Eagle Knights - Robobulls 2010: Small Size League

Ernesto Torres[1], Paolo Aguilar[1], Alan Córdova[1], Hipólito Ruiz[1], Julio Sánchez[1], Miriam Martínez[1], Marco Morales[1], Matthew Holland[2], Wayne Blackshear[2], Chris Hewell[2], and Alfredo Weitzenfeld[2]

[1] Robotics Laboratory, Division of Engineering ITAM, México City, México
[2] Robotics Laboratory, IT, University of South Florida Polytechnic, Lakeland, FL, USA

Abstract. In this paper we describe the design and implementation of the Eagle Knights – Robobulls Small 2010 Sized League RoboCup Team. We explain the design of the AI and Robot System. We focus on the latest improvements in our sixth generation of robots and how these changes interact with the rest of the elements in the team architecture.

1 Introduction

RoboCup [1] is an international joint project to promote research on artificial intelligence, robotics and related fields. In the Small Size League, two teams of five robots up to 18 cm in diameter play soccer on a 4.05 by 6.05 m carpeted soccer field. As shown in Figure 1, aerial cameras send video signals to a shared vision system[2] that estimates the position of the robots and of the ball on the field. This information is then passed to an AI system that produces control commands that are sent to each of the robots through a wireless link. An external referee box indicates the state of the game to the central computer.

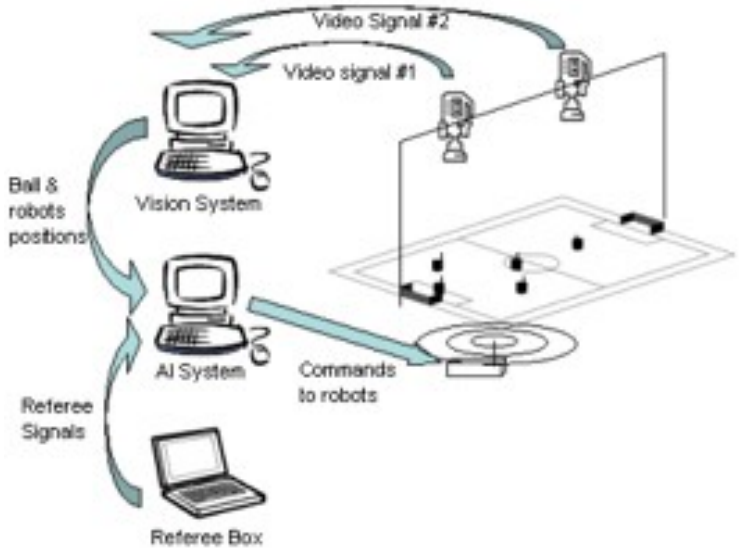

Fig. 1: Typical Architecture of an SSL team

The robot architecture of our team in the Small Size League (SSL) consists of four main components: (1) shared vision system, (2) AI system, (3) robots and (4) referee:

1. The **shared vision system, SSL Vision[2],** digitally processes two video signals from the cameras mounted on top of the field. It computes the position of the ball and robots on the field, including

orientation of robots in our team. Resulting information is transmitted back to the AI system. This year we are using the SSL Shared Vision System.

2. The **AI system** receives the information from the vision system and makes strategic decisions. The actions of the team are based in a set of roles (goalkeeper, defense, forward) that exhibit behaviors according to the current state of the game. These behaviors are encoded as rules of an expert system. A geometrical exploring tree is used to avoid collision with robots of the opposite team [3]. Trajectories are computed based on splines. The AI system sends commands back to the robots through a wireless link.

3. Five **robots**, execute commands sent from the AI system by generating mechanical actions in a 60-times-per-second cycle. Each robot satisfy the constraints set in the SSL rules: they fit inside a 180 mm diameter cylinder and have a height of less than 150 mm, they ensure that more than 80% of the area of the ball when viewed from above is always outside their convex hull.

4. The **referee** can communicate additional decisions (penalties, goal scored, start of the game, etc.) sending a set of predefined commands to the AI system through a serial link.

