Ball Interception Behaviour in Robotic Soccer

João Cunha, Nuno Lau, João Rodrigues

Universidade de Aveiro

Abstract. In robotic soccer the ball is the most crucial factor of the game. It is therefore extremely important for a robot to retrieve it as soon as possible. Thus ball interception is a key behaviour in robotic soccer. However, currently most MSL teams move to the ball position without considering the ball velocity. This often results in inefficient paths described by the robot. This paper presents the CAMBADA solution for a ball interception behaviour based on a uniformly accelerated robot model, where not only the ball velocity is taken into account but also the robot current velocity as well as the robot acceleration, maximum velocity and sensor-action delays are considered. The described work was introduced in the Portuguese robotics open Robótica2009 and RoboCup 2009 and improved the team performance contributing to the first and third places, respectively.

1 Introduction

Robotic soccer, much like human soccer, revolves around a key aspect of the game: the ball. To win, a team must make an efficient use of it, either by passing it to a team-mate, by dribbling it towards the goal, or by shooting it on goal in order to score. However even the most technically evolved robot cannot perform such moves if it doesn't regain possession of the ball as fast as possible. In order to obtain the ball, depending on the situation, a robot must catch a loose ball, receive a pass or tackle it from an opponent. However, most MSL teams, currently move to the ball position, without considering its velocity. Given the increasingly dynamic aspect of the game over recent years, the ball is constantly moving therefore this method doesn't provide the most efficient path to regain ball possession. (Referir soluções existentes) .This paper describes the developed solution implemented within the framework of the CAMBADA project at the University of Aveiro.

The CAMBADA is the University of Aveiro robotic soccer team competing in the RoboCup Middle Size League. The project started in 2003 by researchers of IEETA[1] ATRI[2] research group and students from DETI[3] of the University

[1] Instituto de Engenharia Electrónica e Telemática de Aveiro - Aveiro's Institute of Electronic and Telematic Engineering

[2] Actividade Transversal em Robótica Inteligente - Transverse Activity on Intelligent Robotics

[3] Departamento de Electrónica, Telecomunicações e Informática - Electronics, Telecomunications and Informatics Department

of Aveiro. The multidisciplinary project includes diverse research areas such as image analysis and processing, control, artificial intelligence, multi-agent coordination and sensor fusion. Since its origin, the CAMBADA team has competed in several national and international competitions having won the last four national championships as well as the 2008 edition of RoboCup World Championship. Mostly recently, the CAMBADA team placed in third in both RoboCup 2009 in Graz, Austria and RoboCup 2010 in Singapore.

The CAMBADA team is composed by six robots designed to play soccer. Competing in the RoboCup's Middle Size League, the CAMBADA robots must not exceed the maximum dimensions of $52cm \times 52cm \times 80cm$. The rules however don't impose a particular shape leaving that decision to each team. CAMBADA robots have a conical shape with a base radius of $24cm$ and are $71cm$ high as can be seen in Fig 1.

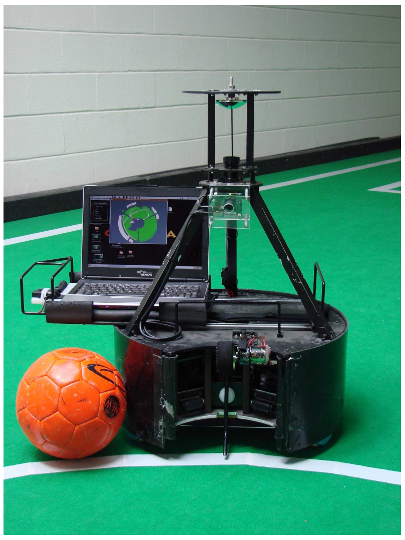


Fig. 1. A CAMBADA robot.

The remainder of this paper is organized as follows: Section 2 describes the CAMBADA software architecture upon which the interception behaviour was developed. Section 3 presents the notion of a behaviour in the CAMBADA context. The interception behaviour implementation is detailed in Section 4. Section 5 discusses the obtained results. Finally, Section 6 presents the conclusions.

2 CAMBADA Architecture

The CAMBADA robots were designed and built at the University of Aveiro. The hardware is distributed in three layers which facilitate replacement and maintenance.

The top layer has the robot's vision system. The CAMBADA robots have an omni-directional vision obtained by means of a CCD camera pointed upwards towards an hyperbolic mirror which enables a robot to see in 360 degrees[1][2].

