Behavior Control for a Humanoid Soccer Player using Webots

J.M. Ibarra Zannatha †, L.E. Figueroa Medina ‡, R. Cisneros Limón †, P. Mejía Álvarez ‡

† *Department of Automatic Control, Cinvestav*
‡ *Department of Computer Systems, Cinvestav*

jibarra@ctrl.cinvestav.mx, lfigueroa@cs.cinvestav.mx, rcisneros@ctrl.cinvestav.mx, pmalvarez@delta.cs.cinvestav.mx

***Abstract*—In this paper the behavior control system developed for a humanoid soccer player is presented, according to the current rules established by the International RoboCup Federation. It is important to point out that this work deals only with field players, excluding the goalkeeper. The behavior control is modeled as a finite-state automaton, which comprises activities as: visual perception (self localization, ball localization, movement estimation), behavior control and game strategies. The developed system is implemented in Webots, which is the official simulation platform used in the RobotStadium league of RoboCup. The results obtained are shown and the path to improve the developed system in our humanoid soccer players is established as future work.**

***Keywords*—Nao, Webots, RoboCup, Automata**

I. Introduction

Since 1996 the International RoboCup Federation organizes soccer matches between teams of robots. At the begining there were only two categories of wheleed mobile robots being considered (small and medium size), but soon legged robots' categories were also of considered: AIBO, a four-legged from Sony and humanoid robots.

By the year 2000, the RoboCup Initiative stated its ultimate goal: That by the year 2050, a team of fully autonomous humanoid robot soccer players shall compete, and even win, a soccer game against the winner of the most recent World Cup, considering the rules stated by FIFA. Nowadays, RoboCup considers five categories (or leagues) of humanoid robots: RobotStadium, Standard Platform, Kid Size, Teen Size and Adult Size [1]. We are interested in the Kid Size category, in which we have competed twice along with La Salle University: RoboCup 2009 held in Graz, Austria (15th place) and RoboCup 2010 held in Singapore (10th place).

P. Mejía is a professor at the Department of Computing Systems in Cinvestav, place that is attended by the student L.E. Figueroa. The last one is carrying out his Master's Thesis at the Robotics and Artificial Vision Laboratory of the Department of Automatic Control at the same institution, under the supervision of J.M. Ibarra and R. Cisneros. The first one is a professor in that department, meanwhile the second one is working there as a research assistant.

Considering the las three categories, each team can compete with prototypes that can be acquired or developed. In either case, these robots have to meet the current regulations. With respect to the Standard Platform, all the teams use the same hardware (Nao from Aldebaran), Meanwhile, the RobotStadium category is held completely on simulation over the Webots®. However, comparing the five leagues a constant fact is found: robots must have a visual perception system that allows them to evolve efficiently on the field, and a behavior control system.

Indeed, with respect to the context of soccer matches between humanoid robots the visual perception is crucial. In fact, it can be assured that this visual system is one of the most important components in the robot, as practically all the most important tasks it has to perform relay on it. In order to comprehend this statement we can try to imagine ourselves in a blindfolded fashion inside the soccer field trying to find the ball, our team mates, our opponents and, appart from that, locate ourselves inside this field, shoot against opponent's goal and score.

In this way, soccer robots shall have an artificial vision system capable of carrying out at least one of the following high level functions: (i) self localization, (ii) ball localization, (iii) correction of the relative position of the robot with respect to the ball, in order to kick towards the opponent's goal, and (iv) ball's movement estimation. It is worth to point out that some of these tasks imply the need for a tridimensional monocular reconstruction, i.e. based on one frame [2] [3].

With regard to the importance of the Behavior Control System (BCS), try to imagine the result of the game if at least one of the players wouldn't have any idea of what a soccer game is. Probably, this neophyte player won't be able to collaborate with all the other players of his team. He will be just a nuisance with respect to the course of the match. Hence the importance of providing these soccer humanoid robots with a game strategy and the capability of deciding in time what to do given some circunstances in order to win the match.

