Reactive-behavior Learning of Quadruped Robots Playing Soccer

Nikola G. Shakev[1], Andon V. Topalov[1], Okyay Kaynak[2] and Levent Akin[3]

1) Department of Control Systems, Technical University Sofia, Plovdiv branch, 61, St. Petersburg Blvd, Plovdiv 4000, Bulgaria, e-mail: topalov@mbox.digsys.bg

2) Department of Electrical and Electronic Engineering, Bogazici University, Bebek 80815, Istanbul, Turkey, e-mail: kaynak@boun.edu.tr

3) Department of Computer Engineering, Bogazici University, Bebek 80815, Istanbul, Turkey, e-mail: akin@boun.edu.tr

Abstract

The robotic soccer belongs to the class of multi-agent systems and involves many challenging sub-problems. Teams of robotic players have to cooperate in order to put the ball in the opposing goal and at the same time defend their own goal. The paper is concerned with the problem of learning and implementation of reactive behaviors for robotic agents playing soccer. It briefly presents the whole control system designed for the Cerberus'01 Sony legged robot team that participated in RoboCup 2001 competitions in Seattle, USA, and then introduces the developed reactive behavior for interception of the moving ball while avoiding collisions with other robotic players and play field walls. For the implementation of the above behavior a fuzzy-neural trajectory generator (FNTG) has been developed and learned. Genetic Algorithms (GAs) - based approach has been employed to perform the learning process of the proposed FNTG.

Introduction

Robotic soccer as first proposed by [1], is a challenging research domain that is particularly appropriate for studying a large spectrum of issues of relevance to the development of complete autonomous agents [2] – [4]. The Sony Legged Robot League is an international robot competition that has been launched within the RoboCup initiative (http://www.robocup.org). Sony's quadruped robot AIBO has been adopted as a hardware platform, so the competition in this league is on a software level. The "Cerberus'01" team made its debut in RoboCup 2001 competition. This was the first international team participating in the league as result of the joint research effort of a group of students and their professors from Bogazici Univerity (BU), Istanbul, Turkey, and Technical University Sofia, Plovdiv branch (TUSP), Plovdiv, Bulgaria. The project was supported by the Bogazici University Research Fund. Vision, Localization and Decision making modules have been developed by BU while the Locomotion, Behavior and Communication modules have been designed in TUSP. The paper emphasizes on the developed behaviors and the locomotion part, which form together the implementation level of the control system of the Sony's quadruped robot. The robotic dogs AIBO used in the research project have been specially modified by Sony Corp. to allow reprogramming. They provide a completely new test bed for conducting research in the field of servicing and entertainment robotics, artificial intelligence and multi-agent systems. The robots could be used to study many fundamental problems like behavior adaptation, human-like thinking, evolution and learning.

The AIBO dog is a fully autonomous robot. The onboard 64-bit RISK processor provides enough computational power to perform image processing, localization and control tasks in real time. The competition rules do not allow to remotely controlling robots in any way, and acoustic communication is only permitted within this multi-robot system consisting of three robotic players. The only information available for decision-making comes from the robot's onboard TV camera, built in proximity sensors, sensors, reporting the state of the robot's body (like the built in acceleration sensor and gyroscope) and from the messages received from the teammates.

The robot consists of a quadruped designed to look like a small dog. It is approximately 30cm long. Its neck and four legs have 3 degrees of freedom each. Figure 1 shows a picture of the robotic dog AIBO. The three robots constituting the robotic soccer team can all behave as players, or one can act as a goalkeeper.

The field is 280cm in length and 180cm in width. Six unique landmarks are placed around the edges of the field to help the robots to localize themselves on the field. Figure 2 shows the playing field.

