Journal of Computer Science 10 (1): 73-84, 2014
ISSN: 1549-3636

doi:10.3844/jcssp.2014.73.84 Published Online 10 (1) 2014 (http://www.thescipub.com/jcs.toc)

STRATEGY PATTERNS PREDICTION MODEL

Aram Baruch Gonzalez Perez and Jorge Adolfo Ramirez Uresti

Department of Information Technologies and Computation,
Instituto Tecnológico y de Estudios Superiores de Monterrey,
Campus Estado de México, Mexico

Received 2013-05-21; Revised 2013-08-03; Accepted 2013-11-12

ABSTRACT

Multi-agent systems are broadly known for being able to simulate real-life situations which require the interaction and cooperation of individuals. Opponent modeling can be used along with multi-agent systems to model complex situations such as competitions like soccer games. In this study, a model for predicting opponent moves based on their target is presented. The model is composed by an offline step (learning phase) and an online one (execution phase). The offline step gets and analyses previous experiences while the online step uses the data generated by offline analysis to predict opponent moves. This model is illustrated by an experiment with the RoboCup 2D Soccer Simulator. The proposed model was tested using 22 games to create the knowledge base and getting an accuracy rate over 80%.

Keywords: Opponent Modeling, Machine Learning, Case Based Reasoning

1. INTRODUCTION

An agent can be defined as an autonomous entity in an environment with the capacity of taking its own actions in order to achieve a goal (Wooldridge, 2008). Also, multi-agent systems take a set of agents in order to cooperate and achieve a common goal that cannot be completed without the help of other agents.

Multi-agent systems are broadly known for being able to simulate real-life situations which require the interaction and cooperation of individuals. These systems are really good in modeling situations where different autonomous individuals need to interact with each other and their environment in order to accomplish a certain goal.

Due to the multi-agent systems' nature, a common practice is to use them to represent a competitive environment in which two teams play against each other in order to accomplish a goal that directly interferes with the other team's objective. An example of this type of environments is the soccer game. A soccer game features two teams composed by eleven players each where the fundamental objective is to score more goals than the opponent. Using agents to represent each player is a natural way to model these kinds of environments, since most players tend to have similar capacities and in this case, only the goalkeeper has to attend different rules, as it is the only one who can grab the ball with its hands.

It was decided to test our Strategy Patterns Prediction Model (SPPM) in a soccer-like environment (Gonzalez and Uresti, 2011). This research is based on opponent modeling on multi-agent systems on RoboCup 2D Soccer Simulator, an environment where participants are in constant movement and interaction and it is focused on the defensive actions of the team. It is accomplished following a complete cycle that is going to be discussed in the rest of this document.

2. MATERIALS AND METHODS

2.1. Knowledge Base Creation

Knowing how the opponent is going to behave in a competitive environment such as soccer is a great way to increase a team's effectiveness by being able to anticipate the rival's actions.

Corresponding Author: Aram Baruch Gonzalez Perez, Department of Information Technologies and Computation, Instituto Tecnológico y de Estudios Superiores de Monterrey, Campus Estado de México, Mexico

In general, the result of predicting the behavior and movements of other agents and storing them in such a way that it is useful for making predictions is known as Opponent Modeling. Since it does not specify a unique technique to achieve its goal, the algorithms and methods used are chosen by each researcher. It can be implemented in most competitive games that involve two or more participants. While Opponent Modeling is a proven techinque to improve players or teams results (Del Giudice and Gmytrasiewicz, 2009; Richards and Amir, 2007; Parker *et al*., 2006; McCracken and Bowling, 2004; Laviers *et al*., 2009), it also needs a lot of information, in some cases it is needed to create a sub-domain of the original environment to reduce complexity.

Creating a good opponent model is not a trivial task and can take a large amount of processing time because it needs to include as many cases as possible, meaning a great amount of data. This causes that creating a functional opponent model, one that is based specifically on the actual rival and without any previous knowledge, a difficult task inside a dynamic environment such as soccer. It becomes almost impossible because of the little number of interactions that can be generalized into a real model of the entire team including its strategies (Stone *et al*., 2000). In this manner previous knowledge from the opponent is needed (Ramon *et al*., 2002; Kuhlmann *et al*., 2006; Del Giudice and Gmytrasiewicz, 2009).

2.2. Initial Setup

In order to create a useful opponent model for Strategy Patterns Prediction Model (SPPM), it is necessary to take into account previous experiences. In this case, records and logs of past games were used. These logs are automatically generated by the RoboCup 2D Soccer Simulator each time a game is executed and they are saved in a RCG file. The RCG files and team binaries used for this research can be found in the RoboCup (2013). The RCG files are binaries that can be reproduced by the Replay Tool Program (r2play) bundled with the RoboCup 2D Soccer Simulator. The binary files contain all the information necessary to recreate an entire game. Since this type of file is binary, it is difficult to obtain the information inside it because a format specification is not given. Instead of trying to get all the data from the RCG file, it can be converted to a readable XML file where all the info of the corresponding game is contained. For this process we used the rcg2xml tool, also bundled with the RoboCup 2D Soccer Simulator.

