FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

FEUP

Exploiting Opponent Behavior in Multi-agent Systems

João Paulo dos Santos Portela

Master in Informatics and Computing Engineering

Supervisor: Luís Paulo Reis (Ph.D.)

Second Supervisor: Pedro Abreu (M.Sc.)

2010, June

Exploiting Opponent Behavior in Multi-agent Systems

João Paulo dos Santos Portela

Master in Informatics and Computing Engineering

Approved in oral examination by the committee:

Chair: Doutor Armando Jorge Miranda de Sousa (Ph.D.)
Supervisor: Luís Paulo Reis (Ph.D.)
External Examiner: Pedro Miguel do Vale Moreira (Ph.D.)

31st July, 2010

Abstract

In simulated soccer, the level of performance depends on many interrelated subsystems, one of such systems is the analysis and subsequent exploitation of the opposing teams' behaviour. The base code for most simulated soccer teams only partially changes from one competition to another, thus the previous competitions logs and binaries (the historical data) can be used for such analysis. Research in this field is still largely unexplored, not having proved itself in competitions like *RoboCup Soccer Simulation 2D league* yet. In this thesis a three step approach for such system is proposed.

The first component is responsible for generating a human-readable opponent model based on detected events (match statistics), such events are defined by an expert panel and their detection is manually validated, this tool is also used to analyse the matches of the 2009 RoboCup simulation league competition. The manual validation indicates that the tool is capable of successfully detecting the desired events with minimal flaws.

The second component focus on classifying an opponent based on the previously described model, to achieve those results, the opponents are clustered according to their characteristics using the K-means clustering algorithm. Then, based on the previously described model three classifiers are trained to identify to which cluster corresponds each model. The classifiers all obtain a classification rate superior to 80%–SVM (96.4%), Random Forest (93.59%) and Bagging (80.151%). The Friedman rank test rejected the *null* hypothesis of equivalence between the three classifiers (Bagging, Random Forest and SVM) and the Nemenyi test showed that SVM and Random Forest are better than Bagging but cannot show which is better between SVM and Random Forest. In the light of these results and considering that SVM is the fastest of the three to train, it is considered the best.

The final component encompasses all of the previous components and is responsible for advising its team in a real-time scenario. To be able to do this the best tactic to face each opponent cluster is identified before the match, using that information, the SVM classifier and an online generated opponent model, it is able to identify the best tactic for each of the opponent play styles. The results are obtained by comparing the results of the FC Portugal team without and with this module at varying advise intervals. The *null* hypothesis is rejected by the Friedman rank test, but the Bonferroni-Dunn test does not allow to conclude if the changes made result in better or worse behaviour than the control group.

In the future the obtained results can be further studied. The project can be extended by varying more tactic parameters or even applied it to another sport or environment (real instead of simulated).

Resumo

No futebol simulado, o nível de performance de uma equipa depende de varios subsistemas interligados, um desses sistemas é a analise e subsequente exploração do comportamento da equipa oposta. O código base da maioria das equipas de futebol simulado apenas mudar parcialmente de competição para competição, portanto os logs e os binários (dados históricos) das competições anteriores podem ser utilizados para essa análise. A pesquisa neste campo ainda foi pouco desenvolvida, não tendo ainda mostrado provas em competições como a *RoboCup Soccer Simulation 2D league*. Nesta tese uma aproximação em três fases para este problema é apresentada.

O primeiro componente é responsável por gerar um modelo do oponente num formato legível baseado nos eventos detectados (estatísticas do jogo), tais eventos são definidos por um painel de especialistas e a sua detecção é validada manualmente, esta ferramenta é utilizada para analisar os jogos da liga de simulação da RoboCup 2009. A validação manual indica que a ferramenta é capaz de detectar os eventos desejados com falhas minimas.

O segundo componente foca-se em classificar um oponente baseado no modelo previamente descrito, para alcançar esses resultados os oponentes são agrupados de acordo com as suas características utilizando o algoritmo de *clustering K-means*. Posteriormente, baseado no modelo anteriormente definido, três classificadores são treinados para identificar a que *cluster* corresponde cada modelo. Todos os classificadores obteram uma taxa de classificação superior a 80%–SVM (96.4%), Random Forest (93.59) e Bagging (80.151%). O teste de *Friedman* rejeita a hipótese nula de equivalência entre os três classificadores e o teste de Nemenyi mostra que o SVM e o Random Forest são melhores que o Bagging mas não permite concluir qual o melhor destes. Tendo em consideração estes resultados e considerando que o SVM é o mais rapido dos três a treinar, é considerado o melhor.

O componente final engloba todos os componentes prévios e é responsável por dar conselhos a sua equipa numa situação de tempo real. Para permitir isto a melhor táctica para enfrentar cada grupo (*cluster*) de oponentes é identificada antes do jogo, utilizando essa informação, o classificador *SVM* e um modelo do oponente gerado durante o jogo, é capaz de identificar a melhor táctica para cada estilo de jogo dos oponentes. Os resultados obtidos comparando os resultados da FC Portugal sem e com este modulo utilizando vários intervalos de conselho. A hipótese nula de equivalência é rejeitada pelo teste de *Friedman*, mas o teste de *Bonferroni-Dunn* não permite concluir se as mudanças resultaram em melhor ou pior comportamento do que o grupo de controlo.

No futuro os resultados obtidos poderão ser melhor estudados. O projecto pode ser estendido variando mais parâmetros tácticos ou mesmo aplicado a outro desporto ou ambiente (real em vez de simulado).

Acknowledgments

The author would like to thank both Professor Luís Paulo Reis and Researcher Pedro Abreu for the support and overall guidelines.

I would also like to thank my parents and sister for unconditional support and encouragement to pursue my interests.

A very special thanks to my girlfriend for her patience, support, encouragement and help.

To all of the unnamed people on the internet that post useful information on their blogs, forums, or any other available source, I say: “Thank you! You saved me precious time.”

Last, but not least, I say thank you to all my friends that helped me through the course of my academical life, whether they offered me advice or just a healthy discussion.

Thank you all! :)

João Paulo dos Santos Portela

"The only best practice you should be using all the time is ***Use Your Brain****."*

Steven Robbins

Contents

Abstract i

Resumo iii

Acknowledgments v

1 Introduction 1
- 1.1 Scope . . . 1
- 1.2 Research Question . . . 1
- 1.3 Main goals . . . 2
- 1.4 Motivation . . . 2
- 1.5 Document Structure . . . 3

2 Context 5
- 2.1 Agents . . . 5
- 2.2 Multi-agent Systems . . . 6
  - 2.2.1 MAS Coordination . . . 6
- 2.3 Robotic Soccer . . . 7
- 2.4 Simulated Robotic Soccer . . . 7
  - 2.4.1 RoboCup soccer simulator 2D . . . 7
  - 2.4.2 Standard Coaching languages . . . 12
- 2.5 RoboCup . . . 14
  - 2.5.1 RoboCup Soccer Simulation 2D league . . . 15
  - 2.5.2 RoboCup Coach competition . . . 16
- 2.6 Conclusions . . . 19

3 Related Work 21
- 3.1 Statistics Generation Software . . . 21
  - 3.1.1 Professional Soccer Performance Analysis tools . . . 21
  - 3.1.2 Research . . . 22
- 3.2 Historical Data Exploiting . . . 23
  - 3.2.1 Team Behaviour Modeling . . . 23
  - 3.2.2 Coaching in Adversarial Domains . . . 24
- 3.3 Conclusions . . . 25

4 FC Portugal 2D 27
- 4.1 Architecture and Knowledge Structures . . . 27
- 4.2 High-level decision and Cooperation . . . 28
- 4.3 Strategical coordination layer . . . 28

4.4 Formations . . . 29
4.5 Setplays . . . 29
4.6 Configurable Strategy Parameters . . . 31
4.6.1 Formation . . . 31
4.6.2 Setplays . . . 32
4.7 Conclusions . . . 35

5 Statistics Generation and Automatic Simulation **37**
5.1 Statistics Selection and Generation . . . 37
5.1.1 Detected events . . . 37
5.1.2 SoccerScope2 . . . 40
5.1.3 Implementation details . . . 40
5.1.4 Pass and Ball Possession . . . 41
5.2 Matches Samples Generation . . . 43
5.2.1 Example of usage . . . 44
5.2.2 Entities description . . . 44
5.3 Conclusions . . . 45

6 Assistant Coach and Opponent Analysis **47**
6.1 Opponents Classification . . . 47
6.1.1 Statistics . . . 47
6.1.2 Generated Matches . . . 48
6.1.3 Clusters Determination . . . 48
6.1.4 Classification Algorithms . . . 48
6.1.5 Training and Validation . . . 49
6.2 Assistant Coach . . . 49
6.2.1 Best Tactic Identification . . . 49
6.2.2 Advise . . . 50
6.3 Conclusions . . . 50
6.3.1 Match Preparation Overview . . . 50

7 Results **53**
7.1 Statistics Generation . . . 53
7.1.1 2009 RoboCup Matches Analysis . . . 53
7.1.2 Validation . . . 56
7.2 Opponents Classification . . . 58
7.2.1 MARS . . . 58
7.2.2 Clusters Determination . . . 60
7.2.3 Classification Algorithms . . . 60
7.3 Assistant Coach . . . 62
7.4 Conclusions . . . 63

8 Conclusions and Future work **65**
8.1 Statistics Generation . . . 65
8.1.1 2009 Matches Analysis . . . 65
8.1.2 Validation . . . 66
8.2 Matches generation framework . . . 66
8.3 Opponents Classification . . . 66
8.4 Assistant Coach . . . 67

8.5 Future Work 68

References **69**

List of Figures

1.1 Basic operating mechanism. The offline mode feeds information to the online mode. 2

2.1 An agent interacting with its environment. The agent takes input from its sensors and produces output from its actuators. . . . 6
2.2 Diagram of the usual communication sequence (simplified) . . . 9
2.3 Diagram illustrating the consequences of slow performance. . . . 9
2.4 Soccer monitor . . . 10
2.5 RoboCup competition leagues overview. . . . 14
2.6 Quarter finals with double elimination . . . 17

4.1 FC Portugal strategy model. [RLM10] . . . 29
4.2 Defining a formation using matchflow. . . . 30
4.3 Setplays structure [MR07b]. . . . 31
4.4 Formation 433 at the start of a match (left team). . . . 32
4.5 Formation 442 at the start of a match (right team). . . . 32
4.6 Kick off setplays. . . . 33
4.7 Free kick setplays. . . . 33
4.8 Goal-kick setplays. . . . 34

5.1 Soccer field wings and quarters partition . . . 38
5.2 Triangle area in red (the black dot represents the player) . . . 39
5.3 Game structure . . . 40
5.4 Matches generation framework object model, overview. . . . 44

6.1 Data flow through the existing modules. . . . 51

7.1 Zone Dominance for the top three tournament finalists. . . . 54
7.2 Successful to missed passes relation. . . . 55
7.3 Average wing variation per game. . . . 56
7.4 Temporal Sequence of the top four Teams. . . . 57
7.5 Goals and goal opportunities, per game (average). . . . 58
7.6 Goal to opportunity ratio. . . . 59
7.7 Calculate the optimal number for K using the Gap algorithm . . . 60
7.8 The number of clusters with groups of sum of squares . . . 60
7.9 Matches distribution by cluster for each algorithm. . . . 61

LIST OF FIGURES

List of Tables

3.1 Professional Performance Tools Comparison (adapted from [Abr10]) 22
3.2 Research Software Comparison (adapted from [Abr10]) 23

7.1 The number of clusters with groups of sum of squares 61
7.2 Ranks for the Friedman test of the opponents classification. 62
7.3 Ranks of the online results. 63

Abbreviations and Symbols

SSIL	Simulated Soccer Internet League
AI	Artificial Intelligence
MAS	Multi-Agent Systems
FIFA	Fédération Internationale de Football Association (International Federation of Association Football)
LIACC	Laboratório de Inteligência Artificial e Ciência de Computadores (Artificial Intelligence and Computer Science Laboratory)
IEETA	Instituto de Engenharia Electrónica e Telemática de Aveiro (Institute of Electronics and Telematics Engineering of Aveiro)
SBSP	Situation Based Strategic Positioning
DPRE	Dynamic Positioning and Role Exchange
SVM	Support vector machines
MARS	Multivariate adaptive regression splines
UDP	User Datagram Protocol
IP	Internet Protocol

Chapter 1

Introduction

1.1 Scope

In the context of Multi-agent systems in adversarial environments many techniques can (and are) employed in order to improve its performance. Those improvements range from enhancing each agent capabilities as an individual [RMM+01] to using advanced high level coordination techniques [RLO01, SV99].

RoboCup is an international research and education initiative whose goal is to foster artificial intelligence and robotics research [Rob] and the *de facto* standard in MAS systems AI research, which makes it the ideal research platform for the developed work.

1.2 Research Question

This research question as with many other studies in this field started just as **"How to improve a simulated robotic soccer team performance?"**. But this question is too broad since the simulated robotic soccer is a such a multidisciplinary field and a team performance can be improved in many ways. These ways range from low-level individual behaviours [YNN+05, RHLL08] to high level coordination techniques [CW01, MR07a]. Before we advance any further two relevant observations must be brought to light.

- Although most teams change (improve) their teams from one competition to another the problem at hand is so large and complex that these changes are usually small compared to the complete system that is the team.
- Most competitions publicly provide game logs of each of the games played, as well as the binaries for the participating teams. It is the case of SSIL and RoboCup among others.

Having this in mind, an approach using the opponent binaries to optimize our own team strategy is proposed.

The proposed tool contains several distinct components, each with its own research merit. The first is a statistics calculation module which identifies game events and gives their statistics.

Figure 1.1: Basic operating mechanism. The offline mode feeds information to the online mode.

Next a module capable of generating the opponent team game logs from its binary (this module can be surpassed if enough game logs already exist). After that, the game logs are classified by the classification module. The previously mentioned modules allow for the online component (assistant coach) to choose the best strategy at the start and during the match. These components are throughly described in chapter 5 and 6.

1.3 Main goals

The main goal of this project is to obtain a tool capable of using an opponent team binaries to generate historical data (game logs) which can be analyzed and categorized in order to improve its own team performance in a simulated robotic soccer match against such a team.

To attain such a goal the need to both specify a strategy at the start of the game and making the necessary adjustments to the team tactics during the game must not be overlooked. With that in mind it is at least required that the system supports both online (playing) and offline (training) operating modes, having the offline mode generating information to the online mode (Figure 1.1).

The proposed approach is to use simple statistics for determining the opponent behaviour instead of the more complex modeling techniques described in chapter 3. This approach is expected to be sufficient to attain proposed results and while greatly simplifying the problem. The use of statistics also has the advantage of being in a more human-understandable format.

1.4 Motivation

Although there is some research done in the same area this thesis focusses on (as documented in section 3), approaches that use historical data to propose high level strategy are still scarce. Being most of the research focused on either statistics generation without the capability of making adequate strategic suggestions (section 3.1), low-level optimization only (section 3.2) or just opponent modeling (section 3.2.2).

The motivation of this work is to dispel such gap in the existing research, as well as complementing the research in all the areas it transverses: game events to detect are defined; statistics generation techniques are explored and validated; classification methods are evaluated; advice giving in adversarial domains analyzed. Giving true insight on methodologies that can provide performance improvements in the context of the *RoboCup Soccer Simulation 2D league*, by utilizing the opposing teams available data.

1.5 Document Structure

Besides this introduction the thesis contains 6 more chapters. In chapter 2 some of the necessary context necessary to understand this thesis is presented, there concepts such as Agents and Multi-agent systems are presented, robotic soccer as a standard problem for AI research is also laid out as well as its simulated counterpart. Finally RoboCup is exhibited as being one of the most prominent robotic soccer competitions. In chapter 3 other work developed in this area is analysed and reviewed, ranging from statistics generation to historical data analysis. In chapter 4 the object of study–FC Portugal–is presented and described. Chapter 5 exposes the work in statistics generation and matches samples generation. Classification methods employed and the subsequent use in the assistant coach are addressed in chapter 6. Chapter 7 contains the obtained experimental results for each component. Finally, in chapter 8 some conclusions are presented and the future work is discussed.

Chapter 2

Context

In this section many concepts related with the object of study are presented and briefly explained. The concept of intelligent agents is explained as well as Multi-agent Systems. Robotic Soccer is presented as a replacement for chess as the standard problem for AI research and RoboCup as its flagship. Since this thesis will focus mostly on Simulated Robotic Soccer it will be more throughly presented than the other types of robotic soccer.

2.1 Agents

An agent, is defined as an autonomous entity, contained in a given environment in which it has perceptions through sensors, and acts autonomously through its actuators (figure 2.1) performing the task for which it was designed [RN03, Wei99, Woo01]. It is important to note that the notion of agent is just a tool to aid in system analysis and comprehension, not an absolute characterization that divides the world into agents and non-agents [RN03].

An intelligent agent is a computer system that is capable of autonomous action in order to meet its design objectives [JW98]. This autonomous action should be:

Responsive - Agents should perceive their environment and act in a timely fashion.

Proactive - Agents should not only act in response to their environment, but also take initiative when appropriate.

As an intelligent agent, for each perception sequence it should do whatever action is expected to maximize its performance measure, based on the evidence provided by its perceptions and built-in knowledge. The performance measure is supposed to be objective and defined by some authority.

Through this thesis when the term “agent” is used, it should be considered as an abbreviation of “intelligent agent”.

Figure 2.1: An agent interacting with its environment. The agent takes input from its sensors and produces output from its actuators.

2.2 Multi-agent Systems

In many areas of application, such as simulated soccer, the agents are grouped into Multi-Agent Systems (MAS), where each autonomous entity must interact with the other agents in the system in order to achieve its own goals (it is common that in cooperative environments different agents share some common goals). The agents interaction defines *social* interaction protocols, which require coordination, cooperation and negotiation [Rei03, Woo01].

2.2.1 MAS Coordination

MAS coordination can be seen as the problem of managing inter-dependencies between the activities of the agents [Woo01]. Being simulated robotic soccer such a complex, dynamic and partially adverse domain the applied coordination methodologies must reflect these domain characteristics. Some of these methodologies are:

- Strategical coordination.
- Partially hierarchic coordination.
- Coordination through communication.
- Coordination through Intelligent perception.
- Coordination by mutual modelling.

2.3 Robotic Soccer

Since robotic soccer—particularly in the form of RoboCup competition—has been proposed as a replacement for chess as a standard problem for Artificial Intelligence (AI) research [KAK+95], robotic soccer has become largely more popular.

