VIENA: VIsual ENvironment for Analyzing *RoboCup* Soccer Teams

José Antonio Iglesias
Carlos III University
Madrid
jiglesia@inf.uc3m.es

Miguel Angel Palacin
Carlos III University
Madrid
mpalacind@gmail.com

Agapito Ledezma
Carlos III University
Madrid
ledezma@inf.uc3m.es

Araceli Sanchis
Carlos III University
Madrid
masm@inf.uc3m.es

Abstract

Nowadays, one of the main subjects of research in multi-agent systems deals with the development of autonomous agents that can interact efficiently with the other agents in the environment. Opponent modeling is a technique in game-playing which attempts to create a model of the behavior of the opponent. This model can be used to predict the future actions of the opponent and generate appropriate strategies to play against it.

RoboCup is an international joint project that aims to foster Artificial Intelligence (AI) and intelligent robotics research by providing a standard problem. *RoboCup* offers different challenges for intelligent agent researchers in a dynamic, real-time and multi-agent domain. One of these challenges, especially in the *Simulation League*, is the opponent modeling, which is crucial for the ultimate goal of the *RoboCup* project: *develop a team of fully autonomous*. This aspect has motivated the construction of a tool for analyzing the behavior of a soccer team: VIENA

Although there are several tools in the RoboCup environment for analyzing a soccer games from a *logfile* and displaying statistical data (a survey of these tools is presented in this paper), VIENA analyzes a soccer team behavior using different methods proposed for the user.

1 Introduction

Humans usually try to recognize the behavior of others in order to interact with them efficiently. This prediction is also very interesting in the multi-agents systems with adversary agents, in which the agents need to interact and cooperate with other agents in order to increase its ability to compete. Opponent modeling is a technique in computer game-playing which attempts to create a model of the behavior of the opponent. Nowadays, opponent modeling in multi-agent systems is increasingly becoming more significant and complex.

Robot World Cup (*RoboCup*) was proposed in 1994 by Hiroaki Kitano as a new standard problem for Artificial Intelligence (AI) and robotics research by providing a standard problem where wide range of technologies can be integrated and examined. Initially, one of the challenges proposed by *RoboCup* was the opponent modeling: *Agent modeling -modeling and reasoning about other agent´s goals, plans, knowledge, capabilities, or emotions - is a key issue in multi-agent interaction. The RoboCup opponent modeling challenge calls for research on modeling a team of opponents in a dynamic, multi-agent domain. The modeling issues in RoboCup can be broken into three parts: On-line tracking, On-line strategy recognition and Off-line review* [1].

Because of the dynamical and adversarial nature of a soccer play, opponent modeling has been very relevant in the different *RoboCup* leagues. Specially, in the Simulation League,

this aspect plays a crucial role and lot of research has been done using this simulated environment. In 2001, a new competition, called *RoboCup Coach Competition* was created. In this competition, an on-line coach was able to act as an advice-giving agent [2]. In order to improve the behavior of the coached team, the coach could receive a global view of the world environment and could communicate with the team [3]. *RoboCup Coach Competition* changed in 2005 in order to emphasize opponent-modeling approaches. The main goal of that new competition was to model the behavior of a soccer team. A play pattern (way of playing soccer) was activated in a test team and the coach should detect this pattern and then, recognize the patterns followed by a team by observation. That competition was held in 2005 and 2006 and the proposed environment has been used for lot of work in opponent modeling. Nowadays, there is not an special competition for this task and it is normally integrated in the robot soccer teams.

As the significance of the opponent modeling is crucial in the *RoboCup* environment, we present in this paper a tool that we have created for analyzing the behavior of a simulated soccer team. This tool is called VIENA: VIsual ENvironment for Analyzing *RoboCup* Soccer Teams. Although there are several tools for analyzing games from *logfiles* and displaying statistical data (a survey of these tools is presented in the paper), the main difference between VIENA and those other tools is that VIENA analyzes a soccer team behavior using different methods proposed for the user. Then, the user can easily analyzes that models and use them to create the appropriate strategies to play.

