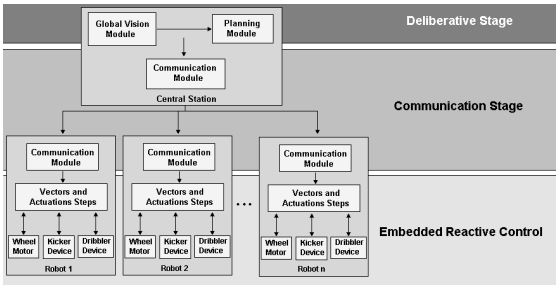

Fig. 2. Our architecture with three main Stages.

3 A Deliberative Central Stage

It is assumed that robots and ball are agents. A state machine is associated with each agent. A Central Deliberative System perceives the environment (agent states) and plans actions and tasks associated with each member team. This stage has two main modules: the Global Vision Module and the Planning Module.

3.1 The Global Vision Module

In order to provide the information not only of the robots but of all the game scenario, the global vision was adopted. Global vision works with a set of n cameras fixed above the field of play, giving the pose of robots, ball and all the relevant visual information. Through a set of image processing techniques, the system performs a two main steps: *i.* radial distortion correction and *ii.* segmentation based either on the HSV or normalized RGB color space analysis. These steps are described as follows next.

The Correction of Radial Distortion Radial and tangential distortions caused by lens at the image must be corrected to improve the reliability in the correspondence of image and real positions of the agents. As the radial

component causes larger distortion, most of the literature work concerns this problem [8]. The two following equations 1 are used to correct radial distortion.

$$x_u = x_d + k_1 x_d \left(x_{d^2} + y_{d^2}\right), y_u = y_d + k_1 y_d \left(x_{d^2} + y_{d^2}\right) \tag{1}$$

where (x_u, y_u) are the corrected coordinates of the distorted point (x_d, y_d), and k_1 is the first term of the radial correction series, truncated at its quadratic term.

The Color Classification

The images acquired from the cameras are captured in RGB (Red Green Blue) format. Our proposal uses a normalized RGB approach. The normalization is done as shown in the figure 3, in which one of the normalized components assumes the minimum value (0), a second component assumes the maximum value (1) and the third one assumes an intermediate value between 0 and 1.

$$\begin{array}{lll} R1 = R / 255 & G1 = G / 255 & B1 = B / 255 \\ R2 = R1 / \max(R1, B1, G1) & G2 = G1 / \max(R1, B1, G1) & B2 = B1 / \max(R1, B1, G1) \\ R3 = 1 - \dfrac{\overline{R2}}{\max(\overline{R2}, \overline{B2}, \overline{G2})} & G3 = 1 - \dfrac{\overline{G2}}{\max(\overline{R2}, \overline{B2}, \overline{G2})} & B3 = 1 - \dfrac{\overline{B2}}{\max(\overline{R2}, \overline{B2}, \overline{G2})} \end{array}$$

Fig. 3. RGB Normalization

In this context, a color circle (see figure 4) divided into intervals with well defined boundaries is used [9]. Considering the minimum and maximum normalized components, the chosen interval defines two possible colors as candidates to the correct classification. At this point, a second intermediate component is located inside this interval and thus the distance of this component is measured in relation to the interval boundaries. The minor distance indicates the closest boundary, selecting which of two colors is definitely the right one.

Image Segmentation This stage is actually started before the color classification. Based on previous acquired frames from the empty field of play, each new frame is subtracted of the empty ones [10]. This technique reveals the areas of the new image substantially different of the old ones in view of diminish the amount information to be processed. After the image subtraction, only the pixels considered relevant to the process undergoes color classification. These pixels are grouped into previous defined intervals of colors named color classes. These classes are of two types: (i) primary classes: orange (ball color), blue and yellow (team colors); (ii) secondary classes: light green, cyan and light pink. With the classification finished, the image scan is started and each pixel belonging to one of the described color classes starts a recursive process in search of similar pixels. This process is known as region growth, and near pixels belonging at the same color class defines a *blob*. As long as all blobs are found, the image segmentation is done.