Robotic soccer comprises many complex and interrelated problems/challenges for artificial-intelligence and robotics, such as design principles of autonomous agents, multi-agent collaboration, strategy acquisition, real-time reasoning and sensor fusion. In this thesis the focus is on simulated robotic soccer. The RoboCup soccer simulator was the chosen platform.

In the next two sections soccer simulation (section 2.4) and the RoboCup competition (section 2.5) are described in more detail.

2.4 Simulated Robotic Soccer

Simulated Robotic Soccer is a particular case of robotic soccer where the environment is purely virtual. The advantages of a purely virtual environment are: allowing the abstraction of implementation specific details allowing to focus on a specific research subject (a common example is abstracting the lower level characteristics like joint control or image processing to focus on coordination); test new approaches before applying them to the real environment; easier to do hundreds of automated runs; among others.

In simulated robotic soccer—like in human soccer—each team is constituted of eleven players (virtual agents) and a coach (also a virtual agent) with different individual goals but with a common collective goal: to score at least one more goal than the opponent, respecting the established rules.

The simulated robotic soccer is a very compelling research field because it provides a standard problem in which several methodologies, algorithms, architectures and theories can be evaluated and compared with each other [Rei03]. Simulated robotic soccer also raises interest among researchers due to the popularity of human soccer throughout the world [Dun99].

The RoboCup soccer simulator 2D is the most popular platform for simulated robotic soccer.

2.4.1 RoboCup soccer simulator 2D

The RoboCup soccer simulator 2D is a client-server platform capable of simulating a soccer game between distributed agents of two competing teams. It is the official simulation tool for the RoboCup soccer simulation 2D league (described in section 2.5.1).

The soccer simulator has the following characteristics: a fully distributed multi-agent domain with both teammates and opponents; hidden state, which means that the agents do not have entire world view at any moment; noisy sensors and actuators, which give them imprecise perception of the environment and inaccurate acting capabilities; limited communication opportunities, forcing the agents to decide before communicating; dynamic, making the agents take into consideration that the environment is changing while they determine their next action; continuous sensor and actuator data, making the number of possible world states virtually infinite;

The RoboCup soccer simulator is composed of three main modules: the server (*soccerserver*), the viewer (*soccermonitor*) and the log player (*logplayer*) [CFH+]. The *soccerserver* the most important of the three, because it is responsible for implementing the game logic. The *soccer monitor* is used to view the game while it is being played by the agents. The *logplayer* is used to review games that have been recorded to a log file. This application appeared in 1998 and since then it has been used in education [SPS04] intelligent robotic research, and international competitions, like RoboCup [KAK+95, Kit97].

As previously mentioned, each team is composed of 12 agents. Each of these agents (24 total) connects to the *soccerserver* to play a virtual soccer game. There are three main agent roles whose actions and perceptions are significantly different and distinguished by the simulator:

Coach

The coach has full view of the field and is able to communicate with its players but has a limited number of communication opportunities and the messages have delay.

Player

The player is modeled by the server to be similar to a human soccer player. It has limited vision of the field. A energy model, that accounts for its stamina, recovery capabilities and exerted effort. And acting capabilities like running, turning, dashing and kicking.

Goalie Player

The goalie player is similar to the regular player but can also catch the ball, when inside the penalty area.

The connection is done using `UDP/IP` sockets. The messages are in plain `ASCII` and follow a format that which allows for easier understanding and debugging by humans, as well as structured parsing and easy processing by computers (the messages format is described in section 2.4.1.4).

2.4.1.1 Soccer Server

The soccerserver is the component that implements the game rules and mechanics enabling two teams to play a game of simulated soccer. The server was built according to a client-server architecture. This means that there are no restrictions on how the teams are built, the only requirement is that it supports `UDP/IP` protocol for connecting/communicating with the server. This is due to the fact that all communication between the server and its clients is done using `UDP/IP` sockets.

Each client corresponds to an agent and is a separate process that connects to the server through a specified port. The agents send requests to the server regarding the actions they want to perform (e.g. kick the ball, turn, run, etc.). The server receives those messages handles the requests and updates the environment accordingly. Besides that, each cycle, the server sends the players their sensory information (visual information about the position of the ball, the position of the other players, etc.). As already implied the server is a real-time system working in discrete intervals (called cycles). Each simulation cycle has a specified duration (usually around $\frac{1}{10}$ of a second). The typical communication sequence is illustrated in figure 2.2. The cycle duration means that

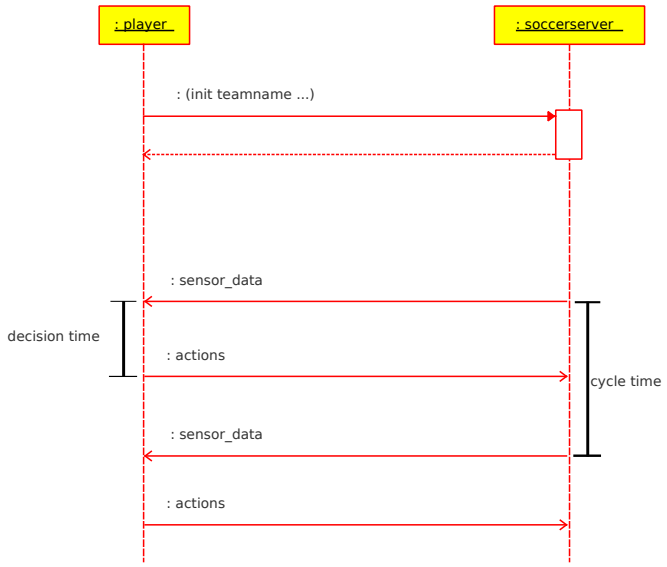


Figure 2.2: Diagram of the usual communication sequence (simplified)

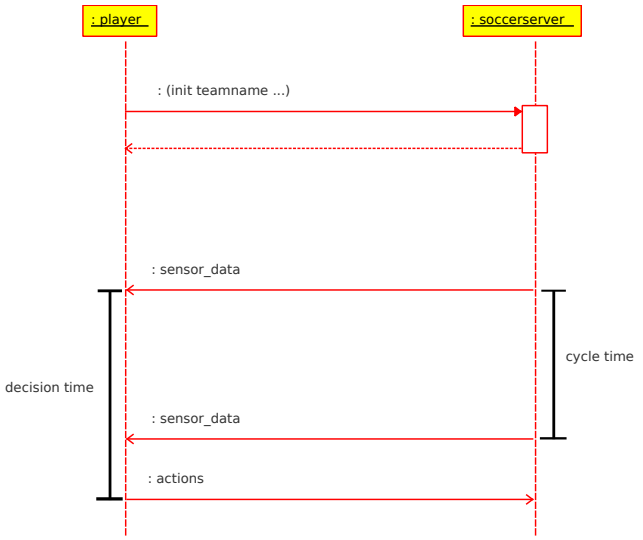


Figure 2.3: Diagram illustrating the consequences of slow performance.

slow performance results in missed acting opportunities which consequently has a negative impact on the team (see figure 2.3).

2.4.1.2 Soccer Monitor

The monitor is a visual tool that allows humans to easily know what is happening in the game (figure 2.4). The information shown includes the score, team names, and the position of the ball and the players, for each game cycle. Other information can be turned on and off like the players vision cone and the players stamina. The monitor also includes a simple interface to control the

Figure 2.4: Soccer monitor

server (like the kick-off button to start the game). The monitor is not required for running a soccer game in the server.

2.4.1.3 Log Player

The log player is used to replay matches from the log files generated with the *soccerserver*. Some options can be set in the server to enable/disable it from storing all the data for a given match. The log files are the files stored by the server and contain information about the server configuration, detailed information about the state of the movable objects in each cycle (like the position, the stamina in the case of the players, etc.), in sum everything necessary to accurately replay the match.

The log player opens a windows similar to the viewer but that can forward and rewind the game. The log player is very useful to review previous games.

2.4.1.4 Protocol Overview

For a better understanding of how the *soccerserver* works a brief overview of protocol format is presented in this section. The messages format is *s-expression* (similar to the LISP programming language). The messages roughly follow this format: **(msgtype** *Value***)** ; where **msgtype** is the type of the message (indicated by the small case) and *Value* is a variable value of that type of message (indicated by the upper case of the first letter). Some examples of messages are:

(dash *Power Direction***)**

Issues the **dash** command to the server. *Power* and *Direction* are arguments that define the power and the direction of the player dash respectively.

(kick *Power Direction***)**

Issues the **kick** command, making the player kick the ball with *Power* power in *Direction* direction.

(say *"Message"***)**

Sends *"Message"* for the other players to hear.

(turn *Moment***)**

Turns the player *Moment* degrees in relation to the direction it was facing.

As previously mentioned each cycle the player receives sensory information from the server. Examples of these perceptions are presented next:

(hear *Time Sender "Message"***)**

Is the message received by the player when a *Sender* communicates. *Time* is the simulation cycle when the message was received. *Sender* is the message sender(**online_coach_left**; **online_coach_right**; **coach**; **self**). *"Message"* is the payload.

(hear *Time Online_Coach Coach_Language_Message***)**

Hear a message from the *Online_Coach* formated in the coaching language format, contained in *Coach_Language_Message*. More about coaching languages in section 2.4.2.

(hear *Time Direction* **our** *UniformNumber "Message"***)**

Hear a message from teammate *UniformNumber* that is in the direction *Direction*, with *"Message"* being the payload.

(hear *Time Direction* **opp** *"Message"***)**

Hear *"Message"* from an opponent that is in the direction *Direction*.

(see *Time ObjInfo*$^+$**)**

The **see** message can be one of the most complex ones since it is composed of many *ObjInfo s-expressions*, one for each object —objects are the goal poles, the players, flags, lines, and the ball—the player can see. The *Time* is the simulation cycle as usual.

ObjInfo ::= **(** *ObjName Distance Direction DistChange DirChange BodyFacingDir HeadFacingDir***)**

ObjInfo contains every possible information about an object. *ObjName* identifies the type of object, can be the player, the ball, the goal poles, the field flags and the field lines. *Distance* is the distance that of the object is from the player. *Direction* is the relative direction of the object from the player. *DistChange* and *DirChange* are the linear and angular velocity of the object, when applicable. *BodyFacingDir*, *HeadFacingDir* only apply when the object seen is a player.

Besides the types of messages previously mentioned (actions and perceptions) there are other types of messages that serve mostly error handling and configuration purposes that are unimportant in the context of this thesis. For a more complete description please refer to [CFH$^+$] (a complete documentation but not up to date) and [Aki] (a more up to date reference but incomplete).

2.4.2 Standard Coaching languages

Coaching languages were built to allow for a standardized coach-player communication with concern in tactics, formations and play type.

Since the online coach receives noise-free information about the movable objects on the field (the players and the ball position) it has the potential of micro-managing every player on its team. Because of that some restrictions to coach-players online communications were imposed: the messages must follow a specific format, the number of messages is limited and the messages are delayed before delivering. Online communication is when during a game the mode is **`play_on`**. When the game mode is not **`play_on`** the number of messages is less restricted, there is no message delay and can be in any string format (with a slightly restricted char set).

Through the rest of this section some most prominent coaching languages are described.

2.4.2.1 CLang

CLang is the standard message type of the soccer simulator 2D, and the only type of message allowed for online communication (a complete language description is done by Chen et al. [CFH+]-(section 7.7)). This language was built to allow coaches from different research groups to work together. As a consequence the main focus was on clear semantics, to prevent misinterpretation from both the players and the coach.

There are five message types:

Info messages are used to inform the players of something that the coach believes is relevant. The messages can contain information about the opponent or even its own team.

Advice messages are used to tell the players what the coach believes they should do in the situations described in the message.

Define messages define shorthand names for regions, directives, conditions, and actions. These can be used in the other messages, allowing the coach to refer to common patterns that appear in the messages, simplifying the communication and also possibly making the messages more understandable by humans.

Meta messages are used to carry meta-level information, and are used mostly for debugging and upward-compatibility.

Freeform allow arbitrary short strings, when the game mode is not **`play_on`**.

When it first appeared in 2001 *CLang* received some criticism due to the lack of high-level concepts like tactics, formations, player types, time periods or situations, “essential to define the behaviour of a soccer team”. Nowadays some of these concerns have not yet been addressed and not allowing the use of alternative coaching languages during online communication is limiting.

2.4.2.2 COACH UNILANG

Even though *CLang* is the only online communication language allowed in official RoboCup competitions there are other standard languages. One of such languages is *COACH UNILANG* described in [RL02], which is the predecessor of CLang. This language allows the use of high-level as well as low-level concepts for coaching a team. Another strong point is its flexibility which allows the definition of new high-level concepts. The messages, much like CLang, are formated in *s-expressions*.

As mentioned *COACH UNILANG* allows coaching at different abstraction levels:

1. **Instructions** are intended for use with players that are used to play together. Enables high-level coaching but is not very flexible.

2. **Statistics + Opponent Modeling**. This level is intended to assist a coach in training a team of players. Its semantics are closely related to the needs of communication between an assistant coach and the main coach. This semantics favor logical separation between the assistant coach and the main coach, and although no competition allows for more than one coach agent these can be implemented as a single agent.

3. **Definitions + Instructions** is the most flexible coaching level. It allows the coach to (re)define soccer the standard soccer concepts at a lower level. Consequently allowing for both high-level and low-level coaching.

For easier understanding and a more adequate usage *COACH UNILANG* is based on several human soccer concepts, these include:

- **Field Regions** correspond to areas inside the field.

- **Time periods** correspond to time intervals in the game.

- **Tactics** enable high-level configuration of team behaviour by defining mentality, game pace, pressure placed on the opponents, formations used for different situations, positioning exchange, etc..

- **Formations** are the spacial distribution of the players in the field.

- **Situations** describe game states.

- **Player Types** define players behaviors while in the possession of the ball and without the ball.

This multi-level approach, flexibility and completeness makes *COACH UNILANG* a very useful language that has even already been successfully used in the context of the Soccer Simulation 2D league by FC Portugal. At the time FC Portugal used this language free-form messages were still allowed during online games.

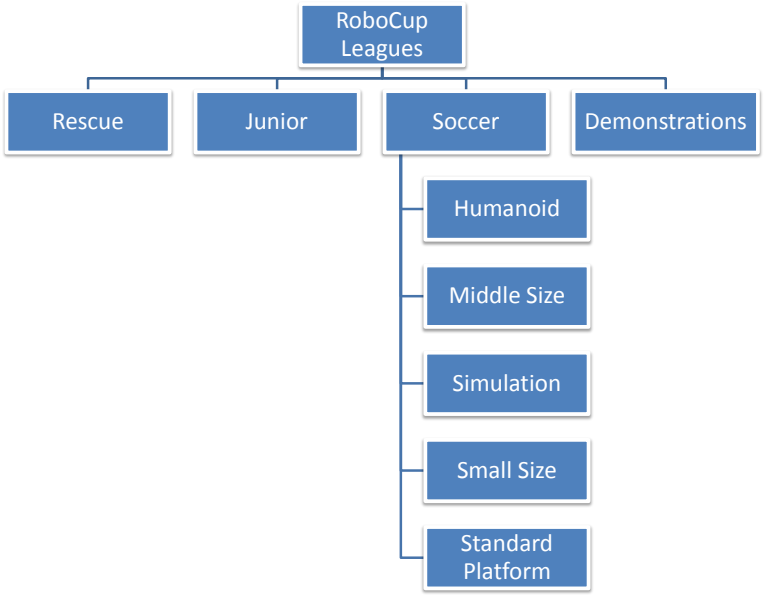


Figure 2.5: RoboCup competition leagues overview.

The change in the rules since this language was created, difficult its usage nowadays. Even so, it can be used as a complement for *CLang*, by using it for configuring the team prior to the games and when free-form messages are allowed. The rest of the game the coach has no choice but to use *CLang*.

Summarizing, even though *COACH UNILANG* was defined back in 2001 its concepts still seem more adequate for usage in the Soccer Simulation 2D competition than *CLang*. This is because it encompasses concepts expected to be used by the most advanced teams like mentality, positioning exchange and game pace.

2.5 RoboCup

Generally speaking RoboCup is an international research and education initiative whose goal is to foster artificial intelligence and robotics research [Rob].

This is attained by providing a standard problem where a wide range of methodologies and technologies related to the field can be applied. Soccer is an attractive field for this kind of development because of its previously mentioned characteristics and overall popularity.

RoboCup is composed of many different leagues which emphasize different topics. The existence of different leagues is justified by the need to parallelly research different fields. For instance the focus of the simulation league is MAS coordination while the small size league focus is on the quick and precise robots control [Rei03, Sin]. An overview of the RoboCup leagues can be seen in figure 2.5.

The focus of this thesis will be on the RoboCup Soccer Simulation League, which is described next. This thesis is also about historical data exploiting, some of the work done in the context of historical data exploiting was done for the Coach competition, for that reason this league will also be described. Both leagues use the RoboCup soccer simulator 2D (described in section 2.4.1) as their simulation tool.

2.5.1 RoboCup Soccer Simulation 2D league

The RoboCup Soccer Simulation 2D league is one of the most well known leagues. This league uses the RoboCup soccer simulator 2D has the official simulation tool [KAK+95, Kit97]. The rules of this league are similar to those of human soccer: two teams compete against each other in a simulated environment with the objective of scoring at least one more goal than the opponent. The soccer rules are enforced by the RoboCup soccer simulator 2D [CFH+]. For those reasons, it was chosen as a test bed for the developed work.

The main focus of this league is the development of coordination strategies, learning and multi-agent planning [Rei03, CFH+]. This league started in 1998 and since then has been used as a proving ground in several robotic soccer research work.

Each team is composed of twelve agents (eleven players plus one coach). Each agent is connected to the simulator using `UDP/IP` sockets sending commands (acting) and receiving the world state (sensing) using `ASCII` strings in a specific format (described in more detail in section 2.4.1). The rules are imposed by the soccer simulator and a human referee. These rules are similar to those used in professional human soccer and defined by FIFA (International Federation of Association Football). Each game has the duration of 6000 cycles (10 minutes), each cycle lasts $\frac{1}{10}$ of a second. The simulation is responsible for detecting foul situations and to apply the corresponding sanctions.

However, since some situations are too hard to be detected by the simulator, official league games also have a human referee, who awards a free kick to the disadvantaged team. Reasons for the human referee to call foul are:

- One team surrounds the ball so that the other cannot kick.
- The goal is blocked by so many players that it cannot go in (like a wall of players).
- If a team intentionally blocks the movement of the opponents.
- Any unforeseen situation that violates fair play may also be called a foul. These situations must be consulted with the rule committee.