Among the most important functions to be performed by the BCS are to decide what to do on the field considering the rules of the game, the signals issued by the referee, the global situation of the game and the game strategy previously defined. What follows is a series of particular problems to be solved by the BCS presented, starting from the general functions just established.

A. Problem statement

In order to have a humanoid robot capable of becoming a good soccer player, a BCS with a minimum number of functions working in an efficient manner is needed: (i) self visual localization (determined by the position and orientation of the robot in the field of play); (ii) ball localization and its movement estimation; (iii) generation of specific movement patterns (simple gait over a straight line, orientation and turning, kicking, rising, complex trajectories, throw-in, etc.); (iv) driving the right movement pattern needed in every moment; (v) reception and listening to the signals sent by the referee box. If, moreover, we are capable to solve the problem of detection of opponent robots, the localization of our team members and being able to communicate with them, then even more advanced game strategies would be feasible to be implemented.

In this work we develop a methodology for perception, decision making and determination of movements and control, that were implemented in simulation with the aid of Webots®. This software is the official one used at the 3D simulation tournament in RoboCup. Inside this simulation software it is possible to implement since techniques for image analysis to the ones needed for behavior control as well as a low-level articular control, all of these with the aid of a 3D graphical user interface (GUI).

Inside this GUI it is possible to visualize the geometric model of standard mobile robots and created ones. This will let us simulate the dynamics of the robots that are already in construction. The use of this GUI is an efficient and objective way to evaluate the algorithms developed by our team concerning perception, movement control and behavior control.

B. Proposed methodology

The developed BCS is based on a finite-state automaton that comprises 8 states and 13 transitions, covering in this way the great majority of game conditions, without being exhaustive. The proposed states call a series of movement routines, proceedings for image analysis and algorithms for trajectory calculations and localization by triangulation. Besides, the use of the professional simulation platform Webots® is proposed, inside wich all the situations for a real game are considered, including the dynamic reactions calculations based on ODE. The cameras used in simulation have also image analysis problematics that arise in practice, even though it is not necessary to implement advanced image processing filters as there are no ilumination problems as in practice, situation that simplifies the developed artificial vision schemes [4].

C. Paper organization

After a brief introduction, section II of this paper presents a short description of soccer game attending to the current RoboCup rules, whereas section III is used to expose, in a superficial manner, the simulation platform being used, that is Webots®. Section IV presents the proposed solution to control the behavior of the player in the field, which is based on a finite-state automaton. In section V the implmentation of all algorithms and proceedings used for the movement and visual percepction of the robot are shown. And finally, in section VI the conclusions and the future work that we will follow on are both presented.

II. Description of the rules stated by RoboCup

The aim of this section is to present the current soccer game's rules between humanoid robots as stated by RoboCup, such that we can plan the behavior that our players must exhibit in order to effectively compete in the corresponding tournaments governed by these rules.

It is worth to mention that RobuCup's rules pretend to guarantee a fair competition that promotes the creativity and technological development. These rules are based on FIFA's ones but they are still very different. However, in the future both must be equal in order to fulfill RoboCup's main objective by the year 2050. Nowadays, two main technological challenges are the robot's dynamic stability for every movement, as well as the correct coordination between perception and locomotion.

A. About the field of play

According to our league of interest, the field size is $4m \times 6m$. It consists of a flat and even ground wich is covered with a green carpet. The field is delimited by white lines that are $5cm$ wide, which are also used to mark the central line and circle, the goal areas and the penalty-kick mark, as it can be seen in Figure 1. The goals are made of tubes of 10 cm diameter, one painted of yellow and the other painted of blue. There are, besides, two posts at each end of the central line which are made of the same material as the goals. Each tube, known as *beacon*, is 45 cm height and is divided in three equal segments, as it can be seen in Figure 2. The lowest and highest segments are colored in the same color as the goal at its left side. The middle segment is colored in the same color as the goal at its right side. More details of the field of play can be found on [5].