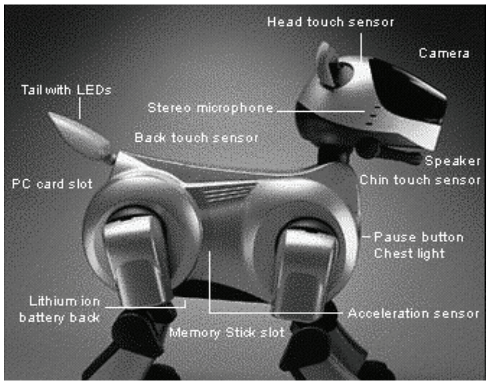


Figure 1 AIBO dog

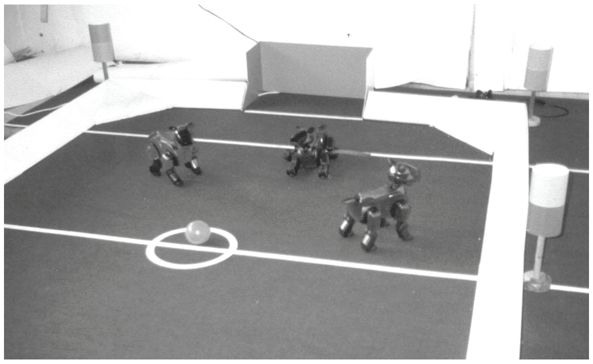


Figure 2 The robots playing soccer

Architecture

The main objective for our team was to build a research platform, which allows robust and efficient carriage of robots playing football. A modular architecture has been adopted and implemented. The following are the main system components: Vision, Localization, Communication, Planer, Behaviors, and Locomotion. The relationship between them can be seen in Figure 3.
The perception level of the control system consists of the Vision module and Localization module. The Vision module accepts images from the camera and identifies objects on the field that are visible. It delivers to the Planer module and Localization module the identified objects as well as the distance to the object and its relative direction. The Localization module receives from the Vision module the relative position of the landmarks around the play field. This information is used to approximate the position and heading of the robot on the field.
The Planer and Communication module form the logical level. It incorporates the memory of the robot, decision-making and inter-robot communication. The Planer module collects the information about the visible objects from the Vision module and stores it in the local map. The local map presents the objects around the robot in polar coordinates. Also the Planer receives from Localization the position and heading of the robot. Using this information it decides what have to be done and invokes the appropriate behavior. Communication module enables an acoustic communication between teammate robots. Exchanging of information can be used to fill the local map or to facilitate the cooperative operations.

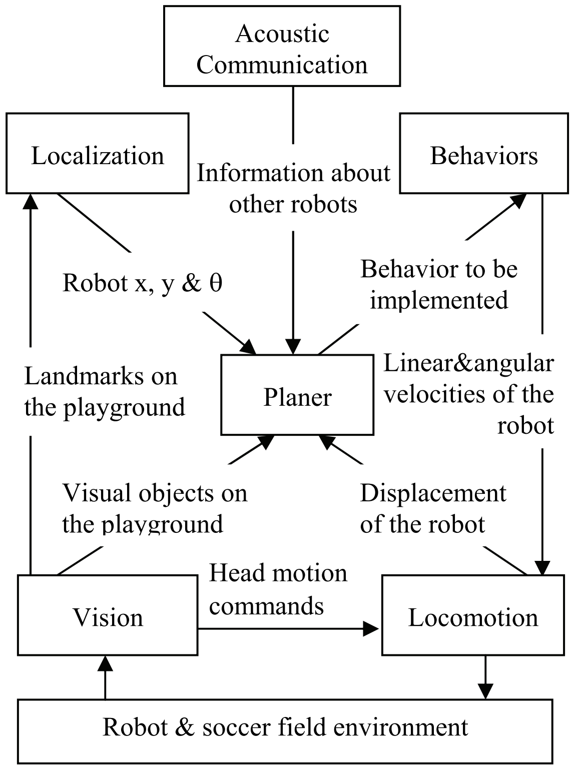

Acoustic Communication
Localization
Information about other robots
Behaviors
Robot x, y & θ
Behavior to be implemented
Landmarks on the playground
Planer
Linear&angular velocities of the robot
Visual objects on the playground
Displacement of the robot
Head motion commands
Vision
Locomotion
Robot & soccer field environment

Figure 3 The architecture of the control system

The implementation level consists of the Behavior module and Locomotion module. The Behavior module supports a set of reactive behaviors that can be implemented. It accepts abstract commands like "Dribble with the ball" and transforms them in values of angular and linear velocities of the robot. The Locomotion module receives the values of linear and angular velocities that have to be implemented and make the appropriate movement. It also delivers to the Planer module an information about the displacement of the robot in order to be actualized the information in the local map.