Since the tournament rules try to mimic professional soccer rules, the competition takes the format of a tournament [Com07].

2.5.1.1 General Tournament Rules

The tournament rules are defined [Com07]. The competition usually consists of 16 teams. The setup of the tournament is as follows: A team is awarded three points for a victory and one point for a draw. A forfeit will record a result of 3:0 against the forfeiting team or the score of the forfeiting team's other game in that round with the largest differential, if larger than 3.

The competition is divided into three rounds:

- The first round consists of 4 teams of 4. The first teams are defined according to the last years results and SSIL, these teams are distributed each into one group. The rest of the teams are assigned randomly.
- In the second round the teams are distributed into 2 groups of 8, where each team plays against all other teams of the same group. The assignment to a group is based on the results of the first round.
- The first four teams proceed to the quarter finals. The quarter finals are in the double elimination final (explaining double-elimination is off the scope of this thesis but it roughly means that you are in the tournament until you lose twice, see figure 2.6).

In double-elimination each game must have a winner. If necessary 3000+3000 cycles of extra time will be played according to the golden goal rule. If there is still no result the game is decided by penalty shootouts.

2.5.2 RoboCup Coach competition

The RoboCup coach competition also uses the soccer simulator has its official simulation tool. This league was introduced in 2001 to promote the research of team and opponent modeling as well as team advise. Instead of creating a full team of agents the entrants are create a single coach agent that as omniscient view of the field but that the only possible action is to give verbal advice to its team [KKS06].

The coach has three main advantages over a normal player. First, it has noise-free full view of the playing field at all times. Second, since the players are autonomous, the coach is not expected to act in every single simulation cycle, allowing it to allocate more resources to high-level deliberations. Third, in the competition the coach has access to the logs of past games [KKS06].

On the other hand the coach also has some limitations in order to force its role as an adviser instead of a centralized controller. There is a limit to how often the coach can communicate with its team players, forcing it to deliberate the utility of its communication prior to communicating. To further avoid for centralized control by the coach the messages have delay. Also, the coach is supposed to give advice to players that are developed independently, possibly by other researchers. To make this communication possible a standard coach language must be used (CLANG) [CFH+, Com06].

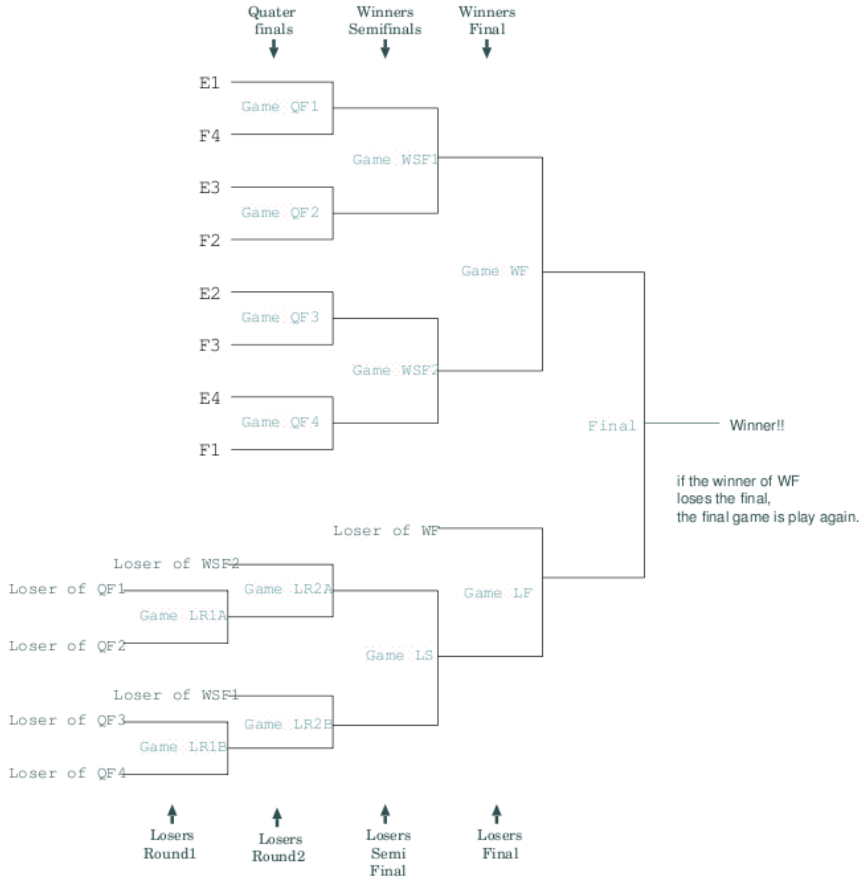


Figure 2.6: Quarter finals with double elimination

Since it appeared the competition suffered several changes, one of such changes radically changed the format of the competition. The competition format before those changes will be referred as "First format" while the format after the changes will be referred as "Second format".

2.5.2.1 First format

In its first format, the coach competition provided a 24 hours time window to analyse the officially released opponent log files (recordings of the past games). As a result, coach developers were able to manually modify their coaches (without opportunities to actually practice against the opponent) to help the *coachable* players defeat their opponents. Although the competition main focus was supposed to be on automated opponent modeling and consequent use of such information to attain play advantages, most entrants created their coaches by hand. Two notable exceptions to this rule are [RVK02, KSL05].

For the first four years of the competition the coaches were evaluated on their performance according to their coached team results. Although the intention of this competition was to nurture the research on opponent modeling in order to find a team-specific strategy, many coaches

performed well by using highly tuned general purpose strategies [KKS06].

2.5.2.2 Second format

In order to address the shortcomings of the original competition in terms of the research focus the competition format had to be changed. In 2005 the competition underwent major rule changes. The coaches started to be evaluated on their ability to model specific weaknesses [KKS06]. After modeling the opposing team the coaches try to predict such weaknesses during a game. To be successful a coach must isolate these weaknesses (called *patterns*) from the rest of the of the team behaviour (called *base strategy*). The competition rules define these terms as:

patterns a simple behaviour that is predictable and exploitable.

base strategy the general strategy of the team, independently of any present pattern.

The competition is divided into two rounds. In the offline round the coach reviews the game log files in order to construct its pattern models. The game log file contains only low-level information such as ball and players position and velocity as well as some player actions such as kick and grab. Higher-level information such as passes, dribbles and shots on goal must be inferred. A coach must model 15 to 20 patterns. For each pattern the coach is given two log files: one from a team that displays the *base strategy* and another from a team with the same *base strategy* but that additionally displays the *pattern*. Each file is labeled with the corresponding pattern name. According to the rules the coach has 5 minutes per log file to construct its models and save the created model to be used in the next phase.

In the online round the coach observes a game of a team that exhibits an unknown combination of 5 of the patterns observed in the offline round. When a pattern is recognized it is announced to the server by the coach. At the end of the game the patterns that the coach detected together with their announcement times are entered into a scoring function to rate the coach performance. A simplified view of how the scoring function works follows. A correct pattern declared at time, t_d, earns the coach the following points:

$$(9000 - t_d) \times \frac{N_T - N_A}{N_T} \tag{2.1}$$

Where N_T is the total number of patterns (15–20), and N_A is the number of currently activated patterns (5). An incorrect pattern declaration, at time t_d, earns the coach the following penalty:

$$9000 - t_d \tag{2.2}$$

The scoring function was build so that the expected value for random guessing is 0. [Com06]

The online coach can give advice to its own team. This is done by specifying the team behaviour previously to the game and communication during the game. As previously mentioned, to allow interoperability this communication is done using a standard language, the CLANG [Com06]. Although the coach is not evaluated according to its team performance, this advise

is still expected to happen because a coach can use it to force the opposing team to expose the patterns to be detected. This competition no longer exists being 2006 the year of its last occurrence.

2.6 Conclusions

Intelligent agents are an adequate abstraction for many problems faced nowadays. Although proper construction of such agents is important for many applications, it rare that those agents are isolated and more common that those agents need to interact with each other in what are now called to multi-agent systems. Those systems require mechanisms to manage agent interactions, i.e., coordination.

Robotic Soccer is a compelling research field in which, nowadays, many researchers focus their work, most of that work in done in the context of the RoboCup competition since it provides not only a common field in which researchers can test their work but also a competition/challenge that nurtures healthy competition.

Chapter 3

Related Work

In this section a brief analysis of existing statistics generation software is done, as well as existing research in the field of historical data usage.

3.1 Statistics Generation Software

Being soccer one of the most popular sports in the world [Dun99], it has aroused interest as a standard problem for scientific research as well has a potential market for professional soccer analysis tools. In this section statistics generation tools were split into two groups:

- **Professional** Performance tools used by real soccer coaches;
- **Research** software, in particular the ones used in the context of RoboCup Soccer Simulator 2D.

3.1.1 Professional Soccer Performance Analysis tools

Today, the competitiveness in professional soccer is higher than ever so, even the slightest detail can influence the match result. As a consequence of that every team must prepare itself the best it can to face an opponent, not only improving their weaknesses but also studying its opponent limitations. In order to achieve that goal professional soccer coaches use automatic tools capable of generating tactical and technical information about his own team and his next opponent. In this particular section four distinct software tools were analyzed (table 3.1.1). Three of the four analyzed softwares capture the match images through video cameras. Usually, that cameras are spread over the soccer field and normally there are at least eight cameras. Mostly due to occlusion problems some images need to be treated manually.

In spite of these systems being able to calculate a large number of statistics none of them provide complete enough statistics to enable the construction of either player or team models, negating the ability to accurately define, by themselves, effective strategic options.

Table 3.1: Professional Performance Tools Comparison (adapted from [Abr10])

Name	Strengths	Weaknesses
Amisco	-Statistical treatment on individual and collective level	-The need for manual treatment of the images after acquisition -Non-existence of player and team modeling
Prozone	-Statistical treatment on individual and collective level	-Non-existence of player and team modeling -Inability to show the real video feed together with 2D analysis
Ascencio Match Expert	-3D game viewer where the user can see the soccer match (from previously collected images)	-Poor statistical treatment when compared to the previous software analysed
Match Vision Studio	-Rich statistical treatment	-No model construction of the players and teams despite the statistical data allowing it

3.1.2 Research

Over the years researchers have tried to develop software capable of:

- Modeling human behaviors in computational agents [Hil01, Sil02, BD04, GM05, SM05].
- Modeling of agent behavior based only on observation [LASB04a, SRV00].
- Simulating specific environments, which can be a natural catastrophe [KT01], a sport, a supply chain management [AMSV07], among others.

For this research work the focus will be specially on the simulation soccer environments (table 3.2). Being the RoboCup Soccer Simulator 2D one of the more prolific applications in its field, its shortcomings such as the inability to generate statistical data have been surpassed by complementary software solutions, like *SSIL* (Simulated Soccer Internet League) statistics or team assistant which have the capability to calculate a huge amount of individual and collective statistics, relying on the RoboCup Soccer Simulator 2D for the simulation part.

Among these SoccerScope2 [IotUoEC] stands out because even though its generated statistics have little relation to the team performance, its simple architecture, source code availability and visual debugging capabilities make it a suitable option for a base code when building a statistics generation software.

Table 3.2: Research Software Comparison (adapted from [Abr10])

Name	Strengths	Weaknesses
SSIL statistics	-Capable of calculating many different statistics -XML statistics output includes a XSLT for convenient HTML visualization	-The set of generated statistics is still incomplete and did not represent a good spectrum of team performance
Team Assistant	-Both individual and collective statistics -Realistic 3D viewer	-Camera sync not yet perfected
Logalyzer	-Capable of calculating many different game statistics	-Generated statistics still not accurate enough for representing a team performance
SoccerScope2	-Includes a visual debugger	-Generated statistics are still very basic

3.2 Historical Data Exploiting

Using historical data to exploit opponent behaviour in favor of our own team goals is not a new approach to the robotic soccer team optimization problem, in particular using the soccer simulation 2D server [NKTN05, KKS06, LASB04b, FMMM06]. This historical data can be used to mimic opponent behaviour, identify and exploit individual players behaviour, and identify and explore team behaviour.

The approach documented by Nii et al. [NKTN05] propose an action rule discovery technique that analyzes the game log files to discover and mimic the most relevant goal scoring actions contained in the log files. The actions to mimic are passes and shots that are classified into success or failure, and the rules of when and where to pass are learned according to that classification. Although valuable this approach is different from the desired approach because it focusses on atomic actions instead of strategic advice.

In the work by Ledezma et al. [LASB04b] a three phase approach to a single low-level opponent modeling based on historical data is proposed. In the first phase a classifier to label the opponent actions based on observation is built. Afterwards, the agent observers the opponent and labels its actions using the classifier. From these observations an opponent model is built. Finally, the built model is used to anticipate the opponent actions. The build model although valuable cannot be used to provide high-level advice by means of a coach.

3.2.1 Team Behaviour Modeling

The approaches documented Kuhlmann et al. [KKS06] and Fathzadeh et al. [FMMM06] were created in the context of the RoboCup coach competition (see chapter 2.5.2), a competition whose goal is not to make our team score more goals than the opposing team but to detect the highest

possible number of the game patterns. This competition format was proposed in an attempt to foster the opponent modeling research.

In [KKS06] an approach to opponent team joint behaviour modeling is presented. This model can be important for determining an agents future actions. The proposed approach grants an autonomous agent the ability to analyze past games of the current opponent, advising its own team of how to play against this opponent, and identifying patterns or weaknesses of the opponent. The proposed approach attained good results in the RoboCup 2005 coach competition, finishing in first place.

The approach documented in [FMMM06] presents a 3-tier architecture for opponent modeling, opponent pattern detection and team advice, by analysing the opponents past games. This approach is developed in the context of the RoboCup simulation coach competition. In the first stage, through log analysis, sequential players events are identified. Then, the pattern is detected, a pattern being a sequence of events occurring a sufficient number of times during the game. After the completion of this stage the base and the patterns models are created and the final stage consists of calculating the difference between the base model and the patterns model and storing it as the final model. Based on this an appropriate strategy is defined. The defined model is then used in the online competition in order to identify the occurrence of patterns. The proposed approach attained good results in the RoboCup 2006 simulation coach competition, finishing in first place.

3.2.2 Coaching in Adversarial Domains

In this section the state of the art work in coaching in adversarial domains will be presented. Since coaching in adversarial domains can be seen as providing advice to the distributed players in order to help the team maximize the effectiveness of its response, it is very important in the context of this thesis.

One interesting approach is described in [RVK02]. It was developed in the context of the coach competition (first format). This approach is focussed in 3 points:

- **Learning Formations**. This formation concept is used to identify the "home area" of a player. The home area is the region of the field where a player should generally be. In this approach the formation is represented as a rectangle for each player of the opposing team. From the formation general player roles inferred. Those roles are midfielder, defender or attacker.

- **Set plays**. Set plays are related to the times when the ball is stopped and one team has time to prepare before kicking the ball. The coach uses this time to make a plan for the agents to follow.

- **Rule Learning**. Using a combination of *clustering* and *C4.5*, rules that describe the passing behaviour of the opponent are generated. The attributes of the rules are the locations of the passer and receiver (using the regions learned when learning formations) and the relative

position of all teammates and opponents. These rules are then transformed into *CLang* for usage.

The described approach is very interesting but still has some details that result in flaws or less performance, in particular, against teams that use dynamic positioning or role exchange.

Another approach is the one described in [KSL05]. This approach treats coaching as a learning problem, and breaks the problem down into learning three components:

Offensive advice

The offensive component is related to learning how to score. This component is learned from observing games where their future opponent loses and trying to imitate the behaviour their opponent (the future opponent opponent). This is done by modeling the player with the ball and then creating a decision three that describes its behaviour related to two possible actions (passing or shooting).

Defense advice

The defensive component is related to learning how to act near our own goal. In case of the defensive advice the approach focuses on modeling the future opponent defense and then attempt to foil the predicted strategy, based on the games the future opponent wins. This is done by predicting how a given player will acquire the ball.

Formation advice

Describes where the players should position them by default. Similar to the offensive advice, this component is learned from observing the games that the future opponent loses and then tries to imitate their opponent behaviour. The formation is defined as a home position (X,Y) and a ball attraction vector (BX,BY) for each player. The X and Y values are calculated as the average position of the player during the game. The BX and BY values were manually defined.

This accounts for role exchange by updating the rules when role exchange occurs. In addition, in this research was identified that the formation advice was more effective than the other advices.

3.3 Conclusions

In area of statistics generation software both professional and research software exist but unfortunately none of them was found to offer a complete set of statistics sufficient to model an opponent. Analysing existing work related to historical data exploiting reveals existing work that is capable of building opponent models that are not applicable to high-level advice, or building opponent models that can only be used to identify patterns and not to give online advice, finally, the work that is found to help the team maximise the effectiveness of its response does not use high-level models such as match statistics.

Chapter 4

FC Portugal 2D

FC Portugal 2D—presented by Reis et al. [RLM10] —is a robotic soccer team jointly developed by *LIACC* and *IEETA* since February, 2000 [RL] to participate in the RoboCup soccer simulation 2D competition, whose main goal is the development of a formal model for the concept of team strategy in the context of a partially adversarial environment, this model should also be general enough to allow it to be used in various dynamic competitive domains. Its research is also concerned with developing general decision-making and cooperation models. The preferred method for low-level skills tunning is online optimization.

4.1 Architecture and Knowledge Structures

Performing cooperative multi-agent tasks, like playing simulated soccer, in a partially cooperative, partially adversarial environment requires a lot of knowledge, which makes definition and usage of good knowledge structures essential. The paradigm chosen partitions the knowledge in three levels: individual action execution; individual decision-making; and cooperation. Individual action execution is concerned with the actual commands needed to perform a low-level action. Individual decision-making determines the action that an agent chooses to execute (from an available set of actions). Cooperation knowledge is concerned with tactics, situations dynamic formations, roles, dynamic plans, and communication protocols [RLO01, LR07].

The team architecture is based around the idea of a common framework for Cooperative Robotics [MRdB06, MR08]. This architecture exploits the common characteristics of cooperative multi-agent systems in order to provide a unique framework for different problems. In order to control different robots (simulated and real) the common framework is completed with specific components that deal with the agents action and perception capabilities. These low level skills and perceptions mechanisms are defined for each type of robot in each type of league, while the high-level actions are chosen though the same decision-making component. The low-level components are integrated with the high-level decision making component through an action vocabulary

that enables the low-level components to understand the high-level decision-making. A perception vocabulary addresses the representation of state-of-the-world information. In order to make the common framework more flexible, allowing the replacement of components in real-time, it requires a flexible architecture the can be modified in both real-time and compile-time.
The high-level Common Framework allows the same high-level controller to decide independently from the low-level skills and perception frameworks. This allows the players to rely on different, and redundant, low-level implementations [CLR07a, CLR07b].