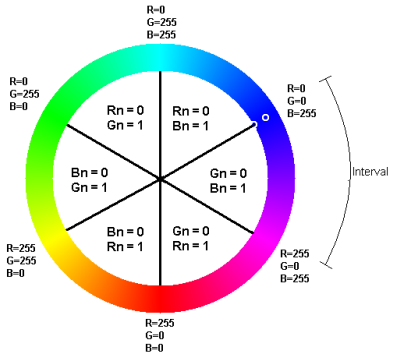

Fig. 4. Color Circle

Agents Localization Finally, all blobs found in the segmentation stage have their center calculated in an effort of perform the blobs bellonging to the process. This process connects some secondary blobs to one primary blob through the search of blob centers whose distance to the center of a main blob is smaller than the radius of the robots. In this way, all robots are recognized and have their positions defined, as well the ball. All the information extracted throughout the vision system is sent to the strategy system which will plan and take decisions that will guide robots through the field of play.

With the current and past positions, Planning Module plans the actions associated with the team members.

3.2 The Planning Module

The planning module is based on a world model which models the state of each agent in the game, like in [11]. We use a set of state machines whose nodes are related to the state of the players and ball and the transitions are given in function of the dynamics of the game.

A Perception Step This step transforms position and velocity information into states associated with each agent. A set of states and transitions (actions) were defined. See table 1 for ball states, and a set of actions (transitions). They are defined based on the relative positions between robots and ball.

State	description	Actions
DISPUTED	two adversary robots are near to the ball	to retrieve the adversary ball
OWNERSHIP	only team members are near to the ball	to move (with ball)
FREE	nobody near to the ball	to move (without ball)
ADVERSARY	only opposite robots are near to the ball	to follow an adversary
GOAL	the ball is into goal limits	to kicker

Table 1. The ball states and Robot Actions.

The robots and ball will assume topological labels, called areas, that identify their localization inside the field: for x axis (that joins both goals) they might be either in defense areas, halfway or attack; for y axis (perpendicular to the x): left side line, right side line and halfway.

A Role Assignment Step With the ball state and topological labels already defined, the Planning Module calculates a set of actions to be achieved by each team member. For that, three kinds of roles are defined: the goal-keeper, the defenser and the attacker. Each role has a own state machine and a different strategy to move, dribble, and shoot (see Figure 5 and 6).

For instance, figure 5(a) shows the goal-keeper role. This strategy calculates an intersection between a vertical straight line (parallel to the middle line) that cross its center and a straight line gotten from the current and previous position of the ball. The intersection point is where the robot must move itself.

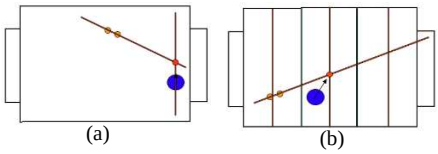

Fig. 5. (a) Goal-Keeper Strategy (b) Defense Strategy

The defenser role uses as information the topological label of the ball, to move the robot to defense the goal, see Figure 5(a). Each area has a vertical straight line, which represents its center. In Figure 5(b) we can see a red center line of the area besides the current ball area (in direction to adversary goal). This strategy calculates the intersection between this red line and an straight line created from the current and previous position of the ball. The intersection point is to where a defense robot must go.

The attacker role is divided in three different strategies, according to ball and robots localization.

The first one is applied when the nearest robot to the ball can attack. It activates the dribbler device and go ahead loading the ball or kicking it into the opposite goal direction. It happens when a straight line from the center of the ball and the center of this robot cross the opposite goal vertical straight line resulting in a point inside of the goal limits.

The second attack strategy occurs when the previously described intersection results in a point out of the limits of the opposite goal. In this case the robot must be posed in a valid position so that it can load the ball to the goal. This position can be gotten by the intersection of a straight line that joins the adversary goal and the ball centers with a circumference of pre-defined radius centered at the ball.

