GUARANÁ Robot-Soccer Team: Some Architectural Issues

A.H. Reali Costa[1]
anna@pcs.usp.br
R. Pegoraro[2]
pegoraro@bauru.unesp.br
G. Stolfi[3]
gstolfi@lcs.poli.usp.br
J.S. Sichman[1]
jaime@pcs.usp.br
F.M. Pait[3]
pait@lac.usp.br
H. Ferasoli Filho[2]
ferasoli@bauru.unesp.br

[1] University of São Paulo, Department of Computer Engineering
[3] University of São Paulo, Department of Electronic Engineering
Av. Prof. Luciano Gualberto, 158 trav. 3, 05508-900, São Paulo, SP, Brazil
[2] University of the State of São Paulo, Department of Computer Science
Av. Luiz E.C. Coube s/n, 17013-360, Bauru, SP, Brazil

Abstract This paper describes the GUARANÁ robot-soccer team that was vice-champion of the FIRA'98 Games in the MiroSot category. An overview of the vision and the strategy modules is presented, and the hardware and software architectures of the main computer and of the robots are described.

1 Introduction

Soccer enthusiasts consider the sport an exciting strategic and artistic game, which is usually decided by the technical and physical abilities of the players as well as by their collective strategy. Likewise, robot soccer is an exciting strategic game that serves to develop applications and test techniques in computer vision, artificial intelligence and multi-agent systems. Speed, ball handling, decision making, and the team's capacity to cooperate determine the result of the game.

The MiroSot category of FIRA games is played between two robot-soccer teams, each one comprising three robots. On top of each robot is a label — blue for one team and yellow for the other — through which the vision system can identify and locate robots. Vision tasks are performed by a computer using images originating from a single camera hanging over the center of the game field. The resulting information is used by a strategy module which commands the robots via wireless communication.

This paper describes the GUARANÁ team, worldwide vice-champion team of FIRA'98 in the MiroSot category. Section 2 presents the architecture of the team and gives an overview of the vision and strategy modules. Section 3 describes its hardware and software and discusses engineering issues regarding the central computer and the robots.

2 Team architecture

The GUARANÁ team architecture is composed of vision, prediction, strategy, and communication modules. In the next subsections, we describe each of these modules.

2.1 Vision

Image interpretation — essentially recognition of players and ball — is based on the colors of the image captured by the camera. We used two rectangular labels with a black margin on top of each GUARANÁ player. The first label has the team color — either blue or yellow — and the second, pink label assists the vision module in determining the robot's orientation. The ball used in the game is an orange golf ball. Figure 1 shows the labels used on top of the GUARANÁ robots.

A preliminary image processing phase consisting in subtracting an image of the empty field from the captured image is used to simplify and speed up interpretation. This subtraction occurs only in R and G channels of the image, since these are sufficient to detect any variation of every necessary color in the application. Pixels for which the result of the subtraction is bigger than a pre-established threshold are considered possible elements of the game (ball or labels). Their color patterns are analyzed using values taken from both R/G and G/B relations, in order to reduce influences of ambient illumination. Color identification is

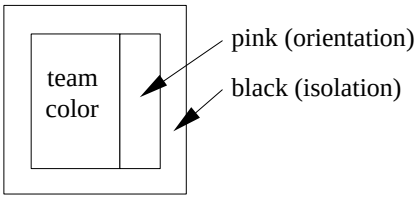


Figure 1: Robot identification labels.

carried out by matching R/G and G/B relations with predefined color *bands*. We have restricted this analysis to the colors which have meaning in the game (yellow, blue, orange, and pink).

Centroid

Neighboring pixels with similar colors are considered components of one same *element*. The elements considered are pink, yellow, blue and orange patches, the orange one being circular. The coordinates (x, y) of the centroid (center of mass) of each element are computed from the pixel coordinates using the formula (**?**):

$$x = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \text{and} \quad y = \frac{1}{n}\sum_{i=1}^{n} y_i$$

where (x_i, y_i) are the pixel coordinates and n is the number of neighboring pixels used to group the pixels into an element.

Identification and Tracking

To send a command to a robot, the central computer must know its *state*, consisting of position, orientation, and speed. This requires a procedure for identifying each robot.