4.2 High-level decision and Cooperation

The **Dynamic Positioning and Role Exchange (DPRE)** mechanism [RLO01, LR07] is extended from the work done by Stone et al. [SV99]. This enables the agents to switch their relative positions (for a given formation) and roles (that define agent behaviour at several levels), at run-time, on the field.
The **Situation Based Strategic Positioning (SBSP)** mechanism is used to dynamically spatially position a team using different formations for different situations. This mechanism is based on the distinction between active and strategic situations [RL01, RLO01]. If an agent is not involved in an active situation it tries to occupy its strategic positioning that changes according to the situation of the game. Situation is an high-level game analysis concept (attacking or defending for example). For each situation a formation must be manually defined. SBSP was one of the main innovations brought forth by the FC Portugal team.
These two concepts are now widely spread and used in various teams.

4.3 Strategical coordination layer

FC Portugal utilizes a multipurpose, adaptable, strategical coordination layer that allows the management of heterogeneous teams in both centralized and decentralized teams, using reduced communication [CLR07a, CLR07b]. The strategical model uses a multi-level hierarchical approach (seen in figure 4.1). The lower level uses the concept of roles to reflex the agent's usual activities. The middle level introduces a sub-tactic that aggregates agents with various roles to solve partial objectives [CLR07a, CLR07b]. On top of the sub-tactics, the concept of formations is used to distribute agents through the sub-tactics. Over that level a tactical level employs an hybrid method to switch formations. This method is based on events situations and precedences. On top of all those levels a strategical level is defined which allow communication between tactics according to scenario conditions.

These methodologies have proven themselves in various competitions and controlled experiments.

The downside of this approach is that it requires to manually define each component, often resulting in defining only a single highly tunned multi-purpose tactic, that requires less effort to maintain than several different ones (this was the case of FC Portugal during the RoboCup 2010

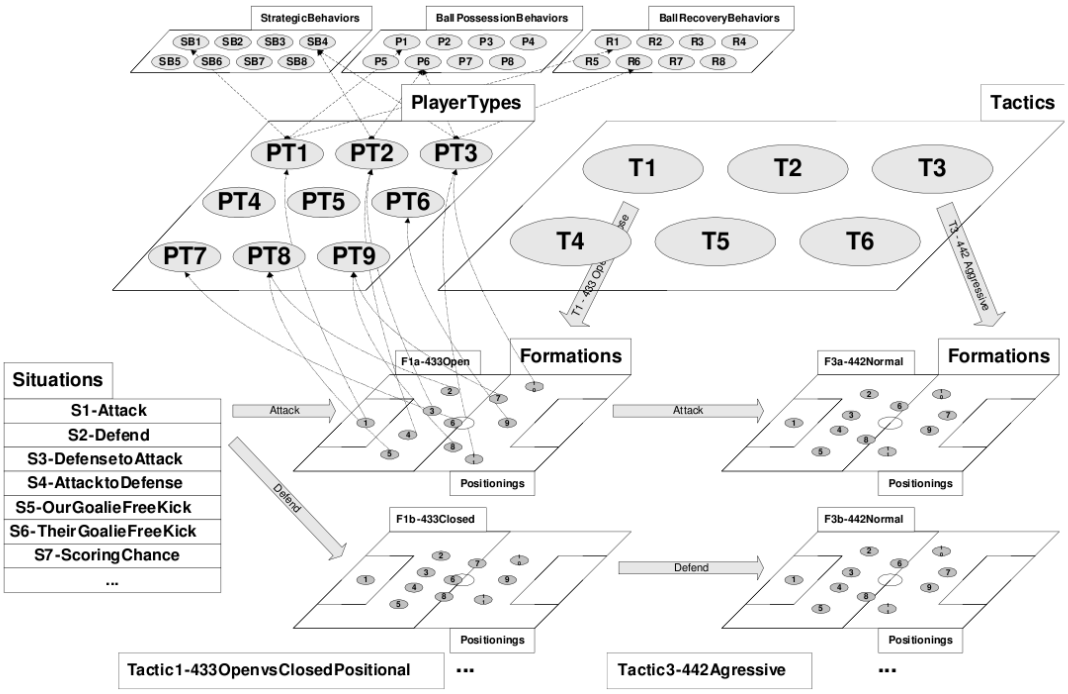


Figure 4.1: FC Portugal strategy model. [RLM10]

classification). A typical exception to this rule occurs when new approaches are being studied (since matchflow tool and new setplays were being studied, in the 2010 RoboCup competition both of those were defined before a match according to the opponent).

4.4 Formations

The notion of formation is ubiquitous to the agent, having a part in its architecture, cooperation and coordination. A formation describes how the players in a team are positioned on the field. Different formations can be used depending on the style of play a team wants to have.
Being formation such an important concept a special tool called matchflow is used to define the formations. This tool allows its user to define the position of each player relative to the ball position, to do so key ball positions and the corresponding players positions are defined, the players positions not defined are obtained by finding the closest defined three points by using delaunay triangulation and the final position of the players is found through interpolation of those three points 4.2.

4.5 Setplays

Setplays are commonly used in many (human) team sports such as soccer, rugby, handball, basketball. Although there are several important differences between human soccer and robotic soccer

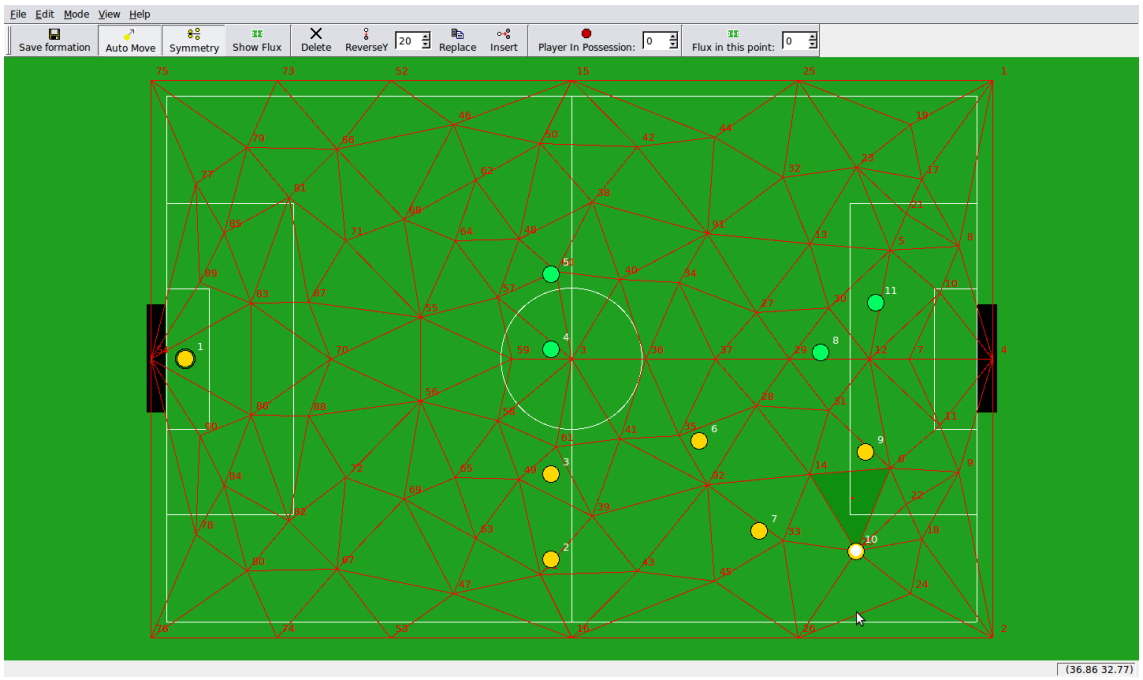


Figure 4.2: Defining a formation using matchflow.

both can benefit greatly from the use of setplays as a tool for high-level coordination and cooperation.

Being setplays such an important tool, a framework for setplay definition and execution applicable to any RoboCup cooperative league and similar domains was developed. The framework is based in a standard, league-independent and flexible language that defines setplays that can be interpreted and executed in real time [MRdB06]. The framework general structure is shown schematically in figure 4.3.

At top level a *Setplay* is identified by a name, and has parameters, which can be simple data types like integers and decimals, or more sophisticated concepts like points and regions. Setplays also have *PlayerReference*s, which identify players taking part in the setplay. Player references can point to specific players, or be *PlayerRoles*, that is, abstract representations of a particular role in the setplay. Parameters and player roles are instantiated at runtime.

Steps are the main component of a setplay. A *Step* can be seen as a state in the execution of a setplay. To control the execution of the setplay each *Step* needs a wait time and abort time in order to control its execution. A *Step* also has a condition, that must be satisfied before entering the step. The possible ways out of a step are defined as *Transition*s. Transitions have a condition, that determines when the *Transition* should be followed. Transitions are divided into three types: *Abort*, which aborts the setplay; *Finish*, that represents the end of the setplay; and *NextStep*, that links sequential steps. The *NextStep* transition besides indicating the following *Step* contains a list of directives that will be applied in order to accomplish the transition.

The setplays have a graphical definition tool called playmaker (described by Reis et al. [RLML10])

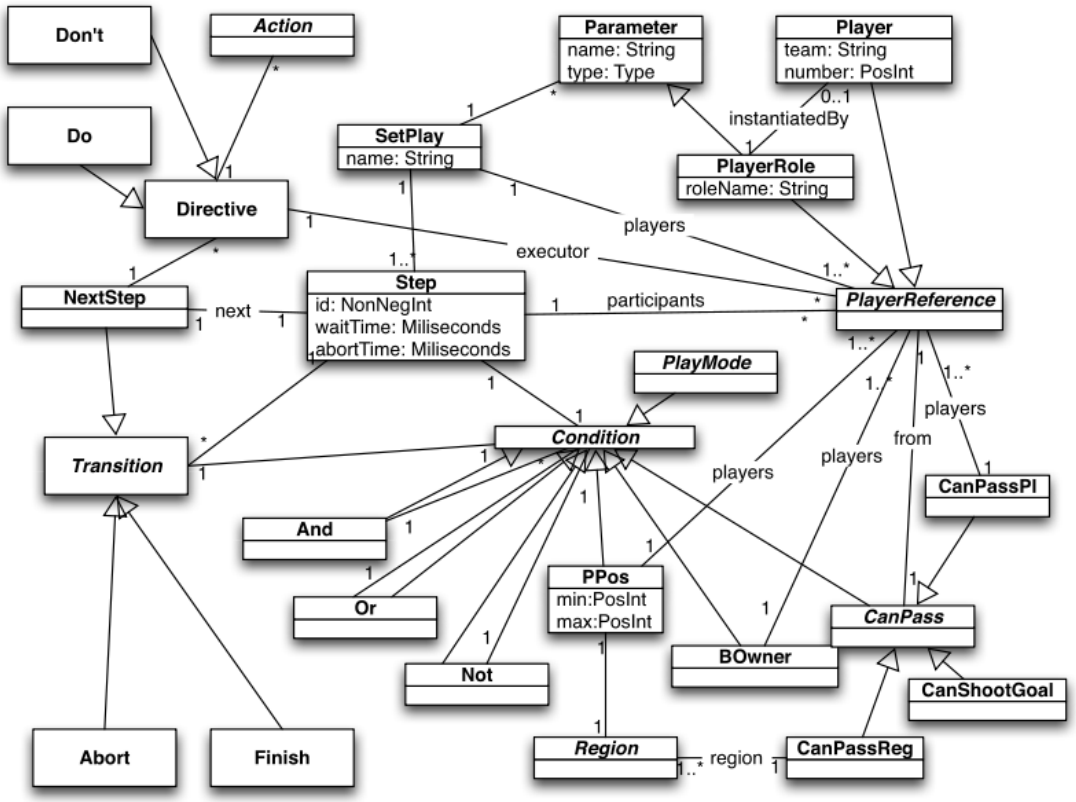


Figure 4.3: Setplays structure [MR07b].

that allows easier definition of the setplays. Even so, setplays definition can be done directly by editing a text file, because its syntax is in easy to understand, human readable *s-expressions*.

4.6 Configurable Strategy Parameters

The team has many dynamic/configurable parameters which were previously described. In this section the values defined for each of those parameters in the context of the matches generation (section 5.2) are described.

4.6.1 Formation

Two different formations were defined using the matchflow tool (presented in section 4.4). These are commonly know as 433 (figure 4.5) and 442 (figure 4.4). It is important to note that for each formation to fully exploit the power of SBSP it must be separately defined for each of the following situations [Rei03]: Defense; Attack; Defense-Attack transition; Attack-Defense transition; Dangerous Attack (by own team); Free kick (by own team); Drop ball(by own team); Goal kick (by own team); Kick-off (by own team); Throw-in (by own team); Corner kick (by own team); Penalty (by own team);

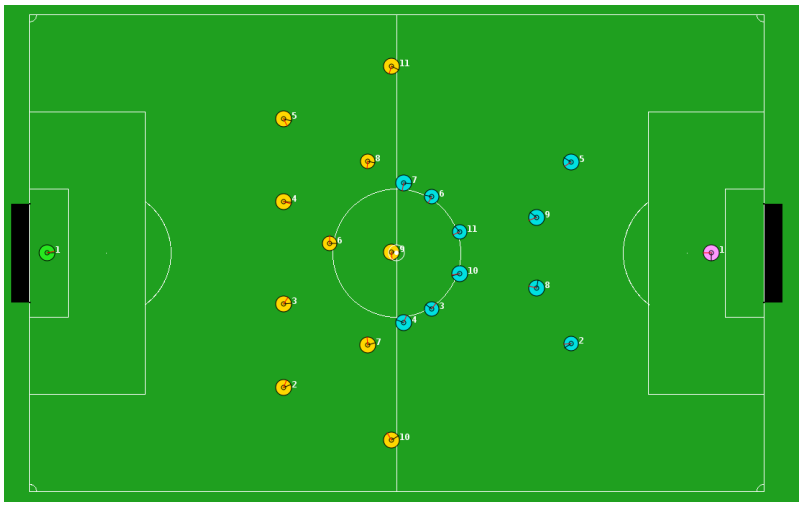


Figure 4.4: Formation 433 at the start of a match (left team).

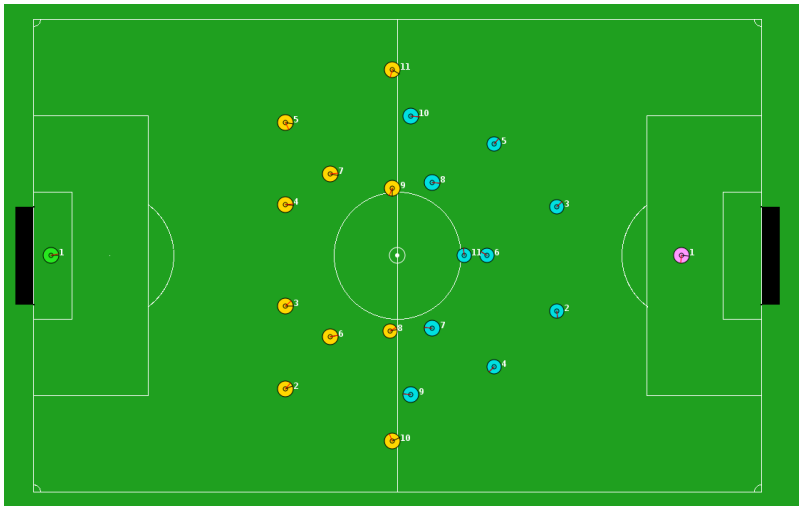


Figure 4.5: Formation 442 at the start of a match (right team).

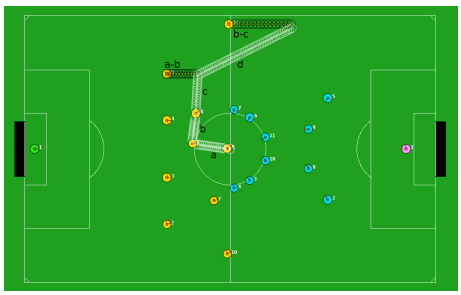
4.6.2 Setplays

The 8 setplays defined were equally divided into 4 groups according to their first step condition:

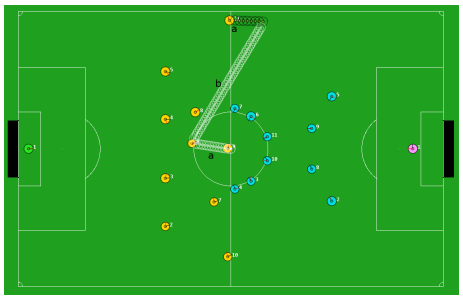
1. **Kick-off** is a situation that characterizes the beginning or resumption of a soccer match (e.g. after a goal scored) [dFA10, CFH+]. In this work two set plays related to this game situation were used:

   Kick-off to winger (4 steps): 4.6(a)

   (a) The kick-off taker passes the ball to the midfielder, while the defender positions himself to the left of the midfielder.

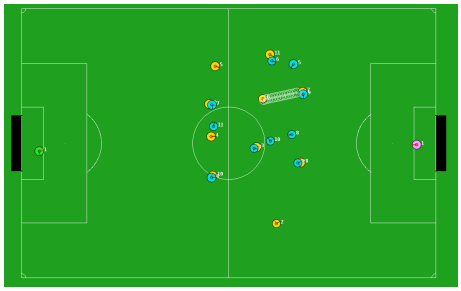


(a) Kick-off to winger (4 steps).

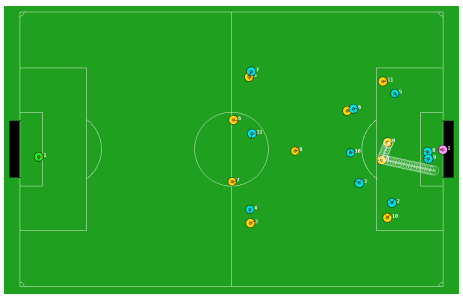


(b) Kick-off to winger (2 steps).

Figure 4.6: Kick off setplays.



(a) First free kick setplay.



(b) Second free kick setplay.

Figure 4.7: Free kick setplays.

(b) The midfielder passes the ball to the attacker, while the winger moves itself near the offside line to the left. The defender keeps himself to the left of the midfielder.