The last approach happens when the intersection of the first approaching strategy results in a point out of the limits of the opposite goal and the ball is

not between the robot and the opposite goal. In this case, an intersection between a vertical straight line that pass in the center of the ball and a circumference with a pre-defined radius centered at the same one is carried through. After, the second and first strategies of approach will be applied.

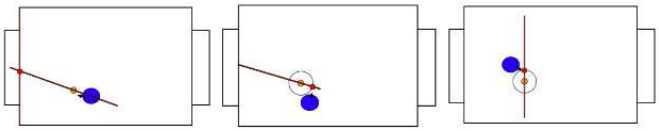

Fig. 6. First, second and third attack strategies, respectively

Each one of these three basic roles supplies a target position to where each robot must move itself. In addition, the planning module decides when a robot should rotate or activate the kicker and dribbler devices. These actions happen when the robot is close enough of the ball to lead or kick it. In order to dribble or to kick, the robot must turn to position the devices in front of the ball. The robot spins if the angle between its front and the correct position is less than a constant defined in the interface.

A Path Planning Step The Path Planning Step defines the robot motion to arrive to target position avoiding obstacles.

Thus, it is first checked whether a straight-line trajectory to the target position is possible without any collision. Then the trajectory and obstacles in the field are regarded as polygons. If one of the vertices of any polygon that represents an obstacle is inside the trajectory polygon, there will be a collision.

If the straight-line path is not possible, we apply the approximated cell decomposition method. This approach allows a robot trajectory planning without any collision. This method was chosen for being simple, to supply a uniform decomposition and to be suitable for make possible the attainment of accuracy (resolution) [12].

The method divides the field in three possible cells: empty, full or mixing cells. The empty cells do not contain obstacles inside. The full ones are completely filled by obstacles. The mixing contains some part filled by obstacles and some empty part. Mixing cells are gotten dividing the main frame by backtracking in cells until it gets a minimum size of cell [1] or until it gets either an empty or full cell. If we choose a good minimum size, enough to avoid obstacle, we have a reasonable processing time.

Starting of the principle that in the end of the process, the cells that had been divided are empty or full, a graph is created connecting the empty neighboring cells. To a cell be neighbor of another one a common point is enough. Later, is executed a shortest path algorithm that uses Dynamic Programming Dijkstra [13]. In our graph, this algorithm gives the shortest path between two nodes

[1] In this case, a minimum size mixing cell is gotten in a full cell.

(empty cells), giving an optimized planned trajectory for each robot, without collision.

The path planning is converted to Velocity Vectors and send to the robots.

4 The Communication System

The CS broadcasts a set of packets containing the PWM levels, dribbler and kick informations and specific ID robot number. The robot owner of the packet must then extract the PWM levels from the protocol and validate it, sending this information to the Control.

The communication protocol consists a header containing the owner of the packet and the data about the PWM levels. The information about the owner of the packet is sent n times by the workstation, so if a robot does not receive this information $\lambda * n$ [2] times, it is discarded. This approach is an attempt to treat noises on the wireless link.

After validation, the Communication Module signals the Control System on the arrival of a new PWM levels. Each robot has its own Communication Module.

5 The Embedded Reactive Control

The Embedded Reactive Control System is responsible for the reactive behavior, receiving low level sensor signals and sending the control to the motors and actuators. This system is composed by the main processor, power stage, motors, gearbox reduction, low level sensors and kicker signals.

The control receives the PWM levels data coming from the Communication Stage, process them and set the PWM signals to the motors. These signals are calculated based on a pre-calculated table with the voltage curve of each motor attached to its gearbox reduction.

Each robot has a kicker and dribbler devices. We use a infrared sensor to detect the ball. This reactive stage activate this devices when some detection happens, enabling it only when the robot has the ball.

6 Implementation and Results

Building a robot team to play soccer is a big challenge in different fields. The range of technologies spans AI, robotic research and embedded system design. Therefore robots are ideal demonstrators for a number of research activities since they offer opportunities to evaluate various strategy-theories, software algorithms, hardware-architectures and design techniques.