In our system, the labels on the robots are similar, and therefore can not be used to distinguish a robot from the other members of the team. The black border is used to avoid misidentifying two robots as one single element when they are in contact.

When the game is started each robot is identified by the user of the system. From then on identification is done via a tracking algorithm which uses exhaustive optimization to find the minimum cost solution of the following system:

$$s(i, j, r) = d(e_{\text{team}}(i), e_{\text{pink}}(j)) + d(e_{\text{team}}(i), e_{\text{prev}}(r))$$

subject to

$$\begin{aligned}
li_{\text{team}} &\leq ne_{\text{team}}(i) &&\leq ls_{\text{team}};\\
li_{\text{pink}} &\leq ne_{\text{pink}}(j) &&\leq ls_{\text{pink}};\\
0 &\leq d(e_{\text{team}}(i), e_{\text{prev}}(k)) &&\leq 14; \quad \text{and}\\
4 &\leq d(e_{\text{team}}(i), e_{\text{pink}}(j)) &&\leq 7
\end{aligned}$$

In the expressions above $r = 1, 2, 3$ is the number of the robot considered; $i = 1, 2, \ldots, n_{\text{team}}$; $j = 1, 2, \ldots, n_{\text{pink}}$; and

- $s(i, j, r)$ is the objective function to be minimized;
- $d(e_1, e_2)$ is the cartesian distance between elements e_1 and e_2 in the image coordinate system;
- $e_{\text{team}}(i)$ is the centroid of the ith element with the team's color;
- $e_{\text{pink}}(j)$ is the centroid of the jth pink element;
- $e_{\text{prev}}(r)$ is the centroid of the rth team color element identified in the previous image;
- $ne_{\text{team}}(i)$ is the number of pixels in the ith team color element;
- $ne_{\text{pink}}(j)$ is the number of pixels in the jth pink element;
- li_{team} and ls_{team} are upper and lower limits in the number of pixels, which define the valid range of the size of a team color element. These values change according to the image resolution used. We used $li_{\text{team}} = 60$ and $ls_{\text{team}} = 80$;
- li_{pink} and ls_{pink} are upper and lower limits in the number of pixels, which define the valid range of the size of a pink element. We used $li_{\text{pink}} = 20$ and $ls_{\text{pink}} = 30$;
- n_{team} is the number of team color elements in the image and
- n_{pink} is the number of pink elements in the image.

This optimization algorithm is applied on each of the three robots in the team. As a result, the system will have three different pairs (team color element, pink element), each associated with one of the robots.

Although correctness of this tracking procedure was not formally proved, in practice it was shown quite robust. We did not experience identification loss or swapping of robot tracking during the games.

Orientation and Position

The robot's orientation is defined by the angle α ranging from field's axis x to the vector (pink centroid, team color centroid), as illustrated in Figure 2. For the robot dimensions and image resolution used, the maximum deviation observed was 10 degrees.

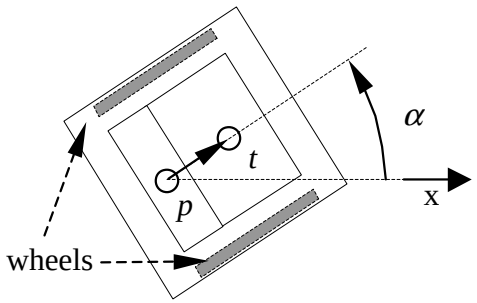


Figure 2: Robot orientation from labels.

The position (x, y) of the team robots results from the weighted average of the positions of the centroids of the team color and pink labels, defining the robot's geometric center. The equations used are:

$$\alpha = \tan^{-1}\frac{y_t - y_p}{x_t - x_p},$$

$$x = \frac{}{c_1 + c_2} \quad \text{and} \quad y = \frac{}{c_1 + c_2},$$

where (x_t, y_t) are the coordinates of the centroid of the team color element; (x_p, y_p) is the coordinate of the centroid of the pink element; and c_1, c_2 are factors that depend on the label size and image resolution. In our case, $c_1 = 7$ and $c_2 = 3$.