The original XML file obtained from an RCG files describes not only the server parameters, but also each of the ball's position, each player's position and actions across the game. Some information is presented even if the element that it is describing did not change across time. Keeping the data as it is presented can cause a big overhead of unnecessary information.

The default parameters create a standard soccer field with some flags that allow players to locate themselves and the other elements inside the field. **Figure 1** shows the default and official soccer field generated by the RoboCup 2D Soccer Simulator, it is the one used in competitions and for this research.

The prediction model intends to forecast the ball position when it is in the adversary's possession, for this reason not all the information contained in the XML is useful. In order to reduce the time needed to create the knowledge base, the unnecessary information inside the XML files is completely removed. This leaves only the data corresponding to the players' actions, players' positions, ball's position and game status.

To reduce the complexity of the opponent modeling process, it was decided that the field must be divided into zones. This division allows the system to be tolerant to the noise generated by the environment.

While dividing the field into zones has been done before (Arias and Uresti, 2008; Berger and Herfert, 2009), there is not a related work on optimizing the field division based on any criteria so we had to create a division that would serve the SPPM's purpose.

The division was made in such way so that the size of each one of the blocks generated is large enough to reduce system complexity and small enough to keep the prediction relevant. The division's size decision was made based on the fact that having a division consisting of small zones would give us too many combinations for search, resulting in no real advantages for creating a division at all. Creating big division zones ends up giving us a small search space allowing to reduce the time employed looking inside the search tree for possible solutions but it also affects the prediction's precision and therefore its usefulness.

This resulted in the soccer field being divided in medium-sized zones that allowed us to generate a grid consisting of 60 zones which is enough to keep the prediction relevant and the search tree in a reasonable size. Creating a different division with more zones ended creating a bigger decision tree because some patterns were divided in different leaves since the previous zones were divided. On behalf, reducing the number of zones ended in grouping patterns together and in cases creating an over generalization of the patterns. The final division is shown in **Fig. 2**. The number of zones was determined by trial and error. The field division also takes advantage of the flags provided by the RoboCup 2D Soccer Simulator which serve as reference points for the agents inside the game.

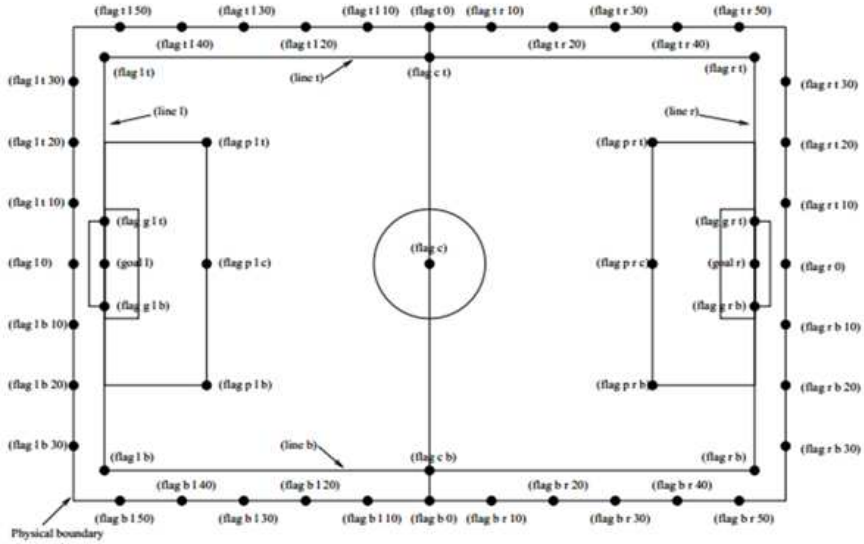

Fig. 1. RoboCup 2D Soccer Simulator official field

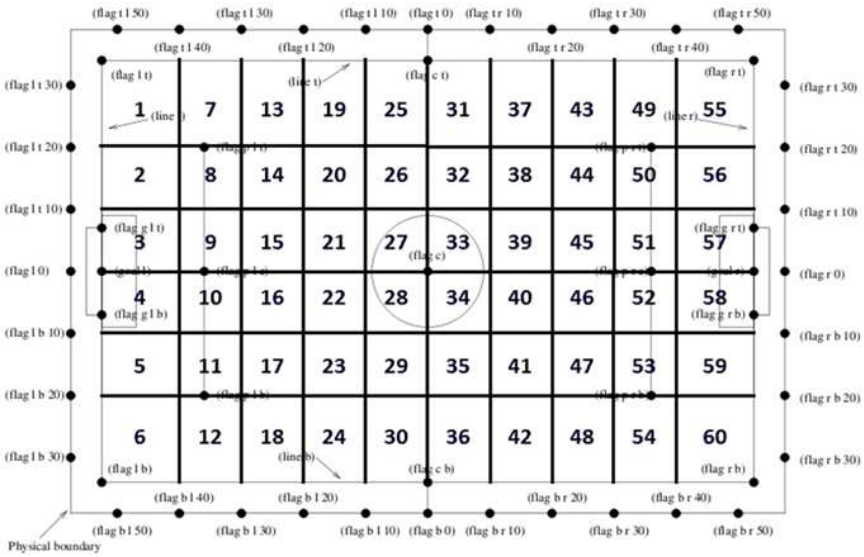

Fig. 2. Field divided into zones

2.3. Patterns

The knowledge base contains a series of patterns that are obtained from one or more RoboCup 2D Soccer Simulator log files. A pattern is defined as the route that the ball follows inside the game while one team keeps it. A team gets the possession of the ball when a member of his team kicks it and then loses it when a member of the opposing team kicks it or when an event that alters the game status change is presented, such as: A goal is scored, the ball goes out of bounds, the play time is over or a foul is committed. An example of a full pattern is shown in **Fig. 3**.

