Generating Dynamic Formation Strategies Based on Human Experience and Game Conditions*

John Atkinson and Dario Rojas

Department of Computer Sciences
Universidad de Concepcion, Concepcion, Chile
atkinson@inf.udec.cl

Abstract. In this paper, a new approach to automatically generating game strategies based on the game conditions is presented. A game policy is defined and applied by a human coach who establishes the attitude of the team for defending or attacking. A simple neural net model is applied using current and previous game experience to classify the game's parameters so that the new game conditions can be determined so that a robotic team can modify its strategy on the fly. Results of the implemented model for a robotic soccer team are discussed.

Keywords: Robotics Game Strategies, Team Formation, Multi-Agent Systems.

1 Introduction

A team's playing strategy is a human football team's main asset. For human players, the strategy is fixed by a coach who defines the players' positions and roles on the football field based on his/her perception on the game conditions and the players' abilities. Accordingly, providing an adaptive playing strategy should involve defining and obtaining the game's current conditions. In order to decide which actions and formation must be taken (and therefore, which low-level behaviors must be accomplished) a team must gather information to determine whether this is doing well or not.

For human football teams, it is relatively easy to determine the game conditions. Several criteria are taken into account including the chances to score, the position on the field, the number of catchings, the score, etc. However, processing this perception information on autonomous robots is not that easy as there are diverse constraints such as processing capabilities, available time, errors with sensors and those of Multi-Agent Systems (MAS) running on a dynamic environment [1].

From a MAS perspective, the playing strategy for the 4-legged robotic competition becomes a significant component due to recent advances on robust vision

* This research is partially sponsored by the National Council for Scientific and Technological Research (FONDECYT, Chile) under grant number 1070714 *"An Interactive Natural-Language Dialogue Model for Intelligent Filtering based on Patterns Discovered from Text Documents"*.

U. Visser et al. (Eds.): RoboCup 2007, LNAI 5001, pp. 159–170, 2008.

and localization techniques which currently provide a more accurate and precise perception from the environment. Because of the dynamic and underlying uncertain nature of the league (i.e., there is distributed autonomous intelligence rather than centralized control), new methods are required to generate effective playing strategies which should not be resource-demanding. To this end, a new adaptive approach to team formation using simple and efficient connectionist techniques is proposed to enable a robotic team to effectively adapt its strategy and positions as the game goes on, depending on diverse parameters obtained from the environment's sensorial information.

This paper is organized as follows: in section 2, related robotics and simulation approaches to team formation are discussed, section 3 proposes a new neural net model for team formation and role assignment based on the conditions of the game, in section 4 the main experiments using our model and different role selection strategies are discussed. Finally, section 5 highlights the main conclusions, drawbacks and issues of this research.

2 Related Work

Designing perception systems for autonomous robots participating in robotic soccer competitions is one of the most important challenges. Recent advances on hardware and software for these robotic applications such as perception and locomotion have been promising in terms of providing powerful and robust robotic control systems.

Nevertheless, no significant progress has been reported in team adaptation and cooperation (4-legged) for behavior-based systems. Most of the research on MAS for the 4-legged robotic competition focuses on solving individual problems for each agent (i.e., decision making, navigation, etc). Hence the domain knowledge is indirectly being considered in a nearly reactive way, that is, decisions are purely made based on explicit triggering rules specified by the programmers into the agents' code.

Since that there is no deep analysis of the game conditions, current playing strategies are only based on the ball's current position, and no cumulated experience or human feedback is considered to improve the agents' performance. Furthermore, most of the state-of-the-art research on four-legged robotic teams use machine learning approaches so as to provide agents with some individual basic skills (i.e., reactive tasks) or cooperative capabilities but no efforts are put into getting information regarding the conditions of the game.

A novel approach to decision making based on roles assignment from auctions is proposed by [2]. Agents bid for their roles on the field, and a captain agent (the *bidder*) determines which agent offers the most for some role. Each bid includes the agent's position and the distance to the ball. Offers are then optimized by using Genetic Algorithms which allow the agents to bid for the best possible offer as the system evolves. However, the agents' actions are not taken into consideration and no estimates regarding the conditions of the game are provided.

