19

Event-driven Hybrid Classifier Systems and Online Learning for Soccer Game Strategies

Yuji Sato
Hosei University
Japan

1. Introduction

The field of robot soccer is a useful setting for the study of artificial intelligence and machine learning. By considering the learning processes used in multi-agent systems, such as cooperative action learning with multiple agents, optimization of strategies for new opponents, robust handling of noise and other disturbances, and real-time learning during live gameplay, it is possible to grasp real-world problems in a fairly abstract way. For this reason, there has recently been active research and exchange of information concerning a robot soccer game contest called RoboCup. Meanwhile, in simulations using robots, it is necessary to tackle noise and to address the issues involved in processing the signals obtained from multiple sensors, and it is not always possible to evaluate and analyze this information effectively. When focusing on game strategy learning, it is often effective to perform a priori evaluation and analysis by computer simulation. In this section we introduce an idea for autonomous adaptive evolution with respect to the strategies of opponents in games, and we present the results of evaluating this idea. Specifically, we start by introducing a hybrid system configuration of classifier systems and algorithmic strategies. Then, with the aim of implementing real-time learning in mid-game, we introduce a bucket brigade algorithm which is a reinforcement learning method for classifiers, and a technique for restricting the subject of learning depending on the frequency of events. And finally, by considering the differing roles assigned to forwards, midfielders and defenders, we introduce a technique for performing learning by applying differences to the reward values given during reinforcement learning. We pitted this technique against soccer game strategies based on hand-coded algorithms, and as the results show, our proposed technique is effective in terms of increased win rate and the speed of convergence on this win rate.

2. Soccer Video Game and Associated Problems

2.1 Overview of Soccer Video Game

The type of soccer game that we will deal with here is a software-driven video game with soccer as its theme in which two teams battle for the most points. Figure 1 shows a typical game scene targeting the area around the current position of the ball. The screen also

Source: Robotic Soccer, Book edited by: Pedro Lima, ISBN 978-3-902613-21-9,
pp. 598, December 2007, Itech Education and Publishing, Vienna, Austria

includes a diagram showing a total view of the game in the lower right hand corner. The size of the field was set to 20.0×110.0 grid world, considering that the size of the actual soccer pitch is 20 m long ×110 m wide. Positions on the field and the locations of objects placed on the field are defined using three-dimensional coordinates in the width (x), length (y) and height (z) directions. Data on the field is all processed using floating-point values. The movement of the soccer ball is controlled by physical computations. The ball state is determined by its position vector V_{bp}, direction vector V_{bd}, speed V, and acceleration V_a. The position vector V_{bp} and speed V are updated in each cycle of the environment according to Equation (1) below:

$$V_{bp} = V_{bp} + V_{bd} \times V \qquad \text{and} \qquad V = V + V_a. \tag{1}$$

The soccer players can be in any of three states — stationary, accelerating, or moving — and are assigned a position vector V_p, a direction vector V_d, and information about the actions they are performing. Changes of state occur when a player takes some kind of action based on the information input from the environment.

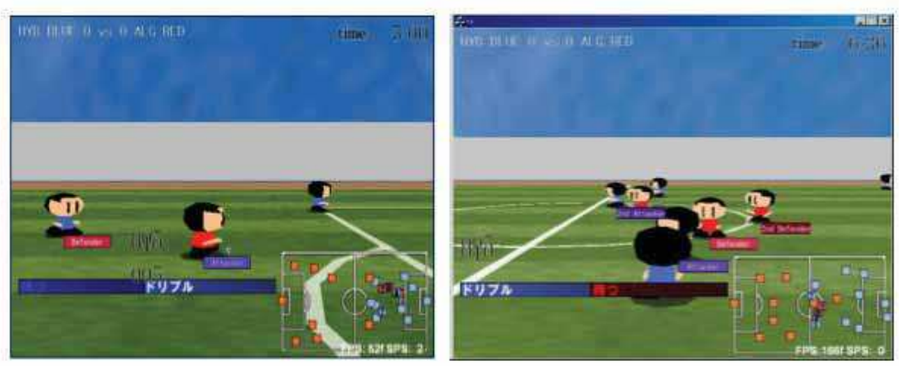


Fig. 1. Example of typical game scene targeting the area around the current position of the ball

Each team has 11 players, and the movements of the 11 players of one team are controlled by computer. The algorithm to control player action is thought up beforehand by a game designer and programmed as a set of control rules in IF-THEN (condition-action) format. Figure 2 shows an example of a rule written in IF-THEN format and the corresponding scene. The rule states "If the ball is right in front of me while I am in front of the goal and if two players of the other team are between me and the goal, then pass the ball to an unmarked player on my team." The program for determining player action consists of a detector, a decision-making section, and an effector. Based on information input from the environment, the detector determines the position and state of each player, the position of the ball, the distance between a player and the goal, etc., and passes these results to the decision-making section. This section then determines player actions according to an algorithm described in IF-THEN format as described above. Examples of player actions

include kick, trap, and move in accordance with current circumstances. The effector finally executes these actions in the environment based on instructions received from the decision-making section. Now, the operation of all or some of the 11 players making up the opposing team is performed by the user, that is, the game player. If the game player is in charge of operating only some of the players on his team, the actions of the remaining players will be controlled by the same algorithm as that of the team controlled by computer.

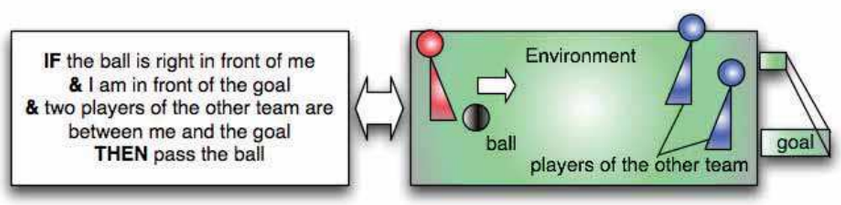


Fig. 2. Example of a rule written in IF-THEN format and corresponding scene

2.2 Problems with Conventional Technique

As described above, the conventional approach to producing a soccer video game is to have a game designer devise the algorithm for controlling player action and to then describe and program that algorithm as a set of rules in IF-THEN format. Recently, however, the Internet is making it easier for anyone to participate in video games and the number of game users is increasing as a result. This development is generating a whole new set of problems. First, the increasing number of users means that the differences in strategies that users prefer and excel in can no longer be ignored and that multiple strategy algorithms must be simultaneously supported. Second, the appearance of users with advanced techniques has generated a need for decision-making algorithms under even more complicated environments. And finally, as the Internet makes it easy for new users to appear one after another, it must be possible to provide and maintain bug-free programs that support such complex decision-making algorithms in a time frame much shorter than that in the past. In other words, the human- and time-related resources required by development and maintenance work are increasing dramatically while the life cycle of each game is shortening. The conventional technique is hard pressed to deal with this situation.