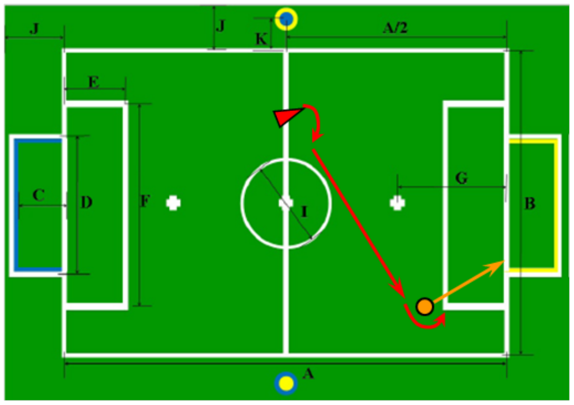

Fig. 1. Field of play according to RoboCup's rules

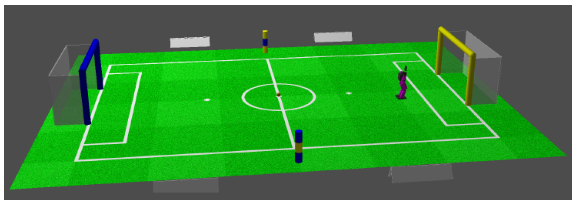

Fig. 2. Detail of goals and beacons

B. About the players

Nowadays, each team is made up of three robots: one goalkeeper and two field players. The main restrictions in the Kid Size category with respect to the size of the robots are the following [5]:

- Height of the robot H_{TOP} and height of its center of mass H_{COM}, such that if we define $H = \min\{H_{TOP}, 2.2H_{COM}\}$ then $30cm < H < 60cm$.
- The size of the feet. Each one must fit into a rectangle of area $H^2/28$, considering that the ratio between the longest side of this rectangle b and the shortest one a shall not exceed 2.5; i.e. $b < 2.5a$.
- The extension of the arms maximally stretched in horizontal direction must be less than 1.2 H.
- The length of the legs H_{leg} must satisfy $0.35H < H_{LEG} < 0.7H$.
- The height of the head, measured from the first joint at the shoulder, must be $0.05H < H_{HEAD} < 0.25H$.

The robots must be black or dark grey and not reflective. Only 10% of its surface can be of color white or light grey and reflective, and only 1% of its surface can be of any other color. Arms, legs and any link of the robot must be of solid shape appearance. These robots must be marked with a color (cyan or magenta) that identify all members of a team. The team's color is decided prior to a every match. Besides, each player is identified with its name or number. The goalkeeper must be clearly identified.

With respect to the perception system, any active external sensor is prohibited. The only active sensorial information allowed is audio on the frequencies and amplitudes that are audible to the average human. Eyes (monocular or stereoscopic cameras), ears and voice (speakers) shall be located at the head, altough the last one can be located at the trunk. Touch, force, temperature, voltage, current, acceleration, velocity or movement sensors can be located everywhere else. The field of view of the robot is limited at any time to 180 degrees, while the neck's movements must have a range similar to a human. Robots can communicate via WLAN using a limited bandwidth (1 MBaud) provided by the referee box. The last one gives all control orders of the game (kick-off, penalty, free-kick, etc).

Regarding security aspects, robots shall not cause any damage to people, other robots or the field of play. Otherwise, they might be expelled from the game, or even from the tournament. Finally, robots must mantain structural integrity throughout the game. Ther perception system must be able to tolerate significant levels of noise and disturbance caused by other players, the referees, robot handlers, and the audience [5].

C. About the game's rules

The game is played with a standard tennis orange ball into two equal periods of 10 minutes each considering a half-time of 5 minutes. For knock-out matches ending in a draw after regular time, extra times of 5 minutes and even penalty kicks will be used to determine the winner of a match. If the current score in a match has a goal difference of 10 goals, then the referee will terminate the match and the score will be recorded as the current one. The rules include the referee's role and everything regarding to fouls, goals, etc. We have only presented a brief summary, so we refer the reader to reference [5].