A slightly different role assignment strategy is proposed by [8] in which decisions are distributedly made, this is, no coordinator or captain is designed. For this, each agent reacts to the team mates' positions and the ball's position by minimizing a distance function so that the closest agent will be given the attacker role whereas the others take defense and support roles. The distributed nature of the model provides a fair approximation of the overall view of the world among agents which allows them to determine the ball's position more precisely.

In order to deal with some of these issues, [9] developed a simulated training agent capable of classifying the opponent's model using a Bayesian method which is previously trained with manually defined rules. Based on the real-time opponent's model and the environment's status, the training agent creates centralized plans using adaptive *Simple Temporal Networks* to be distributedly executed by the players in which best plans are selected by using a hill-climbing search strategy. Overall, the approach does not provide a clear notion of the conditions of the game hence a team can not determine how good/bad the executed actions are. This problem is partially overcome by using a multi-agent system in which agents cooperate to generate different models of the world: a local model and a shared model. However, the approach focuses on error reduction of self-localization rather than cooperation to determine the conditions of the game.

In the context of the *RoboCupSoccer Simulation* league, some approaches determine the strategy of the opponent so to dynamically adapt to the environment. Game conditions are so computed from statistical parameters such as the ball's average position, number of corners, number of goals, etc., which are difficult to obtain accurately. Other approaches explore the generation of agent-agent advises. Here, a training agent is an *adviser* for the other players and produces a set of Markov rules obtained from a set of abstract states contained in the previous games' logs (environment's status, agents' actions, etc). One of the drawbacks is that obtaining these logs involves a global view of the environment which may not be easily available for the four-legged league [5].

Role and task assignment problems have also been formally studied by [3] in the context of MRTA (*Multi-Robot Task Allocation*). One of the drawbacks of this analysis is that the dependence between tasks/roles assigned to robotic agents is not considered. Note that this is a key issue as most of the tasks allocated to one agents has a strong relation with tasks assigned to the other agents of the robotic team. Although the approach can be applied to roles distribution, this always gets the same task distribution to a given scenario as Greedy search is used [6].

3 An Adaptive Model for Computing Game Conditions

One of the most important tasks of a game strategy is to determine whether the executed actions are correct. We need to compute the environments' characteristics which establish when the game conditions are favorable or unfavorable.

There are several approaches to determine the game conditions, however, most of them focus on simulations.

We propose a new model to automatically generate dynamic formation strategies based on the game conditions. The approach computes parameters that determine a good or bad game with no need to manually define input data such as difference of score and number of agents. The relation between these data and the game conditions are automatically obtained by using a learning approach which takes into account human experience, other agents' perceptions and tasks, etc. The agents' experience (i.e., actions performed in a time period) provides criteria on the game conditions with no need to deal with complex perception data.

The *Game Conditions* (GC) of a team establish how favorable/unfavorable the conditions of the game are. The value of the GC can not be obtained instantaneously so the condition can be seen as a sequence of events occurring in a period of time. Thus a GC represents a cumulative form of the current game condition. Unfavorable conditions for the team occur whenever the current game condition shows better choices of losing the game. On the contrary, when favorable conditions occur, the choices of winning the game are better than rather losing it. Near-zero values indicate that the conditions are either uncertainty or balanced.

Computing the GC based on the players' activities provides us with rich information to make further decisions (i.e., difference of score is not a determining factor by itself as a goal can happen by a chance). For instance, if a team is doing very well and even so there is a tie (i.e., there is no useful feedback information on 0-0 scores), then obtaining favorable GC for the team allows it to put more effort into offensive tasks so that this can increase the chances of scoring.