The positions of the adversary robots and of the ball are defined by the centroid coordinates of the adversary color and ball color elements.

Orientations are determined by the direction of movement, given by the difference between the positions of the elements in the current and previous images.

New Coordinate System

After identifying each object in the image, the image coordinate system (in pixels) is transformed into the field cartesian system (unity of 0.5cm), as shown in Figure 3.

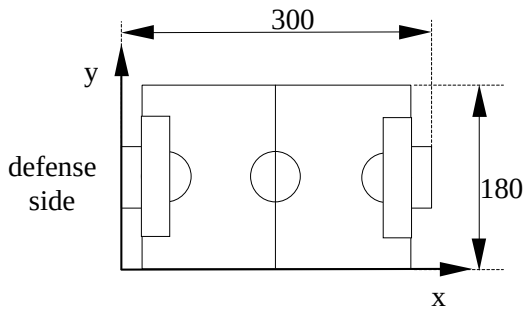


Figure 3: System of coordinates after vision phase.

The vision module implemented was fast (7ms per frame) and robust, allowing variations on illumination conditions and colors of the field, labels, and ball. Setting up the vision system was quick and easy.

2.2 Prediction

A command c_i is sent to a robot r based on the state of all elements observed in the image at instant t_i. Due to the delay involved, it is necessary to predict the future state of the robots and the ball in order to estimate their state at the moment they actually receive the command.

The position, orientation, and velocity of adversary robots and of the ball are predicted based only on the data extracted from the current and previous images assuming constant velocities. Predictions for the team robots are more accurate and take into account their current states and the most recent commands according to a nonlinear dynamical system of the form

$$s_{r_{i+k}} = s_{r_i} + f_r(s_{r_i}, c_i, c_{i-1}, \ldots, c_{i-n})$$

where: $s_{r_{i+k}}$ is the predicted state, after delay of k (sensoring and communication); s_{r_i} is the state at instant i (sensor input); $r = 1, 2, 3$ is the robot considered; $c_i, c_{i-1}, \ldots, c_{i-n}$ are the $n+1$ commands considered; and $f_r(\cdot)$ is an empirically adjusted function.

2.3 Strategy

Every Brazilian is said to be a soccer coach. In human soccer, a stronger team will often defeat another which has superior technique. Trusting our experience, we formulated some simple strategic rules:

- Against stronger opponents, emphasize defense to suffer fewer goals.
- Against slower opponents, attack. There will be time to bring attackers back to help the defense.

Behaviors

There exist three strategical main behaviors that each robot can assume according to its position:

1. *goalkeeper:* The robot is placed in the projected position of the ball on the defensive goal line if the ball is coming towards the defensive region. Otherwise the robot aligns itself with the ball in the y coordinate, always remaining in front of the goal.

   For faster positioning the robot always keeps an orientation of 90 or 270 degrees, as shown in Figure 4.

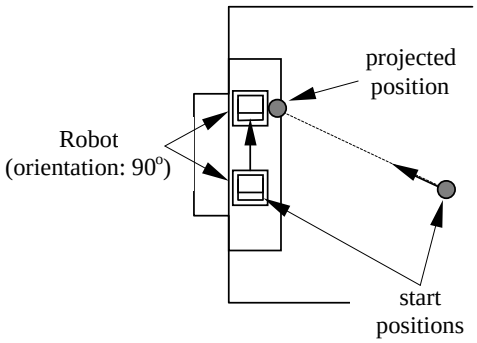


Figure 4: Goalkeeper behavior.

2. *defender:* The robot is placed along the coordinate y of the ball, blocking the ball to prevent it from reaching the goal. It does allow the ball to pass when it is being pushed away from the goal by the attacker. When the defender and the ball are in a favorable situation with respect to the adversary goal it switches behavior with the attacker and tries to score. The favorable situations happen when the robot is the team robot closest to the ball in the transition area, shown in figure 5.
3. *attacker/midfield:* In this position the robot has two distinct behaviors: *(i)* positioning mode, when the objective is to position the robot behind the ball, leading to the goal or blocking the adversary attack. Positioning serves to mount an attack or strengthen the defense. *(ii)* conduction mode, when the robot pushes the ball from behind towards the opponent's goal.