In order to create a visualization of the actual strategy patterns, we defined a short set of symbols that allowed us to follow the pattern development. The set of symbols is presented in **Fig. 4**.

```
[16, 11, 14, 20, 27, 31, 32, 37, 44, 52, 58]
[17, 5, 8, 10, 15, 17, 19, 20, 25, 43, 53]
[42, 9, 11, 30, 31, 38, 53, 54, 55, 57, 58]
[49, 8, 21, 26, 27, 27, 27, 29, 32, 50, 53]
[52, 2, 2, 14, 17, 21, 22, 30, 33, 37, 45]
[41, 1, 13, 18, 19, 29, 35, 38, 45, 48, 53]
[26, 3, 11, 17, 22, 23, 26, 38, 38, 38, 53]
[29, 2, 15, 19, 39, 41, 42, 51, 52, 53, 60]
[39, 3, 4, 20, 20, 28, 28, 35, 46, 50, 52]
[10, 1, 22, 29, 33, 33, 38, 39, 42, 45, 60]
[2, 15, 16, 18, 19, 20, 23, 34, 49, 52, 52]
[43, 3, 4, 9, 15, 38, 38, 41, 45, 47, 47]
```

Fig. 3. A pattern's set

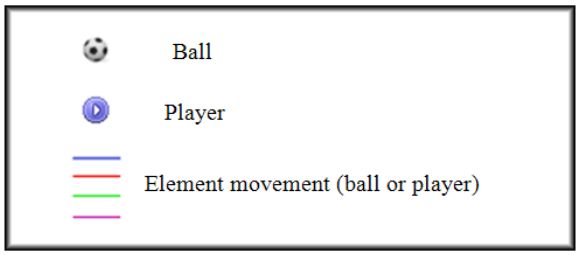

Fig. 4. Symbols for strategy patterns visualization

Each pattern inside the search tree is associated to a full strategy or play. This strategy includes the movement of each of the players that participate in it and the position of the ball. An example of this is shown in **Fig. 5**.

The patterns used to create the knowledge base have a minimum duration of 10 steps but not a maximum duration. The minimum duration restriction was set so that the team is able to search for the similar case or cases inside the knowledge base and still have time to complete a defensive action corresponding to the search result.

2.4. Search Tree

After the knowledge base is created, a search tree is also generated in order to facilitate the comparison between the actual game info and the patterns stored in the knowledge base. The search tree allows the SPPM to have an efficient way to find and compare similar patterns by inside a file that is not only smaller than the knowledge base one but also has the patterns ordered so that it requires less time.

The search tree generated from the knowledge base contains the zones where the ball and the attacking team are positioned at the start of a pattern as shown in **Fig. 6**. The ball's position is used as the first comparison parameter since it determines a team's play and actions. After the ball's position, each following node indicates the zones in which players are located. Since the knowledge base, we only use the players that have a direct interaction with the ball along the strategy pattern in order to reduce both files' size.

Each pattern was sorted (excluding the ball's position since it's always the first value) so that when the tree was created each of the branches keep an ascending order. This was done in order to make comparisons faster and to assure there is no need to implement a sorting method after creating the tree.

The search tree must be stored in order to be read by the team during a play. An XML file is used to do this since both have a hierarchical nature. During a game, the XML file is only read at the beginning in order to reduce the time it takes for the leader to compare the actual game status to the results in the search tree. An example of the XML search tree file is shown in **Fig. 7**.

While reading the tree file from the file system can be done in less than a game cycle, doing this repeatedly can derive in some failures like the system not properly freeing the file so it cannot be read immediately after it is used. Also depending on the implementation, reading the file can saturate the server since there are 6000 game cycles in total and in the worst case scenario the file must be read in each of the 6000 game cycles.

We ended up copying the entire search tree to the leader's memory. This approach did not cause any negative repercussion to that agent's performance during the game and it helped avoiding the above mentioned problems.