III. Webots

Webots® is a robotics simulator developed at the École Polytechnique Fédérale de Lausanne (EPFL), Switzerland by Olivier Michel. This platform is widely used, particulary at the RobotStadium competition in RoboCup. It uses Open Dynamics Engine (ODE) to simulate the dynamics of rigid bodies and collisions between them. It contains a wide library with 3D models of commercial robots and it is relatively easy to add new ones developed by us. We just have to consider geometric properties (shape, dimensions, position, orientation, color, texture) as well as physical ones (mass, friction, spring and damping constants).

Webots® provides us with frequently used sensors in robotics: proximity, light, touch, force, accelerometers, gyroscopes, compasses, cameras, GPS. It provides us also with actuators and servos (rotational or linear), grippers, LEDs, etc.

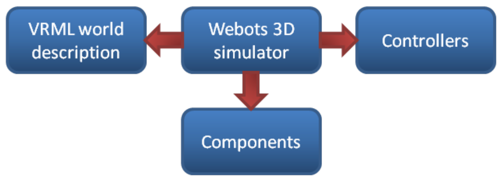

Fig. 3. Structure of Webots®

Webots® features an interactive virtual space based on VRML, which can be programmed using C, C++, Matlab, Python, Java or Urbi. The simulation results can be saved as PNG images (screenshots) or MPEG / AVI videos. The interaction between a user and the virtual world can be made even when the simulation is running [6].

A. Structure

Webots® has a structure that consists of three parts, as can be seen on Figure 3:

1) *The VRML world description*
   VRML is a file format capable of representing interactive virtual worlds which can contain several 3D objects of any complexity. The virtual scenes created for Webots® contain the geometric and physical properties of all objects within it.
2) *Components*
   Every element present in the virtual word (humanoid soccer players, the field of play, goals, ball, etc).
3) *Controllers*
   Binary files that command the behavior of every object within the virtual world. These can be executables, binary files written in Java, C, C++, Python or Matlab, among others.

B. Nao robot soccer application

Our problem consists on simulating a soccer game using Aldebaran's Nao robot as a field soccer player. Using Webots® our simulation project is implemented through 4 levels as described below:

1) *Operating System*
   Any of the three most common OS can be used: Windows, MacOS or Linux. We decided to use Windows because of the advantages that it offers with respect to the familiarity it has with most users and its ease of installation and set up.
2) *Sotware Development Kit*
   Actually, the trend is to produce software development kits (SDK) that simplify application's programming tasks and robotics applications is not the exception. Aldebaran developed two platforms of this type for the Nao robot: Choreographe and Nao SDK; however we decided to use the 3D model of the Nao robot that is included with Webots®.
3) *Programming Language*
   Nao, as every robot simulated in Webots® can be programmed with C, C++, Java, Python or Matlab. We chose C++ because of the convenience of using the object-oriented paradigm when specifying the soccer player's behavior. In this way, the code produced can be conveniently uploaded to the real robot.
4) *Application*
   The developed applications included visual perception (objects recognition, 3D distance estimation, trajectories calculation and visual autolocalization among others) as well as movement applications (walk, kick, stand up).

IV. Behavior Control System

First of all, we distinguish two types of players: goalkeeper and field players, each of them has unique features that distinguish one from another. The goalkeeper functions are to prevent the ball to enter into its goal and keep it away when it is near him. It can touch the ball with any part of its body and retain it for at most 5 seconds. In turn, the field player must score as many goals as it can as a primary function and prevent the opposite team to do the same. It can't touch the ball with any other part of its body rather than its feet. Using the rules of the game, its objectives and the role of the different players it is possible to design a finite-state automaton that implements the activities to be performed by each type of player.