(c) The attacker passes the ball to the defender, while to winger keeps progressing towards the offside line to the left.

(d) The defender passes the ball to the winger.

Kick-off to winger (2 steps): 4.6(b)

(a) The attacker makes a direct pass to the midfielder. The winger positions himself near the offside line to receive a pass from midfielder.

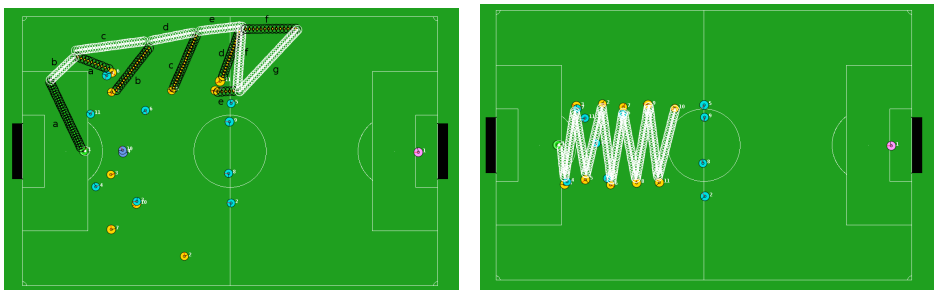
(b) The midfielder passes the ball to the winger.

2. A **Free kick** is awarded to the opposing team if a player commits an offense or foul [dFA10, CFH+]. The free-kick is taken where the offense occurred. Two free-kick situation setplays were defined:

   First free-kick 4.7(a):

   (a) The taker passes the ball to the striker.

   (b) The striker passes the ball to the left-winger or to the right-winger, if not possible, it passes the ball back to the taker.



(a) First goal-kick setplay. (b) Second goal-kick setplay.

Figure 4.8: Goal-kick setplays.

Second free-kick 4.7(b):

(a) The taker passes the ball to a receiver near the goal.

(b) The receiver shoots at the goal.

3. A **Goal kick** occurs when the ball has wholly crossed the goal line without a goal having been scored and having last been touched by a player of the attacking team; awarded to the defending team [dFA10]. The two different goal-kick situation setplays are:

   First goal-kick 4.8(a):

   (a) The goalkeeper and the left defender move to the left region of the field (the left defender more to the left and in front of the keeper).

   (b) The goalkeeper kicks the ball to the region where the left defender is running, the left defender intercepts the ball. The midfielder moves to the left and front of the left defender.

   (c) The left defender kicks the ball to the place the left midfielder is running, the left midfielder intercepts the ball. While the left midfielder intercepts the ball, the left forward moves to the front and left of the midfielder;

   (d) The left midfielder kicks the ball to be intercepted by the left forward. A runner moves to the front of the left forward.

   (e) The left forward kicks the ball forward to be intercepted by the runner.

   (f) The runner kicks the ball to its right that is received by a kicker. The runner keeps running forward.

   (g) The kicker kicks the ball left and forward to be intercepted by the runner.

   Second goal-kick 4.8(b):

   (a) The players position themselves in two rows, perpendicular to the goal line, in front of the penalty area (except the goalkeeper that stays inside the penalty area). The goalkeeper kicks the ball to the player closest to him on his right.

(b) The player that has the ball passes the ball to the player in front of him (on the other row) and slightly more advanced on the field than him (it is a zig zag slalom).

(c) The previous step is repeated until the last player receives the ball.

4. A **Corner kick** is awarded when the whole of the ball passes over the goal line without a goal being scored having last touched a player of the defending team [dFA10].

   First corner kick:

   (a) The taker passes the ball to the receiver that is near him.

   (b) The receiver passes the ball to the left striker that is in front of the opponent goal.

   (c) If the left striker has angle to shoot, it shoots at the goal, else it passes the ball to the center striker.

   (d) The center striker receives the ball and shoots.

   Second corner kick:

   (a) The taker passes the ball to the receiver that is near him.

   (b) The receiver dribbles the ball back to the middle of the penalty box.

   (c) The receiver passes the ball to either the left, right or center attacker.

   (d) Whoever received the ball shoots at the goal.

4.7 Conclusions

FC Portugal is a team with a beautiful, fast, "real soccer like" way of playing simulated soccer. It provides many mechanisms to adjust its strategy whose manual tunning could be partially replaced by automatic mechanisms (such as the one proposed by this thesis). The latest developments should allow it to achieve good results in the forthcoming competitions.

Chapter 5

Statistics Generation and Automatic Simulation

In this chapter the selected match events as well as the statistics generation process are described. The match generation framework is also described. The statistics generation is based upon the detection of match events, those events were selected according to an expert panel. The match generation framework is an another component that allowed for the generation of game logs whenever necessary.

5.1 Statistics Selection and Generation

The decision to identify opponent behaviour based on statistics was done so that the information could be represented in a succinct way that was still understandable by humans and that could be used in contexts other than this work. Although some work has already been done in this area (see 3.1) none of them completely fulfilled the identified requirements.

This led to the statistics generation module described in this section. The base principle of this module is the detection of high-level events from the low-level information existing in the logs. Much like, but not limited to, what is found in [HCH+02]. The high-level events definition and selection was made according to the indications given by a sport expert panel constituted by academical sport professors.

In the remainder of this section the events defined by the experts panel are described 5.1.1, next the tool used as the base for development is presented 5.1.2 and at last the events detection mechanisms are explained 5.1.3.

5.1.1 Detected events

The events listed here were considered the most relevant in the study. Although the events detected in this work are of common use, a brief description of them is included.

Ball Kick A ball kick occurs when a player sends the *kick* command and the ball is within range $[0, kickable\_margin]$.

Ball Possession Ball possession is determined by the ball kicks. If the team of the current player is the same as the previous player kicking then the ball possession did not change, on the other hand when this does not hold (they are not from the same team) the ball possession between those two kicks is considered to be $NEUTRAL$. This ball possession data is extended by contextualizing every instant of possession to the region where it occurred, the zones considered here are the interception of the $WING$ and $QUARTER$ areas as seen in figure 5.1.

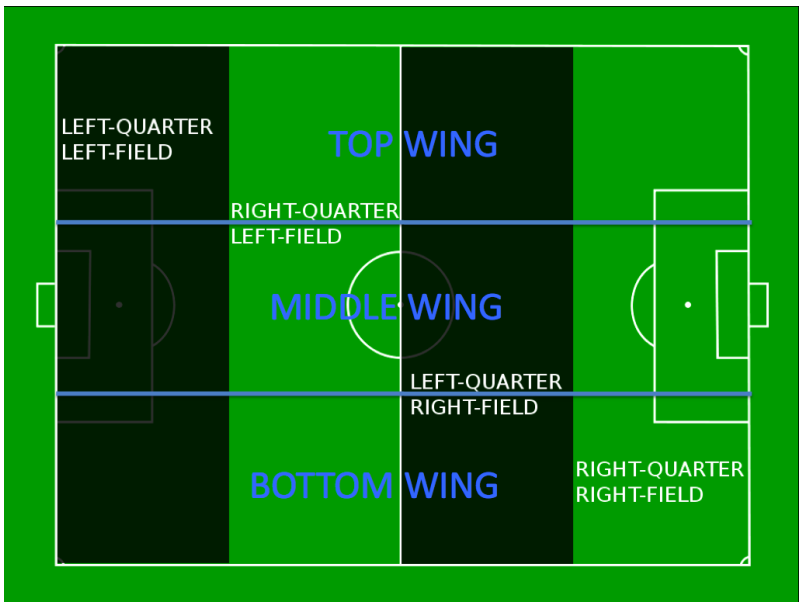


Figure 5.1: Soccer field wings and quarters partition

Pass A pass is when two players of the same team, exchange the ball with each other. One of the players (the kicker) kicks the ball so that his teammate (the receiver) can catch it, if the receiver catches the ball without any of the opponents intercepting the pass was successful on the other hand if the pass is intercepted it is considered a **pass miss**. The pass is considered **offensive** when it ends in the opposing teams $MIDFIELD$. It is a **break pass** when the receiving player of the pass is inside the opposing team $DANGER\_AREA$ and has a goal-scoring opportunity. The goal-scoring opportunity is defined by creating a virtual triangle having the player and the opposing team goal poles as vertexes (figure 5.2) and checking if the number of the opposing team players inside are less than two. A break pass is always an offensive pass.

Goal A goal occurs when the ball crosses the goal line. This verification is executed by the soccer server which has a mode that indicates when a goal occurs. In this research project, an analysis of where the ball is kicked (and originates the goal) is also executed. These

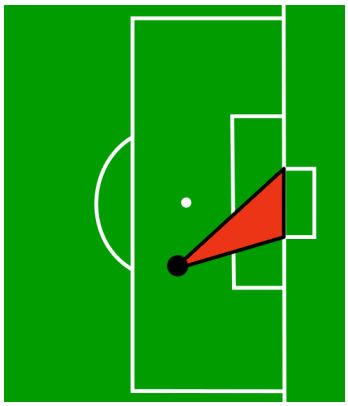


Figure 5.2: Triangle area in red (the black dot represents the player)

areas include the *GOAL_AREA*, the *PENALTY_AREA* or others (designated as *FAR_AWAY* areas).

Goal Miss Is a kick that was either intercepted by the goal-keeper or goes outside near the goal line (the two types of *Goal Miss* detected). Similar to the *Goal* event, in the *Goal Miss* analysis the location of the ball when it was kicked is also detected.

Attack An attack from a team perspective is when the ball advances in the direction of the opposing team field starting by a kick of one of its players. The attack must last at least 30 cycles to be a valid attack. In the attack process if the ball goes back but stays in the last quarter of the opposing team field and in the attacking team possession, the attack remains valid. When the ball reaches the *DANGER_AREA* the attack is classified according to a temporal factor (0.9 for a fast attack, 0.4 for a medium attack otherwise will be a shorter attack. If the ball does not reach the *DANGER_AREA* the attack will be classified as a broken.

Pass Chain A pass chain is a sequence of passes made by the same team without losing ball possession.

Goal Opportunity This occurs when three conditions are met. The player is in kicking conditions (the ball is within range - $[0, kickable\_margin]$). The ball is in the *DANGER_RADIUS* zone, this was defined as being the zone in which the distance to the opponent goal is sufficiently close to represent great danger. The last condition is that the previously mentioned virtual triangle that has the player and the opposing team goal poles as vertexes only has one or zero opposing team player inside (see 5.2).

Wing change This event occurs when a pass starts and ends in different wings (wings are the horizontal partitions seen in figure 5.1). A wing change from/to the *MIDDLE* wing is considered a *partial* wing change. In some special situations two *partial* wing changes could represent a *full* wing change.

5.1.2 SoccerScope2

The statistics generator tool *SoccerScope2* is used as the base code for development. SoccerScope2[1] is a monitor/log player written in Java. It has several monitor/visual debugger features like showing stamina, the offside line, players' kickable area and visibility regions. It also includes some basic statistics calculation, like ball possession and passes velocity.

Internally it reads the logs and creates a list of `Scenes` each corresponding to an instant of the game (as seen on figure 5.3). The size of this list is usually 6000 because is the standard game duration. This structure allows a sequential analysis of the match which, facilitates the adaptation of this software for both offline and online analysis. Another reason for choosing this tool as the base code is its easy expandability, since the addition of new modules just requires the definition of a class that is child of the `SceneAnalyzer` type and subsequent registration as an analyzer.

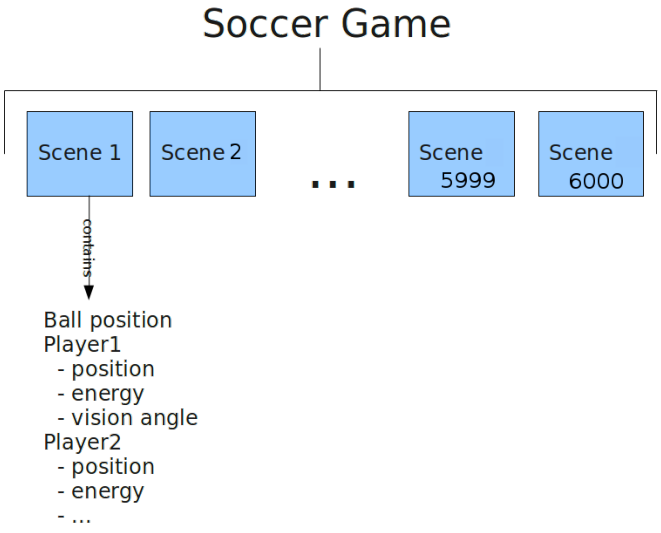


Figure 5.3: Game structure

5.1.3 Implementation details

When implementing the temporal sequential for events detection two distinct approaches were used: Immediate event detection and event detection thought interval analysis.

Immediate event detection is mostly characterized by identifying the events in the cycle they end and usually only needs information from the cycle being analyzed and the previous one. **Interval analysis** is characterized by doing the analysis by request and returning all the events that occurred in that interval, this approach is usually used when information from more then the current and previous simulation cycles is necessary and information from other analyzers is necessary.

The difference between the two is insignificant when doing offline analysis but is relevant when

[1] More information online at `http://ne.cs.uec.ac.jp/~koji/SoccerScope2/index.htm`

doing online analysis. The immediate event detection model is more adequate as it avoids the explicit need to request the analysis of an interval and distributes the processing along the entire match instead of doing all the processing on request.

5.1.3.1 Ball Kick

The ball kick detection is one of the most basic events to be detected. To detect a ball kick it must be checked that the player issued the kick command and was sufficiently close to the ball to hit it (see algorithm 1). This algorithm is implemented using the immediate event detection approach due to its simplicity.

Algorithm 1: Ball kick

```
if player.can_kick() AND player.is_kicking() then
    return true;
else
    return false;
end
```

5.1.4 Pass and Ball Possession

As previously mentioned a pass occurs when two players of the same team, exchange the ball with each other. The ball possession is determined together with the passes because part of their algorithms overlap. At the end of every analysis, if the ball was kicked the variable *kicker* is updated so that in the subsequent cycles the kicker is known. If the kicker and receiver are of the same team the pass succeeded, else, the pass could be a pass miss. To confirm that it is a pass miss, the ball velocity and direction are checked against the teammates current positions and movement speed to confirm that a teammate could have received the ball. The pass event detection

is immediate.

Algorithm 2: Pass and Ball Possession

```
if player.kicked() then
    if kicker != null then
        receiver=player;
        if same_team(kicker,receiver) then
            new_pass(kicker to receiver);
            update_ball_possession(kicker.time, receiver.time, receiver.team);
        else
            update_ball_possession(kicker.time, receiver.time, NONE);
            if kicker.a_teammate_could_receive() then
                new_pass-miss(kicker to kicker.pass_target())
            end
        end
    end
    kicker=player;
end
```

Whenever a player kicks the ball the ball possession table is updated, whether it the ball changed possession or remained on the same team. The ball possession detection is not immediate because the possession can only be determined when a player receives the ball, even so, as soon as the possession can be determined the possession table is updated. The determination of the zones where the ball was is, is done by interval analysis, simply by checking the ball possession and ball position for each cycle upon request (see algorithm 3).

Algorithm 3: Ball Possession by Zones algorithm.

```
for time = start_time to end_time do
    ball=get_ball(time);
    possession_team=possession_at(time);
    foreach zone in Zones do
        if zone.contains(ball.position) then
            zone.increment(possession_team);
        end
    end
end
```

5.1.4.1 Goal and Goal Miss

The goal event is already detected by the server and properly signaled. Even so, it is necessary to check who kicked the ball and which zone the ball was kicked. This is straightforward as can be

seen in algorithm 4.

Algorithm 4: Goal algorithm.

```
if isGoal then
    kicker=get_last_kicker();
    zone=which_zone(kicker.position);
    return new_goal(kicker,zone);
end
```

The goal miss event requires to check if the goalie catched the ball, in which case an event the goal miss event is of the type goalie_catch. The other situation when a goal miss occurs is when the ball goes outside (which is why the goal kick is checked), in this case it is determined where the ball intercepted the goal line through dead reckoning, based on speed and position of the ball, this is necessary to distinguish goal misses that miss by a lot and by a little.

Algorithm 5: Goal algorithm.

```
if goalie.catched() then
    kicker=get_last_kicker();
    if kicker.team != goalie.team then
        zone=which_zone(kicker.position);
        return new_goalmiss(kicker,zone, GOALIE_CATCH);
    end
end
if playmode_changed_to(GOALKICK) then
    predicted_ball_pos=prev_ball.pos + prev_ball.vel ball_mov = (predicted_ball_pos -
    prev_ball.pos);
    inter=intersect(ball_mov, goalline);
    if inter then
        kicker=get_last_kicker();
        zone=which_zone(kicker.position);
        new_goalmiss(kicker, zone, inter.type);
    end
end
```

5.2 Matches Samples Generation

This section contains an overview of the matches generation framework. The matches generation provides an environment that allows simulations to be run programmatically, using a simple framework interface.

In figure 5.4 an overview of the matches generation framework is presented.