The middle layer houses the processing unit, currently a 12" laptop, which collects the data from the sensors and computes the commands provided to the actuators. The laptop executes the vision software along with all high level and decision software and can be seen as the brain of the robot. Given the positional advantage, a ball retention device is placed on this layer.

A network of micro-controllers is placed beneath the middle layer to control the low-level sensing/actuation system, or the nervous system of the robot. The sensing and actuation system is highly distributed, meaning that each node in the network controls different functions of the robot, such as, motion, odometry, kick, compass and system monitor.

The lower layer is composed by the robot motion system and kicking device. The robots move with the aid of a set of three omni-wheels, disposed at the periphery of the robot at angles that differ 120 degrees from each other, powered by three 24 V / 150 W Maxon motors. On this layer there is also an electromagnetic kicking device. Also, for ball handling purposes, a barrier sensor is installed underneath the robot's base, that signals the higher level that the ball is under control.

In the context of an interception, omni-directional vision offers great advantages over other kinds of vision since the robot doesn't need to reposition the camera or itself to see the ball.

The locomotion system is also a factor that greatly affects a wheeled robot ability to intercept the ball as an holonomic motion robot can move to the interception point with the front of the robot oriented towards the ball. It would be far more complex for a robot to intercept a ball using Ackerman steering or differential motion.

It is no surprise that the MSL has evolved towards these types of vision and motion systems, used by almost every team, as they offer significant advantages concerning ball detection and consequent interception.

Following the CAMBADA hardware approach, the software is also distributed. Therefore, five different processes are executed concurrently. All the processes run at the robot's processing unit in Linux.

All processes communicate by means of an RTDB[4] which is physically implemented in shared memory. The RTDB is a data structure which contains the essential state variables to control the robot. The RTDB is divided in two regions, the local and shared regions.

[4] Real-Time DataBase

The local section holds the data needed by the local processes and is not to be broadcasted to the other robots. The shared section is divided between all running agents to contain the data of the world state as perceived by the team. Each sub-divided area is allocated to one robot where it stores the perceived state of the world. There is also one sub-divided area specific for the coach information. As the name implies the shared section is broadcasted through the team, as each agent transmits the owned sub-divided shared section, achieving information sharing between the team.

The RTDB implementation guarantees the temporal validity of the data, with small tolerances [3].

The processes composing the CAMBADA software are:

Vision which is responsible for acquiring the visual data from the cameras in the vision system, processing and transmitting the relevant info to the CAMBADA agent. The transmitted data is the position of the ball, the lines detected for localization purposes and obstacles positions. Given the well structured environment the robots play in, all this data is currently acquired by color segmentation [1][4].

Agent is the process that integrates the sensor information and constructs the robot's worldstate. The agent then decides the command to be applied, based on the perception of the worldstate, accordingly to a pre-defined strategy [5].

Comm that handles the inter-robot communication, receiving the information shared by the team-mates and transmitting the data from the shared section of the RTDB to the team-mates [6][7].

HWcomm or hardware communication process is responsible for transmitting the data to and from the low-level sensing and actuation system.

Monitor that checks the state of the remaining processes relaunching them in case of abnormal termination.

Given the real-time constraints, all process scheduling is handled by a library specifically developed for the task, *pman*, process manager.

The software architecture is depicted in Fig 2

3 Behaviours

The different CAMBADA behaviours represent the basic tasks to be performed by the robot, such as move to a position in the field, dribble or kick the ball. A behaviour can then be seen as the basic block of a CAMBADA robot attitude. A behaviour executes a specific task by computing the desired velocities to be applied at the robot frame, activating the ball handling device and the desired strength to be applied at the kicking system.

The choice of a given behaviour at each instant of the game is executed by a role which is basically a finite-state machine composed of various behaviours that allow the different robots to play distinct parts of the team overall strategy.

The various CAMBADA behaviours are depicted in Fig 3.