In the proposed model, experiences are seen as a set of actions performed by the agents in a period of time T and include:

- ***Player:*** Represents any player but the keeper. The player's experience is computed by counting the following actions: *Actions* (AC) is the total number of actions the player intents to perform on the ball, *Blockades of the ball* (BL), *Shots* (SH) is the number of attempts to score on the opponent's defense zone, *Changes of Positions* (CP) is the total number of changes of positions, *Defense to Attacker* (DA) is the number of changes of position from defense to attack, *Attacker to Defense* (AD) is the number of changes of position from attack to defense, *Total Time* (T) is the elapsed time since the beginning of the sampling, *Defense Time* (DT) is the time spent to perform defense tasks on the defense zone, *Support Time* (ST) is the time spent to perform support tasks on the center of the football field, *Attacker Time* (AT) is the time spent to perform attack tasks on the opponent's defense zone, and *Idle Time* (IT) is the time spent in which no tasks are executed.
- ***Goal Keeper:*** The keeper avoids the opponent team to score and owns an exclusive field's area. Since the goal keeper does not change its position nor perform roles exchanging, only the action parameters on the ball are required. Note that the only enabled actions for the robotic team are catching the ball and shooting the ball, both of which can be considered a kind

of "catching". Hence both actions can be thought as one parameter called *Warning time* (WT) which is the time the goal keeper is in risk.

- ***Captain:*** The team's captain gathers the players' game data and produces its own parameters. In addition, this provides us with information required to calculate the difference of scores (DS) such as the *Scored Goals* (SG) and the *Received Goals* (RG).

Note that we are interested to quantify the intent of acting rather than determining the effect these produce. This outcome will be learned from a Neural Network by putting all the game's parameters together.

The parameters required to obtain the players' experience are sampled every T seconds. Every time a sampling process gets started, the corresponding experience values are set to 0 and data are normalized to values between 0 and 1. Each normalized experience's parameter can be seen in table 1.

Table 1. Experiences Computed for Each Agent's Role

Experience	**Agent Type**	**Meaning**
$E_{BL} = \frac{BL}{AC}$	Player	Blockades
$E_{SH} = \frac{SH}{AC}$	Player	Shots
$E_{DA} = \frac{DA}{CP}$	Player	Defense to Attack change
$E_{AD} = \frac{AD}{CP}$	Player	Attack to Defense change
$E_{DT} = \frac{DT}{T}$	Player	Defense Time
$E_{ST} = \frac{ST}{T}$	Player	Support Time
$E_{AT} = \frac{AT}{T}$	Player	Attacker Time
$E_{IT} = \frac{IT}{T}$	Player	Idle Time
$E_{DS} = \frac{1}{1+e^{SG-RG}}$	Captain	Difference of Score
$E_{WT} = \frac{WT}{T}$	Goal Keeper	Risk Time

The difference of score ($SG(t) - RG(t)$) is then represented as a sigmoid function for differences between -10 and $+10$. Note that the four-legged league does not allow (absolute) differences higher than 10.

In order to compute the GC, a Neural Network (NN) based model is proposed to map players' actions into game conditions. The approach is capable of finding optimum relationships between the players' actions so that game conditions can be computed and then transferred to the team's strategy.

Training of the NN is carried out by performing nine ten-minute robotic soccer simulations. This aimed to obtain initial parameters so to investigate the feasibility of the proposed model. Since that the competition has time and resources constraints, simple NN models have been used. In particular, a Back-Propagation Neural Net was implemented based on [4]. A usual sigmoid function was used as an activation function in the hidden and output layers [4] and this provided fair results on the different configurations. Initially, a full-connected neural network is fed with random values of weights with learning rate values between 0.2 and 0.3. Experiments suggested that learning rates of $\eta = 0.3$ produced the best results as for convergence rates and error drops.

