Eagle Knights 2008: Two-Legged Standard Plaform League

Carlos Ramos, Carlos Rivera, Gibran Rios, Emir Herrera, Marco Morales and Alfredo Weitzenfeld,

Robotics Laboratory Lab, ITAM, Rio Hondo 1,
01000 DF, Mexico
alfredo@itam.mx

Abstract. In this paper we present the system architecture for our Two Legged Standard Platform RoboCup Soccer Team – Eagle Knights. We describe the system architecture for soccer playing in addition to related research projects.

Keywords: Two-legged, standard platform, robocup, autonomous, vision.

1 Introduction

RoboCup is an international effort to promote AI, robotics and related field primarily in the context of soccer playing robots. In the Two Legged Standard Platform League, two teams of four robots play soccer on a relatively small carpeted soccer field.

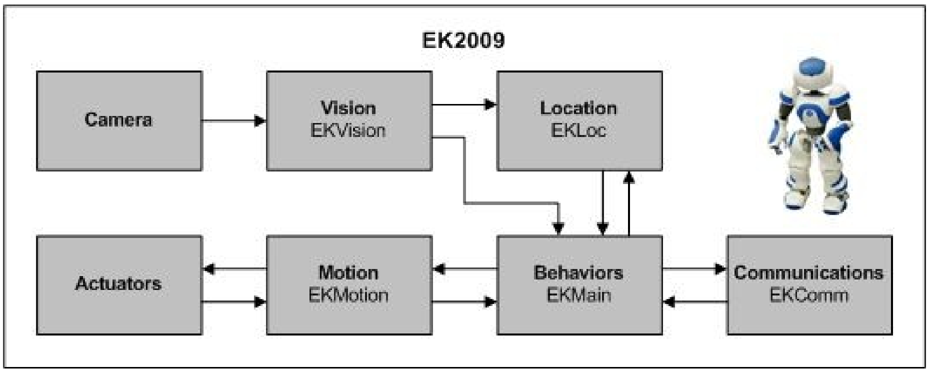


Fig. 1. Two-Legged Standard Plaform System Architecture.

2 RoboCup Two-Legged Standard Platform System Architecture

The Eagle Knights (EK) system architecture is composed of five modules:

- **EKVision**. The vision module receives a raw image from the camera, the main system sensor, and performs segmentation over the image. The module then recognizes objects in the field, including goals, ball and other players.
- **EKMotion**. This module control robot movements, such as walk, run, throw the ball, turn, move the head, etc. It receives commands from the behavior module with output sent to the corresponding actuators representing individual leg and head motor control.
- **EKMain**. This module makes decisions affecting higher level robot actions. It takes input from the sensors and the localization system to generate commands sent to the motion and actuators modules.
- **EKComm**. This module receives all commands from the external Game Controller. The system transmits a data structure between all robots with information such as player id, location of ball if seen, distance to the ball, robot position and ball position.
- **EKLoc**. This module makes all the processing necessary to obtain a reliable localization of the robot in the field. In order to localize, our current model requires the robot to perceive at least two marks

2.1 EKVision

The vision system is divided into three modules: Calibration, Segmentation and Object Recognition as shown in Figure 2.

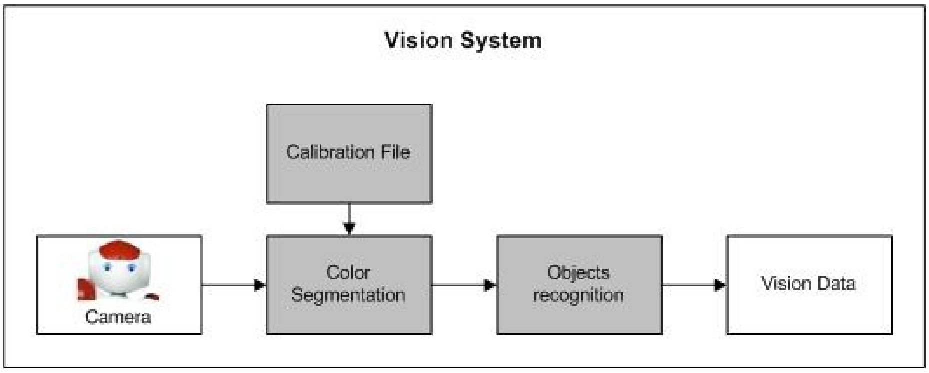


Fig 2. Vision System

Calibration. The system is calibrated prior to the game. The regions of interest are selected in a HSV color space and then transformed to a YUV color space through a GUI that uses pictures previously taken.