3. Hybrid Decision-making System

We have studied the equipping of game programs with machine learning functions as an approach to solving the above problems. This is because incorporating machine learning functions in an appropriate way will enable the system to learn the game player's strategy and to automatically evolve a strong strategy of its own. It will also eliminate worries over program bugs and significantly reduce the resources required for development and maintenance. A number of techniques can be considered for implementing machine learning functions such as neural networks, Q-learning (Sutton & Barto, 1998) and genetic algorithms (GAs), and we have decided, in particular, on incorporating functions for acquiring rules based on classifier systems (Holland, 1992). We came to this decision considering the many examples of applying evolutionary computation to the acquisition of robot decision-making

algorithms (Gustafson & Hsu, 2001; Luke, 1998; Pietro et al., 2002) in the world of robot soccer games such as RoboCup (Kitano et al., 1997; RoboCup), learning classifier systems takes advantage of GAs and reinforcement learning (Sutton & Barto, 1998) to built adaptive rule-based systems that learn gradually via online experiences (Holmes et al., 2002; Huang & Sun, 2004; Kovacs, 2002), and considering the compatibility between the IF-THEN production-rule description format and classifier systems and the resulting ease of program migration.

At the same time, the bucket brigade algorithm (Belew & Gherrity, 1989; Goldberg, 1989; Holland, 1986; Riolo, 1987a; Riolo, 1987b; Riolo, 1989) used as a reinforcement learning scheme for classifier systems needs time to obtain an effective chain between classifiers. As a result of this shortcoming, the bucket brigade algorithm is not suitable for learning all strategies from scratch during a game. A conventional algorithm, on the other hand, provides solid strategies beforehand assuming fixed environmental conditions, but also includes a rule that states that a player encountering undefined environmental conditions must continue with its present course of action. In light of the above, we decided to apply classifier-based learning to only conditions/actions not described by an explicit algorithm. In short, we adopted a hybrid configuration combining a conventional algorithm and a learning section using a classifier system (Sato & Kanno, 2005).

Figure 3 shows the basic idea of the hybrid decision-making system using a classifier system. This hybrid system is achieved by embedding a conventional algorithm into a classifier system as a base. The conventional algorithm is unaffected by learning and is implemented as a set of "privileged classifiers." Specifically, the reliability (credit or strength) of a privileged classifier is set to the highest possible value and is not targeted for updating by learning. If, after analyzing a message list, there are no privileged classifiers in the classifier list that match a current condition, the strength of a classifier that does is updated. Classifiers can also be discovered here using genetic algorithms.

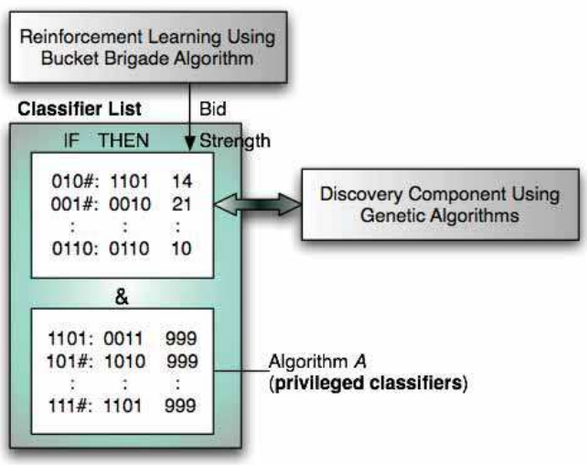


Fig. 3. Basic idea of the hybrid decision-making system using a classifier system

4. Event-driven Learning Classifier System

The preliminary experiments revealed that a hybrid-type system has the potential of exceeding a human-designed algorithm provided that search space can be contracted by limiting the target of learning to actions. On the other hand, having humans select conditions beforehand does nothing to eliminate the problems associated with the conventional way of generating conditions.

To solve this dilemma, we decided to switch the rules to be learned for each game player (user) that the computer opposes. This is because the total possible search space in theory need not be the target of learning if only the strategy of the game player in the current match can somehow be dealt with. Furthermore, it was decided that all of the current player's strategies would not be targeted for learning but rather that the number of events targeted for learning would be limited to that that could be completed in real time. Figure 4 shows the configuration of the proposed event-driven classifier system (Sato & Kanno, 2005). This system differs from standard classifier systems in three main ways. First, the proposed system adds an event analysis section and creates a table that records event frequency for each game player. Second, the classifier discovery section using genetic algorithms targets only actions while conditions are generated by adding new classifiers in accordance with the frequency of actual events. Third, the system updates the strength of classifiers by the bucket brigade algorithm starting with high-frequency events and continuing until learning can no longer be completed in real time. The proposed system also adopts a hybrid configuration combining a conventional algorithm and classifier system as before. Finally, the system provides for two types of rewards that can be obtained from the environment: a large reward obtained from winning or losing a game and a small reward obtained from succeeding or failing in a single play such as passing or dribbling the ball. In short, the above system focuses only on strategy that actually occurs with high frequency during a game and limits learning space to the range that learning can be completed in real time.