We have implemented our proposal with a very limited budget. The Furgbol system was developed in a computer with an Athlon 64 X2 4800++ processor and 2GB of RAM. The Furgbol software has been developed using GNU/Linux

[2] Being $0 \leq \lambda \leq 1$.

operational system and C++ programming language with the QtDesigner development tool [3].

The Deliberative Stage The workstation (CS) is connected to two digital cameras from Samsung (SC-D364 model) with IEEE1394 video outputs. Currently the cameras are connected on VIA1394 Firewire card, VT6306L chipset, with 6x4 input/output, operating with a transfer rate of 400 Mbps (50MB/s).

We are working with three new libraries: libraw1394 and libiec61883 that establishes the communication with the 1394 bus and carries through the data transference; and libdv, that allows the refinement of the received information. The vision system update each robot position in a 20ms rate sample.

All steps of the Deliberative Stage were implemented. The Figure 7 shows a frame with the classified colors, segmented and localized agents.

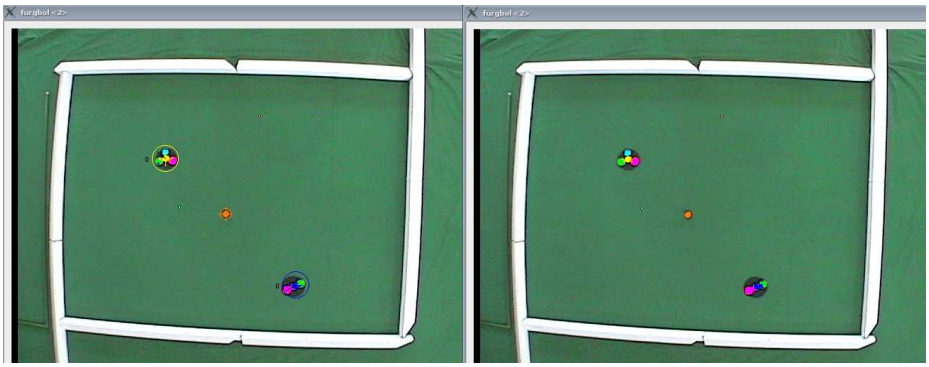

Fig. 7. Interface

A simulator was built in order to allow us to test the strategies and also show the behaviors defined for each agent during the game. With this simulator we can identify errors in the planning module and compare the obtained behavior with the expected result [14]. The Figure 8 displays the simulator interface with the target position defined for a robot (gray), his collision-free trajectory (orange) and identified free cells. Notice that the field was decomposed into full and empty cells.

The Communication Stage The wireless communication is implemented with the Radiometrix's BIM2-433-64-S module and BIM3A-914-64 module, at 433MHz and 900MHz frequency range respectivelly. The workstation broadcasts the packets information about the PWM levels, with a bandwidth of 36600 bps. For instance, the CS sends two times the information about the owner of the packet. Each robot has also its own Communication Module, composed by the BIM2 Transceiver. Currently the communication is one-way only. We use a 19200bps rate to transmit data.

[3] Source codes available in www.ee.furg.br/∼furgbol

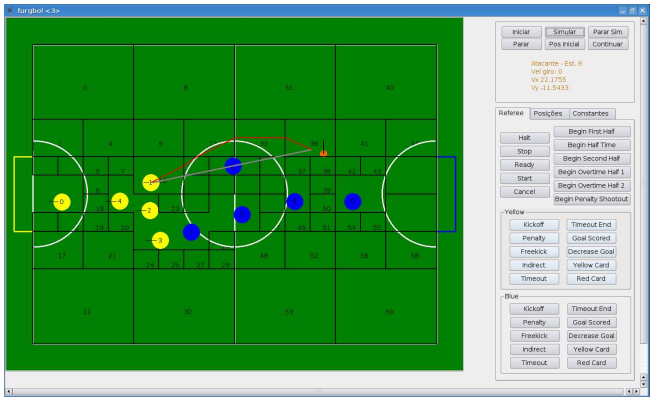

Fig. 8. Interface Simulator

The Embedded Reactive Control The onboard processing is made by a low cost 16 bits RISC microcontroller from the DSPIC30F family, running at 40MHz. The DSPIC family of microcontrollers has a wide range of applications to assist on the programming process. In our project the C programming language was chosen, using the Microchip's MPLAB environment to generate the assembly code.