Since that every agent can provide its own set of experiences, an individual vector was generated to combine the whole set of the team's experiences. To this end, every agent's experiences (figure 1) were averaged by computing the values E_{DS} and E_{WT} so to generate the following input vector to the NN:

$$\mathbf{X_{xp}} = (\sum_{i=1}^{N} \frac{E_{BL}^{i}}{N}, \sum_{i=1}^{N} \frac{E_{TI}^{i}}{N}, \sum_{i=1}^{N} \frac{E_{DA}^{i}}{N}, \sum_{i=1}^{N} \frac{E_{AD}^{i}}{N}, \sum_{i=1}^{N} \frac{E_{TD}^{i}}{N}, \sum_{i=1}^{N} \frac{E_{TS}^{i}}{N}, \sum_{i=1}^{N} \frac{E_{TA}^{i}}{N}, \sum_{i=1}^{N} \frac{E_{TO}^{i}}{N}, E_{DG}, E_{TR}) \quad (1)$$

where i represents the $i-th$ agent, and N is the total number of agent of a team (no keeper is considered).

Determining the game conditions is performed by getting information from a human expert. Real data were obtained by simulating a football game. A total of 9 games of 10-minutes each were performed with no side changes. Every 30 seconds, the game is stopped so to ask the expert to assess the game with a fitness ranging from 0 to 1 concerning the team's performance, which is regarded to as $GC_{hum} \in [0, 1]$. The expert must provide a value close to zero whenever he/she thinks the GC is unfavorable, whereas a value close to 1 is provided whenever the conditions are seen as favorable. For each time interval, the vector generated from equation 1 is obtained so that the corresponding input vector (X_{xp}) and expected output (GC_{hum}) are produced to train the NN.

Using this method, 171 training data were generated from which 40% was used for training purposes and 60% was used for testing the net. Results of the training tasks suggest that the model is well correlated with the expert's score.

Graphics in figure 1 shows the assessment of the trainer GC_{hum} versus the difference of score DS. For GC_{hum} values close to zero, a correlation with the negative difference of score can be observed. In addition, whenever the conditions are favorable, DS gets closer to the maximum (1). However, whenever the difference is minimum, the trainer provides a broad number of assessments. For example, for a low difference of score ($DS = 0.5$), the game conditions have been assessed from 0.1 to 0.9 by the trainer which is far more significant.

A metric for assessing the team's attitude was designed. This represents the attacking and defending efforts of a team for each of the two kinds of games (defense and attack). Attitude provides us with information regarding the defensive or attacking attitude of a team depending on which tasks the team is willing to perform most.

Accordingly, the attitude indicates the proportion of resources the team is willing to spend for defending and attacking during attack/defense games. Thus, the lower the value of attitude, the better the willingness to defend is. Attitude values close to 0 indicate that the team is willing to defend. On the contrary, if the value is close to 1, the team is more willing to perform attacking tasks.

Furthermore, a policy is defined as a team's most preferred attitude based on the GC. A human trainer is responsible to define and set the policy before the game gets started. Thus, a policy is a function on the game conditions and

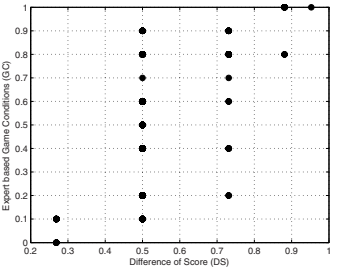

Fig. 1. (a) Difference of Score (DS) versus Game Conditions according to the human expert (GC_{hum})

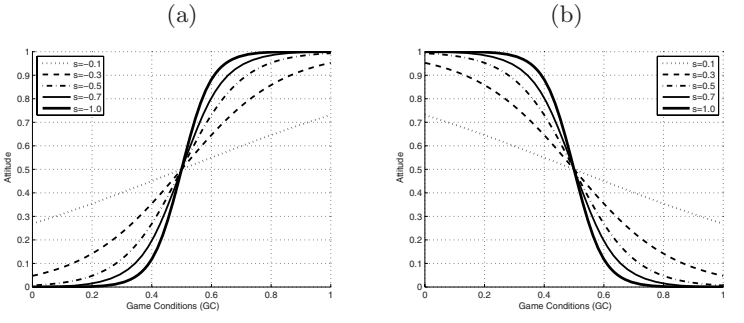

Fig. 2. Proposed Functions for policy. a) Defense Policy b) Offensive Policy.

provides an attitude to be adopted by the team, this is: $attitude = policy(GC) \in [0, 1]$. Graphics in figure 2 show the different policies a team may adopt. For example, figure 2(a) represents a policy generating a defensive attitude providing that GC are unfavorable for the team. A policy that produces an offensive attitude, assuming unfavorable conditions, can be seen at figure 2(b). Here, the human coach aims to set the game policy for the team, so the policy function which fits the expectation is provided to the model.