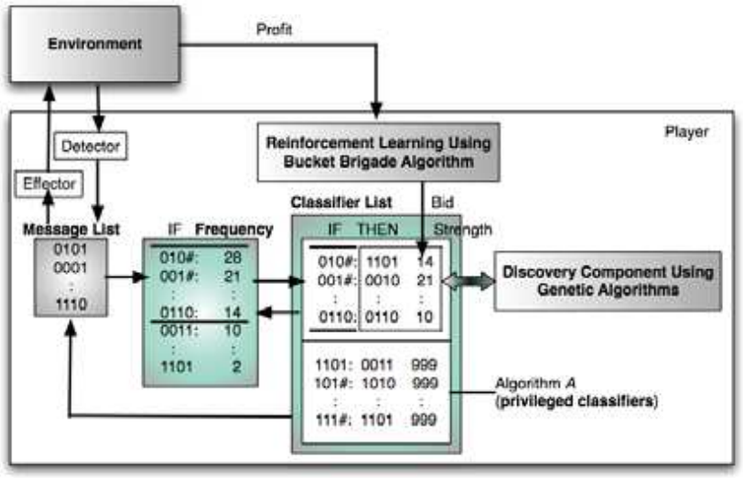


Fig. 4. The configuration of the event-driven hybrid learning classifier system

5. Reward Allotment Based on the Role of Each Position

Table 1 shows the success rewards for each play that were used in these tests. Preliminary tests were performed to make a prior survey of the number of times the players performed pass and dribble actions in a single game, on the basis of which the rewards for passing and dribbling were set so that the product of the success reward for passing and the number of passes made was more or less equal to the product of the success reward for dribbling and the number of dribble actions performed. The goal-scoring success reward was set to a high value because goal-scoring is of great importance to the outcome of a game. For all the in-game agents apart from the goalkeeper, learning was performed by applying success rewards to each play without any particular regard to differences in the role of each position. For example, when a pass was made successfully, bucket brigade learning was performed so that the same reward value (16) was obtained by each player irrespective of whether the player was assigned to a forward, midfielder or defense role. And when a player takes the ball from a member of the opposing team, bucket brigade learning is performed by obtaining the same reward value (15) regardless of the difference in roles between the players involved.

	GETGOAL	DRIBBLE	PASS	GETBALL	LOSTBALL	TOTAL
Forwards	60	4	16	15	-50	45
Midfielders	60	4	16	15	-50	45
Defense	60	4	16	15	-50	45

Table 1. The success rewards for each play that were used in a team *H*1

On the other hand, in real soccer games, the forward, midfielder and defense players are assigned different roles and emphasize different aspects of their play depending on these assigned roles. Accordingly, it is thought that giving different success rewards to each player considering the role assignments of forward, midfielder and defense players might lead to a better game winning rate. These role assignments into consideration might lead result in cooperative learning that contributes to a better winning rate. Table 2 shows the basic concept for determining the success rewards for each player (Sato et al., 2006). For example, a forward should take as many shots at goal as possible in order to gain points. Forwards are therefore given a large success reward for shots at goal, while their reward for stealing the ball from the opposing team is made relatively small. Conversely, the main duty of defense players is to prevent the opposing team from being able to take shots at goal. Defense players are thus given greater rewards for stealing the ball from the other team, and relatively small rewards for successful shots at goal. Meanwhile, the role of midfielders is to move the ball forwards to connect between the defense and forward players, and to act as surrogate defense or forward players when necessary. Accordingly, their success rewards are more evenly spread, with extra emphasis on actions such as passing and dribbling.

	GETGOAL	DRIBBLE	PASS	GETBALL	LOSTBALL
Forwards	Large	Average	Average	Small	Average
Midfielders	Average	Large	Large	Average	Average
Defense	Small	Average	Average	Large	Average

Table 2. The basic concept for determining the success rewards for each player

6. Evaluation Experiments

6.1 Experiment on Event-driven Hybrid Learning Classifier Systems

6.1.1 Evaluation Method

We prepared three strategy algorithms beforehand to help generate data for evaluation purposes. The first one is strategy-algorithm *A* as a product prototype. The remaining two are strategy-algorithm *B* and strategy-algorithm *C* both based on strategy-algorithm *A* but modified to place weight on offense and defense, respectively. These three strategies were made to play against each other beforehand and each was set to have about the same winning percentage.

In the experiments, the outcome of games played between two teams in a soccer environment was observed. The players on one team used one of the above conventional algorithm-type decision-making systems while those on the other team used the event-driven hybrid classifier system proposed in Section 4. The first 20 seconds during a single game was time for learning and applied to constructing a classifier system. Each pair of teams played 10,000 matches and team effectiveness was evaluated from its winning rate R_w defined as the following equation.

$$R_w = N_w / (N_t - N_d), \tag{2}$$

where N_t, N_d, and N_w are total number of matches, number of draws, and number of wins respectively. The experiments evaluated the ability of the proposed event-driven classifier system to deal with a diverse environment and to adapt to a dynamic environment.

First, Fig. 5 summarizes the experiment for evaluating the ability to deal with a diverse environment, that is, the ability to deal with more than one strategy algorithm. Specifically, event-driven classifier system *H*1 incorporating algorithm *A* was made to play against strategy-algorithms *A*, *B*, and *C*, and the outcomes of the resulting matches were observed to see whether learning could be performed to give *H*1 a winning rate better than 50% against all of these strategies.

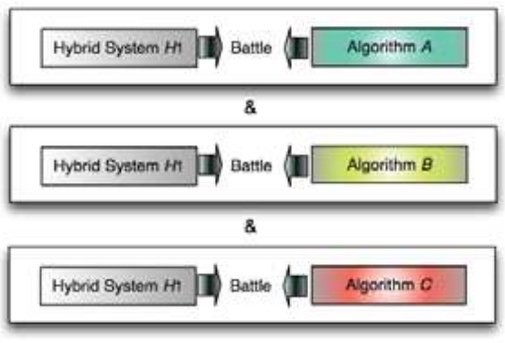


Fig. 5. The experiment for evaluating the ability to deal with a diverse environment

Next, Fig. 6 summarizes the experiment for evaluating the ability to adapt to a dynamic environment, that is, the ability to adapt to changes in strategy. Here, event-driven classifier system *H*1 incorporating algorithm *A* was first made to play against strategy-algorithm *A*. Next, given that system *H*1 had evolved into system *H*1′ having a strategy that could adequately deal with strategy-algorithm *A*, system *H*1′ was made to play against strategy-algorithm *B* to see whether it could further evolve to achieve a winning rate better than 50%. Similarly, given that system *H*1′ had evolved into system *H*1″ having a strategy that could adequately deal with strategy-algorithm *B*, system *H*1″ was made to play against strategy-algorithm *C* to see whether it could again evolve to achieve a winning rate better than 50%. In short, match outcomes were observed to see whether the hybrid system had the ability to adapt to intermittent changes in strategy along the time axis.