In the next sections we describe in more detail each component of our team. While most descriptions are based on our prior generation of robots (Figure 2) [4–7], we are currently designing totally new enhanced robots for RoboCup 2010. The description focuses on the improvements made to our sixth generation of robots and the way these changes interact with the rest of the components in the system architecture.

Fig. 2: Small Size League 2009 Eagle Knights Robots

2 Shared Vision System

The most important change in the SSL League this year is the use of a new shared vision system, SSL-Vision[2]. The new system will be used for the first time during RoboCup 2010. It is the only source of

feedback in the system architecture. If data returned by the vision system is inaccurate or incorrect the overall performance of the team will be severely affected. That is why the vision system should be robust enough to compensate for possible errors.

The main object features used by the vision system are the colors defined in the rules of the SSL [1]. The ball is a standard orange golf ball. Each robot has a 50-mm circular patch, this patch is blue in one team and yellow in the other team.

The main tasks of the vision system are:

- Capture video in real time from cameras mounted on top of the field.
- Recognize the set of colors specified by the rules of the objects of interest on the field (robots and ball).
- Identify and compute the orientation and position of robots in the team.
- Compute the position of robots of the opposite team.
- Track the objects on the field and compute their moving vector.
- Transmit information to the AI system.
- Adapt to different lighting conditions (color calibration procedure).

In the past we have implemented a number of algorithms for adaptability to different light conditions including the use of a neural network to classify camera image pixels to a discrete set of color classes that is robust under different light conditions[8].

3 AI System

The AI System comprises eight modules: artificial intelligence, simulation system, collision detection, transceiver communications, robot control, user interface, vision system communications and game control.

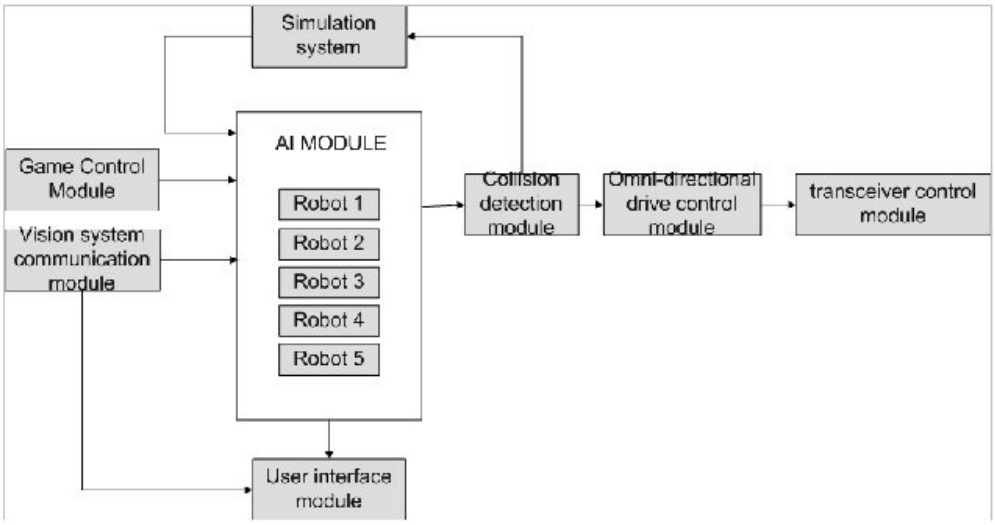

Fig. 3: AI System Architecture

The AI system includes a main thread that loops and calls each of the different modules as shown in Figure 3. The detailed description for each of the modules follows below.

Vision System Communications Module This module provides information about the state of the game scenario corresponding to the position, angles and motion vector of the robots and of the ball. This information is provided through packets.

Game Control Module This module receives referee commands through a serial interface and returns the game state of the game. For 2008 RoboCup we succesfuly tested the Ethernet interface both modes are available.