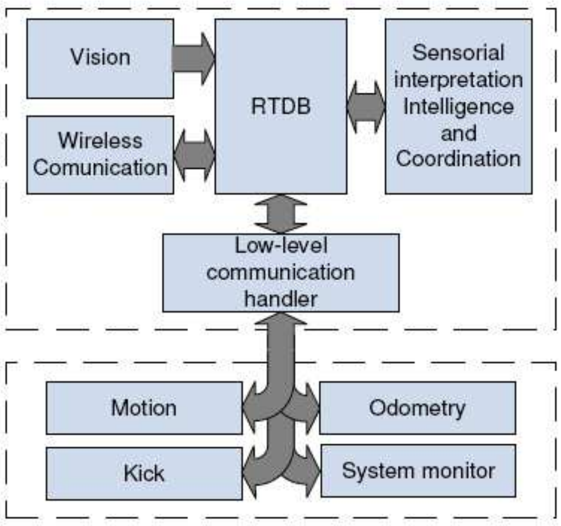

Vision
RTDB
Sensorial interpretation Intelligence and Coordination
Wireless Comunication
Low-level communication handler
Motion
Odometry
Kick
System monitor

Fig. 2. The software architecture, adapted from [8].

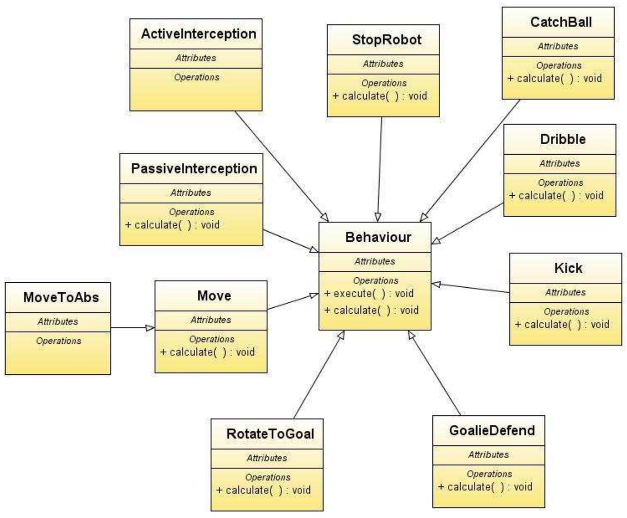


Fig. 3. Class diagram of all CAMBADA behaviours.

All the CAMBADA behaviours derive from a generic behaviour class *Behaviour*. This class implements the method *execute* which inserts in the RTDB the different values that will later be translated to the powers to be applied at the various robot actuators.

Since the control carried out on the ball handling system is on/off, method *grabberControl* implemented on the *Behaviour* class activates the device based on the position of the ball and when the ball is engaged.

All the derived behaviours have the responsability of calculating the desired velocities to be applied at the robot frame and the behaviour *Kick* in particular calculates the strength to be applied at the kicking system.

4 Implementation

The ability to intercept the ball in its path is of major importance in the robotic soccer context. The alternative, moving to the current ball position, is by no means optimal, since a robot takes more time to catch the ball and in some cases it might not even catch it. Fig 4 shows a possible robot path when the it moves to the estimated ball position.

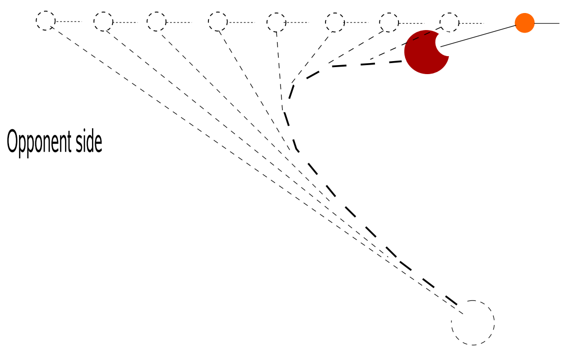


Fig. 4. Described path when the robot moves to the estimated ball position.

When considering that the ball could be dribbled by an opponent robot in the goal's direction, not intercepting the ball could have severe negative consequences. The problem scales as the other teams strive to improve their robot's top speed.

(Describe existing solutions) [9] [10] [11] [12]

The proposed solution assumes an uniformly accelerated kinematic model instead of a uniform movement kinematic model. The value of the maximum acceleration imposed on the robot movement is known, $a_{max} = 3m/s^2$. Therefore the acceleration and current speed are taken into account in the interception point calculation, as well as the robot's maximum speed, $speed_{max}$. Since the maximum velocity depends on the direction of the movement and wheel slippage and slacks, its value was empirically obtained. The value used in the CAMBADA team is $1.8m/s$.

A geometric representation of an interception is shown in Fig 5.