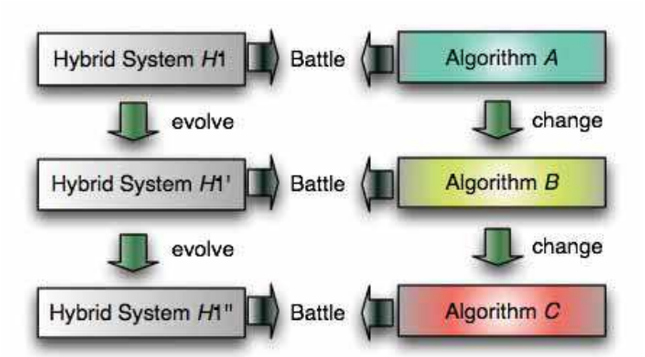


Fig. 6. The experiment for evaluating the ability to adapt to a dynamic environment

6.1.2 Experimental Results and Discussion

Dealing with a diverse environment:
Table 3 shows the results of evaluating the ability to deal with a diverse environment. This table shows the results of 10,000 matches. Against algorithm *A*, system *H*1 won 32%, lost 17%, and drew 51% of the games played. Against algorithm *B*, it won 32%, lost 14%, and drew 54% of the games played. And finally, against algorithm *C*, it won 23%, lost 12%, and drew 65% of the games played. In other words, system *H*1 exhibited a degree of learning resulting in a winning percentage better than 50% against all three algorithms. Figure 7 shows the relationship between number of matches played and winning rate R_w. This figure shows the first 500 matches for each pairs of teams. These results show that system *H*1 could adapt to each of the three algorithms in several ten matches.
The results shown in Table 3 and Fig. 7 tell us that event-driven classifier system *H*1 incorporating algorithm *A* could achieve a winning rate better than 50% against strategy-algorithms *A*, *B*, and *C* by learning. System *H*1 therefore has the ability of dealing with a diverse environment.

	Won	Lost	Drew
Algorithm *A* vs. *H*1	32%	17%	51%
Algorithm *B* vs. *H*1	32%	14%	54%
Algorithm *C* vs. *H*1	23%	12%	65%

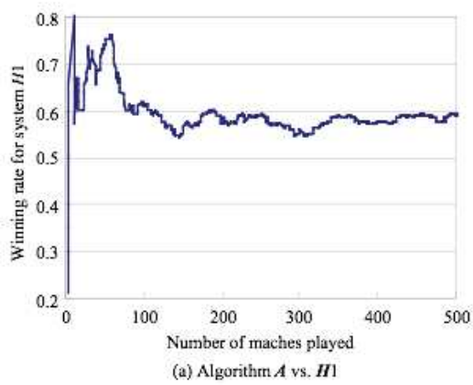
Table 3. Ability to deal with a diverse environment. This table shows the results of 10,000 matches

Adapting to a dynamic environment:
Table 4 shows the results of evaluating the ability to adapt to a dynamic environment. This table shows the results of 10,000 matches. Here, classifier system *H*1′ is the result of learning by playing against strategy-algorithm *A*, and from the table, we see that it also adapted to algorithm *B* by playing against that algorithm to the point of winning 31%, losing 14%, and drawing 55% of the games played. Likewise, classifier system *H*1″, the result of adapting to algorithm *B*, also adapted to algorithm *C* by playing against that algorithm to the point of winning 22%, losing 12%, and drawing 66% of the games played. Figure 8 shows the relationship between number of matches played and winning rate R_w for system *H*1′ with respect to algorithm *B* and for system *H*1″ with respect to algorithm *C*. This figure shows the first 500 matches for each pairs of teams.

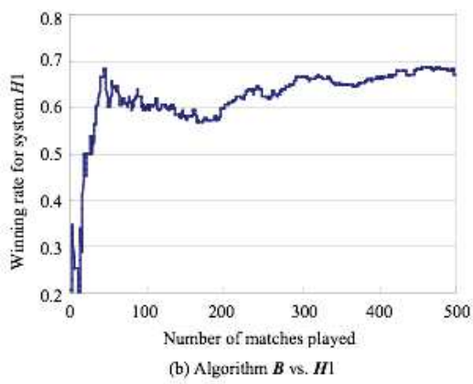
	Won	Lost	Drew
(*A* ->) *B* vs. *H*1′	31%	14%	55%
(*B* ->) *C* vs. *H*1″	22%	12%	66%

Table 4. Ability to dynamic environment. This table shows the results of 10,000 matches

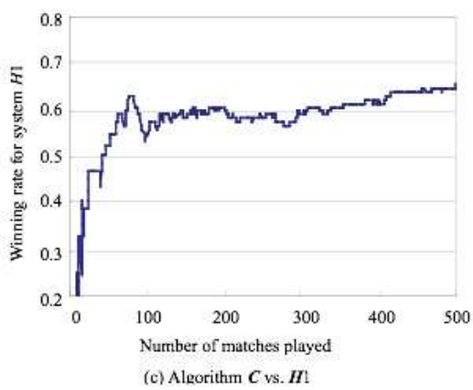
As for the experiment on evaluating the ability to adapt to a dynamic environment, the results of Table 4 and Fig. 8 show that the event-driven classifier system could evolve and achieve a winning rate better than 50% in the face of intermittent changes in strategy along the time axis.
At the same time, Figs. 7 and 8 show that this system requires about 80 matches to evolve to a point where it can either deal with a diverse environment or adapt to a dynamic environment. To achieve a practical, working system, though, it is desirable that the system be able to adapt with fewer learning steps. Making learning more efficient with a smaller number of matches is a topic for future research.
In either case, the event-driven classifier system could adapt to the conventional algorithm in question in about 80 matches. As reference, Fig. 9 shows the relationship between number of matches played and success rate of dribbling, and shooting. These results reveal that an event-driven classifier system can improve the success rate of dribbling and shooting by about 5%.