To select the most suitable team formation according to its *policy*, a selection strategy based on the *Roulette Wheel*, commonly used for some implementation of Genetic Algorithms, was applied [7].

Team formation involves assigning a defined area for the agents when they are not acting on the ball. This of area assignment (aka. *home*) is a specific position in axis X of the field divided into three areas: defense, central and offense. From here, the selection algorithm will pick a home distribution for each agent based on the team's attitude and current formation, keeping in mind that sudden changes on the team formation are not desirable as the agents would need to move a lot in order to reach the home position. Accordingly, a team formation is defined as:

$$\mathbf{F} = (F_x, F_y, F_z) \quad with \quad F_x + F_y + F_z = N$$

where F_x, F_y, F_z represent the number of players having a *defense home*, a *central home* and a *attack home*, respectively, and N is the number of agents in the formation. The fitness of a formation F can be computed based on the team's attitude and the closeness to the previous formation:

$$fitness_{att}(\mathbf{F}) = 1 - |att - (F_x * (\frac{0}{N}) + F_y * (\frac{0.5}{N}) + F_z(\frac{1}{N}))| \tag{2}$$

where *att* is the attitude value obtained from the game conditions and the policy. Next, the closeness-based fitness is determined by calculating the similarity between the current ($\mathbf{F'}$) and the new formation ($\mathbf{F}$). This is computed from the Euclidean distance between both formations and normalized to the maximum distance (i.e., the distance between formations $(3, 0, 0)$, $(0, 0, 3)$ and $(0, 3, 0)$, that is, $\sqrt{18}$):

$$fitness_{sim}(\mathbf{F}, \mathbf{F'}) = \frac{\sqrt{(\mathbf{F} - \mathbf{F'})^2}}{\sqrt{18}} \tag{3}$$

Both fitness functions are weighted according to an expert's defined parameter $\alpha \in [0, 1]$. Thus, the fitness of a formation is evaluated from the current formation and current attitude as follows:

$$fitness(\mathbf{F}, \mathbf{F'}) = \alpha fitness_{sim}(\mathbf{F}, \mathbf{F'}) + (1 - \alpha) fitness_{att}(\mathbf{F}) \tag{4}$$

Our selection algorithm computes the $fitness$ for all the possible formations $\mathbf{F_i}$ with $i \in [1, 10]$. An elitist criterion is used to pick the M formations having the highest $fitness$. Next, the fitness proportional to $fitness_{pi}$ is calculated for each selected formation as: $fitness_{pi} = \frac{fitness_i}{\sum_{i=1}^{10} fitness_i}$

Obtained fitnesses ($fitness_{pi}$) are then sorted and a random number β with uniform distribution is chosen. The algorithm cumulates the $fitness_{pi}$ in descending order until the value is greater or equal to β. Afterwards, the best formation having the last fitness $\mathbf{F_i}$ is selected. The outcome of the algorithm is the new formation $\mathbf{F_{new}}$ which represents the best fitness based on the team's attitude and the expert's criteria.

4 Evaluation and Results

The benefits of using the model for determining the game conditions on the fly were investigated by carrying out a series of experiments. This aimed to assess the robustness of the approach in terms of different time intervals, bandwidth efficiency, and the improvement of the team's performance compared to a different team in which no cooperation strategy is provided.