The implementation level

Dividing the Implementation level to locomotion part and behavior part allows us to write a high-level control laws and behaviors that are relatively independent from the low-level control of the robot. The locomotion part is related to available hardware, so it is specific for the AIBO legged robots. The behavior part is hardware-independent and can be more easily ported to and from other physical platforms, including wheeled robots.
The Locomotion module provides the interface to the physical motion functionality of the robot. The Locomotion module accepts commands from the Behavior module and from Vision module, and translates them to the actual motions. In the current implementation, the Locomotion module accepts commands for head motion, walking commands, and kicking commands. These commands are transformed into motion by controlling 17 joints (head x 3, legs x 4 x 3, tail x 2). Head motion commands include a set of "scan" commands and a "look-to" command that points the head to a specific direction. Displacement commands consist of a walking style (9 walking styles are supported), linear velocities along the principal and lateral axes of the robot, and an angular velocity.
The commands are translated to appropriate set points for positioning of the robot joints. PID-controllers are used to minimize the position error. In this algorithm a build in module OMoNet, supplied by Sony Corp. has been used. The module provides the software model of the robot and enables smooth body movements. Finally, the Locomotion module implements a set of specialized kicking routines and postures for the goalkeeper. The implementation of the kicking commands does not use OMoNet module, so in this case the robot joints are directly controlled.
The Locomotion module is also responsible for providing a rough estimate of the displacement of the robot. This estimate is needed by the Planer module in order to update the position of the objects that are not currently seen.
The Behavior module supports a set of reactive behaviors that can be implemented. It controls all robot movements except the head movements, which are controlled the Vision module. In accordance with the robot's world model, the Planer module has to choose one of the available reactive behaviors. The input data for the Behavior module are the type of the selected behavior and the required information about the environment. The behavior outputs are the linear and the angular velocities required for the implementation of the robot motion and the corresponding walking style. Table 1 presents the list of the behaviors and the input data they need.
The most complex behavior is the BIOA behavior. Having as input data the relative position of the ball and of the nearest obstacle, this behavior can generate the trajectory that will lead the robot to the ball. When the ball becomes closer, the robot must slow down its velocity and finally stop when it reaches the ball. In the next section the BIOA behavior algorithm that converts the input data to a linear and angular velocity will be discussed.

N	Description of behaviors	Needed data
1	Standing	
2	Lateral movement at predefined distance	Distance D; if $D>0$ moving to the right if $D<0$ moving to the left
3	Align with the ball and the goal	Distance to ball - D_b ; Angle to the goal - θ_g .
4	Go straight at predefined distance	Distance - D.
5	Dribble with the ball at distance	Distance - D.
6	Kick the ball	Kick type
7	Ball Interception and Obstacle Avoidance (BIOA)	Distance to the ball - D_b ; distance to the nearest obstacle - D_o ; relative angle to the ball - θ_b ; relative angle to the nearest obstacle - θ_o .
8	Goal-keeper standing	
9	Goal-keeper rotates around the ball	Distance to ball - D_b Angle to the opposite goal - θ_g

Table 1 Behavior types

The fuzzy–neural trajectory generator

The BIOA behavior has been built as a fuzzy-neural network. It uses input data that include only the relative position of the ball and the closest obstacle. Thus the number of the situations that the robotic agent could encounter in its environment has been significantly reduced. The trajectory generator was built as a fuzzy-neural network shown on Figure 4. Its inputs are:

- The distance between the ball and the mobile robot – d_B;
- The distance between the robot and the closest obstacle (it could be the nearest robot or a playground wall depending which of them is closer to the controlled robot) - d_O;
- The relative angle between the azimuth of the robot and the direction of the ball - θ_B;
- The relative angle between the azimuth of the robot and the direction of the closest obstacle - θ_O.

-

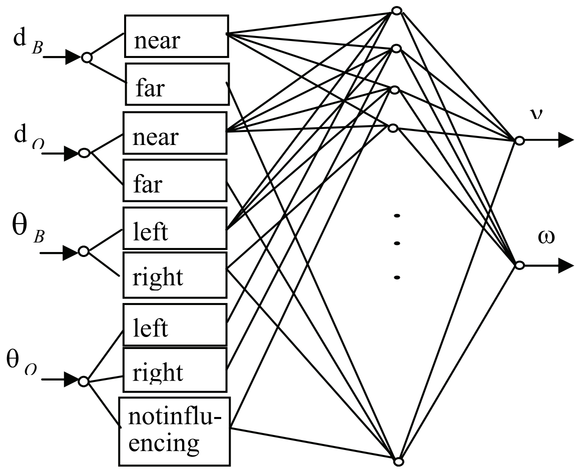


Figure 4 The BIOA fuzzy-neural network.