(a) Algorithm ***A*** vs. ***H***1



(b) Algorithm ***B*** vs. ***H***1



(c) Algorithm ***C*** vs. ***H***1

Fig. 7. The relationship between number of matches played and winning rate of *H*1

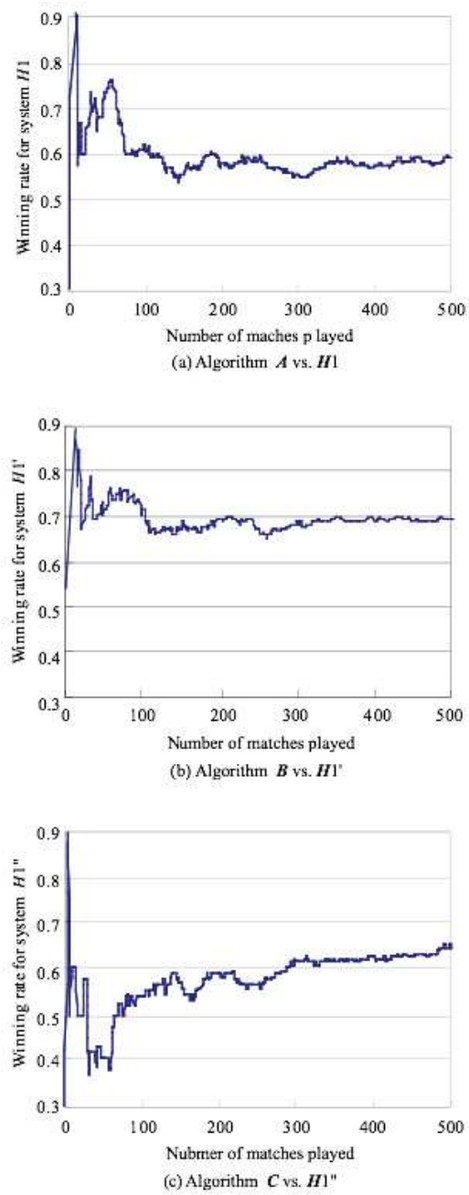


Fig. 8. The relationship between number of matches played and winning rate for system *H*1, *H*1′, and *H*1″

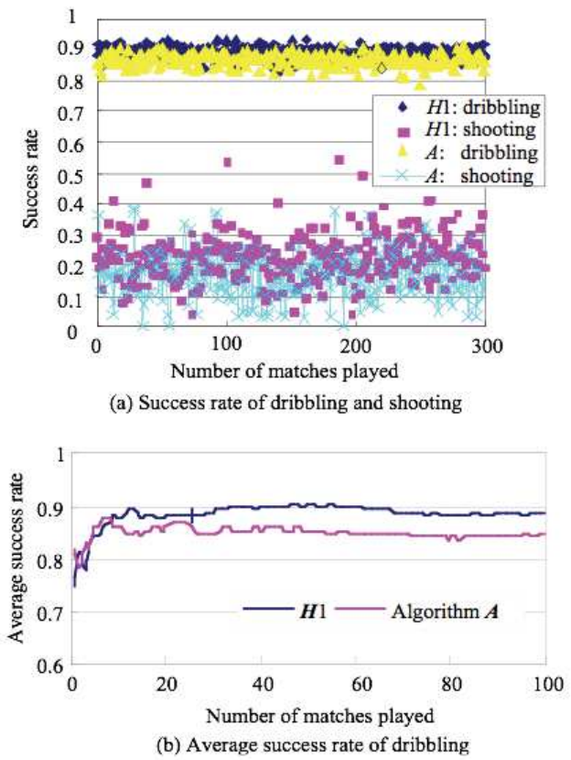


Fig. 9. The relationship between number of matches played and success rate of dribbling and shooting

6.2 Experiment on Reward Allotment Based on the Role of Each Position

6.2.1 Evaluation Method

For the evaluation data, we used three different algorithmic strategies that were employed in earlier trials. The tests involved playing matches between two teams in a soccer environment and observing the number of games won and lost. An algorithmic decision-making system was used for the players of one team, while an event-driven classifier system was used for the players of the other team. The event-driven classifier system was evaluated by using a number of teams in which each player was set with different success reward values for plays such as passing and dribbling, according to the aims of the test.

In practice, we investigated whether or not changes in the game winning rate are caused by giving each player different success rewards based on the role assignments of different positions. We also investigated whether or not there were any changes in the game winning rate by changing the balance of success rewards of each type of play.

Three event-driven classifier systems incorporating algorithm *A* were prepared with differences in the rewards used for bucket-brigade learning. Specifically, these were a team *H*2 in which the success rewards of each player were set considering the role assignments shown in Table 5 in line with the basic policy mentioned above in section 3.3, a team *H*3 in which the success rewards of each player were set as shown in Table 6 based on the opposite idea to the above mentioned basic policy, and a team *H*1 which was set with the rewards used in the prior tests shown in Table 1. These three teams were each made to battle against algorithmic strategies *A*, *B* and *C*, and we comparatively evaluated them by determining the final asymptotic winning rates and the speed with which they converged on these final rates.

	GETGOAL	DRIBBLE	PASS	GETBALL	LOSTBALL	TOTAL
Forwards	80	2	8	5	-50	45
Midfielders	60	4	16	15	-50	45
Defense	40	2	8	45	-50	45

Table 5. The success rewards for each play that were used in a team *H*2

	GETGOAL	DRIBBLE	PASS	GETBALL	LOSTBALL	TOTAL
Forwards	40	2	8	45	-50	45
Midfielders	60	2	8	8	-50	45
Defense	80	2	8	5	-50	45

Table 6. The success rewards for each play that were used in a team *H*3

Next, we will describe the test method used to evaluate the relationship between the winning rate and the balance of success rewards of each type of play. The event-driven classifier system incorporating algorithm *A* provides a total of four teams — two different teams in which the success rewards of each player are set considering their role assignments, and two different teams in which no particular consideration is given to role assignments. Specifically, we provided two new teams — *H*4, in which the balance of success rewards for each play is modified as shown in Table 7 based on the rewards shown in Table 1, and *H*5, in which the balance of success rewards for each play is modified as shown in Table 8 based on the rewards shown in Table 5. Each of these four teams was matched against algorithmic strategies *A*, *B* and *C*, and we compared them with each other in terms of the eventual asymptotic winning rate and the speed of convergence on this rate. We also investigated the relationships between the success rewards and the success rates of each play and between the winning rate and the success rate of each play, with the aim of using this information to analyze the strategies acquired through learning by the event-driven classifier system.