The board is divided in three distinct stages: Communication Stage (detailed earlier in this section), Power Stage and Control Stage. The power circuitry consists of LMD18200T H-bridges.

Power is supplied by twelve AA NiMH batteries, each one able to deliver 1,2V/2800mAh. The Control Stage is responsible for the Actuation Step, which are implemented in the microcontroller program. Nowadays, we use three omni directional wheels in a 120 degree disposition. Let $F0$, $F1$ and $F2$ be the force vectors from each wheel, b the distance from each wheel to the mass center of the robot's chassis, w the robot's angular velocity, v the robot's velocity vector, r the wheel radius, and $w0$, $w1$, $w2$ the angular velocity of each wheel. The angular velocities are defined by equation $w_i = (v \cdot F_i + b \times w)$ for $r, i = 0, 1, 2$ [15]. These values are converted in PWM signals. A onboard pre-defined table maps from PWM setpoint to motor voltage.

We have design and build chassis, wheels, dribbler and kick mechanic devices, see figure 9. The reduction gearboxes are able to rotate the wheels at 300 RPM with DC 12V motors, using omni-directional wheels. These wheels have a diameter of 50mm that makes possible to develop a maximum linear speed of 0.7m/s. In order to support the modifications over the old model, like new gearboxes, motors and wheels, a new chassis made out of aluminum was constructed. This weight is 1.2Kg.

All stages and algorithms run online (except the automatic calibration). See www.ee.furg.br/∼furgbol for a set of videos of FURGBOL performance.

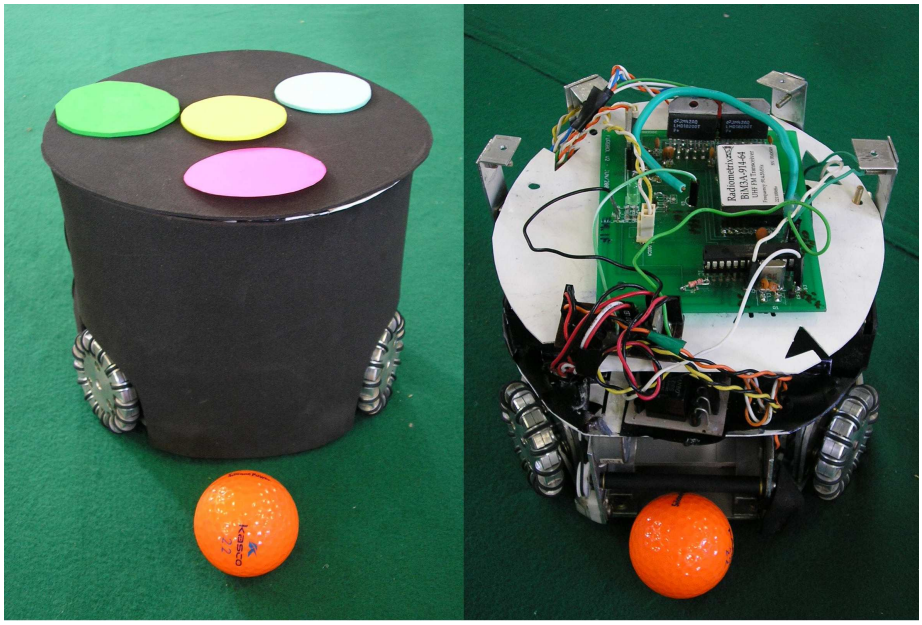

Fig. 9. New FURGBOL robot structure.

7 Conclusions

RoboCup contest is an important test-bed for several areas of the Robotic and Computer Science and Engineering. In addition, for students it is a practical opportunity to develop knowledge in so many areas.