The fourth input signal θ_O is associated with three fuzzy labels: "left", "right" and "not influencing" in accordance with the three different situations considered:

1) θ_O has small positive value, which means that the obstacle is on the left side, so the robot should move to the right in order to avoid collision.

2) θ_O has small negative value meaning that the obstacle is on the right side and the robot should turn to the left.

3) The absolute value of the angle θ_O is big enough and there is no danger of collision with the obstacle.

There are two network outputs providing the linear velocity ν and the angular velocity ω of the robot. These values in addition to the appropriate walking style are sent to the Locomotion module in order to be implemented.

A rule base of Takagi – Sugeno type initially consisting of 24 fuzzy "if – then" rules has been implemented. The *ijkl*-th rule can be expressed as follows:

R_{ijkl} : IF d_B is A_i and d_O is B_j and θ_B is C_k and θ_O is D_l THEN ($f\nu_{ijkl} = V_{ijkl}$) and ($f\omega_{ijkl} = \Omega_{ijkl}$)

where *i,j,k* = 1,2; *l* = 1,2,3; A_i, B_j, C_k, D_l are fuzzy sets and V_{ijkl} and Ω_{ijkl} are constants.

The firing strength of the *ijkl*-th rule is obtained by using a multiplication operator:

$$\mu_{ijkl} = \mu_{Ai}(d_B)\,\mu_{Bj}(d_O)\mu_{Ck}(\theta_B)\mu_{Dl}(\theta_O)$$

The overall outputs of the fuzzy-neural network are computed as a weighted average of each rule's output

$$\nu = \frac{\sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{3}\left(\mu_{ijkl}\; f\nu_{ijkl}\right)}{\sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{3}\mu_{ijkl}};$$

$$\omega = \frac{\sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{3}\left(\mu_{ijkl}\; f\omega_{ijkl}\right)}{\sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{3}\mu_{ijkl}}$$

The membership functions $\mu_{Ai}(d_B)$, $\mu_{Bj}(d_O)$ and $\mu_{Ck}(\theta_B)$ are sigmoid functions expressed as follows:

$$\mu_{Ai}(d_B) = \frac{1}{1+e^{-a_{Ai}(d_B - C_{Ai})}};$$

$$\mu_{Bj}(d_O) = \frac{1}{1+e^{-a_{Bj}(d_O - C_{Bj})}}$$

$$\mu_{Ck}(\theta_B) = \frac{1}{1+e^{-a_{CK}(d_O - C_{CK})}}$$

where *a* and *c* are the parameters to be tuned.

Gaussian functions are used for the membership functions $\mu_{Dl}(\theta_O)$, where:

1) The membership functions for "left" and "right" (l=1 or 2) are chosen to be as follows:

$$\left|\begin{array}{ll} \mu_{Dl}(\theta_O) = e^{\frac{-(\theta_o^2)}{2\sigma_{Dl}^2}} & \text{if } (l=1 \text{ and } \theta_O > 0) \text{ or} \\ & \text{if } (l=2 \text{ and } \theta_O \le 0) \\ \mu_{Dl}(\theta_O) = 0 & \text{otherwise} \end{array}\right.$$

2) The membership function for "not influencing" (l=3) is defined as follows:

$$\mu_{D3}(\theta_O) = 1 - e^{\frac{-(\theta_o^2)}{2\sigma_{D3}^2}}$$

where σ_{Dl} is the parameter to be tuned.

The fuzzy-neural network so proposed had initially 63 tunable parameters. In order to decrease their number, several logical relations have been taken further into consideration:

1) It is obvious that with increasing the distance to the closest obstacle d_O its influence on the generated trajectory will decrease. So the topology of the layer can be simplified taking into account θ_O only for the value "near" of the parameter d_O. In this way the number of the fuzzy rules has been decreased to 16.