The rest of the paper is organized as follows. In Section 2 we provide a brief overview of the related work in opponent modeling, and a survey of the most interesting tools in the RoboCup Environment. Section 3 describes in detail the proposed tool: VIENA. Section 4 contains future work and concluding remarks.

2 Related Work

In general, an opponent model is an abstracted description of the behavior of one or several players in a game. The beginning of opponent modeling is a work done in 1996 by Carmel and Markovitch [4], in which it is introduced the M* algorithm, a generalization of the minimax algorithm that uses an arbitrary opponent model to benefit from its flaws.

2.1 Opponent Modeling in the *RoboCup* Simulation

Although there are lot of dynamic multi-agent domains with adversary agents, in which opponent modeling can be implemented efficiently; in this paper we focus on the opponent modeling in the *RoboCup Soccer Simulation* environment. This environment provides a good platform for modeling a team of opponents in a dynamic, multi-agent domain and a large number of publications related to the *RoboCup Soccer Simulation League* have been publicated[1] from 1996. In this environment many approaches, in which each agent observes and recognizes the behavior of the adversaries [5] have been proposed.

However, a frequently used opponent modeling approach in the *RoboCup* environment is to rely on an omniscient agent (*Coach*) to recognize the opponent behavior and to communicate to the team agents the model of the opponent or an strategy for that model. In addition, in order to focus entirely on opponent modeling the *RoboCup Simulation Coach Competition* was held in 2001, 2002, 2003, 2005 and 2006. This competition is situated within the same soccer server, but instead of creating a full soccer team, a single *coach* agent (which has a full view of the field but only can advice to its team via the standardized language called Clang [6]) must be implemented. The main advantages of the *coach* is that it can *see* the field noise-free and it has access to *logfiles* of past games played by the team to model (opponent).

[1] http://www.cs.utexas.edu/~pstone/tmp/sim-league-research.pdf

In the first three coach competitions (2001, 2002 and 2003), the opponent *logfiles* (recordings of their past games) could be analyzed during 24 hours. Taking into account this competition, interesting researches were carried out: Riley et al. [7] presented several implemented coaching techniques for a simulated robotic soccer domain and justified that coaching can help teams improve in this domain. Kuhlmann et al. [8] presented a multi-faceted learning approach to giving advice in *RoboCup* simulated soccer. Dulalia et al. [9] developed a system (*SimSoccer Coach*) that shows single agent learning by analyzing the fixed opponent´s behavior.

RoboCup Coach Competition changed in 2005 in order to emphasize opponent-modeling approaches. In this competition the coach agents were directly evaluated based on how they model a team which performs a pattern (predictable and exploitable). After modeling the team, the coach is rated on how well it recognizes the pattern. This environment is the base for a large number of works: Kuhlmann et al. [10] modeled a soccer team by characterizing their behavior with a set of features calculated from statistics gathered while observing a game. The winners of the *RoboCup Coach 2007 Competition* (Ramin Fathzadeh et al.) presented in [11] a novel learning architecture for modeling the opponent and a rule based expert system architecture to provide a provoking strategy for opponent players. Recently, Iglesias et al. [12] presented a novel method used by the CAOS team to model and recognize successfully the behavior of a soccer team.

2.2 *RoboCup* Simulation Tools

RoboCup Simulation Competition uses a simulator (*rcssserver*) which is a network-based graphical simulation environment for multiple autonomous mobile robots in a 2D space. *RoboCup* simulation games are recorded as *logfiles*, in which the positions of the ball and all players for both teams at each simulation cycle are stored. The *RoboCup Soccer Simulator Monitor*, called *rcssmonitor* (Figure 1), is used to view a simulation of the game as it takes place by connecting to the *rcssserver* or to view the playback of a simulation by connecting to the *rcsslogplayer*.

Figure 1: The RoboCup Coach Soccer Server Monitor (*rcssmonitor*).