Usually the robots are sent directly towards the target. In the attacker behavior however, a discontinuity in the target position

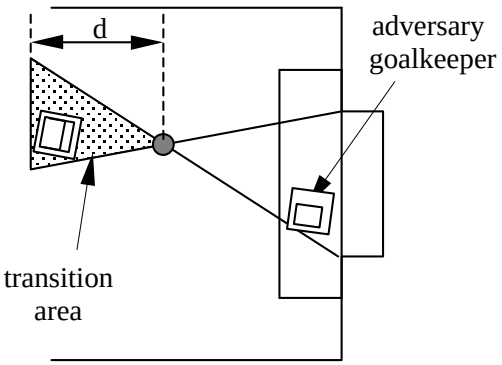


Figure 5: Transition area.

happens during the transition between positioning and conduction modes. This discontinuity is smoothened taking into account the limitations of movement of the robot.

Trajectory planning considers current state s (current speed, position, and orientation) and desired state g (desired speed, position, and orientation) for the robot, as shown in Figure 6a. The dashed line indicates the planned path.

A desired and feasible command which would take the robot to the target state is chosen. The effect of applying the inverse of this command to the target robot state g is computed, generating a new target state g' (Figure 6b). The procedure is repeated using g' as the target state, and so on. The last command thus generated, in fact the first we desire to use, is transmitted to the robot (Figure 6d). The computation is repeated for each new image during transition.

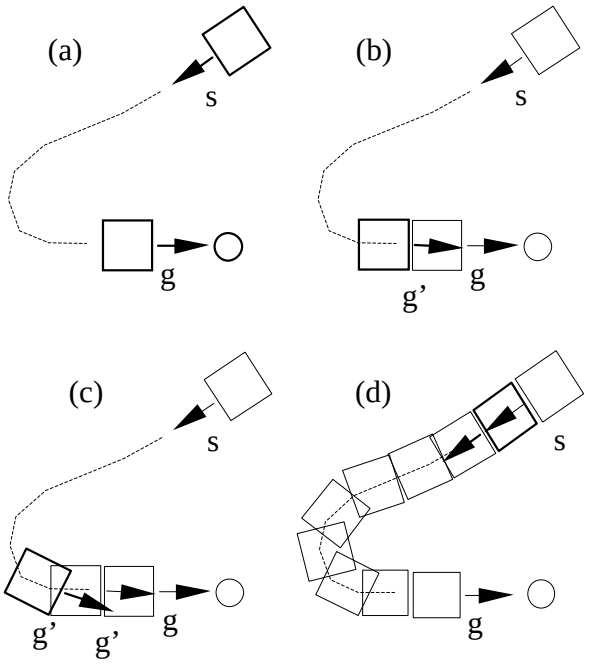


Figure 6: Planning of transition between objectives.

Obstacle avoidance

The robots try to avoid collision by planning a path around obstacles. Due to the dynamic nature of the soccer domain, our robots continually replan their target positions around obstacles via incremental generation of intermediate target points.

A robot r starts by trying to go from its position s towards its target g along a straight line, defining a path $\mathrm{Path}_{s,g}$ from s to g. If following $\mathrm{Path}_{s,g}$ it would come across an obstacle o as in Figure 7a), the robot r aims at an intermediate target g' as shown in Figure 7b, taking into consideration the point P_i where the trajectories of o and r intercept.

The velocity of the objects and the interception point P_i define the collision time t_c. If the collision time t_c is smaller than the time t_g that robot r will take to reach target g (that is, $0 < t_c < t_g$), and if the intersection point P_i of the trajectories of r and o is inside the path $\mathrm{Path}_{s,g}$, then an intermediate target g' is defined. g' is located at a distance d from P_i, perpendicularly to $\mathrm{Path}_{s,g}$. We have used $d = 20$cm. The algorithm also takes into account the walls of the playing field in generating intermediate target points.

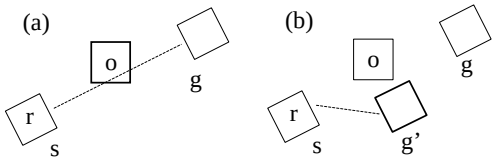


Figure 7: Obstacle avoidance.