A. Deterministic Finite-State Automaton

Deterministic Finite-state machines or automata (FSM) are a very useful tool to specify aspects related to real time, autonomous domains, reactive computing, protocols, circuits, software architectures, etc. The FSM model has a formal syntaxis and semantics that permits the representation of dynamical aspects that wouldn't be possible to represent using any other graphical representation [7].

1) Syntaxis: Deterministic finite-state machines ar defined as a 4-tupla $\langle \sum, S, s_0, \delta, \rangle$, where:

- $\sum$: Input alphabet (a finite, non-empty set of symbols).
- S: Finite, non-empty set of states.
- $s_0 \in S$: Initial state.
- $\delta : S \times \sum \rightarrow S$: State-transition function.

2) Semantics: S represents all possible states of the system being modeled. $\sum$ represents the events that cause a change. Node-transition functions δ determine how each state, given an event, results in any other state.

3) Example: Consider the behavior of a field soccer player as defined in the following subsection by means of Table I. The corresponding FSM is defined as:
$\langle S = \{E_1, \ldots, E_8\}, \sum = \{T_1, \ldots, T_{13}\}, s_0 = E_1, \delta = \{(E_1, T_1) = E_1, \quad (E_2, T_2) = E_2, \quad (E_1, T_{12}) = E_8, (E_2, T_3) = E_3, \quad (E_2, T_{12}) = E_8, \quad (E_3, T_4) = E_3,$

$(E_3, T_5) = E_4, \quad (E_3, T_{12}) = E_8, \quad (E_4, T_6) = E_5,$
$(E_4, T_7) = E_6, \quad (E_5, T_8) = E_3, \quad (E_5, T_9) = E_6,$
$(E_5, T_{12}) = E_8, \quad (E_6, T_8) = E_3, \quad (E_6, T_{10}) = E_7,$
$(E_6, T_{12}) = E_8, \quad (E_7, T_{11}) = E_1, \quad (E_7, T_{12}) = E_8,$
$(E_8, T_{13}) = E_1\}\rangle$

Figure 4 shows the graph that represents this FSM. Formally, the finite-state machine is the 4-tupla just described, not the graph itself. The last one is only a graphic representation that let us visualize its content in a simple manner.

B. FSM for the Field Soccer Player

Considering all the activities that the field soccer player must perform, we have selected the most important ones, those that are essential to ensure a good performance for the corresponding player. These activities are listed below as a FSM whose states and transitions (Figure 4) are described:

TABLE I
STATES AND TRANSITIONS OF THE FSM

States	Transitions
E_1 Seek for the opposite goal	T_1 Goal not found
E_2 Self localization	T_2 Goal found
E_3 Seek for the ball	T_3 Robot located
E_4 Ball localization	T_4 Ball not found
E_5 Walk to the ball	T_5 Ball found
E_6 Adjust position	T_6 Ball is away
E_7 Kick to goal	T_7 Ball is close
E_8 Stand up	T_8 Ball is lost
	T_9 Ball at the feet
	T_{10} Robot is positioned
	T_{11} Succesful kick
	T_{12} Robot has fallen
	T_{13} Robot has stood up

E_1. *Seek for the opposite goal:* Having analyzed the captured image taken at the current robot's position, it has to look for the opposite goal. There are four possible cases: (i) both posts of the opposite goal are found, (ii) only the left post is found, (iii) only the right post is found, (iv) no post is found.

T_1. *Goal not found:* It must be triggered if the test performed during state E_1 arises one of the last three cases posed. A second image must be captured by means of a sweep and then the FSM must return to state E_1.

T_2. *Goal found:* It must be triggered if the test performed during the E_1 state arises the first case posed. So, the FSM must switch to state E_2.

E_2. *Self localization:* In this state, the goal's position in the image is measured and, with this information, the robot's position inside the field of game is calculated [2] [3].

T_3. *Robot located:* This transition switches to state E_3.