The model is capable of operating on different time intervals with no dependency on the sampling period T used for the training phase. The tolerance of the neural net model to different run-time intervals was assessed by performing a series of games under two approaches. The first approach considers sampling with continuous cumulation of experience, and the second one involves sampling with cumulation at independent intervals of experience. For these, nine games

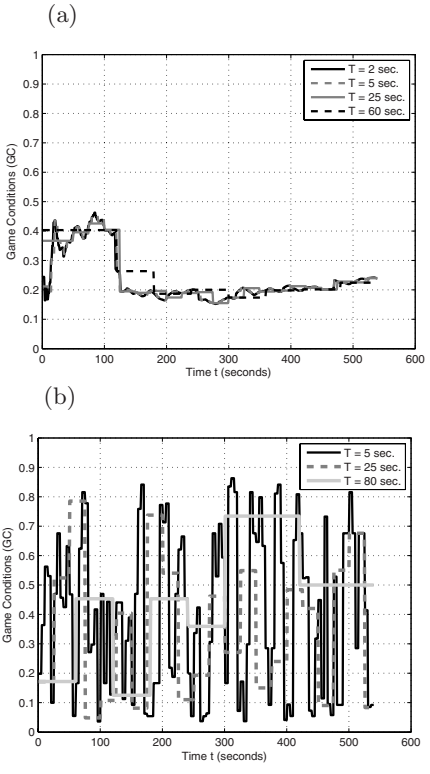

Fig. 3. (a) GC for continuous cumulated experience. (b) GC for independent-interval cumulated experiences.

were performed for training purposes. Changes to the conditions during these games having the worst performance can be seen at figure 3. The games do not use the dynamic formation strategy and so a fixed formation involving two defenders and one attacker is provided only. Runs in figure 3(a) show that despite having different sampling time intervals, the model is still capable of computing a correct value for GC, meaning that for the same game, the sampling time is not significant whenever the approach no.1 is applied.

The effects of the net having experience removed between samplings can be seen in figure 3(b). Based on the approach no. 2, results suggest that agents represent only game conditions from time $t - T$ to t, and accordingly GC is perceived as an *evaluation* of the last sampled time interval.

The efficiency of the model for dynamic team formation was assessed by performing 24 testing games in which 4 unseen opponent agents teams were used. These games used different approaches for cumulating experience, each of which was tested using three time intervals (10, 25, 60). The opponent teams were *Team 1* (potential fields based navigation), *Team 2* (dynamic role assignment with agent always gets the ball), *Team 3* (fixed formation involving two defenders and one central), *Team 4* (stands in the way of the opponent agents).

Table 2. Results of the Testing Games for the Model Team

Team	+ Score	- Score	Difference	Total Score
Baseline	34	36	-2	18
Model	54	38	16	37

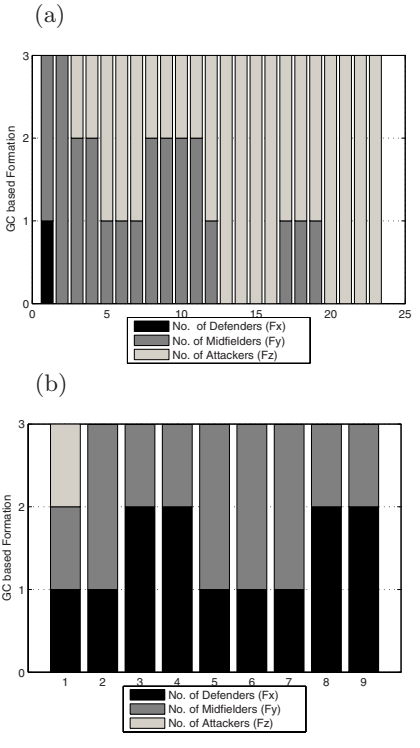

Fig. 4. Two Experimental Formations: (a) Formation with T=25. (b) Formation with T=60.

The model was implemented and based on the simple individual skills of team no. 3 which allowed us to assess the dynamic formation strategy. This team is referred to as the *Model Team* or **mTeam** whose testing parameters were as follows: defensive policy with a stepness of $s = 0.2$, sampling time every T seconds with $T \in \{10, 25, 60\}$, 4 agents. The same 24 games were performed by using the team no. 3 as baseline and having a fixed formation (a central, a defender, an attacker). This team does not use any adaptive formation strategy and is referred to as the *Base Team* or **bTeam**.