Intersection of Trajectories

The interception point $P_i = (x_i, y_i)$ is defined as the point where the trajectory of robot r crosses that of object o. An approximate computation is used. The number of iterations n_x after which the trajectories of r and o will intersect on x and the number of iterations n_y after they will intersect on y are

$$n_x = \frac{x_r - x_o}{dx_o - dx_r} \quad \text{and} \quad n_y = \frac{y_r - y_o}{dy_o - dy_r},$$

where (x_r, y_r) are the robot coordinates; (x_o, y_o) are the object coordinates; and dx_o, dy_o, dx_r, and dy_r are displacements of the object and robot given by the difference between the current position and the previous.

If $n_x < 0$ or $n_y < 0$ there is no interception point. Robot r cannot intercept object o in a number of iterations smaller than $n_i = \min(n_x, n_y)$. When r is far from o, n_x and n_y can be different. However, they iteratively converge to the same value as the system evolves.

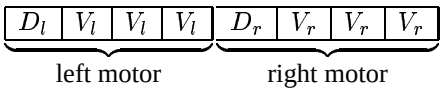
The interception point (x_i, y_i) is calculated as follows:

$$x_i = n_i\ dx_o + x_o \quad \text{and} \quad y_i = n_i\ dy_o + y_o$$

2.4 Communication system

The communication system is composed of one transmitter and three receivers working at 72.830 MHz FSK. They were adapted

from R/C radio in order to transmit digital signals, with transmission rate of 1200 bps, using Manchester code (**?**). The transmitter broadcasts messages to the robots under the following protocol: 2 bits SYNC, 2 bits preamble, 8 bits for robot r_1, 8 bits for robot r_2, 8 bits for robot r_3, and 4 bits for Check Sum. The 8 bits sent to each robot comprise 4 bits for each motor (left and right).



where D is the direction the motor is to turn, and V is the velocity (3 bits).

3 Hardware and software platforms

The software of the GUARANÁ team is centralized and runs on a PC Pentium II-233 MHz, using Windows 95 and generic programming in Borland C 4.0, including some assembler routines in the vision module.

Images are captured by a JVC 3CCD color camera with S-VHS video output and digitalized by a PixelView PV-Bt-848 PCI board. The digitalized image uses resolution of 320x240 pixels and 24 bit quantization in RGB.

Image processing is performed by the vision module, using Microsoft Video for Windows (VfW) (**?**) as interface. The use of VfW provides versatility and transparency since it allows digital image acquisition by any compatible frame grabber. The vision module is able to process 30 frames per second.

The robots use motors adapted from an electromechanical calculating machine. They are small Mabuchi DC motors, rated at about 20 gf×in torque at 5Vcc. A single stage reduction gear is used for each motor/wheel assembly, providing a 5:1 speed reduction.

On each wheel there is an imprinted reflective pattern, coupled to a reflexive infra-red sensor, working as a rotary pulse generator. The pulses from each wheel are fed back to the electronic control unit in order to allow speed control (see Figure 8).

Robots can develop a maximum speed of about 1 m/s. The electronic control unit is composed of an EEPROM-based microcontroller (Atmel 89C2051), running at 14.75 MHz clock. The motors are driven by a full-bridge circuit using complementary MOS transistors. The transistors are switched directly by microcontroller I/O ports, allowing speed and direction control by pulse width modulation.

The robot's receiver is adapted from a FUTABA R/C double conversion servo receiver. The digital signal is obtained at the FM discriminator output, squared, and then sent to the microcontroller, which is responsible for sync and clock recovery and Manchester (split-phase) demodulation.

The robot's body was built with epoxy-fiber glass boards and $1/2$ inch plastic PVC pipe as batteries holders. Electric power was supplied by 4 NiCd AA cells, providing 800 mA×h at 4.8Vcc.

Copper cladding on the robot's top cover acts as the receiving antenna. Owing to the relatively high transmitter power (100 mW) and the highly sensitive receiver, the range of the radio communication was over 50 meters.