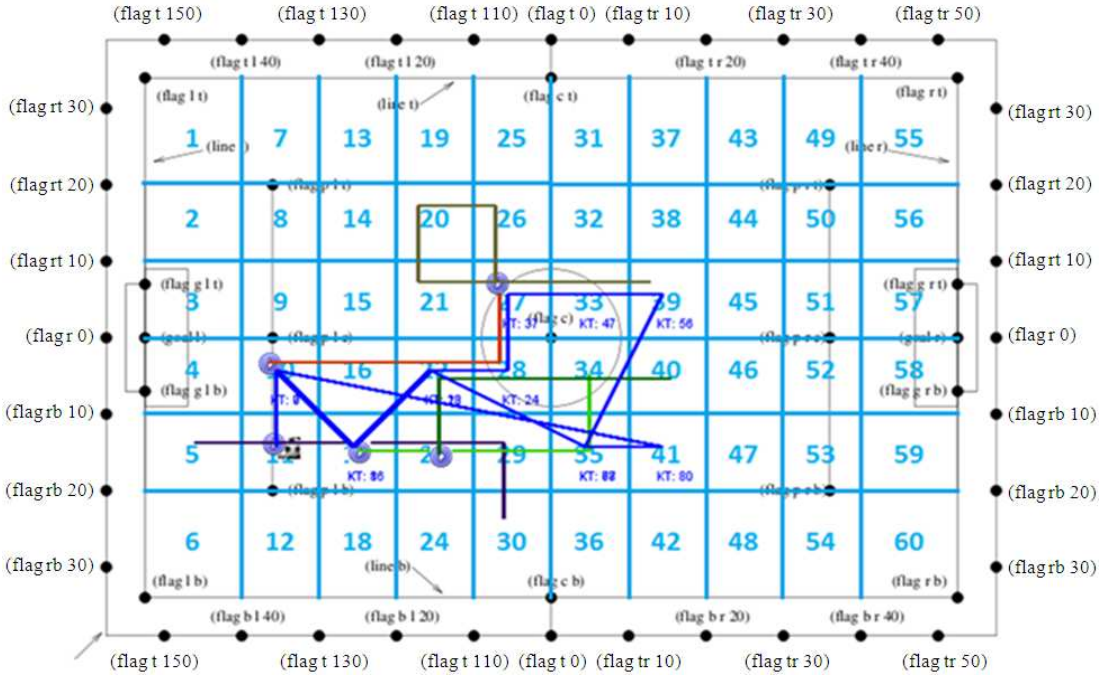

Fig. 5. A full pattern visualization

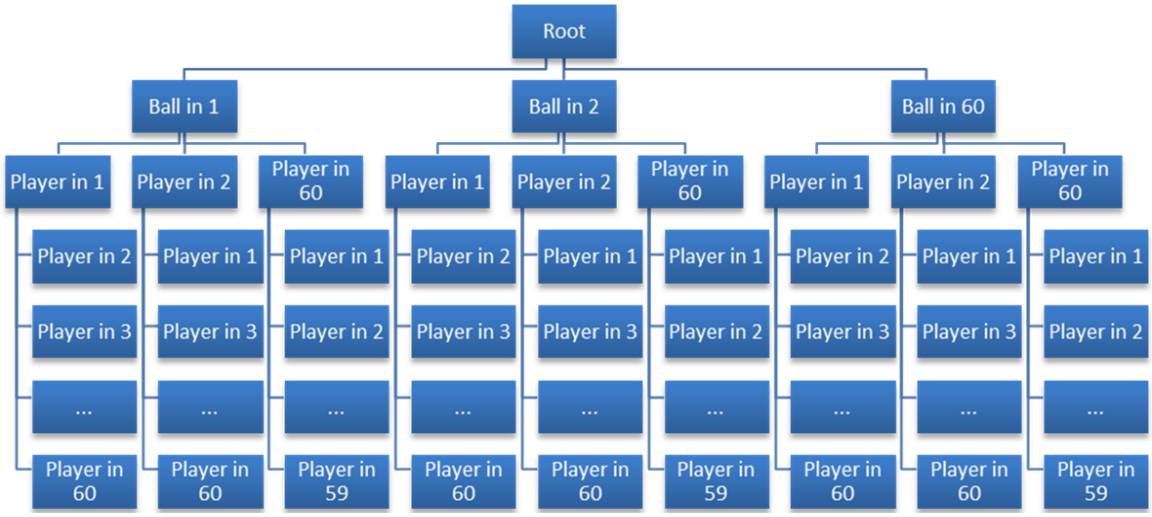

Fig. 6. General structure of the search tree

The rest of the team does not need to have a copy of the search tree since the leader is the only one that uses it to retrieve strategy patterns' information since we follow a centralized approach.

The creation of the knowledge base and the search tree using 22 RCG files took about 35 min. The search time for a single pattern takes less than a game cycle but there are times when a possible result is not found so the SPPM repeats the search progress but it uses the ball's neighbor areas. Searching the neighbor areas can delay the entire progress since it needs to search at least 9 different areas and their combinations.