	GETGOAL	DRIBBLE	PASS	GETBALL	LOSTBALL	TOTAL
Forwards	60	2	8	10	-35	45
Midfielders	60	2	8	10	-35	45
Defense	60	2	8	10	-35	45

Table 7. The success rewards for each play that were used in a team *H*4

	GETGOAL	DRIBBLE	PASS	GETBALL	LOSTBALL	TOTAL
Forwards	80	4	16	5	-60	45
Midfielders	60	8	22	15	-60	45
Defense	40	4	16	45	-60	45

Table 8. The success rewards for each play that were used in a team *H*5

6.2.2 Experimental Results and Discussion

Position role assignments and winning rate:
Figures 10-12 show the results of evaluating the relationships between the position role assignments and winning rates achieved by team *H*1, *H*2, and *H*3. In these figures, each point represents the average result obtained by playing 200 successive games 30 times. From Figures 10-12, the event-driven classifier systems *H*1– *H*5 incorporating algorithm A were able to perform learning to achieve a winning rate of more than 50% with all three of the algorithmic strategies *A*, *B* and *C*. Also, in all the matches with algorithms *A*, *B* and *C*, the winning rates were highest and converged the fastest with team *H*2, where the success rewards of each play were set as shown in Table 5 considering the roll assignments. The lowest rising speed was achieved with team *H*3, where the success rewards of each play were set as shown in Table 6 using weightings opposite to those of the basic strategy. The event-driven classifier system thus seems to be able to contend with opponents having a wide variety of strategies, and it seems that conferring different success rewards to each type of play considering the role assignments of forward, midfielder and defense players results in a better winning rate and faster convergence.

The balance of success rewards of each play and the winning rate:
Figures 10-12 also show the results of evaluating the relationship between the balance of success rewards of each play and the winning rate. In the matches played with all three algorithms *A*, *B* and *C*, team *H*1 ultimately converged on a higher winning rate than team *H*4, and the winning rate also rose at a faster rate. Team *H*2 ultimately converged on a higher winning rate than team *H*5, and its winning rate rose at a faster rate. Our results show that the winning rate and speed of convergence differ significantly when changes are made to the balance of success rewards for each play, regardless of whether or not the position role assignments are taken into consideration.
On the other hand, with regard to the tests for evaluating the relationship between the winning rate and the balance of success rewards for each play, Figs. 10-12 show that differences in the winning rate and the rate of convergence were caused by changing the balance of success rewards for each play independently of whether or not position role assignments were considered. Accordingly, by conferring different success rewards to each type of play by considering the role assignments, and by carefully setting the balance of success rewards for each type of play, it is thought that it is possible to gain further increases in the rate at which games are won and the rate of convergence. On the other hand, with regard to which specific value should be set, no explicitly determined procedure is set in particular. Although it can be determined by trial and error, it is also possible to consider determining the success reward values for each type of play by applying a procedure such as evolutionary computation. Further study will be needed relating to techniques for finding optimal values for the success rewards for each type of play.

Success rate of each play:
Figures 13–15 respectively show the ball possession rates, the number of pass per game, and the number of successful goals per game achieved by each team. The ball possession rates and the number of pass per game exhibit no particular correlation to the winning rate, but were highest for teams *H*1 and *H*2, while teams *H*3 produced lower values of magnitude. As for the number of successful goals per game, all the teams eventually converged on a rate of about 0.6, but our results show that this value rose much faster for team *H*2 which had a high winning rate.
Figures 13 and 14 show that the ball possession rate and the number of pass per game had no particular bearing on the winning rate, with team *H*2 achieving higher values than team *H*1, and teams *H*3 producing low values. On the other hand, in Tables 1, 5 and 6, the sum total of the values of the success rewards for dribbling awarded to forward, midfielder and defense players are found to be 12 for team *H*1, 8 for team *H*2, and 6 for teams *H*3, which corresponds to the order of the ball possession rates in Fig. 13. Also, with regard to the success reward values for passes, the sum total values were 48 for team *H*1, 32 for team *H*2, and 24 for teams *H*3, which corresponds to the ordering of the number of pass per game in Fig. 14. Specifically, it seems that the rate of success of individual play actions has a strong tendency to be dependent on the sum total of the success rewards for each type of play for forward, midfielder and defense players, independently of whether the game is won or lost. On the other hand, the number of successful goals per match was 180 for teams *H*1 and *H*2, and 200 for team *H*3, which does not correspond with the ordering in Fig. 15. In Fig. 15, the goal success rate of team *H*2, which has the highest winning rate, rises the fastest. In order to successfully score a goal, it is impossible to ignore the relationships with other forms of play, and it is thought that rather than being determined solely by the value of the success reward for an individual play, it is strongly related to whether the game is won or lost.

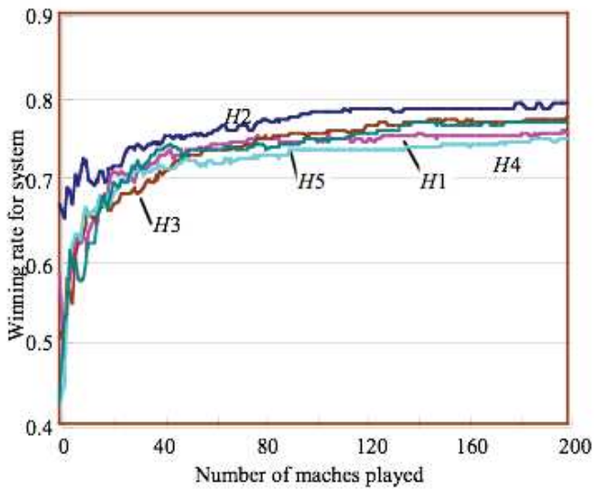


Fig. 10. The relationship between the position role assignments and winning rate (Algorithm *A* vs. *H*1 – *H*5)

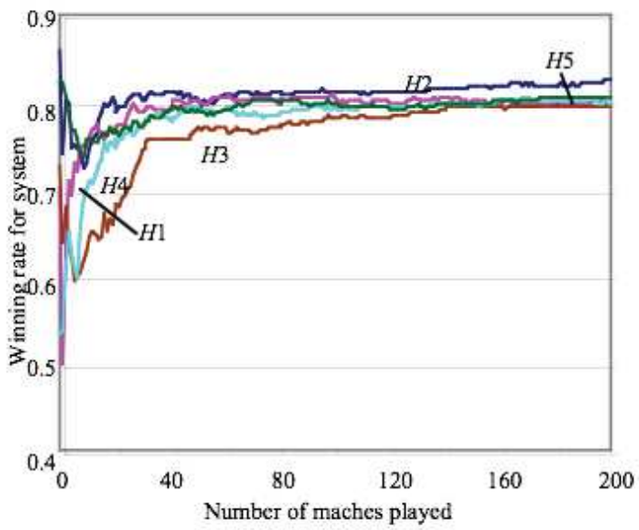


Fig. 11. The relationship between the position role assignments and winning rate (Algorithm *B* vs. *H*1 – *H*5)