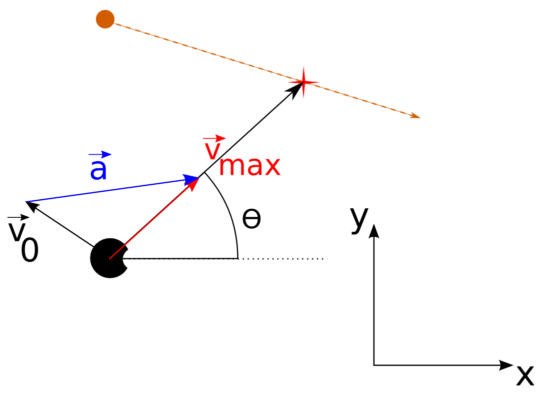


Fig. 5. Geometric representation of an interception.

To determine the interception point, we need to calculate t the time to intercept the ball, but also θ which represents the direction of the interception point in respect to the robot position. In other words the direction to where the robot will. Calculating θ is a crucial step since it gives, not only the maximum velocity of the robot $v_{max}(\theta)$ but also indirectly provides the direction of the acceleration imposed on the robot in order to achieve $v_{max}(\theta)$.

The position of the robot, $\boldsymbol{p}(t,\theta)$, at the time of the interception is given by,

$$\boldsymbol{p}(t,\theta) = \begin{cases} \boldsymbol{p}_0 + \boldsymbol{v}_0 \cdot t + \frac{1}{2} \cdot \boldsymbol{a}(\theta) \cdot t^2 \text{ if } \|\boldsymbol{v_0} + \boldsymbol{a}(\theta) \cdot t\| \leq \|\boldsymbol{v}_{max}(\theta)\|; \\ \boldsymbol{p}_{max}(\theta) + \boldsymbol{v}_{max}(\theta) \cdot t' \text{ if } \|\boldsymbol{v_0} + \boldsymbol{a}(\theta) \cdot t\| \geq \|\boldsymbol{v}_{max}(\theta)\|. \end{cases} \tag{1}$$

where

$$\boldsymbol{v}_{max}(\theta) = \{speed_{max} \cdot \cos(\theta), speed_{max} \cdot \sin(\theta)\} \tag{2}$$

$$t_{max} = \frac{\|\boldsymbol{v}_{max}(\theta) - \boldsymbol{v}_0\|}{a_{max}} \tag{3}$$

$$\boldsymbol{a}(\theta) = \{a_{max} \cdot \cos(\phi(\theta)), a_{max} \cdot \sin(\phi(\theta))\} \tag{4}$$

$$\phi(\theta) = \arctan(v_{0_y} - v_{max_y}(\theta), v_{0_x} - v_{max_x}(\theta)) \tag{5}$$

$$t' = t - t_{max} \tag{6}$$

$$\boldsymbol{p}_{max}(\theta) = \boldsymbol{p}_0 + \boldsymbol{v}_0 \cdot t_{max} + \frac{1}{2} \cdot \boldsymbol{a}(\theta) \cdot {t_{max}}^2 \tag{7}$$

This means that the robot will move according to an uniformly accelerated movement with acceleration $\boldsymbol{a}(\theta)$ until its velocity $\boldsymbol{v_0} + \boldsymbol{a}(\theta) \cdot t$ saturates, which happens at moment t_{max}. At this point the robot should be at $\boldsymbol{p}_{max}(\theta)$. From this moment on the robot will move according to an uniform movement with its maximum velocity $\boldsymbol{v}_{max}(\theta)$.

Since the Eq 1 is non-linear, a numerical method would be required to find an interception point $\boldsymbol{p}(t,\theta)$. To simplify the calculations performed by the robot, an iterative (hill-climbing) solution was developed. The solution tests consecutive points in the ball path. A valid interception point is found when the robot reaches the considered point before the ball.

The consecutive points are generated using a time step of 0.1 seconds. Since the ball is assumed to move according to an uniform movement model with velocity $\boldsymbol{v_b}$, the i-th iteration interception test point $\boldsymbol{p_i}$ is given by,

$$\boldsymbol{p_i} = \boldsymbol{p_b} + \boldsymbol{v_b} \cdot i \cdot 0.1 \tag{8}$$

The time the ball takes to reach the considered is directly obtained by $0.1 \cdot i$. On the other hand by testing a specific point, we can obtain the direction the robot will move, θ, which is given by

$$\theta = \arctan(p_{i_y} - p_{o_y}, p_{i_x} - p_{0_x})) \tag{9}$$

Hence, we only need to determine the time it takes for the robot to reach $\boldsymbol{p_i}$. This is possible by applying Eq. 8 and Eq. 1, $\boldsymbol{p_i} = \boldsymbol{p}(t, \theta)$.