However, during the last years, there have been created many sophisticated tools for analyzing RoboCup simulated games:

Virtual RoboCup [13] is a real-time 3D visualization tool for 2D simulated soccer games as played in the *RoboCup* simulation league. Players are modeled as anthropomorphic animated figures. **LogMonitor** [14] is a tool for analyzing games from *logfiles* and displaying statistical data such as counts of soccer plays. **Team Assistant** [15] is a log-player/debugger/analyzer for *RoboCup* soccer simulation. The analyzer proposed in this tool recognize different actions and graphically display them on the field. **Logalyzer** [16] is a tool for visualization and analysis of *RoboCup logfiles* which provides information about detected actions and several visualizations for the collected data about a soccer game using action graphs.

3 VIENA

RoboCup Simulation games are recorded as *logfiles*, which store all the relevant information about a game (positions of the ball and the players of both teams at every simulation step, stamina of each player and so on). VIENA is a tool for visualizing and analyzing

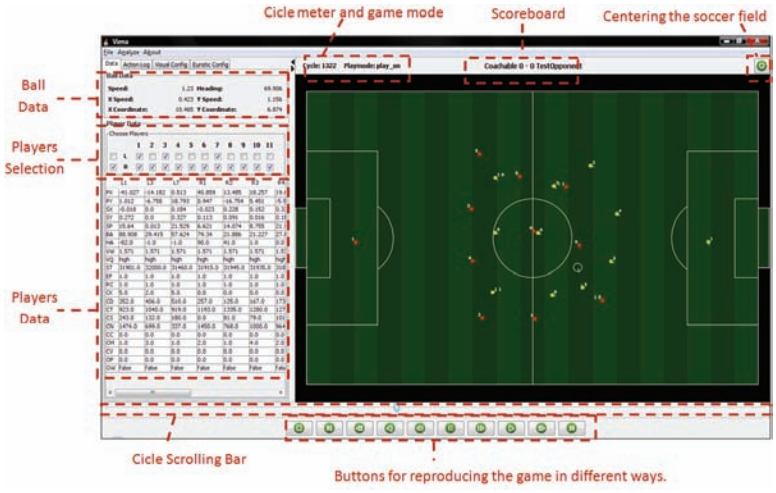

Figure 2: Viena. Soccer field and Data Area.

RoboCup simulation games from those *logfiles*. Thus, VIENA[2] allows: (1) Graphical visualization of a RoboCup Soccer Game using a dynamic data representation. The properties of all the players (position, speed, stamina...) are also shown. (2) Detection of soccer actions. (3) Analysis of a soccer team behavior by using different algorithms.

In the following subsections, these functionalities are explained in detail.

3.1 Graphical visualization of a RoboCup Soccer Game

The *RoboCup Soccer Simulator Monitor* (*rcssmonitor*) displays *on line* a soccer game that is currently simulated. However, VIENA replays *logfiles* of a previously simulated game (Figure 2) and shows the properties of all the players (position, speed, stamina...) in each cycle.

In order to personalize the visualization of the soccer field, VIENA allows to configure dynamically different parameters. For example, the tackle area of a player is represented with a rectangle around the player and if a player is the owner of the ball, it is surround with a triangle (Figure 3), but the size of these areas can be changed by the users.

Figure 3: Viena. Visualization Area.

3.2 Detection of soccer actions

For this functionality, it is necessary to to extract the important information from the *logfiles* to detect the soccer actions during the game. With this purpose, Kuhlmann et al. [8] describe a procedure to identify high-level actions in a soccer game. An action represents a recognized atomic behavior. The goal of their work is to get a coach that learns to predict opponent agents' behavior from past observations and generates advices to improve its

[2]Available at: http://www.caos.inf.uc3m.es/~viena/. A demonstration of how this tool works is available at: http://www.youtube.com/watch?v=BLCu7cWVk3w

team's performance. Based on that work we create the following method to detect soccer actions from the *logfiles* of a soccer game.

At every cycle of the server, each agent updates its world model with data such as positions and velocities, as well as cumulative data such total travel distances and average positions. As the most relevant information of these *logfiles* must be stored, in every cycle the following data is extracted:

- *Cycle*: A number that enables arrange the actions.
- *Ball Position*: Represented by a point in the Cartesian coordinate system.
- *Teammate Positions* and *Opponent Position*: Each teammate and opponent position is represented by a point in the Cartesian coordinate system.
- *Ball Possessor*: A number that indicates who is the owner of the ball.