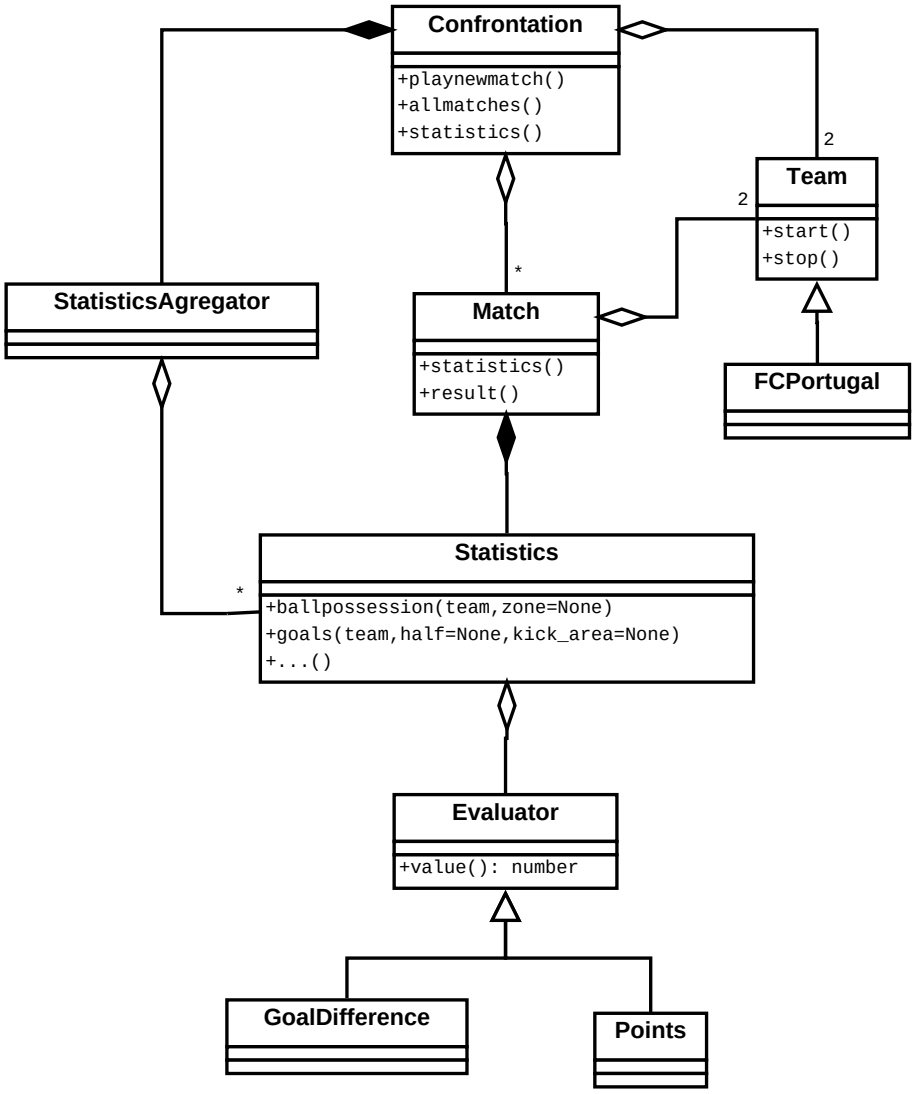


Figure 5.4: Matches generation framework object model, overview.

5.2.1 Example of usage

A typical use of the framework would be: create two *Team* entities and pass them as an argument to the *Confrontation* entity–alternatively the *Confrontation* entity accepts a string instead of a *Team* (as long as it corresponds to an existing team). Then calling the method *playnewmatch()* of the *Confrontation* entity would effectively run a new simulation between the two corresponding teams.

5.2.2 Entities description

The main entities are described in this section.

Confrontation is one of the most important entities since it should be chosen as the preferred interface for interacting with the framework. This entity is responsible for managing all of

the matches between two teams.

Team corresponds to an existing team. It is responsible for managing a team entire life-cycle, enabling the *Match* entity to actually run the simulations. **FC Portugal** is a specialization of the team that allows for specific configurable parameters to be passed.

Match corresponds to a simulation. When it has finished instantiating it always as a finished simulation associated with it. Two ways of instantiating a match exist: giving it only the parameters of the two teams (this will run a new simulation) or giving it a match ID which loads the match from cache.

Statistics entity loads a match statistics into an easy to use entity that contains several easy access methods, to access specific statistics. This entity is returned by the *Match* method *statistics()*. If the match statistics have not yet been calculated the *Statistics* entity automatically uses the match logs and the statistics generation tool to obtain the match statistics.

StatisticsAggregator contains a set of *Statistics* from the same pair of teams and when a statistic is requested it returns the average of the results from its *Statistics* set. This entity is returned from the *statistics()* method of the *Confrontation* entity, and serves the purpose of consulting the statistics from the several matches of one confrontation at once. The *StatisticsAgregator* can be used anywhere a *Statistics* entity is used.

Evaluator The *Evaluator* is an abstract entity whose implementation should define a *value* method that returns an integer value according to a match statistics. The existing concrete *Evaluator*s are: GoalDifference, that returns a value corresponding to the difference of goals between the two teams; *Points* evaluator returns a value corresponding the points a team would have won in a championship match (0 for loss, 1 for tie and three for a victory); The *Mars* entity is an implementation of a Mars evaluation function.

5.3 Conclusions

The implemented statistics generation module detects and generates statistics for a match according to the guidelines provided by an expert panel, its accuracy and application will be studied in chapter 7 and chapter 8.
The match generation framework provides a flexible way to generate a batch of matches without supervision. In the context of this work the match generation framework allowed for the generation of both training and validation sets.

Chapter 6

Assistant Coach and Opponent Analysis

In order to exploit the existing opponent model (match statistics) the opponents need to be identified and classified, that is studied in section 6.1. In order to successfully utilize developed modules an assistant coach capable complementing the existing coach during a match was devised (section 6.2).

6.1 Opponents Classification

Opponents classification consists of employing machine learning techniques to classify the opponent behaviour. The goal of this module is to: identify the existing play styles; identify a which play style an opponent is exhibiting during a match;

With that goal in mind the following steps were taken: Discover which statistics should be used to model the opponent (using MARS algorithm); Using the matches generated by the *match generation framework 5.2* the teams were classified using the K-Means algorithm; Using the same matches three classification algorithms —SVM, Random Forest and Bagging— were trained, these allowed the classification of the opponent play style into one of the predefined clusters, using a different variables then those of K-Means.

6.1.1 Statistics

The set of statistics defined and described in 5.1 is used as the opponent model, since the number of goals is insufficient to assess a team performance. Of the calculated statistics, only statistics that represent the entire soccer match were used. To further reduce the total number of statistics, a feature selection algorithm (MARS) was employed, according to the approach documented by Abreu et al. [ASR10]. This algorithm allows the discovery of which statistics mostly influence the final game result. The obtained formula can be seen in the results section.

6.1.2 Generated Matches

The matches were generated by varying the previously presented parameters 4.6 (a total of 32 possible configurations) against three opponents (WrightEagle, Bahia and Nemesis) these three teams were chosen because they occupied distinct positions in the RoboCup 2009 competition score board, giving a bigger spectrum in the final game simulations Each possible strategy-opponent combination was repeated 10 times in order to achieve higher reliability and reduce the effect of outliers. This gives a total of 960 simulations ($32 * 3 * 10$).

6.1.3 Clusters Determination

The K-means [Jai10] algorithm tries to group data in different sets (designated as clusters) according to their similarities. An implementation of the K-means algorithm in R software[1] was used to determine which clusters should be used.

The data set used for classification consists of the goal difference of each team for each match.

One of the problems with using this algorithm is selecting the right K value (number of clusters). Several approaches to this problem exist but the one chosen was a specific mathematical heuristic described by Tibshirani et al. [TWH00] which consists of picking the k value for which $\log \omega_k$ falls the farthest below in the reference curve. The ω_k is defined as 6.1.

$$\omega_k = \sum_{r=1}^{k} \frac{1}{2n_r} D_r \tag{6.1}$$

Where n is the sample size and D is the sum of the pairwise distance for all points in cluster.

6.1.4 Classification Algorithms

The classification algorithms were built using three different algorithms according to the guidelines presented in [MLH03]. The three classification algorithms are Random Forest, SVM, and Bagging.

6.1.4.1 SVM

Support Vector Machines (SVM) represent a group of supervised learning methods with the aim of finding an optimal separating hyperplane so as to separate two classes of patterns with maximal margin [Vap99]. In this research work a kernel method is used as well as 0.02 for gamma value (obtained through experimental results).

6.1.4.2 Random Forest

Random forest is a method for generating tree models through supervised learning created by Breiman in 2001 [Bre01] and was quickly adopted due to its simplicity (training and tunning) and performance [HTFF05].

[1] More informations available at http://www.r-project.org/

6.1.4.3 Bagging

Bootstrap aggregating (bagging) is a machine learning ensemble meta-algorithm to improve machine learning of classification and regression models. Bagging was created by Breiman [Bre96] [Bre98] in 1994, this method combines multiple predictors improving the performance of tree models.

6.1.4.4 Expected Behaviour

A trained classification algorithm should be able to find the cluster defined by the K-means that a team in a match belongs to, but instead of using the goal difference it can only use the statistics that were not filtered out in section 6.1.1.

6.1.5 Training and Validation

Training and validation are done using 10 fold cross-validation. The training/validation data is the same used by the K-means, presented in section 6.1.2 since new data since new data would not contain information about which cluster (defined by the K-means) each team belongs to.

6.2 Assistant Coach

The module referred as the assistant coach is the apogee of the modules described in the previous sections because it allows for the extension of the FC Portugal team making productive use of the information obtained by the other modules. To make it possible to advise our team of the best tactic to face each opponent with a certain play style, it is necessary to map each $(opponent, playstyle)$ to the best *tactic* against it (in this context play style is the same as cluster). With that in mind the existing set of matches was consulted and the mapping was made according to what is described next (section 6.2.1). The classification algorithm and the mapping were included in the statistics module. That Java application was then extended so that it could connect to the FC Portugal coach using a specially developed module that communicates using *UDP/IP* sockets.

6.2.1 Best Tactic Identification

Using the previously trained classification algorithm (section 6.1.4) one can determine the play style of a team, but is unable to decide the best tactic to use. With that in mind three criteria were used (in order):

1. **Goal difference** The tactic that attained the biggest goal difference is chosen.
2. **Number of samples** If several tactics have the same goal difference the tactic that has more matches with that same goal difference is chosen.
3. **MARS** The last criteria is comparing the match score given by the formula obtained by the MARS algorithm (shown in section 7.2.1).

Consulting the generated matches (described in 6.1.2) for each classification algorithm a best tactic was decided for each team-cluster pair.

6.2.2 Advise

During the match at a predefined interval (500, 1000 or 2000 simulation cycles) the coach analyses the opponent statistics and using the best classification algorithm (in the next chapters it is determined that SVM is the best) finds which cluster the opponent corresponds to, consults the mapping for the best tactic to use and communicates it to the players. Since the classification algorithms were trained using the statistics of a complete match the statistics data needs to be scaled to the whole match, for that the equation6.2 is used.

$$S_v/t * D \tag{6.2}$$

Where current value of the statistic is S_v, t is the current simulation cycle and D is the typical duration of a match and the same duration as the matches used for training (usually 6000).

6.3 Conclusions

The classification module should be capable of classifying an opponent according to its match statistics. To achieve such results three different classifications algorithms were used (Bagging, Random Forest, SVM), the results of which is better are presented in the next chapter. The assistant coach was integrated with the FC Portugal coach in order to complement its online advise capabilities. This assistant coach includes both the statistical calculation module as well as the best classification algorithm.

6.3.1 Match Preparation Overview

When facing an opponent, two distinct phases exist: the preparation (offline phase) and the execution (online phase). The result of the offline phase is a classification of the statistics to a cluster and a mapping of $clusters \rightarrow tactics$. The online module uses the statistics module, the classification algorithm and the mapping to advise its players.
An overview of the data flow can be seen in figure 6.1.

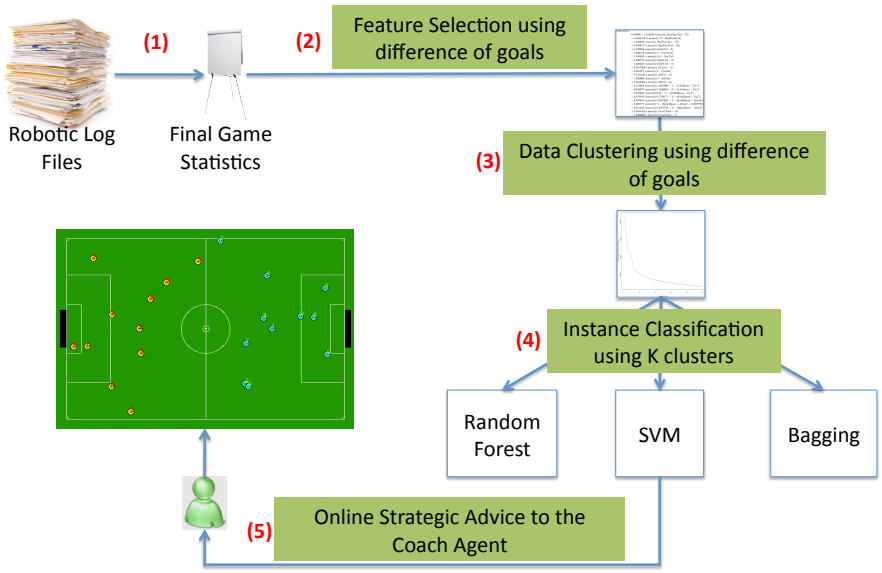


Figure 6.1: Data flow through the existing modules.

Chapter 7

Results

In this chapter the obtained results are presented. The statistics generation module is the first to be introduced, followed by the opponents classification module and finally, the assistant coach. The statistics module was used to analyse the 2009 RoboCup competition logs and was also validated to ensure its proper operation. The opponents classification algorithms results are presented. The assistant coach results are exposed.

7.1 Statistics Generation

The statistics generation module is capable of calculating a set of statistics for each team in a match. This tool is used to analyse the RoboCup 2009 matches and that analysis results are presented. Next, the results of manual validation are presented.

7.1.1 2009 RoboCup Matches Analysis

The developed tool was used to generate the statistics for the teams that participated in the 2009 Soccer Simulation 2D. The obtained results are presented here. This analysis is important to understand the potential of this tool before the other modules are developed. To simplify the analysis only a subset of the calculated statistics is presented here, those are the ones considered to be the most significant.

7.1.1.1 Zone dominance

The zone dominance calculates the average ball possession in each field region per team. In the figure 7.1, the zone dominance of the top three teams in the RoboCup Soccer Simulation 2D 2009 are shown (*Wright Eagle* in red, *Helios20090* in yellow and *Oxsy* in blue respectively). This zone is displayed as if they were attacking in the direction of the arrow shown in the figure (from the bottom to the top of the field). As expected all of the 3 top teams have great control over the

opponent team field, of these three teams *WrightEagle* is the one that also retains more control in its own field, in particular the middle field. This dominance is closely followed by *Helios2009* and at last we have *Oxsy*. This order is the same as the competition results.

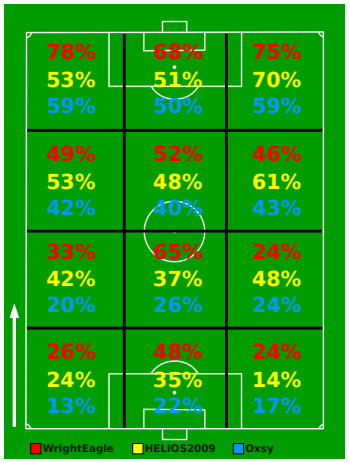


Figure 7.1: Zone Dominance for the top three tournament finalists.

7.1.1.2 Passes

The ratio between the successful and missed passes is calculated and shown in figure 7.2. Regarding that figure, it is easy to note that even the robotic team that presents the lowest ratio (successful versus missed pass) have a percentage of successful pass greater than 67% of the total executed passes. Comparing this data with the final classification of each team in the latest RoboCup competition, it is interesting to see that the team that presents the best ratio is the champion team (Wright Eagle). However the Bahia 2D team, which occupied the last position in the tournament presents only the sixth worst registry. Also it is important to note that *Oxsy*, which occupied the third position in the tournament, did had a good ratio between successful and missed passed (sixth worst mark), similar to *LsuAmoyNQ* which occupied the fifth position in the tournament and presents the second worst mark. This fact could indicate that this statistics *per se* did not influence directly the final match result.

7.1.1.3 Wing variation

The number of wing variations in a soccer match, expresses a collective idea of play executed by a soccer team. Usually this variation is an important tool for creating goal opportunities and can cause confusion in the opponent team. Analysing figure 7.3 the teams *Helios2009*, *WrightEagle*, *OxBlue09* are the one with the largest number wing variations in the competition, which is not surprising because the *Wright Eagle* and *Helios2009* teams occupied the first and the second places in the end of the final RoboCup tournament. This result can also point that this type of strategy can be a successful one in the context of the development of robotic teams.

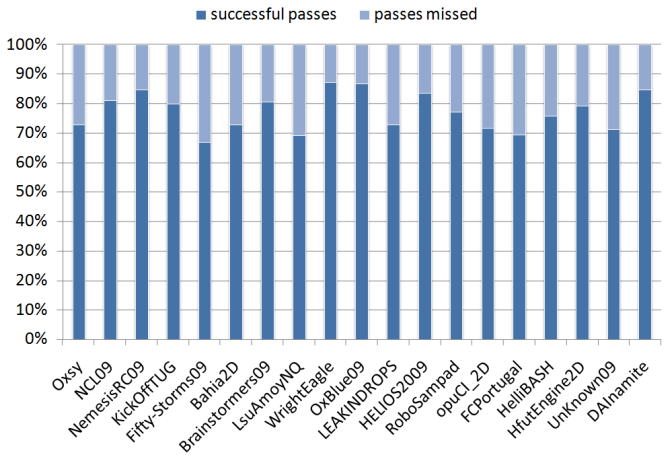


Figure 7.2: Successful to missed passes relation.

7.1.1.4 Temporal Sequence

A temporal sequence is detected when a team has possession of the ball and without losing it, advances in the field with the direction of the opponent goal until achieve a specific zone called "the last third of the field". This sequence is essentially an attack and its classification is done according to the difference between the time it took for the team achieve the zone mentioned before and the time that the same team recovered the ball. If for some reason a team loses the ball before these conditions are met the sequence will be classified as *Broken*.

The figure 7.4 shows the calculus of temporal analysis for the top four teams of the tournament. It is clear to note that the main difference between the top three teams is to the number of break sequences over the competition. This indicator can represent that the first team in the tournament (Wright Eagles) has a huge percentage of success in terms of attack in comparison to other teams. Also it is interesting to note that the team in the fourth place team *Brainstormers* presents the lowest value of *Medium Sequence* which is a peculiar characterization of its game.

7.1.1.5 Goal opportunities

The winner of a soccer match is determined according to the final goal difference (scored - conceded). The team that presents the higher goal difference will be acclaimed as the winner of the match. In order to achieve that primary goal, the creation of goal opportunities constitutes a good indicator for a coach to measure his team performance. Observing the figure 7.5) one can check that three of the four teams that have a large number of goal opportunities, were the three finalist of the tournament and two of them present the higher $(NGoals)/(NOpportunities)$ ratio (up to 40%) in figure 7.6. Analysing the same figure it is curious to note that the team that presents the best goal success ratio was ranked with the ninth place in tournament (*FiftyStorm team*) which means that in spite of having a higher success rate, the number of goal opportunities is still small when compared to other teams.