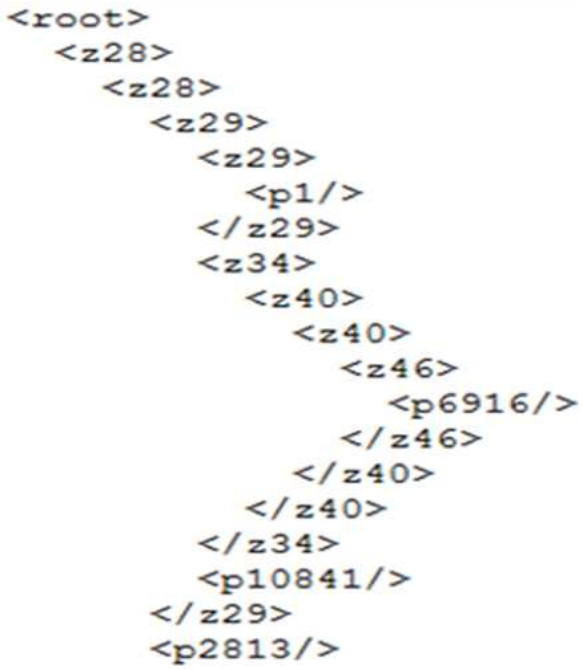

Fig. 7. An example of the XML file for a search tree

2.5. In-Game Features

In order to make use of the knowledge base created in the knowledge base creation phase of the model, SPPM needs to use and test this data into new games. The complete process that must be followed includes: Getting the actual game status information (opponent players' and ball positions), look for similar patterns inside the knowledge base and return the possible zones where the ball will be according to the cases stored in the knowledge base with the potential cover zones that will let the team respond to the rival.

This process is only a part of the complete SPPM, it is the one used during a game and it requires a knowledge base and its corresponding search tree to be created. While the search tree must be created from the knowledge base, the defensive actions taken inside the game can be totally independent of that process. The SPPM process predicts the ball position over time by analyzing and comparing actual game status and past experiences. A defensive action can use this information but it also can be a totally separated process.

To get the complete pattern and, knowing the fact that this is a multi-agent system, it is needed to obtain the partial information that each of the agents knows, due to this situation, the number of messages listened per cycle restrictions and the noise ones were eliminated.

2.6. Decision Process

Even with the modified communication parameters, tryAing to follow a distributed decision process was not possible since it required a full negotiation cycle involving all of the agents. This resulted in time wasted in just trying to coordinate who is going to take the decisions because the simulator is designed to only allow the agents to listen to their teammates' messages a cycle after they have been sent.

Therefore the initial prediction process was planned as follows:

- An agent should identify an appropriate time to begin the prediction process or it can be done in a fixed time. An appropriate time can be defined as a moment when the rival team is not in an imminent scoring chance position. For an imminent scoring change position, the rival team needs to have the possession of the ball and being close to our goal zone
- The agent sends messages to the rest of the team so the negotiation process is started
- Each agent evaluates if it can be the leader (coordinator) for the prediction process. If a leader is chosen then it informs its team about his new acquired role. For electing the leader the following factors were planned to be taken into consideration: Player distance to the ball, player's role (goalkeeper, defense, midfield or attacker), player's distance to enemy's team players and players position
- The leader agent sends messages to the rest of the team in order to obtain their information that includes their positions, the position of the ball and the position of the opponent agents
- The leader agent uses the search tree and gets the possible strategy patterns, then it evaluates each of the search tree results and creates a new pattern that contains the zones the ball is most likely to be located
- The leader sends the prediction pattern to its team

The minimum messages used to decide the leader role are 6 since at least 3 messages are used (proposal, answer and confirmation) and each message takes 1 cycle to be sent and 1 to be received. Another problem with this process occurs when more than one agent has the best possibility to acquire the leader role. Another negotiation process between them must occur and even if they can solve it with at their first try, it would require at least another 4 cycles.

Having a constant time overhead whenever a prediction is meant to be done extends the time the entire process requires and, since RoboCup 2D Soccer Simulator is an entirely dynamic environment, the time consumed in deciding the team's leader is un-viable because the prediction may no longer coincide with the actual field state.

In order to reduce the time spent on negotiation issues, it was decided to follow a centralized approach for decision process. In human soccer, the goalkeeper usually has a complete vision of the field and also is the player that most likely has the fewest interactions with the ball. Based on the characteristics previously mentioned, we decided that it should be the agent who receives all the data and takes all the decisions. The final communication process is shown in **Fig. 8**.

2.7. Prediction

Once all the data is received, it needs to be cleaned and consolidated into a single pattern so that it can be used into the search tree. The leader agent determines the ball's position by calculating an average from the positions received and then a single zone can be assigned. A similar process is followed for each of the opponent players reported and duplicates are eliminated. Then the information is merged in a single list in a format that can be used by the search tree.

Once the result is taken from the search tree, we get all the possible patterns' IDs with the best matches. The criteria to decide those matches is based in the distance between the actual field status and the one contained in the pattern, giving more importance to strategies that involve more players. This is called the similarity measure. The similarity measure formula is the following:

$$s = \left(\left(\sqrt{(X_{br} - X_{bp})^2 + (Y_{br} - Y_{bp})^2}\right) \times \alpha\right) + \sum_{i=1}^{n} \frac{\sqrt{(X_{pri} - X_{ppi})^2 + (Y_{pri} - Y_{ppi})^2}}{n}$$

where, X_{br} and Y_{br} are the X and Y positions of the ball in the actual status and X_{bp} and Y_{bp} are the X and Y positions of the ball inside the pattern. X_{pri} and Y_{pri} are the X and Y position of the players in the actual status while the X_{ppi} and Y_{ppi} indicate the position of the players inside the pattern analyzed. The symbol α represents a constant value used to give the ball's position more importance in the comparison. For our tests we decided to use a value of $\alpha = 1.5$.This process can be compared and is based on Case Based Reasoning (CBR). CBR uses human like thinking in order to react to actual circumstances based on previous experiences. It has been openly used in this domain (Arias and Uresti, 2008; Berger and Herfert, 2009). The basic CBR process is defined as the following actions:

- Retrieve-Similar cases or situations must be retrieved from memory given the actual problem conditions
- Reuse-The retrieved cases must be mapped to the new problem even if they need to be adapted to fit the situation
- Revise-Test and evaluate the possible solutions in the new scenario
- Retain-After adapting, testing and evaluating, store the new solution as a new case in the memory

CBR is considered as a cycle that allows the system to constantly learn new experiences or cases.

Having all the play patterns that coincide with the actual in-game status, the zones where the ball was during those plays can be taken from the knowledge base created from previous data and for the prediction a sample is taken each 5 steps (again this is determined in this case by the distance a player can travel). With this information the probabilities of the ball being in a certain zone can be deducted and this is shown in **Fig. 9**.

Having the zones at a 5 cycle interval step, we can determine which ones need to be covered by our players (the ones with most possibilities of the ball being there) and the ones that can be ignored. Being this a soccer game, the areas that need to be covered are the ones that surround the predicted zones and that are between the ball position and our team's goal box.

The next step is to decide whether it is a viable option to send the players to the zones, if it is already too late or even if the prediction is wrong and therefore there is no point to cover them. Another issue to take into account is that even if the entire search process takes about 1 or 2 steps to be completed, there are cases when it takes about 12 to 15 steps to end the entire process. In those cases, a condition in which the process stops if it has taken too much time or if the ball is close to the goal box must be included. These situations affect the goalkeeper's ability to react to an attack, so it must stop performing the decision process and focus in defending its own goal box.

Fig. 8. SPPM Communication process

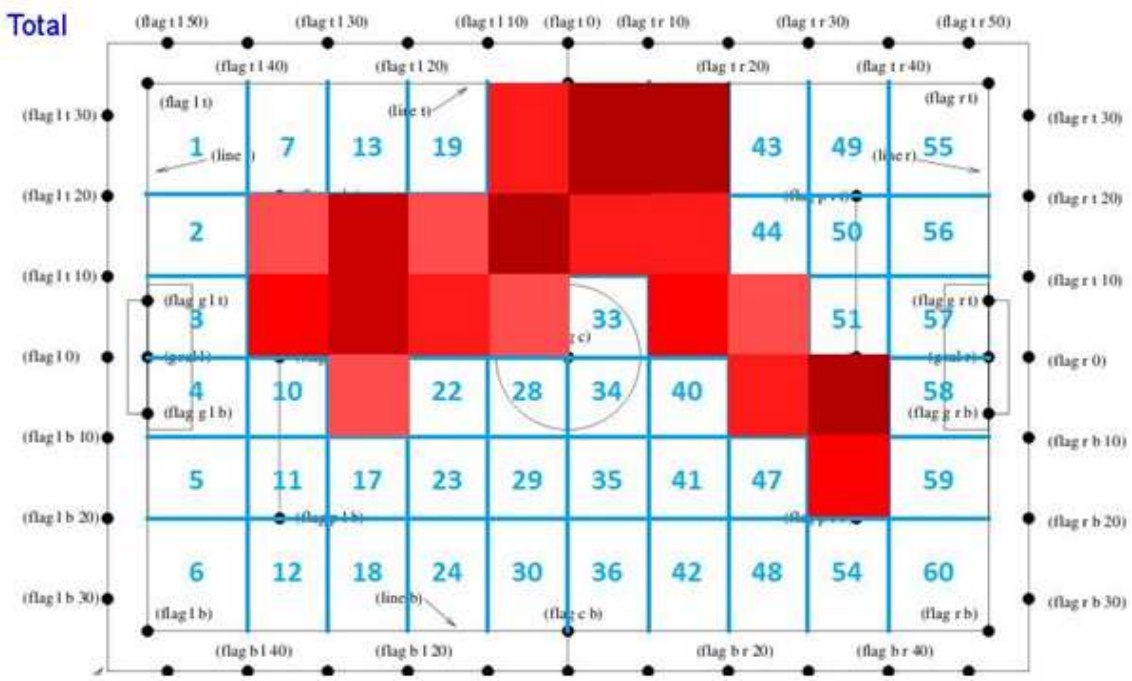

Fig. 9. Darker zones indicate a higher probability

After analyzing all the combinations of possible situations mentioned in the previous paragraph that can be present during a game, it was decided that the following are the only ones that can really affect the team and in particular the goalkeeper's individual goals: Catching the ball, reacting to opponents approaching the goal line and clearing the ball from the penalty area.

If the goalkeeper determines that it is viable to make a defensive action, it communicates it to the rest of the team with the time and zones that need to be covered so that the players that are closest to those zones go to them and try to recover the ball.

If any of the agents perceive that the ball or the play is not similar to the prediction, then an alert message is sent to inform that the covering zones have almost no probabilities to have the ball inside.