After extracting the data from the *logfiles*, what actions have occurred must be inferred. In the work mentioned above, Kuhlmann et al. [8] propose eight soccer actions to create advices in the Simulation Soccer. Taking this work into account, VIENA identifies nine different actions. There is some uncertainty inherent in the results because there are actions very hardly to identify, even if it is done by a soccer expert.

One of the most important data to identify high-level actions is the ball owner every cycle. When a ball possession change occurs, it means that an action is taking place. VIENA can classify the following soccer actions:

- *PassXtoY*: If a player X of the team T kicks the ball and a teammate Y gains possession, then the ball owner made a pass. Perhaps the ball owner did not want to do this pass, but we can not consider this assumption. This action stores both the player who makes the pass and the player who gains possession.
- *InterceptedPassXtoY (T)*: If the player X kicks the ball and the opponent Y gains possession within a reasonable distance of the ball owner, the action is assumed to be an intercepted pass.
- *StealXfromY (T)*: If a player X kicks the ball and an opponent Y gains possession, then the opponent stole the ball from the ball owner.
- *FoulX (T)*: If the player X commits a foul.
- *ShootX*: If the player X kicks the ball and at the end of the interval, the ball is close to the goal.
- *GoalX (T)*: If a player X kicks the ball and at the end of the interval, the ball is in the goal, the action is classified as a goal by the player X.
- *KickInX*: If the player X kicks the ball when the state of the game is "kickIn".
- *KickOffX*: If the player X kicks the ball when the state of the game is "kickOff".
- *CornerX*: If the player X kicks the ball after other player shoot the ball.
- *Free Kick (T)*: If goalkeeper of the team T kicks the ball after holding it.

3.3 Analysis of a soccer team behavior by using different algorithms

Several research works propose different methods for analyzing and modeling a soccer team behavior. One of the goals of VIENA is to create human-understandable models of a soccer simulated team from the *logfile* of a game. As these models are evaluated and analyzed to get information about the behavior of the opponent, it is important to create human-understandable models of a soccer team.

VIENA proposes two different methods to create a soccer simulated team model. However, it is important to note that the way the tool has been developed ables to add new methods for creating behavior models made by any user.

The two methods developed in VIENA are: TF-IDF and Trie-ChiSquare. These methods are briefly described as follows:

3.3.1 TF-IDF

This method is based on the relevance feedback algorithm proposed by Rocchio [17]. TFIDF (Term Frequency - Inverse Document Frequency) is a common method often used in Information Retrieval (IR) problems. This method is based on a statistical measure (*weight*) used to evaluate how *important* a word is to a document in a collection: First, it evaluates the word frequency in the document (the more a word appears in the document, the more it is considered to be significant in the document). In addition, IDF measures how frequent a word is in the collection.

The method proposed in VIENA evaluates the frequency of an specific soccer action but also how many users have executed that action. This measure allows to detect the soccer actions that are not executed by all the players and those that, in our case, are considered as relevant because can define the behavior of the team. Thus, the *TFIDF* weight is calculated as follows:

$$TFIDF_{A,T} = tf_{A,T} * \log \frac{players_T}{players_{T,A}}$$

where $TFIDF_{A,T}$ is the TFIDF measure value for the Action A in the team T. $tf_{A,T}$ determines how many times the action A has been executed by the team T. $players_T$ determines the number of players of the team T (in this case is always 11). $players_{T,A}$ determines the number of players of the team T which have executed the action A.

This measure considers as important those actions which have been executed by a reduced number of players. Therefore, the model of a team consists of those action with a high relevance, it means, with a high value for this measure (TFIDF).

3.3.2 Trie-ChiSquare

The actions performed by a soccer team are inherently sequential, and this sequentiality is considered in this modeling method proposed. This method is explained in detail and evaluated in [18, 12]. In this method, the temporal dependencies are very significant and it is considered that a current action depends on the actions that have happened before and it is related to the actions that will happen after. Taking this aspect into account, the input of this method is the sequence of all the soccer actions performed during the game. In this case, to get the most representative set of sequential actions (subsequences) from the acquired sequence, the use of a *trie* data structure [19] is proposed.