5 Results

To test the performance of the proposed solution, we conducted experiments in the CAMBADA training field, by releasing a ball down a ramp in order to achieve a desired velocity. The experiments were conducted with three different ball velocities, $1m/s$, $2m/s$ and $2.5m/s$. The obtained results are classified in three different classes: failure, if the robot fails to intercept the ball, contact, if the ball touches the front of the robot but bounces away out of the range of the robot control and success, if the robot is able to intercept the ball and keep it under control.

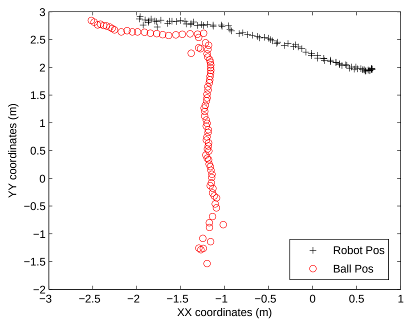
Table 1 presents the obtained results in the tests performed on the real robots.

Ball Velocity(m/s)	Failure	Contact	Success
1	0%	0%	100%
2	0%	20%	80%
2.5	0%	100%	0%

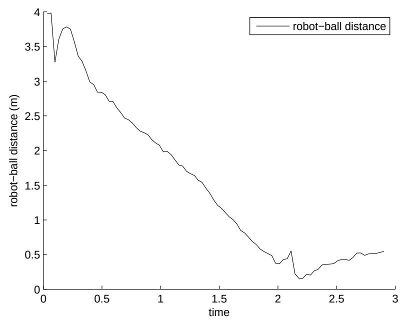
Table 1. The results obtained in the interception tests performed on the CAMBADA robots.

Furthermore, experiments were conducted to test the performance of the developed solution against the strategy of moving towards the ball without considering its velocity. Using the same initial conditions such as the robot position and velocity and ball position and velocity, a robot intercepts the ball in under

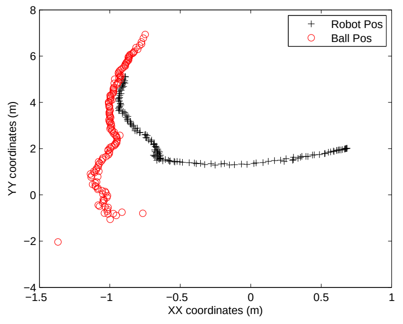
2 seconds while a robot moving towards the ball is not even able to come in contact with the ball. Figure 6 presents the paths performed by the robot using both strategies. For visualization purposes the evolution of the distance between the robot and the ball during the course of the experiment is presented. Keep in mind that the robot has an approximate radius of $25cm$. Thus the distance between the ball and the robot cannot be smaller than this value.



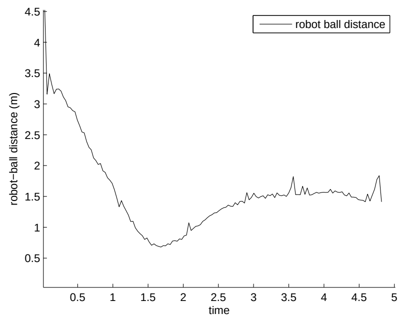
a)



b)



c)



d)

Fig. 6. Examples of interception and moving to the ball strategies. The circles represent the ball positions while the crosses represent the robot positions. **a)** the paths described by the robot and the ball using the developed interception algorithm. **b)** the corresponding distance between the ball and the robot throughout the experiment. **c)** the paths described by the ball and the robot when the latter is naively moving the the ball position. **d)** the corresponding distance between the ball and the robot.

6 Discussion

In robotic soccer, ball interception skills are of major importance and provide serious advantages during a game. Although the RoboCup Simulation League has a variety of solutions that address this problem, the transition to real systems presents some issues that hinder such effort. This paper presented a solution that solves the ball interception problem based on the Newton's solution considering an uniformly accelerated with saturation robot motion model. The obtained results in Table 1 show that the robot is able to consistently intercept the ball in its path. However the performance decreases as the ball velocity increases. This is due to the fact that the obtained solution is the shortest time interception point. This causes the robot to intercept the ball as soon as possible without attempting to absorb the impact of the ball. Hence the ball will bounce away in such situations.