The logs generated from the test files can be analyzed and included inside the knowledge base; this allows the system to evolve and to get more information about different and new situations. This feature allows our SPPM to respond better in next matches and to evolve across time.

3. RESULTS

The tests made during this research were divided into three types: Test with our team against teams that were previously analyzed and included inside the knowledge base, tests against teams not included in the knowledge base and games where our team was not involved at all.

The first test type was intended to prove the system reliability against opponents' movement of already known teams. The second one does the same but with other teams and situations that are unknown for the system. The third set is made to test it the prediction's accuracy in general.

Twenty-seven analyses were done to get the results presented here, where 12 analyses were done over previously analyzed teams, 3 over not previously analyzed teams and 12 over games that did not involve the team developed for this research.

The following was used in order to create the knowledge base for these results:

- 22 games were used (143 MB in GZ files)

- An XML file was created for each of the 22 games (1.4 GB)
- A single knowledge base created with 604 strategy patterns (1.06 MB XML file)
- A single search tree was created (62.6KB XML file)

The entire knowledge base creation was done in a 2.0 GHZ dual core PC and it took about 35 min for the entire process to be completed.

Overall our SPPM achieved to get a prediction of the ball position with a precision over the 80% in an acceptable range defined by being in a distance equivalent to at most one zone far from the real ball position. This lets the team define coverage zones along time so that the adversary team can be stopped and the ball recovered.

The prediction accuracy was probed in three different groups of tests. The first group involved testing the same teams used to create the knowledge base against our team (PA-Previously Analyzed), the second group follows a similar mechanic but whit teams that were not included in the learning phase (NPA-Not Previously Analyzed) and finally the third group consists of a set of games in where our team did not participate (OT-Other Teams).

The results involving the actual distance and the predicted one are shown in **Fig. 10**. According to the results, taking into consideration the mean distances in X and in Y separately is the best way to get the actual position of the ball.

Given the results generated by analyzing the distances, the predictions obtained during the course of this research where accommodated in their respective group based on the mean distance during the whole play between the prognosticated zones and the real ones.

The **Fig. 11** shows the percentage of the results and the group that they belong to, this graphic only takes into account the results in mean distance in X and Y because of the results previously generated. It is shown that most of the time (more than 80%); the zones predicted are close enough to the real ones to make a defensive action that lets the team try to recover the ball.

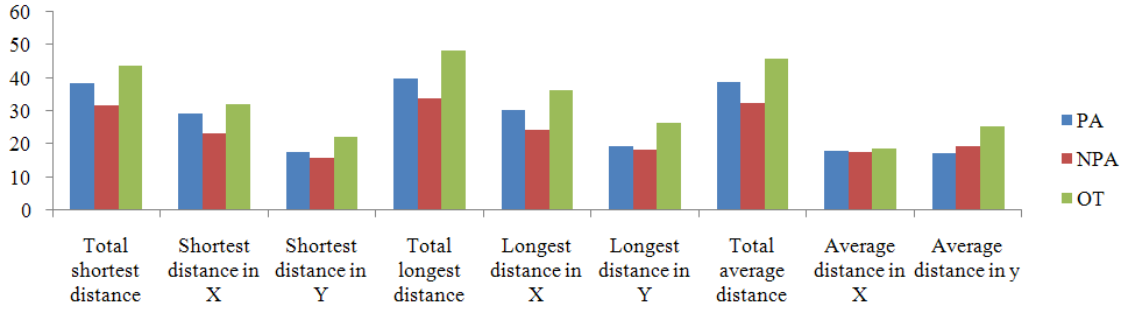

Fig. 10. Distances got in the results for Previously Analyzed teams (PA), Not Previously Analyzed teams (NPA) and Other Teams (OT) that not involve the one developed for this research

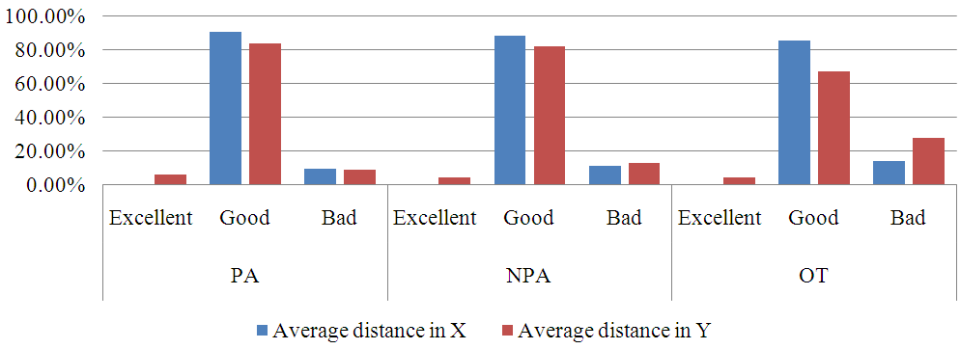

Fig. 11. Percentage of usefulness got in the results for Previously Analyzed teams (PA), Not Previously Analyzed teams (NPA) and Other Teams (OT) that not involve the one developed for this research

4. DISCUSSION

Analyzing the results and the circumstances presented inside the games that were shown the following circumstances were observed.