First, the sequence is segmented into several subsequences. This segmentation is divided by defining an appropriate length and obtaining every possible ordered subsequence of that specific length.

Then, the subsequences of actions are stored in a *trie*, such that all possible subsequences are accessible and explicitly represented. In a *trie*, every node represents an action, and the node's children represent the actions that have appeared following this action. Also, each node keeps track of the number of times an action has been inserted on to it. When a new subsequence is inserted into the *trie*, existing nodes of the *trie* are modified and/or new nodes are created. Moreover, as the dependencies of the actions are relevant in an agent behavior, the subsequence suffixes (subsequences that extend to the end of the given sequence) are also inserted.

Once the *trie* is created, it is traversed to calculate the relevance of each subsequence (path from the *root* node to any other node of the *trie*). For this purpose, a statistical dependency test [20] is used. Therefore, to evaluate the relation between a soccer action and its prefix (sequence of actions previous to an action), it is used one of the most popular statistics: Chi-square test [21]. This statistical test enables to compare observed and expected sequences objectively and evaluate whether a deviation appears. Hence, every action of the trie store a value that determines whether a soccer action is or not relevant with the previous ones in its behavior sequence.

To compute this test, it is necessary a 2x2 contingency table (also known as a cross-tabulation table). This table is filled with four frequency counts, as we can see in the Table

	Event	Different action	Total
Prefix	O_{11}	O_{12}	$O_{11} + O_{12}$
Different prefix	O_{21}	O_{22}	$O_{21} + O_{22}$
Total	$O_{11} + O_{21}$	$O_{12} + O_{22}$	$O_{11} + O_{12} + O_{21} + O_{22}$

Table 1: The contingency table

1. The counts are calculated as follows: The first number O_{11} indicates how many times the current node/action is following its prefix. The number O_{12} indicates how many times the same prefix is followed by a different action. The number O_{21} indicates how many times a different prefix of the same length, is followed by the same action. The number O_{22} indicates how many times a different prefix of the same size, is followed by a different action.

The expected values are calculated as in Equation 1.

$$Expected(E_{ij}) = \frac{(Row_i Total x Column_i Total)}{GrandTotal} \quad (1)$$

The formula to calculate chi-squared value, is giving in Equation 2.

$$X^2 = \sum_{i=1}^{r} \sum_{j=1}^{k} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \quad (2)$$

where: O_{ij} is the observed frequency and E_{ij} is the expected frequency.

This value is calculated for each action (node) of the trie. Hence, after labeling all the subsequences of the *trie*, the model of a soccer team behavior is represented by the distribution of its subsequences. Those subsequences with a higher value are more relevant in the team behavior model.

4 Conclusions and Future work

This paper presents a tool, called VIENA, for modeling a soccer team behavior from observation based on a previous work [12]. This tool can: (1) visualize a RoboCup Soccer Game using a dynamic data representation (2) detect soccer actions and (3) analyze of a soccer team behavior by using different algorithms creating a soccer team behavior model. The main contribution of this tool is that VIENA has implemented two different methods for creating soccer team behavior models. These methods create human-understandable models, which is really important to know how the opponent behave. In addition, the way the tool has been developed ables to add new methods for creating behavior models made by the users.

This tool is very useful because the adaptation and learning abilities are essential for any player in the RoboCup environment. For this reason, VIENA helps to create opponent models which can be considered for adapting the soccer team behavior to the opponent team (e.g. creating counter-strategies).

In this research, we have considered that the behavior of a team does not change over a game. However, in order to construct a soccer team model more realistic, it should be frequently revised to keep it up to date. The use of the created models for carrying out useful actions in the proposed environment is considered as a preliminary work in [22]. However, this aspect is essential and it is considered as the most relevant future work.

Acknowledgments. This work has been supported by the Spanish Ministry of Education and Science under project TRA-2007-67374-C02-02.

References

[1] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, H. Matsubara, Robocup: A challenge problem for ai and robotics., in: RoboCup, 1997, pp. 1–19.

[2] I. Noda, H. Matsubara, K. Hiraki, I. Frank, Soccer server: a tool for research on multiagent systems, in: Applied

AI, Vol. 12, Taylor and Francis, 1998, pp. 233–258.