In this paper, we have described a low cost model underlying the FURGBOL Brazilian autonomous robot F-180 team, its implementation and our experiences with it. From a set of theories and algorithms, we have designed and implemented a real team of robots. We have proposed an architecture composed by three main modules proposed: *i.* a Deliberative Stage, *ii.* a Communication Stage and, *iii.* a Embedded Reactive Control. Relevant aspects of our architecture, like software, complexity, hardware and design issues are presented, detailed and analyzed. Our architecture was implemented using inexpensive and easily extendible hardware components and a standard software environment. And, even a very limited budge (U$ 1500,00), FURGBOL has show to be a relatively successful approach; we are five times champion of Brazilian Robocup and champion of Latin American Robocup last year, and vice-champion of this same tournament twice.

We have a set of future short term and long term perspectives. For this year, we intend to improve our kick device, besides to test the robot's new chassis and wheels under real conditions. All of the recently-developed components have the potential to enhance our game play. We will continue to work intensively during

the next two years, aiming to add local vision and embedded treatment to our robots.

References

1. Parker, L., Schneider, F., Schultz, A.: Multi-Robot Systems. From Swarms to Intelligent Automata. Springer (2005)
2. Shimizu, S., Nagahashi, T. Fujiyoshi, H.: Robust and accurate detection of object orientation and id without color segmentation. In: RoboCup 2005. (2005)
3. Egorova, A., Gloye, A., Liers, A., Rojas, R., Schreiber, M., Simon, M., Tenchio, O. Wiesel, F.: Fu-fighters 2003 (global vision). In: RoboCup 2003. (2003)
4. D'Andrea, R.: Robocup2003 cornell robocup documentation, mechanical group final documentation. In: RoboCup 2003, Springer (2003)
5. Loomis, J., Palmer, J., Pandit, P.: Performance development of a real-time vision system. In: RoboCup 2003, Springer (2003)
6. Kuth, A., Bredenfeld, A., Günther, H., Kobialka, H., Klaassen, B., Licht, U., Paap, K.L., Plöger, P.G. Streich, H., Vollmer, J., Wilberg, J., Worst, R., Christaller, T.: Team description of the gmd robocup-team. In: RoboCup-98: Robot Soccer World Cup II. (1999)
7. Botelho, S.S.C., Alami, R.: M+: a scheme for multi-robot cooperation through negotiated task allocation and achievement. In: ICRA2000. (2000)
8. Bailey, D.: A new approach to lens distortion correction, Proceedings Image and Vision Computing. Proceedings Image and Vision Computing New Zealand (2002)
9. Oliveira, L.M.: Segmentação fuzzy de imagens e vídeos. Master's thesis, Universidade Federal do Rio Grande do Norte (2007)
10. Cerqueira, A.C.T., Lins, F.C.A., Medeiros, A.A.D., Alsina, P.J.: A versão 2006 da equipe poti de futebol de robôs. In: Jornada de Robótica Inteligente. (2006)
11. Laue, T., Burchardt, A., Cierpka, K., Fritsch, S., Göde, N., Huhn, K., N., Kirilov, T., Lassen, B., Miezal, M., Lyatif, E., Schwarting, M., Seekircher, A., Stein, R.: B-smart team description for robocup 2007. In: RoboCup 2007. (2007)
12. Amato, N.: Randomized motion planning. Technical report, University of Padova (2004)
13. Cormen, T., Leiserson, C., Rivest, R., Stein, C.: Introduction to Algorithms. McGraw Hill/MIT Press (2001)
14. Torres, E., Martínez, L.A., Muñoz, M., Rodríguez, J., Soto, M., Silva, J., Murrieta, Y., Faba, A., Chavarría, C., Beebe, J., Castillo, M., Garcia, R., Gill, D., Gupta, S., Gutierrez, A., Lindemuth, M., Kalyadin, D., Kern, J., King, S., Kontitsis, M., Moses, A., Palankar, M., Silverman, A., Tsalatsanis, A., Weitzenfeld, A.: Eagle knights-robobulls 2007: Small size league. In: RoboCup 2007. (2007)
15. Reshko, G., Nourbakhsh, I., Mason, M., Zeglin, G.: Holonomic motion control. In: http://www.cs.cmu.edu/ pprk/physics.html. (2000)