There are some teams less susceptible to fall into the model predictions. This is caused depending on the opponent team's ability to react to our predictions and the capacity of both teams to play the game. There are some strategies that are more likely to be presented in a game with certain circumstances (like a team's dominion over its rival) than in others.

Some predictions do not end in the zone that was supposed to be in because in some cases the opponent changes its route when a member of our team got close. In other cases the opponent directly shot to goal during the circumstance described and sometimes our team managed to get the ball effectively ruining the prediction. All these situations can reduce the system's prediction precision because given the actual game status; those situations can change completely the strategy outcome.

Some teams change their tactics depending on the score and the time left to end the game.

While the prediction process got a good accuracy rate (over 80%), the defensive actions implemented by our team did not allow us to corroborate the entire SPPM's goal. The actual game results did not change in our favor since our team is way below actual competing teams' level.

Even though the considerations described before, it is considered that the predictions have a very good accuracy rate and are done in an efficient time so that the agents can react to it in real time.

5. CONCLUSION

In this study a model for opponent strategies modeling (SPPM) is presented. SPPM obtains information from RoboCup 2D Soccer Simulation log files, converts this information into a knowledge base containing strategy patterns used by opponents. With these strategy patterns a search tree is generated to index all possible cases in such a way that any search of the tree is fast enough to be useful in real-time. When in play, the opponent formation of players and ball position is detected, analyzed for possible matches in our knowledge base, a probability of the ball position in the future states is generated and a final decision on how to defend against this possible strategy is made. All this process is done fast enough so a useful real-time decision is made.

The importance of the creation of the knowledge base that supports the model is also discussed as well as the actions that take place inside the environment that the model is used. The model was tested in the RoboCup 2D Soccer Simulator so that it is proved in a dynamical environment which also has an opponent who takes its own decisions and follows its own course of action. The model discussed in this study gives an accuracy of around 80% in the tests. Taking into consideration the dynamic nature of the environment in which it takes place, it can be said it is a really good percentage.

6. REFERENCES

Arias, M.A. and J.R. Uresti, 2008. Team agent behavior architecture in robot soccer. Proceedings of the IEEE Latin American Robotic Symposium, Oct. 29-30, IEEE Xplore Press, Natal, Rio Grande do Norte, pp: 249-256. DOI: 10.1109/LARS.2008.35

Berger, R. and D. Herfert, 2009. AT Humbolt Team Description 2009. Proceedings of the RoboCup International Symposium, Jun. 30-Jul. 3, Springer-Verlag, Austria.

Del Giudice, A. and P. Gmytrasiewicz, 2009. Towards strategic kriegspiel play with opponent modeling. Proceedings of the AAAI Spring Symposiam, Mar. 26-28, AAAI Press, California.

Gonzalez, A.B. and J.A.R. Uresti, 2011. Strategy Patterns Prediction Model (SPPM). Proceedings of the 10th International Conference on Mexican Advances in Artificial Intelligence, Nov. 26-Dec. 4, Springer Berlin Heidelberg, Puebla, Mexico, pp: 101-112. DOI: 10.1007/978-3-642-25324-9_9

Kuhlmann, G., W. Knox and P. Stone, 2006. Know thine enemy: A champion robocup coach agent. Proceedings of the 21st National Conference on Artificial Intelligence, Jul. 16-20, AAAI Press, Menlo Park, CA., pp: 1463-1468.

Laviers, K., G. Sukthankar, D.W. Aha, M. Molineaux and C. Darken *et al*., 2009. Improving offensive performance through opponent modeling. Proceedings of the 5th Artificial Intelligence for Interactive Digital Entertainment Conference, Oct 14-16, AAAI Press, Darken, Christian, pp: 58-63.

McCracken, P. and M. Bowling, 2004. Safe strategies for agent modelling in games. University of Alberta.

Parker, A., D. Nau and V.S. Subrahmanian, 2006. Overconfidence or paranoia? Search in imperfect-information games. Proceedings of the 21st National Conference on Artificial Intelligence, (CAI' 06), AAAI Press, pp: 1045-1050.

Ramon, J., N. Jacobs and H. Blockeel, 2002. Opponent modeling by analysing play. Proceedings of the 1st Workshop on Agents in Computer Games, Jul. 27-27, Edmonton, Canada, pp: 1-8.

Richards, M. and E. Amir, 2007. Opponent modeling in scrabble. Proceedings of the 20th International Joint Conference on Artifical Intelligence, Morgan Kaufmann Publishers Inc. San Francisco, CA, USA., pp: 1482-1487.

RoboCup, 2013. RoboCup Official Repository.

Stone, P., P. Riley and M. Veloso, 2000. Defining and using ideal teammate and opponent agent models: A case study in robotic soccer. Proceedings of the 4th International Conference on MultiAgent Systems, Jul. 10-12, IEEE Xplore Press, Boston, MA., pp: 441-442. DOI: 10.1109/ICMAS.2000.858515

Wooldridge, M., 2008. An Introduction to MultiAgent Systems. 1st Edn., John Wiley and Sons, ISBN-10: 0470353473, pp: 366.