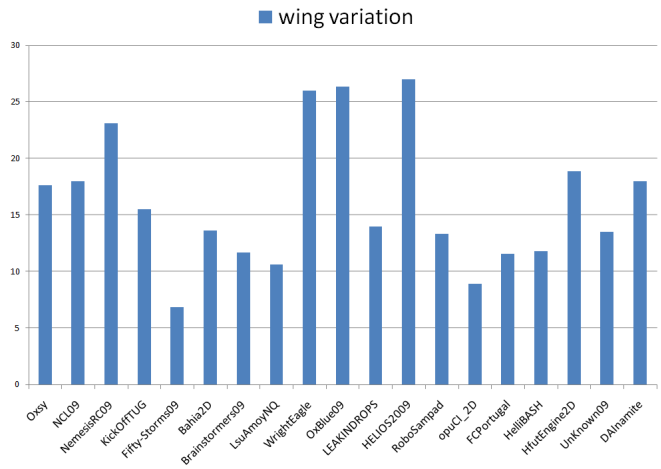


Figure 7.3: Average wing variation per game.

7.1.2 Validation

The statistics module required manual validation to confirm that the detected events exhibited the desired behaviour. With that in mind random match samples were selected from the RoboCup 2009 competition. This approach only used small match samples. When compared with validations using larger data sets yields less accurate results. Yet it was preferred because it allowed for a more granular and precise analysis of the results and consequently allows for better description of the encountered flaws.

7.1.2.1 Randomly selected samples

As previously stated the samples were randomly selected from the RoboCup 2009 competition matches. The selection results are as follows:

- The match selected to detect the **passes** was the final between *WrightEagle* and *HELIOS2009* (match id was 200907051300). The time windows was 1000 cycles and the selected cycles were from 0 to 1000.

- The **ball possession** was validated in the interval 746 to 1346 (600 size) in the Second Round match between *UnKnown09* and *HELIOS2009* (match id 200907021055).

- The **goal misses** were validated using the match that opposed *NCL09* and *LEAKINDROPS* (match id 200907041600), in the interval $[2930, 5930]$ (3000 cycles in size).

- To validate the **attacks** a time frame of size 600 was analysed, corresponding to the match that opposed *NemesisRC09* and *HelliBASH* (match id 200907041817) in the time interval 1805 to 2405.

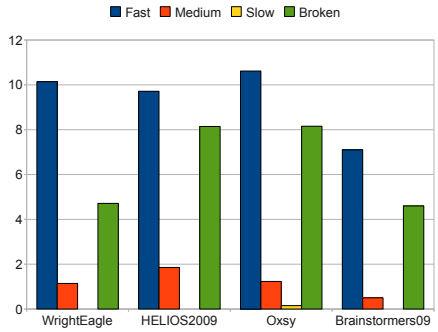


Figure 7.4: Temporal Sequence of the top four Teams.

- Finally, the **goal opportunities** were validated using the log file from the match that opposed *HfutEngine2D* and *opuCI* (id 200907041626). The time interval corresponded to $[4295, 5295]$ (1000 cycles).

7.1.2.2 Passes

The passes detection contain very few errors, the situations that could not be properly detect were:

1. In some situations where two players of opposing teams kicked the ball at the same time the algorithm was unable to consider a pass if one of the players succeeded at passing.

2. Passes from goal kick are not accounted for because the start before the play mode is $PLAY\_ON$.

3. When two players of opposing teams receive the ball. This one is harder to detect than when two player kick the ball because it is necessary to check which of the receivers kept the ball.

7.1.2.3 Ball Possession

The ball possession algorithm also contains very few errors. The ones detected were:

1. Since the algorithm only counts ball possession during for $PLAY\_ON$, when the goalie passes the ball to a player the small amount of time that the game is $PLAY\_ON$ but the ball hasn't reached the player yet is counted as $NEUTRAL$ instead of the goalie team.

2. In some situations where players of both teams kicked the ball a human would count those as $NEUTRAL$ and the algorithm would not.

3. The reverse of the previous is also true: in situations where opposing players kick the ball but the ball possession did not change because they did not lose control of the ball, the algorithm may count as $NEUTRAL$.

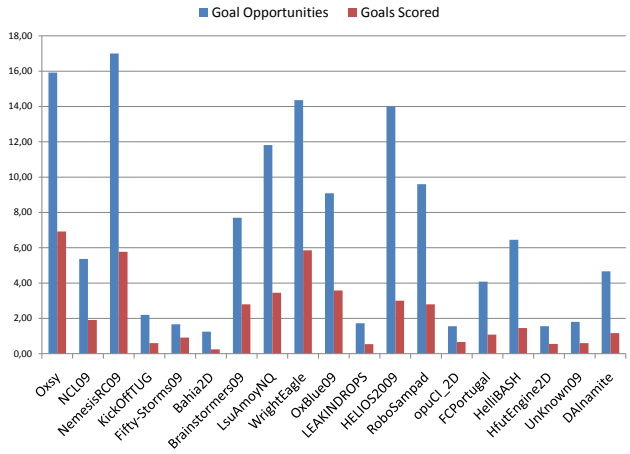


Figure 7.5: Goals and goal opportunities, per game (average).

7.1.2.4 Goal Misses

The goal misses also has very few small flaws. The detected ones are:

1. Doesn't account for situations where the ball was blocked by an opposing player instead of the goalie.

2. The situation where the ball hits the one of the goal poles does not count as goal miss (and should).

7.2 Opponents Classification

In this section the opponents classification module results are presented. The algorithms performance is validated as part of the training process.

7.2.1 MARS

The formula obtained by the MARS algorithm is presented 7.1, as previously mentioned only final game statistics were used. The variables in this expression are: the total number of bad passes (*BadPassTot*), the number of goals (*GoalsTot*), number of outside (*OutTot*), the number of attacks (*AttTot*), the number of pass chains (*PassChain*) and the ball possession by zones (1 –

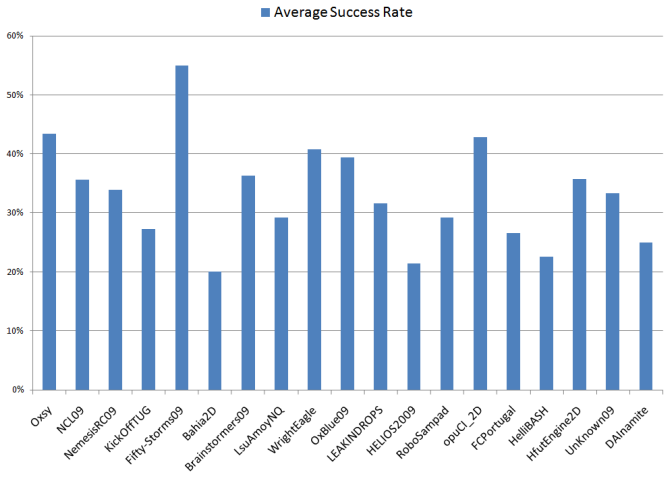


Figure 7.6: Goal to opportunity ratio.

$LeftBposs-Def$, $2-MiddBposs-Def$, $3-RightBposs-Attack$ etc.).

$$
\begin{aligned}
&3.853966 \\
&+1.513163 \times max(0, BadPassTot - 75) \\
&+0.09444147 \times max(0, 75 - BadPassTot) \\
&-1.533902 \times max(0, BadPassTot - 70) \\
&+0.3496171 \times max(0, BadPassTot - 62) \\
&+1.048082 \times max(0, GoalsTot - 2) \\
&-1.626119 \times max(0, 2 - GoalsTot) \\
&-0.1608921 \times max(0, 10 - OutTot) \\
&+2.298775 \times max(0, GoalKick - 3) \\
&-1.620533 \times max(0, GoalKick - 2) \\
&+0.9517624 \times max(0, OffInt - 2) \\
&-2.021077 \times max(0, 2 - FasAtt) \\
&-0.7101195 \times max(0, AttTot - 6) \\
&-1.200986 \times max(0, 6 - AttTot) \\
&+0.7653595 \times max(0, AttTot - 9) \\
&-10.27804 \times max(0, 0.1337386 - \text{“}1 - LeftBposs - Def\text{”}) \\
&-29.29977 \times max(0, 0.1439394 - \text{“}2 - LeftBposs - Def\text{”}) \\
&-2.455354 \times max(0, 0.55 - \text{“}1 - MiddBposs - Def\text{”}) \\
&-32.53045 \times max(0, 0.1794171 - \text{“}2 - MiddBposs - Def\text{”}) \\
&-6.037467 \times max(0, 0.2307692 - \text{“}4 - MiddBposs - Attack\text{”}) \\
&-2.886777 \times max(0, \text{“}3 - RightBposs - Attack\text{”} - 0.2977778) \\
&-8.811459 \times max(0, 0.2977778 - \text{“}3 - RightBposs - Attack\text{”}) \\
&+0.3035423 \times max(0, PassChain - 13) \\
&-0.2490695 \times max(0, PassChain - 7)
\end{aligned}
\tag{7.1}
$$

GCV : 10.1274 RSS : 3129.240 $GRSq$: 0.7827846 RSq : 0.8317884

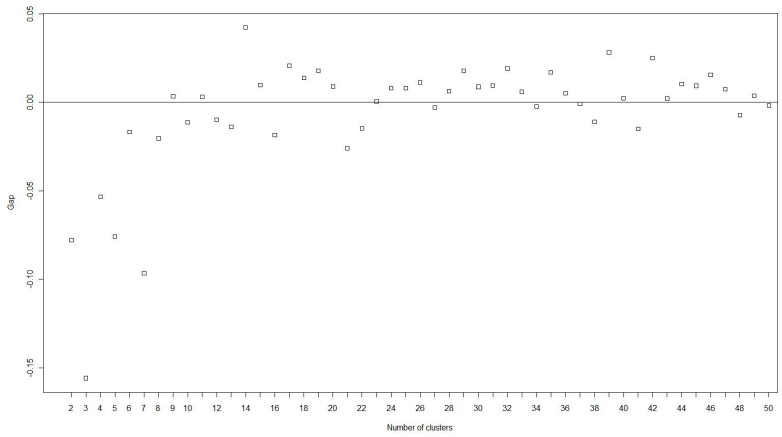


Figure 7.7: Calculate the optimal number for K using the Gap algorithm

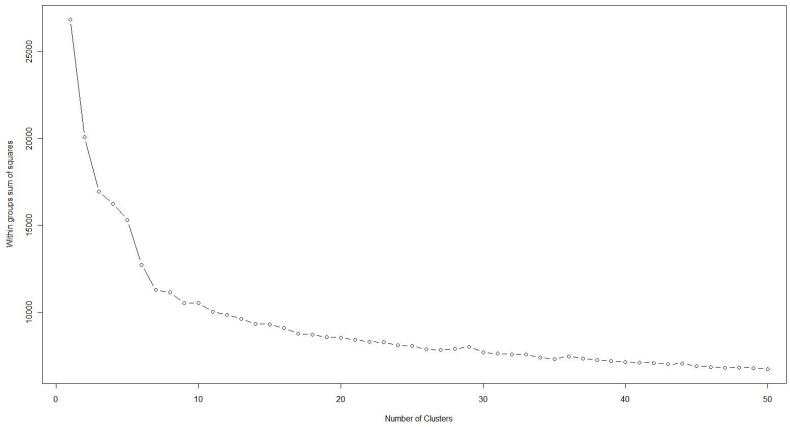


Figure 7.8: The number of clusters with groups of sum of squares

It can be seen in the results that the coefficient of determination is higher that 80% (0.8317884).

7.2.2 Clusters Determination

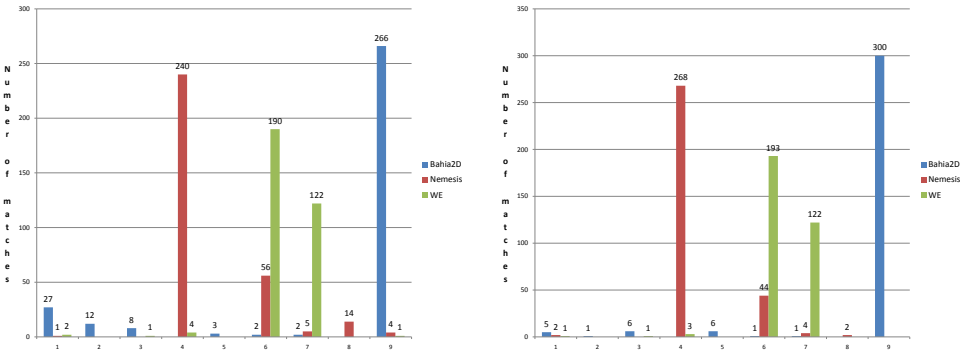
The Gap algorithm was used to determine the optimal number of clusters (figures 7.7, 7.8), the result indicated that **nine** clusters should be used. Subsequently, the K-means algorithm was used to classify the generated data to those 9 different clusters.

7.2.3 Classification Algorithms

The results of the classification algorithms (SVM, RandomForest and Bagging)training are presented in table 7.1. All of the algorithms use a low number of clusters to classify the WrightEagle team strategy, this is not unexpected because this team is the world champion and can win its opponent without changing its strategy much.

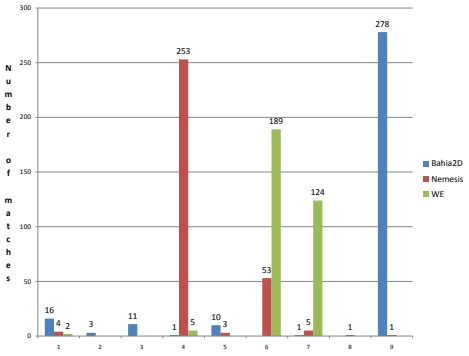
Table 7.1: The number of clusters with groups of sum of squares

	Bagging				Random Forest				SVM			
	Number of Clusters	Median of Games per Cluster	Average of Repeated Games per cluster	Average of Excluded Strategies per Cluster	Number of Clusters	Median of Games per Cluster	Average of Repeated Games per cluster	Average of Excluded Strategies per Cluster	Number of Clusters	Median of Games per Cluster	Average of Repeated Games per cluster	Average of Excluded Strategies per Cluster
Bahia 2D	7	8	4,57	22	7	5	6,53	25	7	10	5,08	23
Nemesis	6	9,5	4,10	19	5	4	5,24	19,8	7	4	4,57	23
WrightEagle	6	3	4,44	20	5	3	4,64	18,2	4	64,5	4,57	14.5



(a) Bagging distribution of matches per cluster

(b) Random Forest distribution of matches per cluster



(c) SVM distribution of matches per cluster

Figure 7.9: Matches distribution by cluster for each algorithm.

Even though the number of clusters is not the same for all three algorithms it is possible to observe that all of them identified one main cluster for both Nemesis and Bahia and two main clusters for WrightEagle (figure 7.9).

The bagging algorithm is the one that presents the highest number of clusters to classified the data. Naturally, its median presents a smaller number of repeated matches per cluster. In terms of excluded games (related to the non use of all combinations (32) in each cluster) all three techniques were very similar and finally regarding the median of games per cluster the technique that presents the highest number was the SVM (in concerns to the Nemesis team, the Bagging

algorithm presented the highest number of matches per cluster (in median)).

Regarding the rate of the results when compared with the expected clusters determined by the K-means, SVM is the one that has the highest result (96.4%), followed by Random Forest (93.59) and Bagging (80.151%).

The statistical validation of these results was done using the Friedman rank test with the statistic derived by Iman and Davenport as described by Demsar [Dem06]. The *null* hypothesis of equivalence between the three predictors is rejected with a $p-value$ of 0.038.

The 960 tested games (1920 rows, because each match has two teams) were divided in 12 groups. For each group an algorithm ranking is performed, ordering the algorithms by percentage rate (results in table 7.2).

Comparing the three predictors with a 5% significance level using the Nemenyi test [Dem06], a *CD* of 0.96 was obtained. The *CD* is the critical value for the difference of mean ranks between the three predictors. Observing the average value obtained by the three predictors one can see that these predictors have significant statistical differences. The only difference between the analysed predictors smaller than the critical difference is between the SVM and Random Forest algorithms (only 0.8334). The difference between the other pairs Bagging/SVM and Bagging/Random Forest is 1.9667 and 1.08333, respectively.

Table 7.2: Ranks for the Friedman test of the opponents classification.

	1	2	3	4	5	6	7	8	9	10	11	12	Means
Random Forest	2	2	2	2	2	2	2	2	2	1	2	2	1.91667
SVM	1	1	1	1	1	1	1	1	1	2	1	1	1.08333
Bagging	3	3	3	3	3	3	3	3	3	3	3	3	3

7.3 Assistant Coach

When the assistant coach was properly integrated with the FC Portugal code, the results needed to be validated. With that in mind 40 simulations were run for each advise interval. The classification algorithm used was SVM. The chosen intervals are: 500; 1000; and 2000 simulation cycles. This means that there are 9 (3×3) possible combinations. The matches generation framework (section 5.2) was once again used to generate the 360 (9×40) required matches. Since a control group is also necessary as a base for comparison another 120 matches ($40 \times against each team$) were also simulated.

After the simulations finished the matches were divided into 12 groups according to the opponent (4 groups for each opponent). The ranks obtained can be seen in table 7.3. The *null* hypothesis of equivalence between the 4 predictors is rejected with a $p-value$ of 0.01665. Comparing the 3 predictors against the baseline/control group for a 5% significance level using the Bonferroni-Dunn [Dem06] the *CD* is 1.26. Since the difference of the averages of each pair is smaller than *CD* it is not possible to conclude if any of the algorithms is better or worse than the control group.

Table 7.3: Ranks of the online results.

	1	2	3	4	5	6	7	8	9	10	11	12	Mean
Baseline	2	2	1	1	4	1	2	1	4	4	4	2	2,33
SVM500	4	3	2	2	2	2	3	4	1	3	3	1	2,50
SVM1000	1	1	4	4	3	4	4	3	2	2	1	4	3,00
SVM2000	3	4	3	3	1	3	1	2	3	1	2	3	2,42

7.4 Conclusions

The obtained results are analysed according to well know methods to ensure that they are valid. In the next chapter the conclusions that come from the analysis of these results are presented, explaining the data obtained here.

Chapter 8

Conclusions and Future work

During the course of this thesis several modules were developed, each of those providing its unique contribution to its work. The match generation framework made it possible to run simulations in order to obtain the much needed match logs. The statistics module provided an opponent model based on the match logs. When the opponents needed to be classified the methods for such classification were 6.1 studied. And finally the assistant coach was able to connect to a match and provide online advice to its team.

In this chapter the conclusion on the role of each of those components are drawn as well as their individual performance.