2) Symmetry between the robot's movement to the left and to the right has been incorporated into the trajectory generation algorithm.

So the entire number of the tunable parameters has been finally decreased to 31

Genetic learning of the fuzzy-neural trajectory generator

The classical optimization algorithms are not suitable for finding the optimal (global) solution in multi-parameter search spaces that are typically subject to discontinuities, multi-modality and nonlinear constraints. Genetic algorithms are adaptive methods widely used in searching the optimal point in a highly nonlinear and complex space. They are parallel, global search techniques that emulate natural genetic operations [Michalewicz, 1992]. Considering the advantages of GAs, a genetic-based approach has been developed to provide learning of the FNTG.

Each parameter of the fuzzy-neural network was encoded by 15 bits. The real values of the parameters were limited to (-5, 5). The population size was taken to be 60. The binary strings used for the representation of the candidate solutions had a length of 465 bits.

Each string was encoded in the following form:

$$a^b_{A_i} c^b_{A_i} a^b_{B_j} c^b_{B_j} a^b_{C_k} c^b_{C_k} \sigma^b_{D_l} f^b_{v_{1111}} \dots f^b_{v_{2223}} f^b_{\omega_{1111}} \dots f^b_{\omega_{2223}} b^b_{21} b^b_{22},$$

where the subscript "b" stands for binary values and b_{21} and b_{22} were the biases of the output neurons.

The performance evaluation was based on the following requirements:

1) For each candidate solution the robot was allowed to move during 220 time steps and some data were collected during the 170-th, 200-th and 220-th step. The purpose of the robot was defined to try to intercept the ball for not more than 200 time steps. For each of the above candidate solutions four trial motions from different starting postures were allowed and the average data were taken as representative.

2) The robot should not collide and should not be in a dangerous proximity with the obstacles during its motion.

3) The robot must stop after it reaches the ball.

The fitness function for the evaluation of the behavior was defined as follows:

$$F = \frac{10}{\gamma + (6d)^2 + d_1 + d_2 + 2v_{fit} + 0.5\omega_{fit} + \delta + 60\phi}$$

where F is the fitness function, γ is a small positive constant (γ was chosen to be 0.001), d, d_1 and d_2 are the average distances (based on four trials) from the robot to the ball at the time step 200, 170 and 220 respectively, $v_{fit} = \frac{v + v_1 + v_2}{3}$ and $\omega_{fit} = \frac{|\omega| + |\omega_1| + |\omega_2|}{3}$ where v and ω, v_1 and ω_1, v_2 and ω_2 are the robot's linear and angular velocity at time step 200, 170 and 220. $\delta = (20(0.145-H))^2$, where 0.145m is the minimal allowed distance between the robot and the obstacle, H = min (*h*, 0.145) and *h* is the shortest distance between the robot and the obstacle achieved during the robot motion. ϕ is a Boolean variable where $\phi = 0$ if linear velocity has achieved during the motion of the robot a fixed minimal value and $\phi = 1$ otherwise. The variable ϕ is included to force the robot to move during the evaluation period. The variables v_{fit} and ω_{fit} were included in the denominator to force the robot to stop when the ball is intercepted.

Initial population was generated randomly. Evaluation was executed and the fitness value of each solution was calculated. The candidate with higher value had the higher opportunity to reproduce a new population by crossover and mutation. During the evolution, the elite preserving strategy was adopted. A new group of candidate solutions was formed at each iteration and its size remained the same as that before the reproduction. The purpose of crossover lies in the combination of useful string segments from different individuals to form new and better performing offspring. A three-point crossover technique was used. Mutation randomly alters each gene in the string with a small probability. Therefore mutation helps ensure that no point in the search space has a zero probability of being explored. It was set to 0.001.

Experimental results

The entire structure of FNTG has been firstly simulated in the MATLAB® environment. Genetic learning with reproduction process iterating until the 95-th population has been implemented. Fig. 5 shows the change of the average fitness and the best fitness function in the population during the generations.