The use of small motors and plastic wheels and body resulted in a lightweight, robust, and low-cost robot.

4 Conclusion

The GUARANÁ team's strong points are its straightforward and efficient tactics and the vision system: simple, robust to environmental variations, and easy to adjust. We believed that the good performance of the team is due to the reasonable balance of our system.

Robot control was shown imprecise and could benefit from improvements. To achieve a more competitive system the robots would need stronger, higher-torque motors.

Acknowledgements

Faculty, researchers, and both graduate and undergraduate students are co-responsible for the successful completion of the GUARANÁ project. We would like to thank Marcelo Nicoletti Franchin (UNESP-Bauru), Tsen Chung Kang (PEE/EPUSP), Reinaldo A.C. Bianchi, Bruno A. Basseto, Renato Mikio Nakagomi, Fernando Luiz Goulart, Guido Marcelo Chagas, Júlio L.R. Monteiro, Ricardo Matsushima Teixeira, Ricardo F. Pereira, Leonardo Scardua, Roberto Barra, Ricardo P. Domenecci, Mário A.N. Solis for their work in several phases of the project.

Dr. J.S. Sichman research is partially funded by CNPq, Brazil, grant number 301041/95-4. His participation in the FIRA'98 was sponsored by FAPESP, Brazil, grant number 98/03489-9. Renê Pegoraro is partially financed by CNPq.

Our participation in MiroSot FIRA'98 was sponsored by the Polytechnic School of the University of São Paulo. We would like to express our gratitude to Prof. Antônio Marcos de Aguirra Massola, Dean of the School, as well as to Prof. Geraldo Lino de Campos, head of the Department of Computer Engineering for their inestimable help.

References

[1] Han, W.-G. et al., Path Planning of Visual-Servoed Multiple Mobile Robots using the Genetic Algorithms, *Proceedings of Micro-Robot World Cup Soccer Tournament*, pp. 57-63, 1996.

[2] Giozza, W.F. et al., *Redes Locais de Computadores: tecnologia e aplicações*, São Paulo, Brazil, 1986 *(in Portuguese)*.

[3] *Video for Windows Documentation* `ftp://ftp.microsoft.com/developer/drg/Multimedia/Jumpstart/VfW11e/DK/VFWDK`

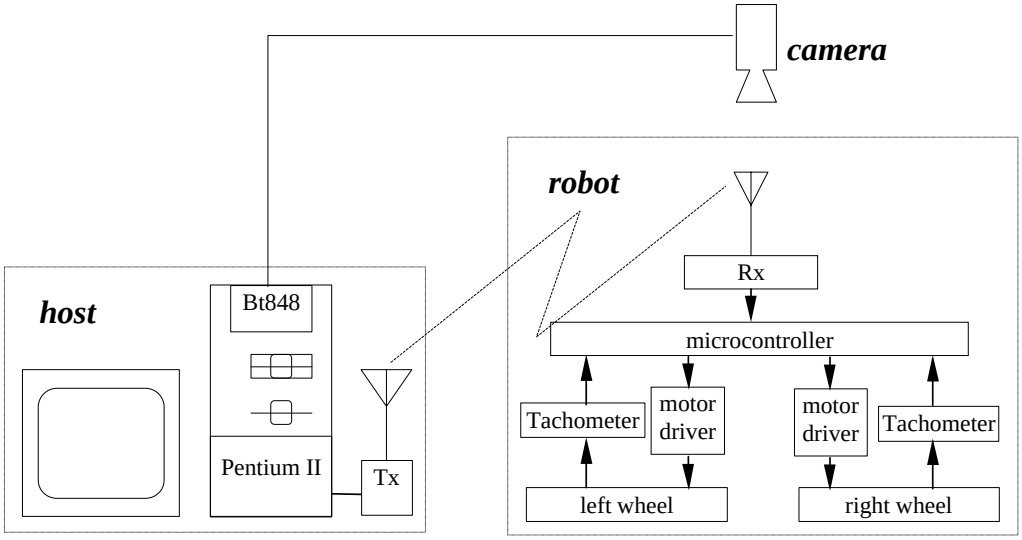


Figure 8: Electronic Scheme.