Performance was then assessed by assigning scores to the $bTeam$ and $mTeam$. The obtained score for each team is counted as for human football: the winner

team gets 3 points and the loser gets 0 points. If there is a tie, each team gets 1 point. Some key observation can be made from results in table 2.

Changes on the game conditions for both teams can be observed from graphics in figure 1. Dark line represents the game conditions for the *mTeam* which uses the dynamic formation strategy whereas the dotted line represents the conditions for the *bTeam*. In both scenarios, the teams' defensive policy allows them to keep the conditions favorable, which can be seen by looking at the scores, the obtained differences, and the game conditions (GC).

Graphics in figure 4 show the changes to the formations during the games. Performance of formations in figure 4(a) suggests that the initial formation is *centralized* with most of the players standing at the center of the field. As the game goes on, the formation becomes more offensive as the game conditions advice them to do so (GC close to 0.8). The fixed and rigid structure of the *bTeam* refrains it from scoring a single point due to the lack of supporters. Furthermore, the *bTeam* frequently tended to lose the ball at the middle of the field.

The performance of the formation of figure 4(b) can be seen in graphics of figure 1(b), in which the initial formation is *uniform* (i.e., there is a defender, an attacker and a supporter).

Our team takes a defensive formation involving two defenders and one central as the opponent team performs most of the offensive and risky actions. As the time goes on, game conditions become favorable for the model team as this keeps the ball most of the time. Overall, the strategy produces a more centralized formation using two centrals and one defender, and as a consequence favorable conditions are kept by scoring 3 goals.

5 Conclusions

A new approach to dynamic team formation using a simple Neural Net based model is described. The model is a mixture of simple neural net approaches, heuristics-based evaluation, multi-agent systems techniques and the human expert's experience so as to provide a robust method to compute game conditions which in turn allows the team to dynamically modify its positions and roles on the fly.

The experiments and real testing show that the neural net's prediction level as being trained by a human expert is well correlated with the automatically trained model. This suggests that the function applied to automatically generate samples is both robust and resource-efficient. The final design and implementation for our team also provides some interesting insights. Neural nets being trained at periods of time in which previous experience was removed before starting a new period of sampling proved to be useful to measure the change of conditions in a specific period of time. Hence this model may be applied to check whether some decision on a performed strategy had an instant effect on the game or not.

References

1. d'nverno, M., Luck, M.: Understanding Agent Systems, 2nd edn. Springer, Heidelberg (2004)
2. Frias, V., Sklar, E., Parsons, S.: Exploring auction mechanisms for role assignment in teams of autonomous robots. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004. LNCS (LNAI), vol. 3276, pp. 532–539. Springer, Heidelberg (2005)
3. Gerkey, B., Mataric, J.: A formal analysis and taxonomy of task allocation in multi-robot systems. The International Journal of Robotics Research 23(9), 939–954 (2004)
4. Hagan, M., Demuth, H., Beale, M.: Neural Network Design. Martin Hagan (2002)
5. Kuhlmann, G., Knox, W., Stone, P.: Know thine enemy: A champion robocup coach agent. In: Twenty-First National Conference on Artifical Inteligence (AAAI 2006), Boston, MA, July 2006, AAAI Press (2006)
6. Lerman, K., Jones, C., Galstyan, A., Mataric, M.: Analysis of dynamic task allocation in multi-robot systems. The International Journal of Robotics Research 25(3), 225–241 (2006)
7. Mitchell, M.: An Introduction to Genetic Algorithms. The MIT Press (1996)
8. Quinlan, M.J., Nicklin, S.P., Hong, K., Henderson, N., King, R.: The 2005 nubots team report. Technical report, School of Electrical Engineering and Computer Science, The University of Newcastle, Australia (2006)
9. Riley, P., Veloso, M.: Planning for distributed execution through use of probabilistic opponent models. In: Proceedings of the Sixth International Conference on AI Planning and Scheduling, pp. 72–81 (2002)