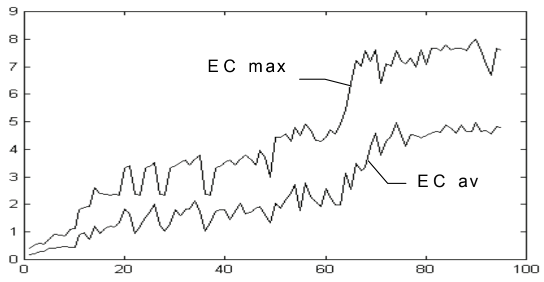


Figure 5 The change of the average fitness of the population (EC av) and the best fitness function in the population (EC max) during the generations.

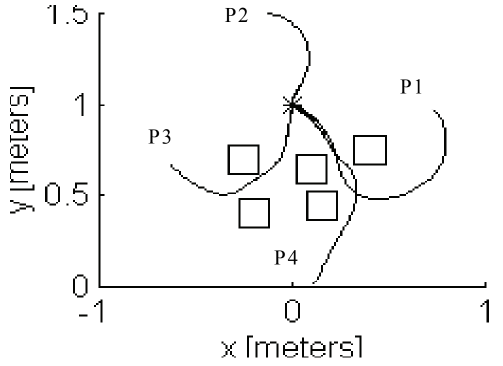


Figure 6 The experimental setup during the genetic learning of the behavior.

Fig. 6 shows experimental setup during the genetic learning. The generated trajectories correspond to the four trial motions from different starting postures using the parameters having the best fitness function in 95-th population. Position of the ball is marked with * symbol.

Fig. 7 shows example of the generated trajectory.

Fig. 8 shows the changes of linear and angular velocities during the motion of the robot pictured on the Fig. 7.

The learned structure has been subsequently incorporated in the AIBO's software

Conclusions

The paper presents a viable solution to the problem of navigating mobile robots in an unknown environment. The learning process has been made off-line by simulations using genetic algorithms. The tuned FNTG has been subsequently incorporated in the robotic dog's software as a single behavior within the Behavior module. The simulation experiments made with FNTG and the real-time tests carried out with AIBO dogs confirmed the efficiency and robustness of the proposed approach.

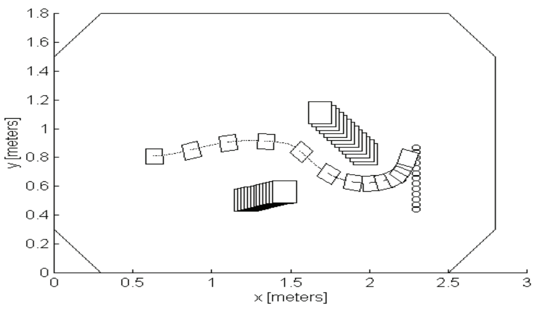
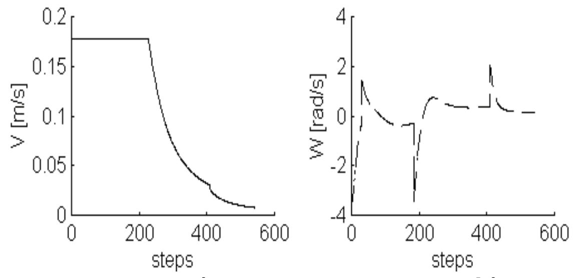


Figure 7 Generated trajectory that shows the robot's BIOA behavior



a) b)

Figure 8 The changes of the robot's linear (8a) and angular (8b) velocities during the tracking of the trajectory pictured on fig. 7.

Since the low-level motion control has been separated from the developed behaviors, the above algorithm can be easily ported and implemented to other hardware platforms too.

References

1. Mackworth, A., K. On seeing robots. In A. Basu and X. Li, (Eds.). *Computer Vision: Systems, Theory, and Applications*. Word Scientific Press, Singapore, 1993, pp. 1-13.
2. Asada, M., Y. Kuniyoshi, A. Drogoul, H. Asama, M. Mataric, D. Duhaut, P. Stone, and H. Kitano. "The RoboCup physical agent challenge: Phase-i." *Applied Artificial Intelligence.* 12, 1998.
3. Arkin, R. C. Behavior-Based Robotics. The MIT Press, Cambridge, Mass., 1998.
4. Topalov, A., Tzafestas, S., G. Layered multi-agent reactive-behavior learning in a Robotic Soccer. Proc. of the 6th IFAC Symposium on Robot Control, SYROCO'00, Vienna, September, 2000, pp.325-330.