E_3. *Seek for the ball:* At the robot's current position another image is captured in order to seek for an orange spot; i.e. the ball. There are two possible cases: (i) the ball is detected and (ii) the ball is not detected.

T_4. *Ball not found:* As the ball was not found, a second image must be taken by means of a sweep of the head (using neck movements). The FSM must return tu state E_3.

T_5. *Ball found:* Once the ball is detected on state E_3 the FSM must switch to state E_4 in order to calculate the ball's position.

E_4. *Ball localization:* The ball's position is calculated with respect to the robot's reference frame as well as the field's absolute one. There are two possible cases: (i) the ball is far away or (ii) the ball is near the robot's feet.

T_6. *Ball is away:* The FSM switches to state E_5.

T_7. *Ball is close:* The FSM switches to state E_6.

E_5. *Walk to the ball:* Once the robot and the ball are located inside the field of game, the robot must walk to the ball. During its journey two different situations may happen: (i) the ball remains in the field of view of the robot; that is, near the vertical line at the middle of the image, or (ii) the robot lose sight of the ball.

T_8. *Ball is lost:* In this case the robot stops its movement and the FSM switches to state E_3.

T_9. *Ball at the feet:* If the ball still remains in the field of view of the robot at the end of the journey the FSM switches to state E_6 in order to kick the ball.

E_6. *Adjust position:* In this state the robot seeks for the first post of the goal and adds an offset to point to the center of it, in order to adjust its position. If the ball remains in the field of view the FSM switches to state E_7 through transition T_{10}. Otherwise, transition T_8 is triggered.

T_{10}. *Robot is positioned:* By means of this transition the FSM switches to state E_7.

E_7. *Kick to goal:* The robot kicks the ball directly to the goal, in the direction previously calculated.

T_{11}. *Succesful kick:* Finally, after switching through al the states, the FSM arrives to the final one: the kick. And then, it returns to the initial state E_1.

T_{12}. *Robot has fallen:* From virtually any state (E_1, E_2, E_3, E_4, E_5, E_6, E_7 and even E_8) the robot may fall or be knocked down. When the corresponding sensors detect this situation the current task (movement or sensorial) must stop and the FSM must switch to state E_8.

E_8. *Stand up:* The robot must stand up by means of a routine that considers which side of it is looking up. Once it has stood up the FSM must switch to E_1 by means of transition T_13.

T_{13}. *Robot has stood up:* The FSM switches to state E_1.

V. Implementation and Results

The automaton just described refers to movments that the robot must execute, to processes of image analysis and certain algorithms that allow robot's visual self localization, as well as ball localization inside the field of game. These are described below:

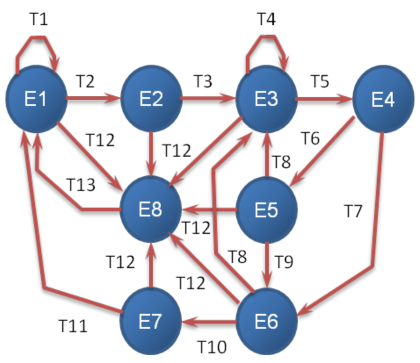

Fig. 4. FSM that represents the soccer player's behavior

A. Movement routines

1) Gait: In order to calculate the number of steps needed to arrive to certain location the desired longitude si divided by the step size and the result is truncated. The joint parameters that determine each step are obtained by using the kinematic simulator ARMS [8] [9]. This routine is based on the one that is delivered with Webots®. This executes one step or more, stopping before walking any more.

2) Position adjustment: A routine that performs a turn without changing position was also designed. This lets the robot to correct its orientation within intervals of 10 degrees approximately. In this way the maximum orientation error is 5 degrees. In order to perform that adjustment little lateral steps are used in combination with a small forward step of one foot and a small backward step of the other, as a function of the desired turn. It is basically a combination of various basic default movements.