AI Module This module receives the robots and ball positions, robots orientations, game state, robots roles, shooting direction and field configuration. It estimates the future position of each robot and the actions they should take. The strategies are programmed as rules on an expert system. The actions are classified according to their importance. For each node of the tree one or more evaluations are used. Each evaluation has a group of possible results associated with a particular score. During the loop of the program the tree is evaluated. The trajectory to take from the root to leaf (final action) depends on the highest score of the evaluation result on each level using a Best First Search method. Once the system has reached a final action (e.g., passing, shooting, or blocking, the robot moving vector) its linear and angular velocity and the use of the kicker and dribbler devices is defined. The robots also include a roll motion to coordinate them in joint actions. The robots are coordinated through different roles: Goalkeeper, Defense, First, Second and Third Forward. The final action for each role is defined using the position of the robots and the ball. With the implementation of the Kalman Filter in the vision system it is possible to know where the ball is moving and the defenses and goalkeeper can block its moving path. It is also used to intercept the ball from the other team, to give or receive passes and to shoot to goal. To define more efficient trajectories we implemented a spline based method to produce smoother routes for each robot.

User Interface Module This module constantly displays important programmed using MFC control. The information includes the robot's position, orientation and speed, the game referee, the control commands to the robots and the configuration of the AI system. The spline trajectory module is implemented in this section to test the result of these new routes.

Simulation Module This module tests system functions of the AI, collision detection and robot control modules without the actual vision system or robots present. It is useful to debug and test actions in the artificial intelligence module. The field is visualized with a two or a three dimension graphics OpenGL[9] interface.

Collision Detection Module This module receives actual and final position of the robots and generates a new path generated by a GET (geometrical exploring tree). With this method it is possible to easily avoid the opposite's robots and goals. The method works in real time (five robots in more than 60 fps). Because the space is constantly changing and the configuration is constantly changing we decided to use exploring trees. A GET constructs a tree during every process iteration. It can combine different types of obstacles with geometrical figures. The robots are represented like circles and the goal like rectangles. To generate the planner the tree starts with a root in the initial point and it is classified as an exploring node. The final point has already been defined by the AI module. The steps of the algorithm are as follows. First select an exploring node of the tree and try to reach the goal advancing a small predefined distance. If there are no obstacles interfering with the new point then the tree is extended and the new point replaces the last exploring node. In order to know if an obstacle interferes between the initial point and the goal the geometry must be considered. A vector "A" is defined from the initial point to the goal. In the case of a circular obstacle a possible intersection between the circle and the vector "A" is calculated like shown in Figure 4. In case of an intersection the distance between the obstacle and the new extension is validated to be smaller than a radio "R".

In the example shown in Figure 4. there are two intersections: "P1" and "P2". When the tree reaches the radio "R" , it will generate two possible routes one to each side of the obstacle. These two points are now considered as exploring nodes. While the exploring node can not freely reach the goal then it continues rounding the obstacle until there is not intersection or another obstacle is found. In this case a new obstacle is defined as the exploring node obstacle and it continues surrounding the new obstacle as shown in Figure 5.

In case of the goals the algorithm works similar but surrounding them with a rectangle. Once the tree has reached a point close enough to the goal the nearest path is chosen and the robot can be sent directly to the first point before intersection or in the direction of the next node towards the goal. This algorithm is repeated every cycle until the robot reaches its goal.

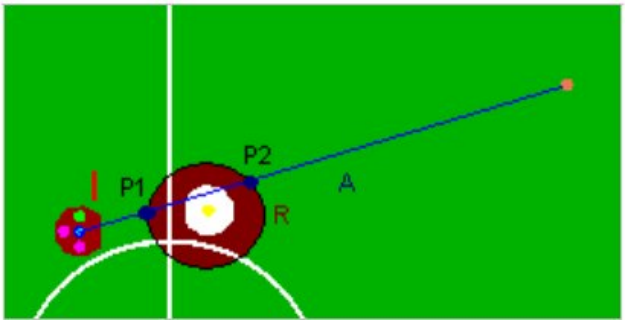

Fig. 4: Collision detection of a circular obstacle and the robot 1 trying to reach the ball

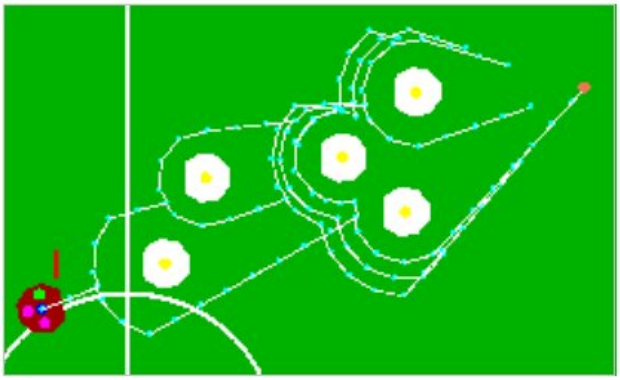

Fig. 5: The tree finds the goal avoiding multiple obstacles

Transceiver Communications Module This module builds the packets sent using our transceiver. The information sent to each robot is the moving vector and the angular speed of each robot.

4 Robots

We designed and built six omnidirectional robots. Each robot has five Faulhaber 2224P0212 motors with gearheads 14:1 (four motors for the wheels and one for the dribbler) [10], a low resistance solenoid, a DSP – Digital Signal Processor, a transceiver, a single printed circuit board and two Lithium Polymer batteries. The height of the robot is 140 mm, the maximum diameter of its projection to the ground is 178 mm, and the maximum percentage of ball coverage is 19%. The robots were designed and manufactured using a the e-Machine Workshop software. The robot receives commands from the AI system in the PC. It includes the following functional elements:

4.1 Robot ID

Each robot incorporates an identification circuit manually setup with a dipswitch making it easy to modify the robot ID if necessary.

4.2 DSP

The robot micro controller is a Texas Instruments TMS320LF2812 fixedpoint single chip DSP. This device offers low power and high-performance processing capabilities, optimized for digital motor and motion con-

trol. The DSP consists of six major blocks of logic: (1) External program and data memory, (2) I/O Interface, (3) Standard I/O, in addition to other modules not currently used in our design. The modules used are:

External program and data memory The RAM module is used in debugging the software with the Parallel Port JTAG Controller Interface.

I/O Interface It contains different kinds of pins: (i) Capture units used for capturing rising pulses generated by the motor encoders which can be used to measure speed and direction of the moving motor. (ii) PWM outputs having an associated compare unit. A periodic value is established to determine the size of the PWM, and the compare value is used to change the duty cycle.

Standard I/O . It is used to read and write values for transceiver communication, motor, kicker and dribbler control.

4.3 Motor Control

The motor encoders generate a number of square pulses for each completed turn. Each pulse is captured using the DSP and the feedback speed is computed into the omni directional-module. To control the motors speed a PWM signal sent back to the motor. This information is obtained by the omni-directional module.

4.4 Wireless Communication

Wireless communication is controlled by two Radiometrix RPC-914/869-64 transceivers with radio frequency at either 914MHz or 869MHz. The transceiver module is a self-contained plug-in radio incorporating a 64kbit/s packet controller with a parallel port interface. Data is transferred between the RPC and the host (either DSP or PC) four bits at a time using a fully asynchronous protocol. The nibbles are always sent in pairs to form a byte, having the Least Significant Nibble (bits 0 to 3) transferred first, followed by the Most Significant Nibble (bits 4 to 7). Two pairs of handshake lines REQUEST & ACCEPT, control the flow of data in each direction.

4.5 Omni-Directional Drive Control Module

This module receives the movement vector including linear and angular velocities from the transceiver. To control the motor speeds two steps are completed: (i) Speeds are read from the motor enconders. The speed of each motor generates the actual linear and angular velocities of the robot. (ii) These velocities along with transceiver velocities are used as inputs to the PID algorithm. There are three independent PID algorithms in the process: the linear speed projection in the x and y coordinates of the robot and the angular velocity. The output of the PID is turned into speeds for each motor (using the motors geometry in the robot) and finally they are controlled to the correct speed with a PWM signal. An illustration of the problem is shown in Figure 6.

4.6 Kicker Control System

Small Size soccer robots use different kicking designs to push the ball. We use a push type solenoid that kicks the ball. Solenoid kicker system needs a high power supply. For size restrictions robots have four 7.4V/ 700mA batteries, equivalent to 31 Watts of power. With this amount of power we obtain less than the solenoid requires for a minimum performance. The main idea in power elevation is to store energy, then discharge it when solenoid is activated. To solve this power problem we implement a fourlayer system as follows:

Voltage transformation The 14.8 dc voltage obtained from the batteries is increased using a (Pico Electronics IRF100S) dc-dc converter [11] to reach 200 volts. The output is used to charge up two 2200mF capacitors. The converter is controlled using a control pin of the DSP with a relay and a transistor. The robot can kick approximately every 25 seconds.

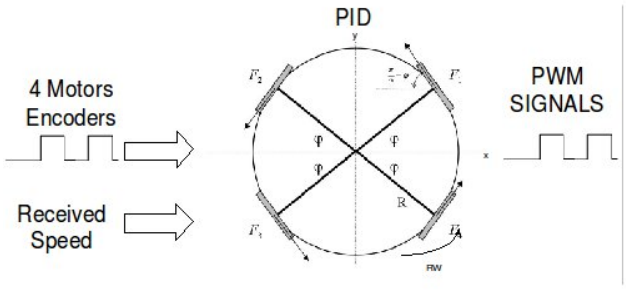

Fig. 6: Motor control using Pulse Width Modulation (PWM) and Proportional-Integral-Derivative controller (PID)

Discharge and solenoid activation An infrared sensor system in the bottom of the robot senses if the robot has the ball. The DSP sends a high-level output bit when the robot is in score position. To discharge the capacitors into the solenoid, the Discharge layer uses both the DSP kick bit and the infrared ball detector output bit to discharge the capacitors. Because the capacitors charge level is very high, the robot discharges it using a power MOSFET. A signal from the DSP, is sent to the RELAY to control the flow of current through it and thus controlling the kick.

5 Conclusions

We presented a software and hardware overview of the SSL Eagle Knights team. Our team has been the first Latin American team consistently obtaining top results in all its regional RoboCup participation, 3rd and 2nd place in US Open 2003 and 2004, respectively, and 1st place in Latin American Open 2004 and 2005. We have also participated in the last five RoboCup competitions: Osaka, Japan 2005; Bremen, Germany 2006; Atlanta, USA 2007; Suzhou, China 2008; and Graz, Austria 2009. We have released the Vision System and documentation of our electronics and DSP software to the public in order to promote the participation of other teams in this initiative. More information can be found at http://robotica.itam.mx/ and http://www.usfrobobulls.org.

6 Acknowledgements

This work is supported by the Asociación Mexicana de Cultura, A.C. and by the Information Technology Department at the University of South Florida Polytechnic.

References

1. RoboCup SSL, laws of the F180 league 2010. http://small-size.informatik.uni-bremen.de/rules:main.
2. S. Zickler, T. Laue, O. Birbach, M. Wongphati, and M. Veloso. SSL-Vision: The Shared Vision System for the RoboCup Small Size League. *RoboCup 2009: Robot Soccer World Cup XIII*, pages 425–436, 2009.
3. Basu A. Elnagar, A. Local path planning in dynamic environments with uncertainty. pages 183 –190, Oct 1994.
4. L.A Martínez-Gómez, F. Moneo, D. Sotelo, M. Soto, and A. Weitzenfeld. Design and implementation of a small size RoboCup soccer team. In *2nd IEEE-RAS Latin American Robotics Symposium*, Sao Luis, Brasil, 2005.
5. David Sotelo. Design and implementation of ITAM's F-180 robots. Technical report, Digital Systems Department, ITAM, Mexico City, Mexico, 2006.
6. Ernesto Torres. Coordination strategies for multi-robot systems in RoboCup small-size league. Technical report, Computer Engineering Dept, ITAM, Mexico City, Mexico, 2009 April.
7. Jesús Rodríguez. Robotics architectures for the RoboCup small-size league. Technical report, Computer Engineering Dept, ITAM, Mexico City, Mexico, June 2009.

8. Ernesto Torres and Alfredo Weitzenfeld. RoboCup small-size league: Using neural networks to learn color segmentation during visual processing. In *ENRI-LARS*, Salvador, Brasil, 2008.
9. OpenGL open graphics library. http://www.opengl.org/.
10. Micromo. http://www.faulhaber-group.com/uploadpk/e_201_MIN.pdf.
11. Pico electronics. http://www.picoelectronics.com/dcdclow/pe88_89.htm.