8.1 Statistics Generation

Based on the previously presented results some conclusions are taken. The purpose of this module was to detect game events present in the game logs and generate the corresponding statistics. This tool is capable of calculating soccer events to help soccer coaches improving their teams performance. The set of statistics are defined by a group of sports researchers and the test data used is the RoboCup 2009 tournament – soccer simulation 2D in particular – logs. Similar to other research studies [CPHMMS+07] in order to detect all of the events, a sequential analysis method was used and proved itself as a good approach for this particular environment. In the next sections (8.1.1 8.1.2) the conclusions regarding the matches analysis and the validation are discussed.

8.1.1 2009 Matches Analysis

Regarding the results obtained previously it is important to note that even some of the most simple statistics seem to yield important clues to a way a team plays or some of the characteristics it could improve. One of such statistics is the goal opportunities versus goal scored, in this score the top teams present excellent results. Some other teams such as *Fifty Storm* and *OPUCI_2D* in spite of having a good ratio, still need to improve their creation of goal opportunities over the game. The

field zone dominance statistics of the three leading teams suggests that dominating the opposing team field is a must, but what seem to set them apart from each other is the ability to also control their own field. Finally, from the sequence analysis point of view the observed results suggest that the fast attacks are the most important of the bunch. The low number of broken attacks of the *WrightEagle* team also point out that successfully reaching the opponents field can be a distinction factor.

It was also curious to note that some important statistics like successful passes to pass misses relation do not seem to demonstrate, by itself, any relation to the final results of the competition. Possible interpretations for this fact could be that the success of the passes is already so high for every team that it loses its' importance or that the statistics should be complemented with further contextual information.

8.1.2 Validation

Regarding the results obtained during the validation process, some unexpected flaws were discovered. The discovery of those flaws was very fortunate because it allows for further improvement of the tool.
Even when accounting for the identified flaws the statistics generation tool provides very accurate results and can be used as is.

8.2 Matches generation framework

This research work demanded matches data for both training and validation. To obtain those matches many simulations were run using different parameters and opponents. To solve that problem the matches generation framework was developed. This framework is programmable, configurable and also capable of automating the task of running the much needed simulations, those characteristics made it invaluable during the course of this work.

In conclusion, the match generation framework is a very useful tool that was even used by the FC Portugal team during its preparation for the 2010 RoboCup competition and hopefully will continue to facilitate peoples work.

8.3 Opponents Classification

As part of this research work, an approach to classify opponent soccer teams using final game statistics was presented. This work is important because it allows for a better informed analysis of the opponent teams.

At the beginning of this work three robotic soccer teams that participated in RoboCup 2009 competition were chosen, according to their results in that same competition: better, similar and worst results than FC Portugal. Then, the MARS algorithm was used to select the most significant statistics (using as a target function the goals difference). After running many simulations using

the simulation framework – 960 simulations varying team strategy and opponents – the K-means algorithm was used to identify the opponent clusters (strategies), and after that three induction algorithms were used. The results obtained proved that it is possible to classify team behaviour using classification. Finally, SVM proved to be the algorithm more able to classify the opponent team behaviours.

8.4 Assistant Coach

As part of this thesis, an approach to an assistant coach capable of providing online advice to its players, based on simple match statistics, was presented. This assistant coach was tested and some results were obtained. These are discussed next.

Since the *null* hypothesis was rejected one can conclude that the novel approach differs, in some way, from the control group (the original FC Portugal team). Unfortunately, using the currently available data it is not possible to conclude whether this novel approach offers a performance gain or loss.

It was expected by the author that this approach offered undeniable improvements to the original FC Portugal team, these test results show, at least, that if those improvements do exist, they fall short from the expectations. The possible reasons for this are explored next:

1. The tactic selection to face a team exhibiting a particular strategy (mapped to a particular cluster by the classification algorithm) may have been severely affected by outliers, since it did not account for the expected value of the match result neither standard deviation of that result.

2. The chosen data set to create the clusters using the K-means algorithm was known to have great influence in the match result. It may also happen that even though it greatly influences the match result it may not offer adequate partitioning when trying to exploit opponent behaviour.

3. Another possibility is that the parameters chosen to vary (formation and setplays) may not offer significant differences in the match results. Meaning that no matter how well they are changed, it remains impossible to improve the match results.

4. The possibility that human error affected the process somewhere along the line and that such error remained undetected cannot be discarded.

5. At last remains the possibility that the chosen high-level statistics cannot be used to identify opponent behaviour in a way that can be properly exploited to adjust the chosen strategy parameters.

The possible reasons explored here are little more than informed guesses but can be valuable when accounting for future work.

8.5 Future Work

In this section the future work is discussed. The first thing to address in the future work would be to explore the listed possibilities of why the obtained results fall short from the expectations. Even though the previously defined list is sorted by likelihood, the first thing to test would be that the chosen parameters actually offer significant differences in the match results, since it is highly testable.

Regarding the statistics generation module, in the future it could be applied to human soccer in order to obtain interesting match statistics for those games–surely adaptation to be used in real soccer will be needed–and more statistics could be calculated in order to identify even better statistics. Still regarding the statistics generation module the encountered flaws should be fixed in order to improve it.

For the classification and the assistant coach modules more opponents could be used instead of just three. Another interesting addition would be changing parameters other than just formation and setplays.

The expansion of this work to another sport or even another area would provide new and interesting challenges.

A slight change in the problem by not using the opponent team binaries but just logs of public matches it played, would increase the difficulty of this work but also its value.

References

[Abr10] Pedro Abreu. Metodologias de inteligência artificial aplicadas na simulação de jogos desportivos colectivos. Work in progress for Faculty of Engineering, University of Porto, 2010.

[Aki] Hidehisa Akiyama. *Users Manual/Soccer Server*. Available at http://sourceforge.net/apps/mediawiki/sserver/index.php?title=Users_Manual/Soccer_Server.

[AMSV07] Pedro Abreu, Pedro Mendes, Daniel Castro Silva, and Vasco Vinhas. Microcredit practices applyed to e-supply chain management. In *Proceedings of IADIS International conference on e-Commerce 2007)*, pages 91–98, 2007.

[ASR10] Pedro Abreu, Daniel Castro Silva, and Paulo Reis. Using Multivariate Adaptive Regression Splines in the Construction of Simulated Soccer Team's Behavior Models. 2010.

[BD04] J. Broekens and D. DeGroot. Emotional agents need formal models of emotion. In *Proceedings of the 16th Belgian-Dutch Conference on Artificial Intelligence (BNAIC 2004, Groningen, The Netherlands)*, pages 195–202, 2004.

[Bre96] Leo Breiman. Bagging predictors. *Mach. Learn.*, 24(2):123–140, 1996.

[Bre98] Leo Breiman. Arcing classifiers. *The Annals of Statistics*, 26(3):801–849, 1998.

[Bre01] Leo Breiman. Random forests. In *Machine Learning*, pages 5–32, 2001.

[CFH+] Mao Chen, Ehsan Foroughi, Fredrik Heintz, ZhanXiang Huang, Spiros Kapetanakis, Kostas Kostiadis, Johan Kummeneje, Itsuki Noda, Oliver Obst, Pat Riley, Timo Steffens, Yi Wang, and Xiang Yin. *Users manual: RoboCup soccer server manual for soccer server version 7.07 and later.* Available at http://sourceforge.net/projects/sserver/files/.

[CLR07a] João Certo, Nuno Lau, and Luis P. Reis. A generic multi-robot coordination strategic layer. In *RoboComm '07: Proceedings of the 1st international conference on Robot communication and coordination*, pages 1–7, Piscataway, NJ, USA, 2007. IEEE Press.

[CLR07b] João Certo, Nuno Lau, and Luis P. Reis. A generic strategic layer for collaborative networks. In *IFIP, International Federation for Information Processing, Volume 243, Establishing the Foundations for Collaborative Networks*, pages 273–282, 2007.

REFERENCES

[Com06] The RoboCup 2006 Simulation League Organizing Committee. *RoboCup 2006 Official Rules for the Coach Competition*, 2006.

[Com07] The RoboCup 2007 Simulation League Organizing Committee. *Rules Soccer Simulation League 2D RoboCup2007 Atlanta*, 2007.

[CPHMMS+07] J. Castellano-Paulis, A. Hernandez-Mendo, Morales-Sanchez, Veronica, and M. Anguera-Argilaga. Optimising a probabilistic model of the development of play in soccer. *Quality and Quantity*, 41(1):93–104, February 2007.

[CW01] André L. V. Coelho and Daniel Weingaertner. Evolving coordination strategies in simulated robot soccer. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, pages 147–148, New York, NY, USA, 2001. ACM.

[Dem06] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, 2006.

[dFA10] Fédération Internationale de Football Association. *Laws of the Game 2010/2011*, 2010.

[Dun99] Eric Dunning. *Sport matters: sociological studies of sport, violence, and civilization*. Taylor and Francis, 1999, 1999.

[FMMM06] Ramin Fathzadeh, Vahid Mokhtari, Morteza Mousakhani, and Fariborz Mahmoudi. Mining Opponent Behavior: A Champion of RoboCup Coach Competition. In *Robotics Symposium, 2006. LARS '06. IEEE 3rd Latin American*, pages 80–83, Oct. 2006.

[GM05] Jonathan Gratch and Stacy Marsella. Evaluating a computational model of emotion. *Autonomous Agents and Multi-Agent Systems*, 11(1):23–43, 2005.

[HCH+02] Jafar Habibi, Ehsan Chiniforooshan, A. HeydarNoori, M. Mirzazadeh, MohammadAli Safari, and HamidReza Younesi. Coaching a soccer simulation team in robocup environment. In *EurAsia-ICT '02: Proceedings of the First EurAsian Conference on Information and Communication Technology*, pages 117–126, London, UK, 2002. Springer-Verlag.

[Hil01] Katrin Hille. Synthesizing emotional behavior in a simple animated character. *Artif. Life*, 7(3):303–313, 2001.

[HTFF05] Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27:83–85, 2005.

[Jai10] Anil K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, June 2010.

[JW98] Nicholas R. Jennings and Michael J. Wooldridge. Applications of intelligent agents. http://www.cs.umbc.edu/agents/introduction/jennings98.pdf, 1998. [Online; accessed June-2010].

REFERENCES

[KAK+95] H. Kitano, M. Asada, Y. Kuniyoshi, L Noda, and Osawa E. Robocup: The robot world cup initiative. *Proceedings of UCAI'95 Workshop on Entertainment and AI/Alife*, 1(1):19–24, 1995.

[Kit97] H. Kitano. RoboCup: The road cup initiative. *Proceedings of the 1st International Conference on Autonomous Agent (Agents'97), Marina del Ray*, 1(1), 1997.

[KKS06] Gregory Kuhlmann, William B. Knox, and Peter Stone. Know thine enemy: A champion RoboCup coach agent. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, pages 1463–68, July 2006.

[KSL05] Gregory Kuhlmann, Peter Stone, and Justin Lallinger. The UT Austin Villa 2003 champion simulator coach: A machine learning approach. In Daniele Nardi, Martin Riedmiller, and Claude Sammut, editors, *RoboCup-2004: Robot Soccer World Cup VIII*, volume 3276 of *Lecture Notes in Artificial Intelligence*, pages 636–644. Springer Verlag, Berlin, 2005.

[KT01] Hiroaki Kitano and Satoshi Tadokoro. RoboCup rescue: A grand challenge for multiagent and intelligent systems. *AI Magazine*, 22(1):39–52, 2001.

[LASB04a] Agapito Ledezma, Ricardo Aler, Araceli Sanchis, and Daniel Borrajo. Predicting opponent actions by observation. In *In*, pages 286–296. Springer, 2004.

[LASB04b] Agapito Ledezma, Ricardo Aler, Araceli Sanchís, and Daniel Borrajo. Predicting opponent actions by observation. In Daniele Nardi, Martin Riedmiller, Claude Sammut, and José Santos-Victor, editors, *RobuCup*, volume 3276 of *Lecture Notes in Computer Science*, pages 286–296. Springer, 2004.

[lotUoEC] Takeuchi laboratory of the University of Electro-Communications. Soccerscope 2 - visual debugger and log player. `http://ne.cs.uec.ac.jp/~koji/SoccerScope2/index.htm`. [Online; accessed June-2010].

[LR07] Nuno Lau and Luís Paulo Reis. FC Portugal - High-level Coordination Methodologies in Soccer Robotics. In *Robotic Soccer*, pages 167–192. I-Tech Education and Publishing, 2007.

[MLH03] David Meyer, Friedrich Leisch, and Kurt Hornik. The support vector machine under test. *Neurocomputing*, 55(1-2):169–186, 2003.

[MR07a] Luís Mota and Luís Paulo Reis. Setplays: achieving coordination by the appropriate use of arbitrary pre-defined flexible plans and inter-robot communication. In *RoboComm '07: Proceedings of the 1st international conference on Robot communication and coordination*, pages 1–7, Piscataway, NJ, USA, 2007. IEEE Press.

[MR07b] Luís Mota and Luís Paulo Reis. Setplays: achieving coordination by the appropriate use of arbitrary pre-defined flexible plans and inter-robot communication. In *RoboComm '07: Proceedings of the 1st international conference on Robot communication and coordination*, pages 1–7, Piscataway, NJ, USA, 2007. IEEE Press.

[MR08] Luís Mota and Luís Paulo Reis. A common framework for co-operative robotics: An open, fault tolerant architecture for multi-league robocup teams. In *SIMPAR '08: Proceedings of the 1st International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, pages 171–182, Berlin, Heidelberg, 2008. Springer-Verlag.

[MRdB06] Luís Mota, Luís Paulo Reis, and Hans dieter Burkhard. Communication challenges raised by open co-operative teams in robocup. In *Proceedings of the Scientific Meeting of the Portuguese Robotics Open 2006*. Estela Bicho et al. (eds.), 2006.

[MSK+06] Kazuyuki Murase, Kosuke Sekiyama, Naoyuki Kubota, Tomohide Naniwa, and Joaquin Sitte, editors. *Proceedings of the 3rd International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE 2005), Awara-Spa, Fukui, Japan, September 20-22, 2005*. Springer, 2006.

[NKTN05] Manabu Nii, Makoto Kajihara, Yutaka Takahashi, and Tomoharu Nakashima. An action rule discovery technique from simulated roboCup soccer logs. In Murase et al. [MSK+06], pages 81–86.

[Rei03] Luís Paulo Reis. *Coordenação em Sistemas Multi-Agente: Aplicações na Gestão Universitária e Futebol Robótico-PhD Thesis*. PhD thesis, Faculty of Engineering, University of Porto, 2003.

[RHLL08] Martin A. Riedmiller, Roland Hafner, Sascha Lange, and Martin Lauer. Learning to dribble on a real robot by success and failure. In *ICRA*, pages 2207–2208. IEEE, 2008.

[RL] Luís Paulo Reis and Nuno Lau. Homepage of FC Portugal. http://www.ieeta.pt/robocup/. [Online; accessed June-2010].

[RL01] Luís Paulo Reis and Nuno Lau. FC Portugal team description: RoboCup 2000 Simulation League Champion. In *RoboCup 2000: Robot Soccer World Cup IV*, pages 29–40, London, UK, 2001. Springer-Verlag.

[RL02] Luís Paulo Reis and Nuno Lau. Coach unilang - a standard language for coaching a (robo)soccer team. In *RoboCup 2001: Robot Soccer World Cup V*, pages 183–192, London, UK, 2002. Springer-Verlag.

[RLM10] Luís Paulo Reis, Nuno Lau, and Luís Mota. FC portugal 2d simulation: Team description paper. 2010.

[RLML10] L. Reis, R. Lopes, L. Mota, and N. Lau. Playmaker: Graphical definition of formations and setplays. In *CISTI 2010*, pages 582–587, 2010.

[RLO01] Luís Paulo Reis, Nuno Lau, and Eugenio Oliveira. Situation based strategic positioning for coordinating a team of homogeneous agents. In *Balancing Reactivity and Social Deliberation in Multi-Agent Systems, From RoboCup to Real-World Applications (selected papers from the ECAI 2000 Workshop and additional contributions)*, pages 175–197, London, UK, 2001. Springer-Verlag.

REFERENCES

[RMM+01] Martin A. Riedmiller, Artur Merke, David Meier, Andreas Hoffman, Alex Sinner, Ortwin Thate, and R. Ehrmann. Karlsruhe brainstormers - a reinforcement learning approach to robotic soccer. In *RoboCup 2000: Robot Soccer World Cup IV*, pages 367–372, London, UK, 2001. Springer-Verlag.

[RN03] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003.

[Rob] RoboCup. RoboCup: What is RoboCup. http://www.robocup.org/about-robocup/. [Online; accessed June-2010].

[RVK02] Patrick Riley, Manuela Veloso, and Gal Kaminka. An empirical study of coaching. In H. Asama, T. Arai, T. Fukuda, and T. Hasegawa, editors, *Distributed Autonomous Robotic Systems 5*, pages 215–224. Springer-Verlag, 2002.

[Sil02] B. Silverman. Human behavior models for game-theoretic agents: Case of crowd tipping. *Cognitive Science Quarterly*, 2(3/4):18, 2002.

[Sin] RoboCup 2010 Singapore. RoboCup 2010 Singapore. http://www.robocup2010.org/. [Online; accessed June-2010].

[SM05] D. Shell and M. Mataric. Behavior-based methods for modeling and structuring. In *Cognition and Multi-Agent Interaction)*, pages 279–306. Cambridge University Press, 2005.

[SPS04] Elizabeth Sklar, Simon Parsons, and Peter Stone. Using roboCup in university-level computer science education. *J. Educ. Resour. Comput.*, 4(2):4, 2004.

[SRV00] Peter Stone, Patrick Riley, and Manuela Veloso. Defining and using ideal teammate and opponent agent models: A case study in robotic soccer. *Multi-Agent Systems, International Conference on*, 0:0441, 2000.

[SV99] Peter Stone and Manuela Veloso. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork. *Artif. Intell.*, 110(2):241–273, 1999.

[TWH00] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the Number of Clusters in a Dataset via the Gap Statistic. *Journal of the Royal Statistical Society, Series B*, 63:411–423, 2000.

[Vap99] Vladimir Vapnik. *The Nature of Statistical Learning Theory (Information Science and Statistics)*. Springer, 2nd edition, November 1999.

[Wei99] Gerhard Weiss, editor. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1999.

[Woo01] Michael Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons, Inc., New York, NY, USA, 2001.

[YNN+05] Satoshi Yokoyama, Naoki Namikawa, Tomoharu Nakashima, Masayo Udo, and Hisao Ishibuchi. Developing a goal keeper for simulated roboCup soccer and its performance evaluation. In Murase et al. [MSK+06], pages 75–80.

REFERENCES