3) Stand up: Another routine that lets the robot to stand up after it has fallen was also developed. This routine was programmed by means of a manual adjustment of the articular configuration, in an heuristic way. It was one fo the most complicated tasks and includes a rolling routine that is performed when the robot faces the ground. That is because it can stand up only when it faces up.

4) Kick: Webots® includes a default kick routine that was improved in order to achieve a larger kick distance (from 1.3 m to 2.2 m). This default routine performs a kick with only one foot. By means of a program in Pearl the corresponding movements were mirrored in order to obtain a kick routine for the opposite foot. Besides, a lateral kick routine using the internal part of the foot was also programmed.

5) Sweep: A sweep routine of the head to seek for the goals was also programmed. This is performed by turning the head about the azimuthal coordinate by an angle of 40 degrees during each iteration, in order to ensure an overlap of successive images (taking into account that the field of view is 46 degrees). By doing this there is a possibility that both posts of a goal can't be simultaneously seen in only one image, so every time a post is detected its location is saved, as well as the joint configuration of the head.

To seek for the ball this sweep routine is also executed. And in the case that the ball can't be found an angle of 30 degrees is added to the elevation coordinate of the head, and the azimuthal sweep is executed again.

6) Ball tracking: Once the ball is detected, it must be tracked to the center of the image. This is done by performing the right neck movements.

B. Perception routines

1) Goal: The robot's position (x, y) is unknown, but the image frame is parallel to the robot's frame and then an image is captured. The optical axis is completely horizontal, so all vertical lines of the scene (the posts of the goals) preserve its parallelism on the captured image. So, it's enough to explore a narrow horizontal fringe (between 5 and 11 pixels wide) at the center of the image in order to look for the corresponding color of the opposite goal (yellow or blue). When the color sought is found then the robot knows that a post has been found; so if it explores above this zone then it will be able to know which post it is looking at (the right one or the left one), by means of a comparison of the pixels on the left and the right of each side of this segment (Figure 5).

2) Ball: Having oriented its optical axis down (45 degrees below an horizontal line) and to the front, an image is captured to seek for the orange spot; that is, the ball, by means of a color segmentation process.

3) Robot has fallen: The robot has accelerometers that permit to detect if it has fallen as well as the pose it has when it is on the floor. If it faces up then he can stand up; otherwise, he must roll in order to do it.

C. Localization algorithms

1) Robot's localization: Using vision, it is possible to calculate the relative position of the posts of the goal with respect to the image frame and, as the joint variables of the neck are known, the posts' position can be described in the robots main frame. Besides, the map of the field of play is known (Figure 1). In this way, it is possible to apply simple triangulation (or trilateration) algorithms to obtain the absolute position of the robot as well as its orientation [2] [3].

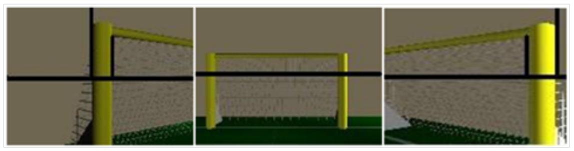

Fig. 5. Three typical images of a goal

2) Ball's localization: Once the robot has tracked the ball to the center of the image using the head's azimuthal and elevation coordinates (α, β), it is possible to find the ball's position (x_p, y_p) with respect to the robot's frame (Figure 6):

$$\begin{aligned} \rho &= h \sin \alpha \\ x_p &= \rho \cos \beta \\ y_p &= \rho \sin \beta \end{aligned} \tag{1}$$

The height h of the camera is obtained by calibration, the angles (α, β) correspond to the azimuthal and elevation coordinates of the head, and ρ is the distance between robot and ball. As the position of the robot with respect to the field of game is already known, it is possible to calculate the ball's absolute position with respect to the field's frame.

3) Kicking direction: Once the robot and the ball have been located inside the field of game, it is necessary to define the kicking direction. To do it it is necessary to determine: (a) the angle and distance that the robot has to walk in order to get into a feasible kicking position, and (b) the kicking direction needed with respect to that position to head the ball to the first post plus an offset to the center of the goal. Using this information the robot is commanded to walk and kick the ball once it is properly oriented. Figure 1 shows an example of this proceeding.