Although the interception behaviour is not ideal for receiving a pass from a team-mate, this skill is still very useful in defensive situations in order to stop an opponent dribbling the ball. This advantage is clearly depicted in Fig. 6 where the robot using the interception behaviour is able to intercept the ball. On the other hand a robot that moves towards the ball position is unable to catch the ball before it leaves the field.

The developed solution has been successfully integrated in the CAMBADA competition strategy helping the team to reach important results such as winning the National Championships, Robotica'2009 and Robotica'2010, and achieving third place in the World Championships, RoboCup'2009 and RoboCup'2010.

Acknowledgements

Alguem quer agradecer à familia?

References

1. A.J.R. Neves, G. Corrente, and A.J. Pinho. An omnidirectional vision system for soccer robots. In *Proc. of the EPIA 2007*, volume 4874 of *Lecture Notes in Artificial Inteligence*, pages 499–507. Springer, 2007.
2. B. Cunha, J.L. Azevedo, N. Lau, and L. Almeida. Obtaining the inverse distance map from a non-svp hyperbolic catadioptric robotic vision system. In *Proc. of the RoboCup 2007*, Atlanta, USA, 2007.
3. L. Almeida, F. Santos, T. Facchinetti, P. Pedreiras, V. Silva, and L.S. Lopes. Coordinating distributed autonomous agents with a real-time database: The CAMBADA project. In *Proc. of the ISCIS*. Springer, 2004.
4. A.J.R. Neves, D.A. Martins, and A.J. Pinho. A hybrid vision system for soccer robots using radial search lines. In *Proc. of the 8th Conference on Autonomous Robot Systems and Competitions, Portuguese Robotics Open - ROBOTICA'2008*, pages 51–55, Aveiro, Portugal, April 2008.
5. N. Lau, L.S. Lopes, and G. Corrente. Cambada: Information sharing and team coordination. In *Proc. of the 8th Conference on Autonomous Robot Systems and Competitions, Portuguese Robotics Open - ROBOTICA'2008*, pages 27–32, Aveiro, Portugal, April 2008.
6. F. Santos, L. Almeida, P. Pedreiras, L.S. Lopes, and T. Facchinetti. An Adaptive TDMA Protocol for Soft Real-Time Wireless Communication among Mobile Autonomous Agents. In *Proc. of the Int. Workshop on Architecture for Cooperative Embedded Real-Time Systems, WACERTS 2004*, 2004.

7. F. Santos, G. Corrente, L. Almeida, N. Lau, and L.S. Lopes. Selfconfiguration of an Adaptive TDMA wireless communication protocol for teams of mobile robots. In *Proc. of the 13th Portuguese Conference on Artificial Intelligence, EPIA 2007*, 2007.
8. J.L. Azevedo, B. Cunha, and L. Almeida. Hierarchical distributed architectures for autonomous mobile robots: A case study. In *Proc. of the 12th IEEE Conference on Emerging Technologies and Factory Automation, ETFA 2007*, pages 973–980, 2007.
9. Frieder Stolzenburg, Oliver Obst, and Jan Murray. Qualitative velocity and ball interception. In Matthias Jarke, Jana Koehler, and Gerhard Lakemeyer, editors, *KI*, volume 2479 of *Lecture Notes in Computer Science*, pages 283–298. Springer, 2002.
10. Patrick Riley, Peter Stone, David A. McAllester, and Manuela M. Veloso. Att-cmunited-2000: Third place finisher in the robocup-2000 simulator league. In Peter Stone, Tucker R. Balch, and Gerhard K. Kraetzschmar, editors, *RoboCup*, volume 2019 of *Lecture Notes in Computer Science*, pages 489–492. Springer, 2000.
11. Heiko Müller, Martin Lauer, Roland Hafner, Sascha Lange, Artur Merke, and Martin Riedmiller. Making a robot learn to play soccer using reward and punishment. In Joachim Hertzberg, Michael Beetz, and Roman Englert, editors, *KI*, volume 4667 of *Lecture Notes in Computer Science*, pages 220–234. Springer, 2007.
12. Bob van der Vecht and Pedro U. Lima. Formulation and implementation of relational behaviours for multi-robot cooperative systems. In Daniele Nardi, Martin Riedmiller, Claude Sammut, and José Santos-Victor, editors, *RobuCup*, volume 3276 of *Lecture Notes in Computer Science*, pages 516–523. Springer, 2004.