[3] P. Carpenter, P. Riley, M. M. Veloso, G. A. Kaminka, Integration of advice in an action-selection architecture, in: RoboCup 2002: Robot Soccer World Cup VI, 2002, pp. 195–205.

[4] D. Carmel, S. Markovitch, Incorporating opponent models into adversary search, in: Proceedings of the National Conference on Artificial Intelligence, Portland, Oregon, 1996, pp. 120–125.

[5] M. Wünstel, D. Polani, T. Uthmann, J. Perl, Behavior classification with self-organizing maps, in: RoboCup 2000: Robot Soccer World Cup IV, Springer-Verlag, London, UK, 2001, pp. 108–118.

[6] M. Chen, E. Foroughi, F. Heintz, Z. Huang, S. Kapetanakis, K. Kostiadis, J. Kummeneje, I. Noda, O. Obst, P. Riley, T. Steffens, Y. Wang, X. Yin, Soccerserver manual ver. 7 (2002).

[7] P. Riley, M. Veloso, G. Kaminka, An empirical study of coaching, in: Distributed Autonomous Robotic Systems 5, Springer-Verlag, 2002, pp. 215–224.

[8] G. Kuhlmann, P. Stone, J. Lallinger, The champion UT Austin Villa 2003 simulator online coach team, in: RoboCup-2003: Robot Soccer World Cup VII, 2004.

[9] C. L. Dulalia, P. S. L. Go, P. V. C. Tan, M. Z. I. O. Uy, R. de Dios Bulos, Learning in coaching, in: Proceedings of the fourth IEEE International Workshop, WSTST-05, 2005, pp. 1338–1347.

[10] G. Kuhlmann, W. B. Knox, P. Stone, Know thine enemy: A champion robocup coach agent, in: Proceedings of the National Conference on Artificial Intelligence, 2006, pp. 1463–68.

[11] R. Fathzadeh, V. Mokhtari, M. R. Kangavari, Opponent provocation and behavior classification: A machine learning approach, in: RoboCup 2007: Robot Soccer World Cup XI, 2007, pp. 540–547.

[12] J. A. Iglesias, A. Ledezma, A. Sanchis, CAOS Coach 2006 Simulation Team: An opponent modelling approach, Computing and Informatics 28 (1) (2009) 57–80.

[13] B. Jung, M. Oesker, H. Hecht, Virtual RoboCup: Real-Time 3D Visualization of 2D Soccer Games, in: Proceedings of the International Workshop on RoboCup, IJCAI'99, 1999, pp. 121–126.

[14] T. Takahashi, Logmonitor: From player's action analysis to collaboration analysis and advice on formation, in: RoboCup-99: Robot Soccer World Cup III, London, UK, 2000, pp. 103–113.

[15] R. S. F. A. S. Eslam Nazemi, Amir-Reza Zareian, Team assistant - team description paper, in: RoboCup 2002: Robot Soccer World Cup VI, 2002.

[16] A. Bezek, Modeling multiagent games using action graphs, in: Proceedings of Modeling Other Agents from Observations, 2004.

[17] J. Rocchio, Relevance feedback in information retrieval, in: SMART retrieval system: Experiments in Automatic Document Process, 1971, pp. 313–323.

[18] J. A. Iglesias, A. Ledezma, A. Sanchis, G. A. Kaminka, Classifying efficiently the behavior of a soccer team, in: W. B. et al. (Ed.), Proceedings of Intelligent Autonomous Systemsb, 2008, pp. 316–323.

[19] E. Fredkin, Trie memory, Comm. A.C.M. 3 (9) (1960) 490–499.

[20] Z. Huang, Y. Yang, X. Chen, An approach to plan recognition and retrieval for multi-agent systems, in: Proceedings of AORC, 2003.

[21] C. L. Chiang, Statistical Methods of Analysis, World Scientific, 2003.

[22] J. A. Iglesias, J. A. Fernández, I. R. Villena, A. Ledezma, A. Sanchis, The winning advantage: Using opponent models in robot soccer, in: Proceedings of the IDEAL 2009, 2009.