D. Results

To test the functionality of this algorithm the ball was manually placed into an arbitrary position and the robot into another one (Figure 1). The algorithm was always able to score a goal. During the robot's gait the ball was frequently changed from one place to another. However, the robot was able to change its path and score again, even when it sometimes fell down.

VI. CONCLUSIONS AND FUTURE WORK

A. Conclusions

In this paper a behavior control scheme for a robot soccer player was presented (field soccer player), based on a finite-state automaton that was implemented on Webots®. This project was developed as a part of a Master's Thesis [10].

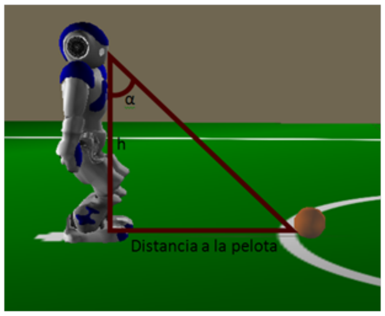

Fig. 6. Ball's position with respect to the robot

The number of states considered for the automaton is relatively small (only eight), and not all the posibilities are considered. However, the robot is capable of score starting at any location inside the field of game and any ball's location, exhibiting an appropriate behavior to perform its tasks.

The use of Webots® as a development platform has the following added advantages: (i) to compete in the RobotStadium league in RoboCup, which consists of a soccer game inside Webots®, (ii) upload our behavior control system to the real robot Nao and compete in the Standard Platform league, or (iii) use this automaton within the robots already developed to compete in the Kid Size league. Considering the last two cases, all the image processing algorithms that were not used with the Webots® version must be included. That's because the images in the last one are artificial, so it is enough to use simple image analysis schemes. On the other hand, real images depend on ilumination and other elements that cannot be controlled.

B. Future work

This project is still trying to improve the performance of the behavior control scheme. Firstly, we have to develope the goalkeeper version for the automaton and add more states in order to include all the game's possibilities: in particular, all the ones that refer to the referee box (start of the game, goal, throw-in, end of the game, etc). Besides, we shall work on topics as: collaboration between robots and reaction behavior caused by the opposite team. The gait's stability must be improved and robot's displacements must be optimized, i.e. using more elaborate trajectories (not just simple turns and straight movements that requires more time). Finally, we must use more elaborate self localization schemes to achieve better results and to improve the image processing and analysis schemes.

REFERENCES

[1] Robocup. http://www.robocup.org, may 2010. (last checked).
[2] Felipe Lara and Andrés Espínola. Image processing and analysis for humanoid soccer players. *CoMRob*, 2010.
[3] Juan Manuel Ibarra, Rafael Cisneros, Ángel Gómez, Eric Hernández, Enrique Figueroa, and Felipe Lara. Moocular visual self localization for humanoid soccer players. *CONIELECOMP*, 2011. (waiting for acceptance).
[4] Cyberbotics. http://www.cyberbotics.com, may 2010. (last checked).
[5] Robocup soccer humanoid league: Rules and setup for the 2010 competition. http://www.tzi.de/humanoid, mar 2010.
[6] Cyberbotics. Webots reference manual, apr 2009.
[7] E. Roche and Y. Schabes. *Finite-State Language Processing*. The Massachusetts Institute of Technology, 1997.
[8] Juan Manuel Ibarra Zannatha and Rafael Cisneros Limn. Forward and inverse kinematics for a small-sized humanoid robot. *CONIELECOMP*, 2009.
[9] Rafael Cisneros Limón. Strategies of kinematic modeling and simulation of humanoid robots. Master's thesis, Automatic Control Department of CINVESTAV, 2009.
[10] Luis Enrique Figueroa Medina. Behavior simulation for humanoid robots inside a soccer game, 2010.