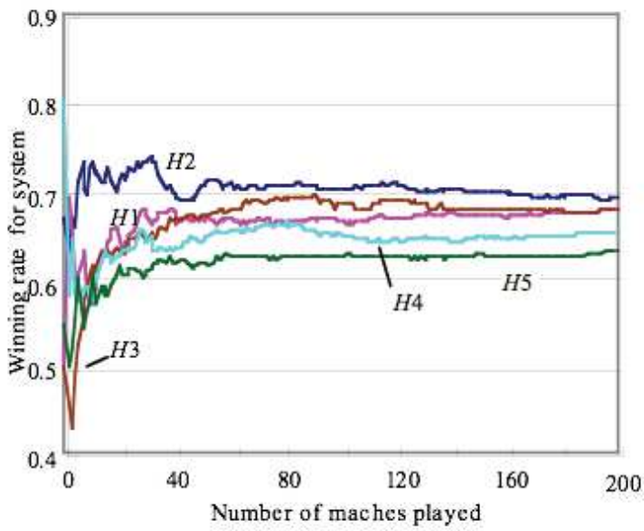


Fig. 12. The relationship between the position role assignments and winning rate (Algorithm *C* vs. *H*1 – *H*5)

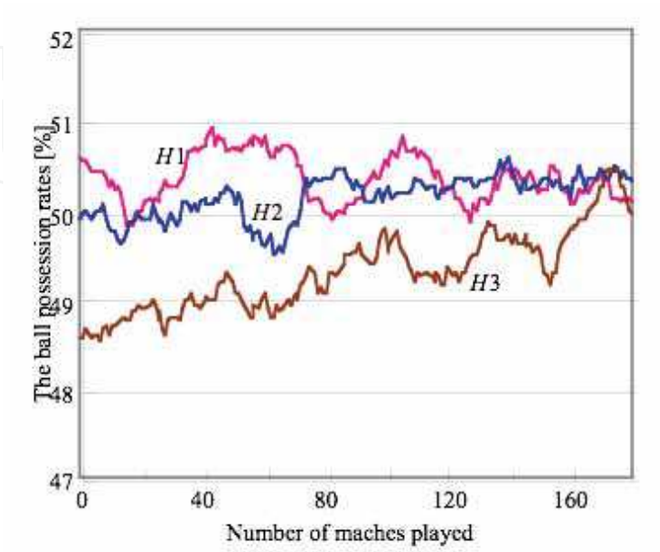


Fig. 13. The ball possession rates achieved by team *H*1, *H*2, and *H*3 (vs. Algorithm *A*)

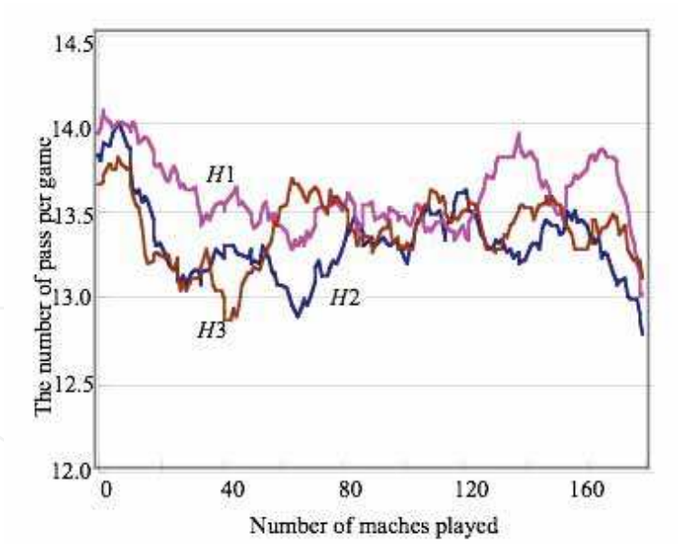


Fig. 14. The number of pass per game achieved by team *H*1, *H*2, and *H*3 (vs. Algorithm *A*)

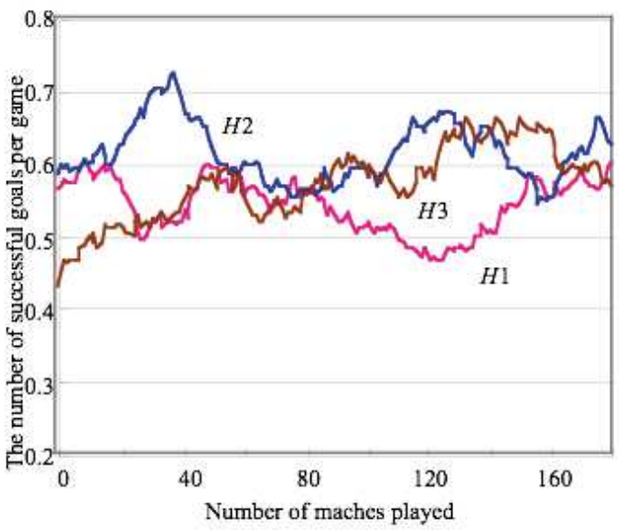


Fig. 15. The number of successful goals per game achieved by team *H*1, *H*2, and *H*3 (vs. Algorithm *A*)

7. Conclusion

In this section we have reported on the results of applying a classifier system to the acquisition of decision-making algorithms by agents in a soccer game. First, we introduced the hybrid system configurations of the existing algorithms and a classifier system. Then, in order to implement real-time learning while a game is in progress, we introduced a bucket brigade algorithm that implements reinforcement learning for the classifier, and a technique for selecting the subject of learning depending on the frequency of events. And finally, we introduced a method for performing learning by awarding players different reward values during reinforcement learning depending on whether they are assigned the role of forward, midfielder or defender. We played this technique against an existing soccer game with hand-coded algorithms, and we evaluated the win rate and the speed of convergence. As a result, we demonstrated that this is an effective means for autonomous adaptive evolution to deal with the opponent's strategies in mid-game. It should be stressed that the technique introduced here has only been evaluated by computer simulation in a video game. When it is applied to a robot soccer game, there are other factors that have to be considered, such as processing information input from multiple sensors and dealing with noise. However, by employing an algorithm that was effective in previous RoboCup contests as the existing algorithm implemented inside the hybrid system, it ought to be an effective technique even in robot soccer games.