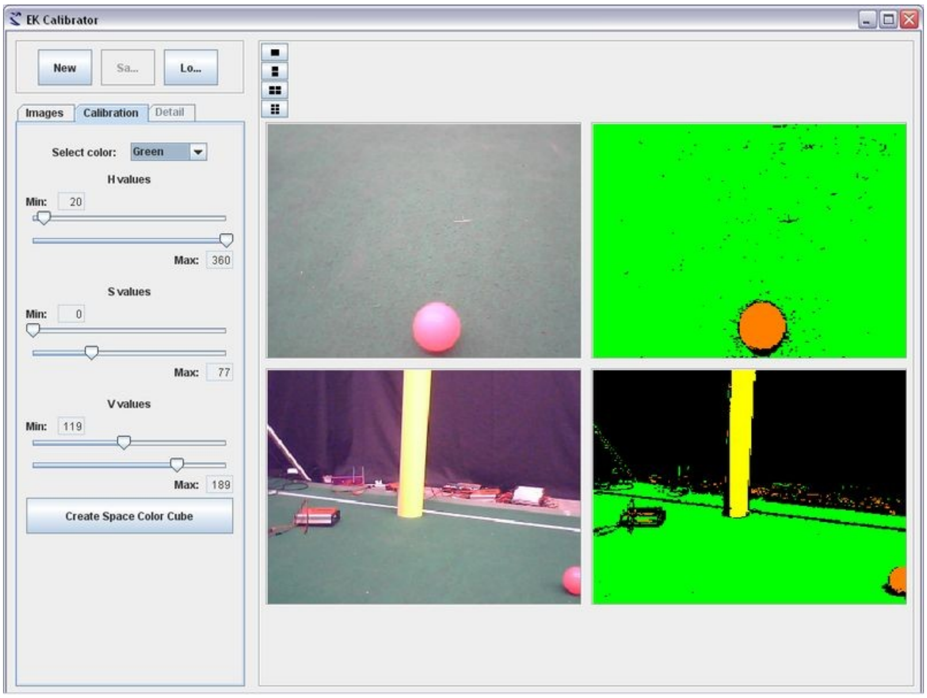
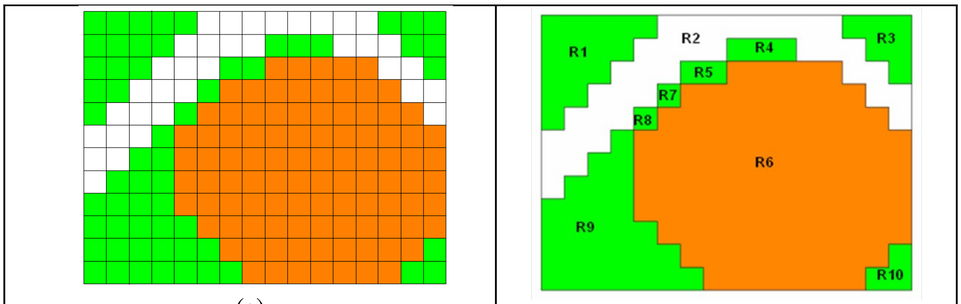


Fig. 3. Software EKCalibrator

We use the HSV model to define initial color range useful during the segmentation process. The result is a 2MB file that contains a table that lists possible combinations of values in the YUV model and colors useful for object recognition. This file is copied to the flash memory of the robot. Figure 3 show the system interface used for the calibration process.

Segmentation. Using the previously defined values, the 7 main colors defined for the game are assigned to objects in each image taken from the field. The segmented pixels are grouped by regions (blobs) for further processing.



(a) (b)

Fig 4: Color segmentation process: (a) individual pixels, and (b) blob regions.

Examples of the color segmentation process are shown in Figure 4: (a) individual pixels, and (b) blob regions. Objects can be recognized after color regions are processed. Objects in the field must fulfill certain requirements in order to allow some confidence that the region being analyzed corresponds to the object of interest. For example, the ball must have green in some adjacent area with similar criteria used to identify goals.

2.2 EKMotion

This module is responsible for robot motion control, including in particular walking and kicking. We have developed a number of different routines depending on team roles. For example, the goalie has different motions in contrast to other team players. This also applies to different head kicks and movements in general.

2.3 EKMain

The behavior module receives information from sensors and localization, sending output to robot actuators. In defining our team robot behaviors, we specify three types of players: Attacker, Defender and Goalie. Each one has a different behavior that depends on ball position and Game Controller.

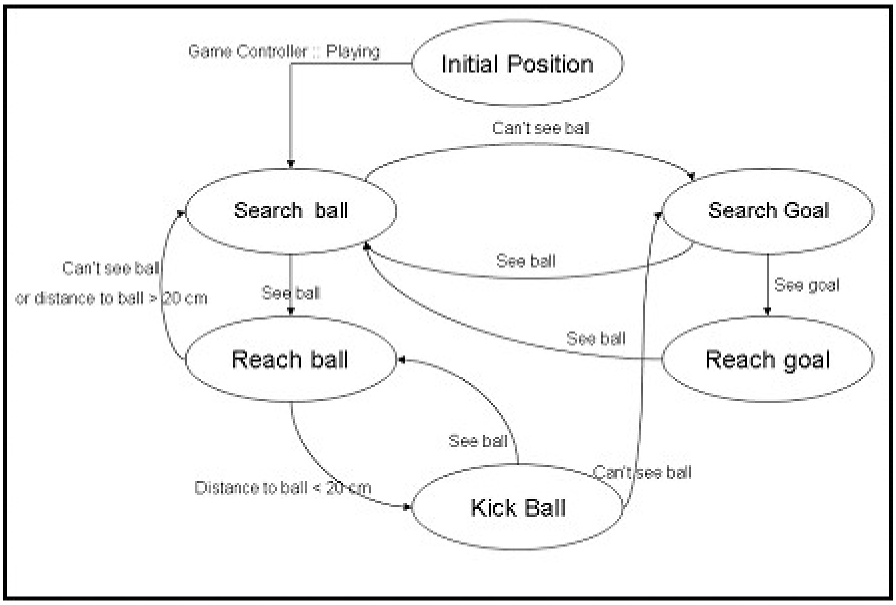


Fig. 5. Basic Individual Goalie State Machine.

Goalie basic behavior is described by a state machine as shown in Figure 5:

- **Initial Position**. Initial posture that the robot takes when it is turned on.
- **Search Ball**. The robot searches for the ball.
- **Reach Ball**. The robot walks towards the ball.
- **Kick ball**. The robot kicks the ball out its goal area.

Attacker basic individual behavior is described by a state machine as shown in Figure 6:

- **Initial Position**. Initial posture that the robot takes when it is turned on.
- **Search Ball**. The robot searches for the ball.
- **Reach Ball**. The robot walks towards the ball.
- **Kick Ball**. The robot kicks the ball towards the goal.
- **Explore Field**. The robot walks around the field to find the ball.

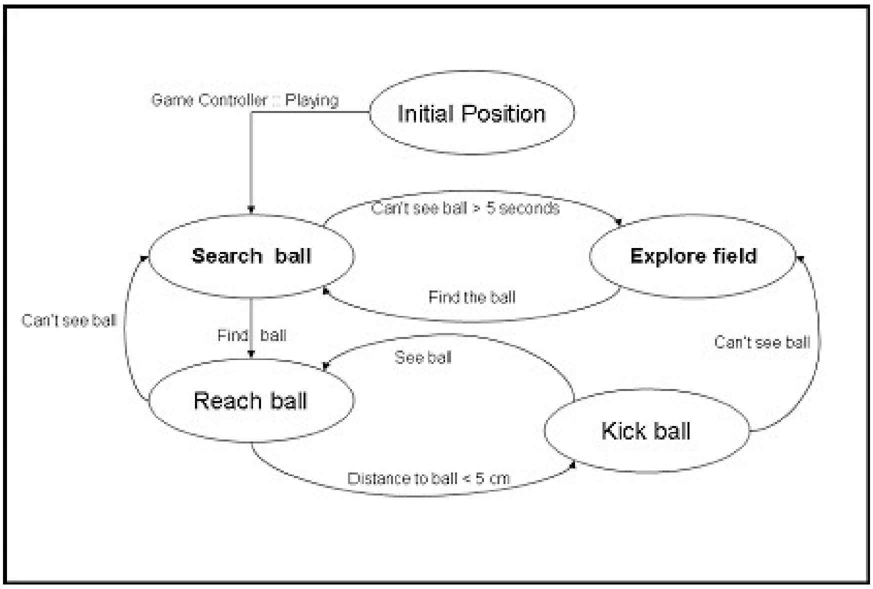


Fig. 6. Basic Individual Attacker State Machine.

2.4 EKComm

This module receives commands from the Game Controller and passes them to the Behaviors module accepting connections using either TCP or UDP protocol. This module is used to cooperation between robots for have information about the state of the game and the world.

2.5 EKLoc

An important part of playing soccer is being able to localize in the field in an efficient and reliable way. Localization includes computing distances to known objects, use of a triangulation algorithm to compute exact positioning, calculate robot orientation angles, and correct any resulting precision errors. A block diagram for the algorithm is showin in Figure 7.
In the next section we describe our Localization system based on 2008 rules. We are currently working on a new version that will adapt to the new rules of the league. The new Localization system is extended from the previous versions and will take into account probabilistic methods in conjunction with odometry data.

Distance to objects

The first step in localizing is to obtain the distance between identified objects and the robot. It is important that the robot can distinguish at least two marks when it starts. After making several experiments, we developed a simple algorithm that computes distances to objects by using a cubic mathematical relationship that takes as parameter the object area and returns as result the distance to the object. In order to obtain this relation, we took a large number of measurements at different distances from the object that we are interested in. The distance range used went from 15 centimeters to 4 meters. Beyond four meters it became very difficult to distinguish between objects and noise.

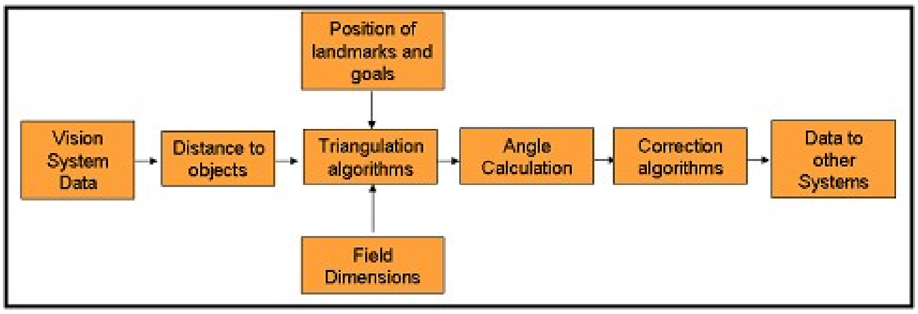


Fig. 7. Localization System block diagram.

Figure 8 shows the area versus distance function together with the resulting standard deviation for this function.

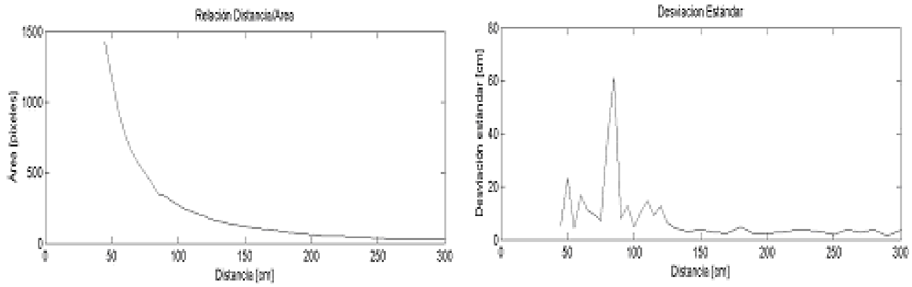


Fig. 8. Upper diagram shows relation between area (Y axis) and distance (X axis), while lower diagram shows the resulting standard deviation.

We chose an interpolation function to match our data. Using Matlab we calculated the coefficients for the cubical segments of the interpolation (splines). Also, with this method, we only need to calculate the coefficients offline and load them in memory when the robot starts playing. When we want to calculate a distance to an object, we just have to evaluate a polynomial expression with the appropriate coefficients and according to the following equation:

$$s(x) = ax^3 + bx^2 + cx + d \qquad (1)$$

We calculated the cubical function coefficients and tested them using different distances. Results from these computations are shown in Table 1.

Table 1. Results of the interpolation function.

Real Distance [cm]	**Computed Distance (average) [cm]**	**Error [cm]**
35	40	5
50	52.07	2.07
65	67.47	2.47
80	82.76	2.76
95	96.98	1.98
110	112.70	2.70
125	124.54	0.45
140	140.39	0.39
155	151.46	3.53
170	170.66	0.66
200	204.63	4.63
230	225.69	4.31
260	250.73	9.26
290	277.49	12.51

Triangulation Algorithm

Following distance computation we apply a triangulation method from two marks to obtain the position of the robot on the field. Triangulation results in a very precise position of the robot in a two dimensions plane. If a robot sees one mark and can calculate the distance to this mark, the robot could be anywhere in a circumference with origin in the mark, and radio equal to the distance calculated. While a single is not sufficient for the robot to compute its actual position, recognizing two marks can already help compute a specific location from the intersection of two circumferences. Note that the robot could be in one of two intersection points in the circumferences.

In Table 2 we show the results of the triangulation algorithm. To test the algorithm we put the robot in an arbitrary position in the field. Then we computed the average distance obtained from multiple measurements followed by an average error calculation. Note the large difference between the true position and the computed average.

Table 2. Results of triangulation algorithm.

Real Position [cm]	Average [cm]	Error [cm]
(50,80)	(80.06,139.28)	(30.06,59.28)
(160,90)	(163.86, 104.44)	(3.86, 14.44)
(265,80)	(278.08,108.39)	(13.08,28.39)
(65,185)	(95.04, 195.87)	(30.04, 10.87)
(170, 190)	(175.25, 196.10)	(5.25, 6.10)
(280, 210)	(301.64, 222.44)	(21.64, 12.44)
(70,290)	(80.93, 303.52)	(10.93, 13.52)
(186,300)	(195.75, 313.52)	(9.75, 13.52)
(285,300)	(301.17, 315.51)	(16.17, 15.51)
(65,380)	(68.53, 414.64)	(3.53, 34.64)
(165,400)	(180.04, 421.87)	(15.04, 21.87)
(290,350)	(312.87, 362.08)	(22.87, 12.08)

Angle Calculation

Once we find the robot position we need to find its orientation to complete localization. We refer to two vectors whose origin is the robot location and the end points are the coordinates of the marks that we use as references for the triangulation.

Correction Algorithms

While testing our algorithm with a moving robot, we noticed that in many occasions our data was not consistent between two contiguous frames. To fix the problem we added a correction algorithm taking historical data from positions already calculated by the robot in obtaining the average of these measurements. We reduced the variation of the output signal for the triangulation algorithm by using the following average filter function:

$$s(x) = \frac{\sum_{i=0}^{n} x(i)}{n} \tag{2}$$

Figure 9 shows sample output from this filter correction. Our original signal produced variations of approximate 10%. By applying this filter we managed to reduce this variation to less than 3%. See [7] for more details.

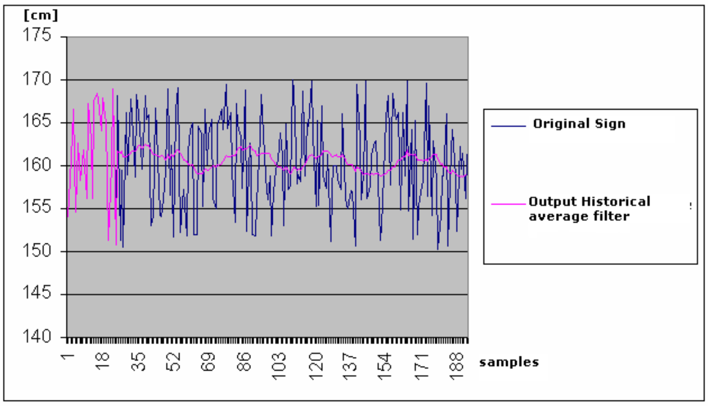


Fig. 9. Historical Average Filter.

After applying this correction algorithm, we tested the system and obtained good results without affecting the performance of the system. These results are shown in Table 3.

Table 3. Final results for the localization system.

Region	Real Position [cm]	Average [cm]	Error [cm]
1	(50,80)	(69.6, 124.8)	(19.6,44.8)
2	(160,90)	(160.27, 92.95)	(0.27,2.95)
3	(265,80)	(270.83, 100.35)	(5.83, 20.35)
4	(65,185)	(81.76, 188.81)	(16.76, 3.81)
5	(170, 190)	(170.47, 193.41)	(0.47, 3.41)
6	(280, 210)	(294.86, 216.79)	(14.86, 6.79)
7	(70,290)	(75.09, 293.53)	(5.09, 3.53)
8	(186,300)	(189.9, 303.53)	(3.90, 3.53)
9	(285,300)	(289.36, 303.19)	(4.36, 3.19)
10	(65,380)	(65.63, 413.74)	(0.63, 33.74)
11	(165,400)	(170, 415.22)	(5.33, 15.22)
12	(290,350)	(305.8, 355.25)	(15.80, 5.25)

Note that errors were computed by field regions, where the complete field was divided into twelve similarly sized areas, as shown in Figure 10. In some regions errors were larger due to changes in illumination. Yet, when compared to the 10cm approximate robot size, worst errors were a bit less than a full body length. Current work focuses in dividing the field into differently sized regions depending on required localization precision.

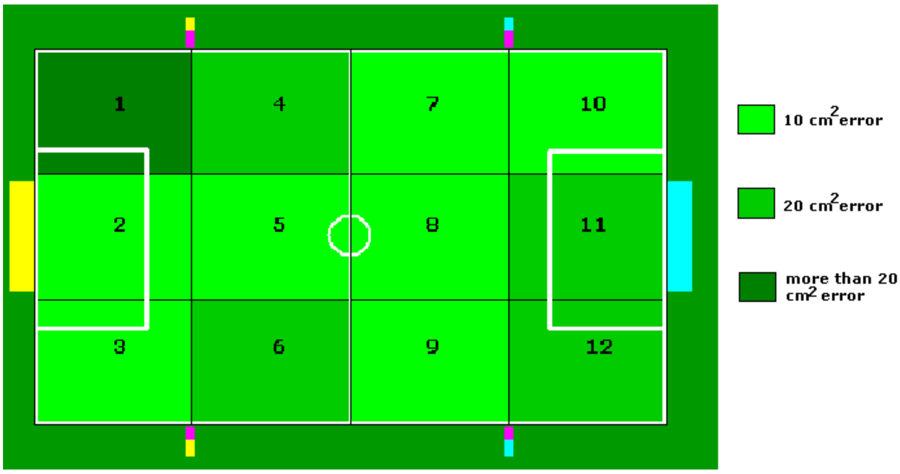


Fig. 10. Localization results by field region.

4 Conclusions

We have presented the system architecture for the Eagle Knights Two-Legged Standard Plaform team with special emphasis on our real time localization system. The system calculates distances with the help of an interpolation function producing good results and in real time. The algorithms used allowed us to estimated a reliable position for the robot without using probabilistic methods like other teams do. We are currently incorporating localization information as part of our game playing strategy while adapting the algorithm to specific regions in the field to produce better qualitative game playing results as opposed to costly numerical accuracy. Our team started competing in 2004 and has since then continuously participated in regional or world events.

This work is part of broader research we are pursuing in the robotics laboratory at ITAM. Other research areas include:

- Spatial cognition in rats and robots (see [1-5]).
- Visual saccades in monkeys and robots (see [6]).
- Prey catching and predator avoidance in frogs and robots (see [8]).
- Robot soccer coaching in standard platform league (see [9-10]).
- Wolf pack hunting in robots (see [11]).

More information can be found in http://robotica.itam.mx.

Acknowledgments. This work is supported in part by "Asociación Mexicana de Cultura, A.C." in Mexico.

References

1. Barrera, A., and Weitzenfeld A., 2006, Return of the Rat: Biologically-Inspired Robotic Exploration and Navigation, BioRob 2006, Feb 20-22, Pisa, Italy.
2. Barrera, A., and Weitzenfeld A., 2006, Biologically Inspired Neural Controller for Robot Learning and Mapping, IJCNN – International Joint Conference on Neural Networks, Vancouver, Canada, July 16-21.
3. Barrera, A., and Weitzenfeld A., 2007, Rat-inspired Robot Spatial Cognition and Goal-oriented Navigation, MED 2007, Athens, Greece, June 26-29.
4. Barrera, A., and Weitzenfeld, A., 2008, Computational Modeling of Spatial Cognition in Rats and Robotic Experimentation: Goal-Oriented Navigation and Place Recognition in Multiple Directions, BioRob 2008, Oct 19-22, Scottsdale, AZ, USA.
5. Barrera, A., and Weitzenfeld, A., 2008, Biologically-inspired Robot Spatial Cognition based on Rat Neurophysiological Studies, Journal of Autonomous Robots, Springer, ISSN 0929-5593.
6. Flores Ando, F., and Weitzenfeld, A., 2005, Visual Input Compensation using the Crowley-Arbib Saccade Model, Proc. International Conference on Advanced Robotics ICAR, Seattle, USA, July 17-20.
7. Martínez-Gómez, J.A, Weitzenfeld, A., 2005, Real Time Localization in Four Legged RoboCup Soccer, Proc. 2nd IEEE-RAS Latin American Robotics Symposium, Sao Luis, Brasil, Sept 20-23.
8. Weitzenfeld, A., 2008, A Prey Catching and Predator Avoidance Neural-Schema Architecture for Single and Multiple Robots, Journal of Intelligent and Robotics Systems, Springer, Vol. 51, No. 2, pp 203-233, Feb, ISSN 0921-0296.
9. Weitzenfeld, A., and Dominey, P., 2007, Cognitive Robotics: Command, Interrogation and Teaching in Robot Coaching, RoboCup 2006: Robot Soccer World Cup X, G. Lakemeyer et al. (Eds.), LNCS 4434, pp. 379–386, Springer-Verlag, ISSN 0302-9743.
10. Weitzenfeld, A, Ramos, C, Dominey, P, 2008, Coaching Robots to Play Soccer via Spoken-Language, RoboCup Symposium 2008, July 14-20, Suzhou, China.
11. Weitzenfeld A., Vallesa, A., and Flores, H., 2006, A Biologically-Inspired Wolf Pack Multiple Robot Hunting Model, LARS 2006, Santiago Chile, Oct 26-27.