8. References

Barry, A. (2003). Limits in Long Path Learning with XCS, In *Proceedings of the Fifth Annual Genetic and Evolutionary Computation Conference.* Vol. 2, pp. 1832-1843, LNCS 2724, Springer-Verlag, Berlin, Heidelberg.

Belew, R.K and Gherrity, M. (1989). Back Propagation for the Classifier System, In *Proceedings of the Third International Conference on Genetic Algorithms*. pp. 275-281, Morgan Kaufmann Publishers, CA.

Bull, L., Wyatt, D., and Parmee, I. (2002). Towards the Use of XCS in Interactive Evolutionary Design, In *Proceedings of the Fourth Annual Genetic and Evolutionary Computation Conference*. pp. 951, Morgan Kaufmann Publishers, San Francisco, CA.

Butz, M.V., Goldberg, D.E., and Lanzi, P.L. (2004). Gradient-Based Learning Updates Improve XCS Performance in Multistep Problems, In *Proceedings of the Sixth Annual Genetic and Evolutionary Computation Conference*, Vol. 2, pp. 751-762, LNCS 3103, Springer-Verlag, Berlin, Heidelberg.

Dawson, D. (2003). Improving Performance in Size-Constrained Extended Classifier Systems, In *Proceedings of the Fifth Annual Genetic and Evolutionary Computation Conference*, Vol. 2, pp. 1870-1881, LNCS 2724, Springer-Verlag, Berlin, Heidelberg.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

Gustafson, S.M., and Hsu, W.H. (2001). Layered Learning in Genetic Programming for a Cooperative Robot Soccer Problem, In *Proceedings of the Fourth European Conference on Genetic Programming*, pp. 291-301.

Holland, J.H. (1992). *Adaptation in Natural and Artificial Systems*, The MIT Press, Cambridge, MA. The University of Michigan Press, Ann Arbor, 1975.

Holland, J.H. (1986). Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems, In Michalski, R.S. et al. (eds.): *Machine Learning II*, pp. 593-623, Morgan Kaufmann Publishers, CA.

Holmes, J.H., Lanzi, P.L., Stolzmann, W., Wilson, S.W. (2002). Learning Classifier Systems: New Models, Successful Applications. *Information Processing Letters*, Vol. 82, pp. 23-30.

Huang, C-H. and Sun, C-T. (2004). Parameter Adaptation within Co-adaptive Learning Classifier Systems, In *Proceedings of the Sixth Annual Genetic and Evolutionary Computation Conference*, Vol. 2, pp. 774-784, LNCS 3103, Springer-Verlag, Berlin, Heidelberg.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawa, E., and Matsubara, H. (1997). RoboCup: A challenge problem for AI, *AI Magazine*, Vol. 18, pp. 73-85.

Kovacs, T. (2002). What Should a Classifier System Learn and How Should We Measure It? *Journal of Soft Computing*, Vol. 6, No. 3-4, pp. 171-182.

Luke, S. (1998). Genetic Programming Produced Competitive Soccer Softbot Teams for RoboCup 97, In *Proceedings of the Third Annual Genetic Programming Conference*, pp. 204-222, Morgan Kaufmann Publishers, San Francisco, CA.

Pietro, A.D., While, L., and Barone, L. (2002). Learning in RoboCup Keepaway using Evolutionary Algorithms, In *Proceedings of the Fourth Annual Genetic and Evolutionary Computation Conference*, pp. 1065-1072, Morgan Kaufmann Publishers, San Francisco, CA.

Riolo, R.L. (1987a). Bucket brigade performance: I. Long sequences of classifiers, genetic algorithms and their application, In *Proceedings of the Second International Conference on Genetic Algorithms*, pp. 184-195, Lawrence Erlbaum Associates, Publishers.

Riolo, R.L. (1987b). Bucket brigade performance: II. Default hierarchies, In *Proceedings of the Second International Conference on Genetic Algorithms*, pp. 196-201, Lawrence Erlbaum Associates, Publishers.

Riolo, R.L. (1989). The emergence of coupled sequences of classifiers, In *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 256-263, Morgan Kaufmann Publishers, CA.

RoboCup web page. http://www.robocup.org/

Sato, Y., and Kanno, R. (2005). Event-driven Hybrid Learning Classifier Systems for Online Soccer Games, In *Proceedings of the 2005 IEEE Congress on Evolutionary Computation*, pp. 2091-2098, IEEE Press, Edinburgh, Scotland.

Sato Y., Akatsuka Y., and Nishizono T. (2006). Reward Allotment in an Event-driven Hybrid Learning Classifier System for Online Soccer Games, In *Proceedings of the 2006 Genetic and Evolutionary Computation Conference*, pp. 1753-1760, ACM Press, Seattle, WA.

Sutton, R.S., Barto, A.G. (1998). *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge, MA.

Wilson, S.W. (1995). Classifier Fitness Based on Accuracy, *Evolutionary Computation*, Vol. 3, No. 2, pp. 149-175, The MIT Press, Cambridge, MA.

Wilson, S.W. (1998). Generalization in the XCS Classifier System, In *Proceedings of the Third Annual Genetic Programming Conference*, pp. 665-674, Morgan Kaufmann Publishers, San Francisco, CA.

Robotic Soccer

Edited by Pedro Lima

ISBN 978-3-902613-21-9

Hard cover, 598 pages

Publisher I-Tech Education and Publishing

Published online 01, December, 2007

Published in print edition December, 2007

Many papers in the book concern advanced research on (multi-)robot subsystems, naturally motivated by the challenges posed by robot soccer, but certainly applicable to other domains: reasoning, multi-criteria decision-making, behavior and team coordination, cooperative perception, localization, mobility systems (namely omni-directional wheeled motion, as well as quadruped and biped locomotion, all strongly developed within RoboCup), and even a couple of papers on a topic apparently solved before Soccer Robotics - color segmentation - but for which several new algorithms were introduced since the mid-nineties by researchers on the field, to solve dynamic illumination and fast color segmentation problems, among others. This book is certainly a small sample of the research activity on Soccer Robotics going on around the globe as you read it, but it surely covers a good deal of what has been done in the field recently, and as such it works as a valuable source for researchers interested in the involved subjects, whether they are currently "soccer roboticists" or not.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen