ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE
ENGINEERING AND TECHNOLOGY

DISTRIBUTED TEAM FORMATION
FOR ROBOT SOCCER

M.Sc. THESIS

Onuralp ULUSOY

Department of Computer Engineering

Computer Engineering Programme

JANUARY 2012

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

DISTRIBUTED TEAM FORMATION FOR ROBOT SOCCER

M.Sc. THESIS

Onuralp ULUSOY
(504091521)

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor: Asst. Prof. Dr. Sanem SARIEL-TALAY

JANUARY 2012

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ROBOT FUTBOLU İÇİN
DAĞITILMIŞ TAKIM FORMASYONU

YÜKSEK LİSANS TEZİ

Onuralp ULUSOY
(504091521)

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Yrd. Doç. Dr. Sanem SARIEL-TALAY

OCAK 2012

Onuralp Ulusoy, a **M.Sc.** student of ITU Graduate School of Science Engineering and Technology student ID 504091521, successfully defended the thesis entitled "**DISTRIBUTED TEAM FORMATION FOR ROBOT SOCCER**", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor : **Asst. Prof. Dr. Sanem SARIEL-TALAY** ....
İstanbul Technical University

Jury Members : **Asst. Prof. Dr. Sanem SARIEL TALAY** ....

Asst. Prof. Dr. Hatice KÖSE-BAĞCI ....

Asst. Prof. Dr. Gülay ÖKE ....

Date of Submission : 19 December 2011
Date of Defense : 26 January 2012

FOREWORD

I would like to express my gratitude to my thesis advisor Asst. Prof. Dr. Sanem Sarıel-Talay for her interest, guidance and encouragement during the development of this thesis. I would like to thank my family for always being there for me and supporting all of my endeavours throughout the years. I am also grateful to team members of beeStanbul RoboCup 3D soccer simulation team, for their friendship and support during my research.

I would also like to thank to The Scientific and Technological Research Council of Turkey (TÜBİTAK) for its financial support via the National Scholarship Programme for M.Sc. Students.

December 2011 Onuralp ULUSOY

TABLE OF CONTENTS

Page

ABBREVIATIONS

CBR	**:**	Case-Based Reasoning
DR	**:**	Dominant Region
DPVC	**:**	Dynamic Positioning based on Voronoi Cells
FA	**:**	Fortune Algorithm
MAS	**:**	Multi-Agent Systems
PFS	**:**	Partial Fourier Series
PNP	**:**	Petri Net Plans
SBSP	**:**	Situation Based Strategic Positioning
SPL	**:**	Standard Platform League
SSL	**:**	Soccer Simulation League

LIST OF TABLES

Page

LIST OF FIGURES

Page

DISTRIBUTED TEAM FORMATION
FOR ROBOT SOCCER

SUMMARY

The field of multi-agent systems (MAS) has been an active research area for the last two decades. Multi-agent systems consist of intelligent agents and their environments. Although multi-agent systems can include environments with robots or humans, they mostly refer to software agents within the software simulations. Researches on MAS mostly focus on obtaining better results with multiple agents in communication, multi-agent learning, coordination and cooperation than single agents.

RoboCup competitions are suitable testbeds for multi-agent systems. Annual RoboCup competitions consist of various leagues such as robotic soccer leagues in different sizes, soccer simulations in both 2D and 3D environments, robot rescue environments and at home leagues. Specifically, simulation competitions are suitable for analyzing complicated team strategies in the face of realistic constraints such as limitations on observability, communication and teleoperation. As of 2011, RoboCup 3D Soccer Simulation League (SSL) hosted 9×9 agent games on a 14×21 m field and is a good example of simulations with constraints mentioned above.

Just like real soccer games, the main objective of a humanoid soccer team is scoring goals against an opponent team. Efficiency of cooperation is an important key factor to win a game. There are mainly two behaviors which involve cooperation issues, namely, passing the ball to a teammate or spreading out to the field of play to gain control of the ball whenever needed. Both behaviors require agents to be in appropriate positions to achieve the desired outcomes. These positions usually belong to special formations which may dynamically change their shapes for different situations during a game. The performance of the overall team is highly dependent on these adaptive formations and the corresponding positions of robotic agents. Contrary to human soccer games, there are not generic formations for humanoid soccer especially because these are also dependent on the underlying motion model. Therefore, the set of mobility constraints of a team plays an important role in the selection of an appropriate team strategy.

The purpose of this thesis is to present efficient team strategies which consist of two important parts: role allocation and team formation of the agents in a simulated soccer game. Experiments are set in the RoboCup 3D SSL environment, Simspark. Different methods for role allocation and team formation are developed and compared with previous works in the area, in terms of ball possession and ball position.

The methods developed in this thesis mainly aims distributed decisions to be made by the agents for the team strategy and to keep computational costs at minimum. Robustness of the method against the communication failures is another issue and it is also considered in the development. Experiments are set to analyze the developed methods and some earlier methods. Some of these earlier methods use supervisors while some of them make autonomous decisions. Results show that team strategies with distributed decisions for team formation and role allocation outperform the existing methods.

ROBOT FUTBOLU İÇİN
DAĞITILMIŞ TAKIM FORMASYONU

ÖZET

Çoklu etmen sistemleri son yirmi yılda oldukça aktif bir araştırma alanı oluşturmuştur. Etmenler, belli hedefleri gerçekleştirmek amacıyla otonom kararlar alıp uygulayabilen bilgisayar sistemleri olarak tanımlanabilir. Etmenler hedeflerini gerçekleştirmek için bulundukları çevreden duyularıyla algıladıkları bilgileri göz önünde bulundurarak kararlar alıp bunları eyleme dönüştürürler. Akıllı etmenlerden oluşan çoklu etmen sistemleri robotların veya insanların bulunduğu çevreleri de kapsamasına rağmen, çoğunlukla yazılım etmenlerini barındıran benzetim ortamları için geliştirilmektedir. Çoklu etmen sistemlerinde etmenler karar verirken, bulundukları çevrenin yanısıra diğer etmenlerin davranışlarını da göz önünde bulundurmak zorundadır. Bazı hedeflerin gerçekleştirilebilmesi için diğer etmenlerle iletişim kurarak birlikte hareket edilmesi gerekmektedir. Çoklu etmen sistemlerindeki araştırmalar çoğunlukla birden fazla etmenle iletişim, çoklu etmen öğrenmesi, organizasyon, uzlaşma, dağıtılmış problem çözümü, hatalara karşı dayanıklılık, koordinasyon ve işbirliği gibi konularda tek etmenli sistemlerden daha iyi sonuçlar elde etmeye yoğunlaşmıştır.

RoboCup yarışmaları çoklu etmen sistemleri için uygun sınama alanlarıdır. RoboCup organizasyonu 1997 yılında kurulmuştur ve çeşitli kategorilerde düzenlediği yarışmalarla yapay zeka konusundaki çalışmalara ilgiyi arttırmayı amaçlamaktadır. Her yıl düzenlenen RoboCup yarışmaları farklı boyutlarda robot futbolu, 2 boyutlu ve 3 boyutlu ortamlarda futbol benzetimi, robot kurtarma ortamları ve ev uygulamaları gibi çeşitli dallarda ligler içerir. RoboCup'ın temel hedefi 2050 yılında FIFA Dünya Kupası'nı kazanan futbol takımını yenebilecek tamamen otonom insansı robot takımını yaratabilmektir. Robot futbolu yarışmalarının bazıları çeşitli boyutlarda insansı robotlarla (Standart Platform Ligi, 3 boyutlu futbol benzetim ligi vb.) bazıları ise insansı olmayan ve daha hızlı hareket edebilen robotlarla (Orta boyutlu robot ligi, küçük boyutlu robot ligi vb.) yapılmaktadır. Bazı liglerde yarışan takımlar robotlarını kendi tasarlamaktadır. Bu tür liglerde takım stratejilerinden çok mekanik ve teknolojik özelliklerin kullanımı önemlidir. Standart Platform Ligi ve 3 boyutlu futbol benzetim ligi gibi liglerde ise tüm takımlar önceden belirlenmiş standart robotları kullanmak zorundadır. Bu yüzden bu tür liglerde birden fazla etmenin birlikte ve uyum içinde hareket etmesini sağlayacak stratejiler önem kazanmaktadır. Özellikle benzetim yarışmaları, gözlem, iletişim ve uzaktan yönetim kısıtları gibi sınırlamalara sahip karmaşık takım stratejilerini incelemek için oldukça uygundur. RoboCup 3 boyutlu futbol benzetim ligi yukarıda belirtilen kısıtlara sahip benzetimlere iyi bir örnektir ve 2011 yılı itibarıyla 14×21 m boyutlarında sahada 9×9 etmenli oyunlara ev sahipliği yapmıştır. Önümüzdeki yıllarda teknik gelişmelerle birlikte etmen sayılarının her takım için 11'e çıkarılması, saha boyutlarının gerçek futbol sahalarının boyutlarına genişletilmesi hedeflenmektedir. RoboCup 3 boyutlu simülasyon ligi,

gerçek robotlar yerine bunların benzetimleriyle oluşturulan yazılımsal etmenleri kullandığı için araştırma maliyetleri açısından da ekonomik bir yapıya sahiptir. Bu yapı sayesinde RoboCup'ın 2050 yılı hedefini gerçekleştirme konusunda insan futboluna en yakın stratejilerin uygulanıp sınanabileceği yarışma ortamı RoboCup 3 boyutlu simülasyon ligidir.

Gerçek futbol oyunlarında olduğu gibi, insansı futbol takımlarının ana hedefi rakip takıma karşı gol atmaktır. Verimli bir işbirliği sağlamak maçı kazanmak için anahtar faktördür. İşbirliği konusunda takım arkadaşına pas atmak veya sahaya gerektiğinde topun kontrolünü sağlamak üzere yayılmak gibi iki temel davranış vardır. İki davranış da etmenlerin istenen sonucu elde etmeleri için uygun konumlarda olmasını gerektirir. Bu konumlar genellikle oyundaki durumlara göre dinamik olarak değişen özel dizilişlerdir. Takımın genel performansı uyarlanabilir dizilişlere ve bunlara ait olan robotik etmenlerin konumlarına bağlıdır. İnsanların oynadığı futbol oyununun aksine, temeli oluşturan hareket modeline bağlılıktan dolayı insansı robot futbolunda genel dizilişler yoktur. Bu yüzden bir takımın hareketlilik kısıtları uygun takım stratejisini seçmede önemli bir rol oynamaktadır. Hızlı hareket edebilen robotlarla daha geniş alanlara yayılarak açık bir takım formasyonu uygulanabilir. Fakat insansı robotlar günümüz teknolojisindeki kısıtlamalardan dolayı çok hızlı hareket edememektedir. Yavaş hareket eden bir robot topa sahip olan takım arkadaşı topu kaybettiğinde uzak bir konumdaysa top kontrolünün rakibe geçmesine sebep olabilir. Aksi durumda topa sahip olan takım arkadaşına fazla yakın duran robot bu robotun hareket kabiliyetini düşürüp hata yapmasına sebep olabilir. Bu yüzden birbirlerinin hatalarını düzeltebilecek kadar yakın ve takım performansını düşürmeyecek şekilde konumlanmak önemlidir. Robotlar, konumlanma sırasında kendi takım arkadaşlarına dikkat ettiği gibi rakip takım oyuncularını da göz önünde bulundurmalıdır. Robotlar, hücum sırasında rakiplerden uzakta durmaya çalışarak rakip kaleye topu taşıma sırasında olası engellerden sakınmalıdır. Savunmada ise robotlar rakip robotlara olabilecek en yakın pozisyona gelmeli ve rakipleri engelleyerek top kontrolünün kendi takımına geçmesini sağlamalıdır. Takım formasyonu oluştururken söylenen tüm hususların dikkate alınması, futbolun temel amacı olan rakip takıma gol atma ve attığı gol sayısından az gol yiyerek oyunu kazanma konusunda takımın performansının en üst seviyeye yaklaşmasını sağlayacaktır.

Bu tezin amacı, benzetim ortamında oynanan futbol oyununda rol ataması ve takım dizilişi olmak üzere iki önemli parçadan oluşan etkili takım stratejileri sunmaktır. Rol ataması ve takım dizilişi için farklı yöntemler geliştirilmiş ve bu alandaki geçmiş çalışmalarla beraber top konumu ve top kontrolü konusunda karşılaştırmalı testler yürütülmüştür. Etmenler saha içindeki rollerine göre (kaleci, savunma, orta saha, hücum) gruplara ayrılmış ve oyuncuların bulundukları gruptaki diğer etmenlerle uyumlu bir şekilde görevlerini yerine getirmeleri hedeflenmiştir. Takımda grup davranışları için savunma ve hücum şeklinde iki grup belirlenmiş ve etmenlerin kendi rollerine ait görevlerin yanında grubun davranışlarına da yardımcı olacak şekilde hareket etmeleri amaçlanmıştır. Etmenlerin gruplara ayrılması ile ilgili farklı yöntemler geliştirilmiş ve bu yöntemlerin sonuçları incelenmiştir. Bu yöntemler rollere atanmış robot sayılarının değişmediği durağan gruplama, çeşitli durumların önceden tanımlandığı ve bu durumlara göre rollerdeki robot sayılarının belirlendiği durum tabanlı gruplama ve Bayes öğrenme yöntemini temel alan olasılıksal gruplamayı içermektedir. Takımın saha içindeki konumlanması ve dizilişi için Voronoi diyagramlarını temel alan bir yapı geliştirilmiştir. Bu yapı ile tüm etmenler dağıtılmış bir şekilde kendi konumlarını hesaplamaktadır. Yöntemi kullanan her etmen ilk önce topa ve kendi konumuna göre bir çokgen oluşturmaktadır. Bu çokgen savunma grubundaki oyuncular için topla takımın kendi kalesi arasında rakibi durduracak şekilde, hücum oyuncuları içinse topun etrafında konumlanarak top kontrolünü sağlayacak şekilde oluşturulmaktadır. Daha sonraki aşamada etmen, saha içinde gördüğü takım arkadaşlarının konumlarını göz önünde bulundurarak Voronoi diyagramlarının kurallarına göre ilk oluşturduğu çokgeni değiştirmektedir. Bu değişim kendi çokgenini hesaplayan etmenlerle diğer etmenler arasında doğru parçaları çizilmesi, diğer etmenlerden bu çizgilere paralel çizgiler çekilmesi, paralel çizgiler çokgeni ikiye bölünüyorsa bu iki parçadan etmeni içerenin alınması gibi adımları içerir. Bu aşama sonucunda elde edilen çokgenin ağırlık merkezi etmenin gitmesi gereken hedefi belirlemektedir. Bu aşamalar çalışma zamanında sürekli güncellenerek dinamik bir formasyonun oluşması sağlanmaktadır. Uygulanan yöntem sayesinde etmenler bulundukları gruptaki diğer etmenlere, onların hareketlerini kısıtlamayacak şekilde olabildiğince yaklaşarak konumlanmaktadır. Takım formasyonu içinde rakip oyuncularla çarpışmaları engellemek amacıyla bazı engel sakınımı yöntemleri de uygulanmaktadır. Saha dizilişi ile amaçlanan topun etrafında etkin bir konumlanma ile top kontrolünü ele geçirmek ve top rakibin kontrolündeyken onları durdurabilecek konumlarda etmenler bulundurmaktır. Geliştirilen yöntemler belli bir yapı üzerinde birleştirilmiş ve RoboCup 3 boyutlu futbol benzetim ligi takımlarından beeStanbul'un yazılımına eklenmiştir. Yöntemlerin başarımı RoboCup 3 boyutlu futbol benzetim liginde yarışmakta olan bazı diğer takımların yazılımlarıyla yapılan maçlarla sınanmıştır. Başarım ölçütleri olarak topun takımlar tarafından kontrol edilme yüzdesi ve topun bulunduğu konumlar incelenmiştir. İlk ölçüt topun etrafında doğru konumlanmanın başarımını ölçerken ikinci ölçüt topun rakip sahaya taşınma yüzdesini incelemektedir. Bu yöntemlerle ilgili deneyler RoboCup 3 boyutlu futbol benzetim liginin yarışma ortamı Simspark'ta uygulanmıştır. Görsel arayüz olarak da RoboCup 3 boyutlu futbol benzetim liginde 2011'de kullanılmaya başlanan Roboviz programı kullanılmıştır.

Bu tez kapsamında geliştirilen yöntemler temel olarak etmenlerin takım stratejisi konusunda ayrık kararlar vermesi ve işlem maliyetlerinin azami seviyede tutulmasını hedeflemektedir. Yöntemin iletişimde oluşabilecek sorunlara karşı dayanıklılığı da geliştirme sırasında dikkate alınmıştır. Rakiplerin saha içindeki dizilişleri, davranışları ve stratejilerine göre uyarlanabilir takım stratejileri geliştirmek de tez içindeki çalışmaların amaçlarından biridir. Bu hedeflere ulaşmada geliştirilen yöntemin başarımı çeşitli deneylerle sınanmıştır. Deneyler önerilen yöntemlerle literatürde mevcut bulunan yöntemler üzerinde karşılaştırmalı olarak yürütülmüştür. Önceki yöntemlerden bazıları denetçi kullanırken bazılarında etmenler bağımsız kararlar vermektedir. Denetçi kullanan yöntemler çoklu etmen sisteminin çevresiyle ilgili bilgileri alıp incelemekte ve bu bilgileri tüm etmenlere iletmektedir. Denetçi olmayan yöntemlerde ise etmenler, çevre ile ilgili kendi donanımlarıyla elde ettikleri ve diğer etmenlerden iletişim yoluyla aldıkları bilgileri kullanarak tamamen otonom karar vermek zorundadır. Sonuçlar rol ataması ve takım dizilişi için ayrık kararlar veren takım stratejilerinin önceki yöntemlere göre daha iyi çalıştığını göstermektedir.

1. INTRODUCTION

Multi-agent environments are formed from agents that are usually autonomous, decentralized and have a local observation view of the system instead of having a global view. Research on MAS focus on providing efficient solutions in topics like coordination, cooperation, communication, learning, planning, problem solving and organization, using multiple agents. Particularly, cooperation and coordination are the key topics for multiple agent systems.

Robot soccer [1] is the official game for RoboCup competitions. It is also a suitable test-bed that involves many issues in MAS research. There are several categories for different robot sizes and types, and also competitions in 2D and 3D simulation environments.

The motivation behind this thesis is the development of efficient cooperation and coordination methods in the RoboCup 3D soccer simulation environment, Simspark. The proposed methods can be applied to any other robot soccer environment. As of 2011, 3D Soccer Simulation League is played with two teams of nine agents. Agents are fully autonomous humanoid robots with no supervisor existence. They have a limited vision of the field and a limited communication ability. The simulation has dynamic noise to make the environment more realistic; actions are also non-deterministic. These challenges and presence of opponents make the 3D Soccer Simulation League a good environment to study issues such as cooperative team strategies.

In this thesis, team formation strategies, role allocation and group behaviors of agents are investigated, and some methods are proposed for forming a winning strategy in robot soccer. Role allocations are made with both communication and observations. Two different groups are determined for different purposes of the game, namely defenders and attackers. Attackers are formed for constructing an attacking formation around the ball and scoring a goal whenever possible while defenders are for blocking and constructing a defensive obstacle against the opponent team. Team formation strategy for agent positioning is based on a distributed Voronoi cell approach, where agents determine their cell discretely. This approach also differs from the classic Voronoi diagrams in construction of the cells. With the proposed approach, agents try to position themselves around the ball to possess it, because it is crucial for winning the game.

This thesis mainly focuses on building a team formation for robot soccer which can adapt itself according to opponent behaviors. Contributions while achieving this goal covers challenges like distribution, less computation and robustness to communication failures. Proposed Voronoi cell approach provides a team formation that positions the agents around the ball to create a crowded area. Crowding around the ball aids the team to be able to possess the ball more while carrying it to the opponent area. The algorithm is distributed so each agent executes the method itself without the need of a supervisor. This adds robustness to communication because agents can stay in the team formation only using self field knowledge. The proposed algorithm's computational cost is also kept low. Another contribution of the thesis is grouping the agents for different purposes. Several methods for this purpose is proposed within the thesis and they provided adaptive grouping of agents into defenders and attackers.

1.1 Organization of the Thesis

The thesis is organized as following: Chapter 2 provides background to present earlier work in the field. This section covers development environments, methods that are used in the development. Chapter 3 explains the developed team formation method, giving the details of distributed Voronoi cell construction and role allocation methods. Chapter 4 gives the experimental results of comparisons among the developed method and previous work. Robustness to communication failures are also investigated in this section. Finally, the thesis is concluded with Chapter 5 and the future work is presented at the end.

2. BACKGROUND AND RELATED WORK

To develop an efficient team formation strategy for robot soccer, some important issues must be taken into account. Role allocation of agents to create and maintain group formations, positioning of the agents, planning of their behaviors and using a simulation environment to experiment the methods are among them. This chapter explains RoboCup competitions, the simulation environment that is used in this thesis and the related work on the issues that are mentioned above.

2.1 RoboCup

RoboCup [1] is an international scientific initiative, which is founded in 1997. RoboCup aims to advance the researches in the intelligent robotics area. Original goal of RoboCup is to form a team of robots that can win against human soccer World Cup champions by 2050. Since 1997, RoboCup expanded into different areas. There are four main branches in RoboCup organizations currently, namely RoboCup Soccer, RoboCup Rescue, RoboCup @Home and RoboCup Junior.

RoboCup Soccer divides into various branches of leagues. Humanoid League is played by robots that modelled with a human-like body structure and human-like senses. This league consists three branches according to the size of robots: Teen Size, Kid Size and Adult Size. Middle Size League is played with a regular size human soccer ball and with robots of no more than 50 cm diameter. In this league, robots can communicate via a wireless network. Simulation League consists two branches, 2D and 3D Soccer Simulation Leagues. These leagues are played by software agents and mainly focus on artificial intelligence and team strategy. 3D Soccer Simulation is played by humanoid agents thus motion models also taken into consideration. Small Size League is a highly dynamic environment. It is a hybrid centralized/distributed system and researches on this league focus on multi-robot cooperation. Standart Platform League (SPL) is played by identical robots, so software development is the only factor for the success of a team. Four-Legged League was the predecessor of SPL, which was played by Sony's AIBO robots. AIBO robots are shaped like a dog, and SPL replaced them with Aldebaran Robotics's humanoid Nao robots.

RoboCup Rescue competitions are for robot rescue studies in disaster scenarios and are made in two branches: Robot League and Simulation League. Robot league aims to increase awareness to the challenges in search and rescue applications. Issues like mobility, mapping, planning etc. are some of the challenges in this league. Simulation League deals with two main challenges. First one is to emulate realistic disaster scenarios. Second challenge is to develop intelligent agents with the capabilities to be the main actors in a disaster scenario.

RoboCup @Home focuses on developing robotic technologies with a high relevance to the future domestic applications. Annual competitions on this category are done with autonomous service robots in a non-standardized home environment, which makes it similar to real world home environments. Researches on RoboCup @Home covers areas like human-robot interaction, localization and mapping, computer vision, object recognition and manipulation etc.

RoboCup Junior is made in three different competitions, namely Soccer Challenge, Dance Challenge and Rescue Challenge. Younger students which can study at any school below grad degree can participate in these challenges. It mainly is an introduction to RoboCup and encouragement for younger students to keep on working on robotics.

2.2 Robot Soccer Simulation Environment

RoboCup soccer simulation competitions have two main branches: 2D SSL and 3D SSL. 2D SSL is a competition which the agents are represented as circles in a two-dimensional field and have the ability to move, dash, turn and kick. They are not representations of any real robots and each agent can execute its own strategy to win the game. 3D SSL is played by humanoid robots in a simulation software called Simspark [2]. APPENDIX A.1 gives a detailed information on Simspark. Visualization of the simulation is provided by a graphical tool called Roboviz [3] since

2011. Roboviz also provides some efficient features like giving the ability to reposition agents and the ball in runtime, changing the play mode and drawing geometric shapes on the field for experimental issues. Figure 2.1 shows a screenshot from Roboviz in a match.

Figure 2.1: Screenshot of an instance in a match from Roboviz. [3]

Soccer competitions are run in rcsserver3d, which is the environment of Simspark for three dimensional matches. Simspark provides an environment for multiplayer soccer games of two competing teams of simulated autonomous humanoid agents (RoboCup 2011 3D Simulation League hosted 9 x 9 agent games on a 21x14 m field.). The team scoring more goals in a ten-minute-long match wins the game. Simspark uses ODE (Open Dynamics Engine) [4] for physical agent simulation of Nao humanoid robots by Aldebaran Robotics [5]. The real Nao robot has a height of 57 cm, a weight of 4,5 kg and 22 degrees of freedom. The robot is equipped with special sensors including a gyroscope, an accelerometer and a force resistance perceptor on each foot. Simspark can simulate all these features and model some realistic limitations including sensor and actuator noise. Figure 2.2 shows the real and simulated version of Nao robots. The simulator also provides limited communication among robots through special effectors and perceptors and visual information in the form of noisy distance and angle values for the objects in the viewpoint of agents.

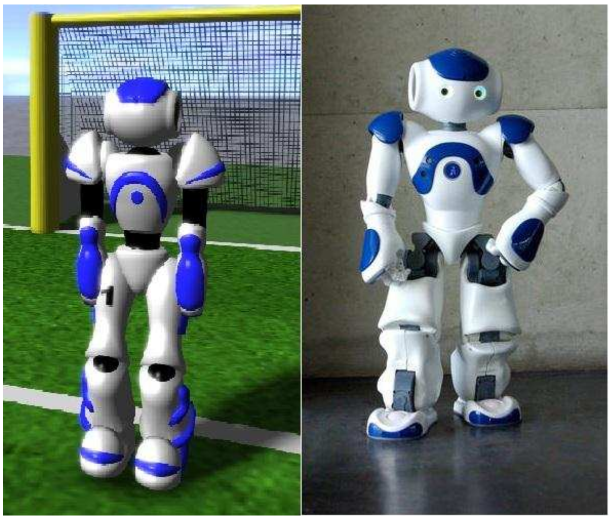


Figure 2.2: Nao robot simulation on Simspark (left) and real Nao robot (right).

Field dimensions of rcsserver3d are 21 m by 14 m. Goals are 2.1 m by 0.6 m and have a height of 0.8 m. The ball has a mass of 26 g and a radius of 0.04 m. Teams don't have a supervisor and agents have to localize themselves according to flag information that is recieved by the game server. There are eight flags which are positioned on each goal's left and right posts and corners of the entire field. Agents have a 120 degree angle of view that is provided by their camera on their heads and only the polar coordinates of the flags they see are retrieved at a given time. Figure 2.3 shows the rcssserver3d's field plan.

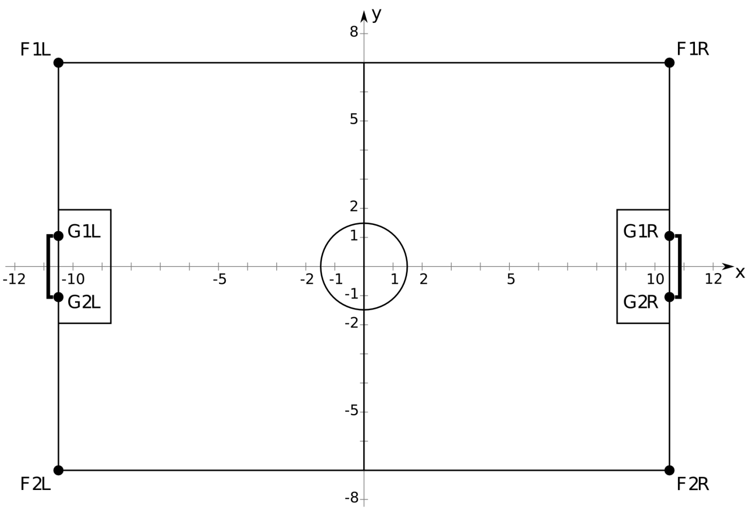


Figure 2.3: Field plan of rcssserver3d [2].

Some rules of humanoid soccer are different from real soccer due to the limitations of mobility of agents. At present, fouls are not penalized in the RoboCup 3D SSL but crowding the ball. According to the crowding rule, at most two players are allowed to be in the 0.8 m radius circle around the ball; only a single player from a team in a circle with a radius of 0.4 m and at most two teammates in a circle with a radius of 1 m. Failure to comply with either of these rules results in a repositioning of an agent out of the field. All these rules should be taken into account in the team strategy for avoiding any penalties.

2.3 Multi-Robot Team Formation Strategies

Dynamic team formation problem has been investigated in earlier work for both humanitarian and military applications ([6], [7]). Successful results of these works have been used in RoboCup environments as well ([8], [9], [10]). Multirobot coordination approaches used in RoboCup environments mostly rely on continuous communication among agents. However, RoboCup 3D SSL doesn't provide a supervisor and communication among agents is limited.

Several team formation algorithms were applied in RoboCup soccer competitions ([11], [12], [13], [14]). [11] use Voronoi cells to position and distribute players in the field for RoboCup 2D SSL. Voronoi diagrams are named after the mathematician Georgy Voronoi, and consist of a given space and a set of regions (r_x) that are shaped according to a set of points (p_x). Voronoi diagrams can be defined as follows: each point p_i in the space have a region r_i and a region is associated with all the points where the distance with p_i and the other points in p_x are equal. Using this approach, each region of a point has the same distance to the other points in the given space. With the method in [11], each agent calculates its own Voronoi cell and moves to the center of its own cell. Figure 2.4 shows the main cell calculation algorithm of [11]. Dynamically calculating the cells ensures that agents scatter throughout the field. After the distribution is achieved, agents move to better positions by attraction vectors and calculate their Voronoi cells dynamically to be distributed in the field again. Even though this method is efficient for fast-moving 2D soccer agents, slower humanoid agents in 3D SSL should maintain proximity to each other in order to gain control of the ball quickly when it is lost. Therefore, distributing the agents throughout the field may not result in the desired outcome in 3D SSL. Figure 2.5 shows the progress steps in calculating the Voronoi cells with the method in [11].

```
// Input: an agent indexed i
// Output: a polygon (cell) as the Voronoi Cell of input agent
// Consider the entire field as the initial Voronoi Cell of i
cell_i ← all the field
Point P_i ← position of i
for all agents j ≠ i do
  Point P_j ← position of agent j
  Line L ← perpendicular bisector of P_i and P_j
  Intersect L with cell_i
  if L intersects cell_i in 0 or 1 points then
    Continue
  end if
  // L cuts cell_i into two polygons (left and right)
  Polygon cell_0 ← right polygon
  Polygon cell_1 ← left polygon
  if cell_0 contains P_i then
    cell_i ← cell_0
  else
    cell_i ← cell_1
  end if
end for
```

Figure 2.4: Voronoi cell calculation algorithm in [11].

The approach in this thesis uses Voronoi cell decomposition as in [11] but differs from this approach by its initial frame construction and its adaptability based on the ball location. The objective is not spreading out all players on the field but constructing a formation around the ball to easily possess it whenever possible.

[12] propose Dominant Region (DR) diagrams to create a formation. DR diagrams look like Voronoi diagrams, but the required calculation is based on the arrival time of all agents to their future positions. Each agent forms its region based on an area where it can reach to faster than its teammates. Players can move in their regions in order to be positioned on the field. With this approach, the agent which is closest to the ball approaches to the ball and the others can follow it while staying in their dominant regions. To calculate full diagram that covers the entire field needs an excessive computation, so rough calculations are suggested by [12] to decrease the computational cost. Figure 2.6 shows the results of full calculation and the rough

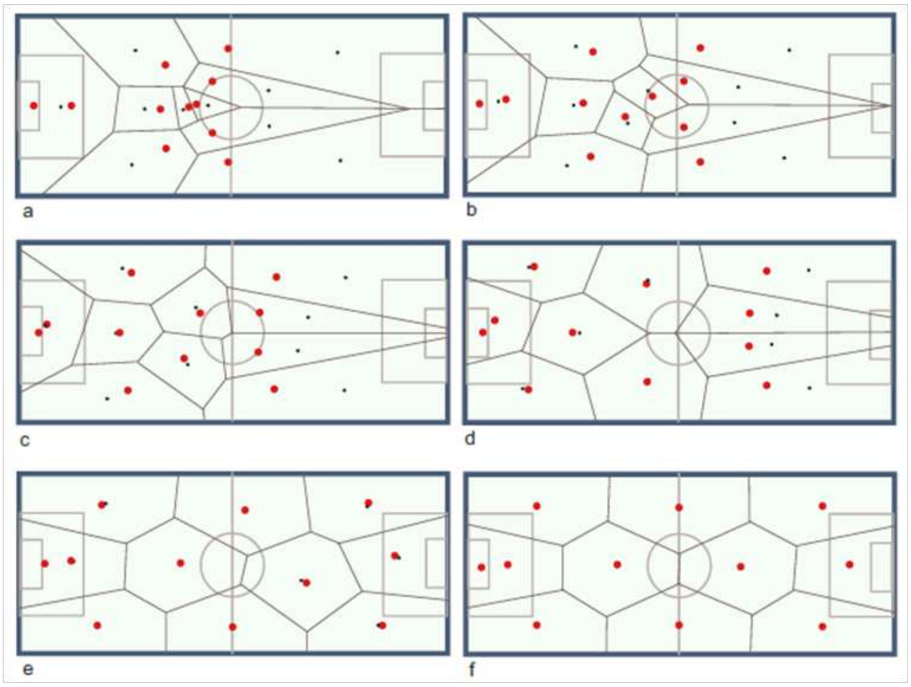
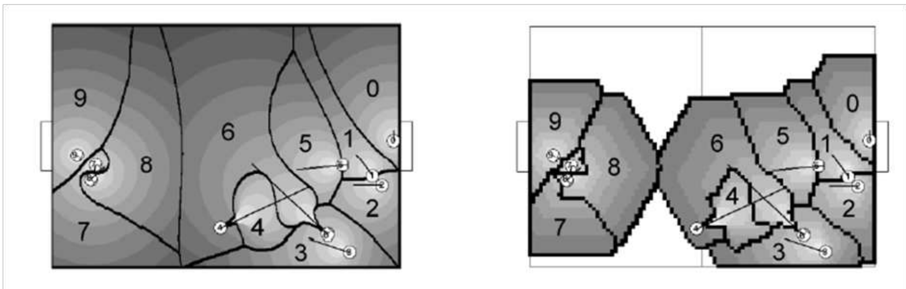


Figure 2.5: Steps of the progress in Voronoi cell calculation of [11].

calculation for an instance. This method can be useful in 3D humanoid soccer but needs a supervisor or a high communication bandwidth among the agents to calculate a general DR diagram. Therefore, it may not be suitable for environments with limited communication.



(a) Detailed calculation (b) Rough calculation

Figure 2.6: Dominant Region diagrams with full and rough calculations [12].

Situation Based Strategic Positioning (SBSP) [13] is another team formation approach which uses game information including the current position of the agent and its current role, the selected formation for the team and the positions of others. Maintaining this information, agents move to their positions according to their roles. Figure 2.7 shows some positioning examples according to the ball position that are provided in [13] This method requires dynamically assigning roles to the agents during the game. SBSP suffers from a complicated rule-based algorithm to reach a final formation in non-deterministic and noisy environments like RoboCup 3D SSL due to the computation requirements.

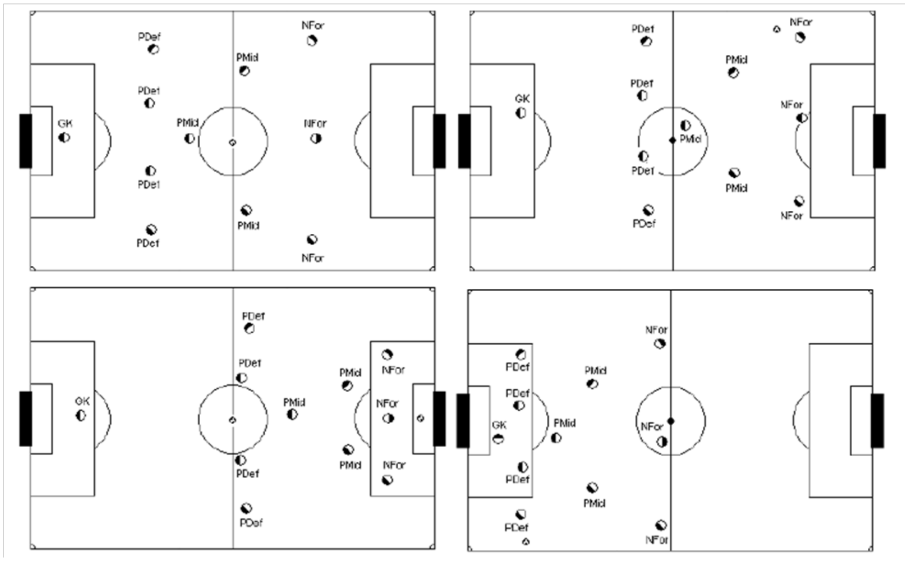


Figure 2.7: Situation based positioning examples according to the agent roles and ball position [13].

[14] propose a Case-Based Reasoning (CBR) method to position the agents. Cases represent both the action sequences and the formations by keeping the game situations including positions of the agents, game time, current score etc. CBR is an applicable formation method but in some of the CBR methods, the cases should be hand-coded before and usually the number of them is limited. Some of the CBR approaches update their case libraries in runtime to modify cases but this is a costly process.

2.4 Role Allocation

[15] presented a distributed play-based system which equips robots with plays. A play refers to an alternative team strategy. With the play-based role assignment algorithm robots autonomously adapt their strategy based on the current state of the environment, behavior of the opponents and the game. This method assigns roles to robots in a fault-tolerant manner that minimizes role switching and synchronization problems. It puts high priority on the communication of task relevant data. The contributions in [15] are used in RoboCup four-legged league (Two teams of four Sony AIBO robots against each other in a soccer game) in 2005 by CMDash team.

Instead of mapping between states and joint actions, [15] provide a play-based approach, which consists of plays. Plays provide set of roles assigned to the robots. A play specifies a plan for the team under applicable conditions and provides sequence of steps for the team to execute. Weights of plays should be adapted to match an opponent. Also the team can change plays relying to score or the time left in the game. Three important things in this approach are applicability conditions, role assignments and weights for play selection. Applicability conditions show if a play is usable for execution. Each condition is a conjunction of binary predicates. Plays with the greatest weight are chosen and in each play robots are assigned to appropriate roles. When a play is selected, it continues until a new play with a greater weight is available or the current one is no longer applicable. To prevent rapid changes between plays, applicability conditions shouldn't rapidly osscillate between true and false (e.g. time and game score can't rapidly oscillate so they can prevent rapid changes between plays if they are selected as applicability conditions). Figure 2.8 shows an example play used by [15]. In this "guard" play, robots play more defensive. Not conceding a goal is more important than scoring a goal. This play is applicable when the team is winning and has less players than the opponent, or the team is winning by two goals or more and second half is played currently. ROLES tag shows the role assignments according to the current count of teammate robots and WEIGHT tag shows the play's weight value (3 for guard play). Play selector algorithm of [15] is executed on the team leader robot. Play selector selects the play to be performed. The leader broadcasts the current play and the robots' roles periodically. If a robot can't hear the broadcasted play due to failure of the team leader or a network problem, it runs a default play selected before. To avoid executing this less efficient plays, team leader should be changed in failure situations with a leader selection algorithm.

```
PLAY Guard
APPLICABLE winning fewerPlayers
APPLICABLE secondHalf winningBy2OrMoreGoals
ROLES 1 Goalkeeper
ROLES 2 Goalkeeper Defender
ROLES 3 Goalkeeper Defender Independent
ROLES 4 Goalkeeper Defender Midfielder Independent
WEIGHT 3
```

Figure 2.8: An example play from the playbook in [16].

2.5 Group Formation and Planning

Forming groups with different purposes in the team of agents usually results in a better team performance. [16] introduce a method which coordinates the agents within each group by explicit communication. Forming groups with optimal number of agents can prevent unnecessary crowds. Therefore, agents in different groups can achieve multiple tasks which might help completing those tasks faster. While inter-group communication is kept limited, intra group communication demands are high.

[14] presents a case-based approach for cooperative action selection, which relies on storage, retrieval and adaption of example cases. The main focus is on cases of coordinated attacking passes between robots in the presence of defending opponents in a robot soccer game. A case describes the state of the environment and the actions to perform in that state. Case representation is distinguished between controllable and uncontrollable indexing features corresponding to the positions of ball, team and opponents. Uncontrollable features are opponent positions and the continuous ball features. Controllable features are teammates' positions and the cost of moving robots from current situation to match the case. In Case-Based Reasoning cases are recorded and state of the world is modeled at a given time and a succesful action is prescribed. Solution description in the cases indicates which actions the robot should perform. Retrieval process assigns robots to actions and then the robot shares its individual intentions to act via coordination mechanism. Case retrieval and reuse steps are done with robots' communication and sharing their internal states and actions. Figure 2.9 (a) shows an example case and it's symmetrics in the work of [14]. Figure 2.9 (b) describes the case representation P (B: ball position, G: Goal's colour, Tm: Teammate positions, Opp: Opponent positions), Solution Description S and case scope K. Case scope gives two radius values to shape an elliptic area for the given elements of a case. If that element is inside its scope, than that case becomes applicable for planning.

In [17], Degree of Possession notion is used with case-based reasoning, which measures the nature of possession and who possesses the ball. These degrees added to the case similarity measurements for case selection. There are six degrees with integer parameter values. Three degrees are considered as no position, opponent's clear control of the ball and team's clear control of the ball with appropriate parameters. The other

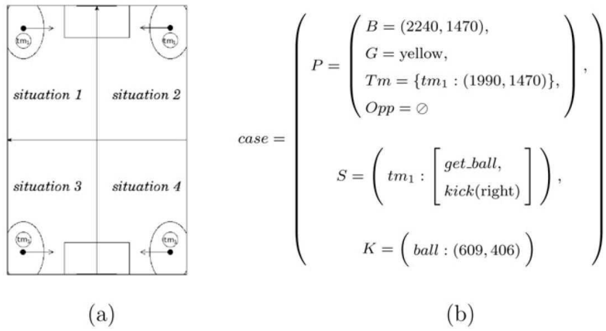


Figure 2.9: An example case and it’s symmetric cases in [14].

three are the same considerations but with a scrum in position, not a clear position. So the latter three has lower Degree of Possession values for case similarity. [17] divided the soccer field into 30 regions. The purpose of the team is moving from region 0 (team’s goal) to region 29 (opponent’s goal) with reasonable cases.

[18] used case-based reasoning within an adaptive manner. In this work, agents are taught with the behaviors of agents from previous matches using the match log files. The agents calculate a case distance value for an instance to compare with it’s case library and tries to use the most similar case with the other agent.

[19] used a method similar to [14] but added some features to cases instead of just using positions and scopes. These features are obtained through observations and used in case representation. Some features from [19] are like "defenders back" which holds the number of defenders in defensive zone, "effective imbalance" which holds the difference of numbers between team defenders-opponent attackers and team attackers-opponent defenders and "effective attackers" which holds the number of attackers that can affect the game.

In [20], a hybrid method of case-based reasoning and genetic algorithms is proposed. Figure 2.10 shows the structure of the hybrid approach in [20]. Represented cases are evaluated with the conventional genetic algorithm to obtain better cases. If genetic algorithm gives a good result, that case is stored. Case retrieval and reuse steps are performed after the genetic algorithm. In this hybrid method, there is also a rule-based structure for robustness. Rules are given by humans and if the best retrieved case is under a threshold value, a plan from the rule base is selected for performing instead of the case.

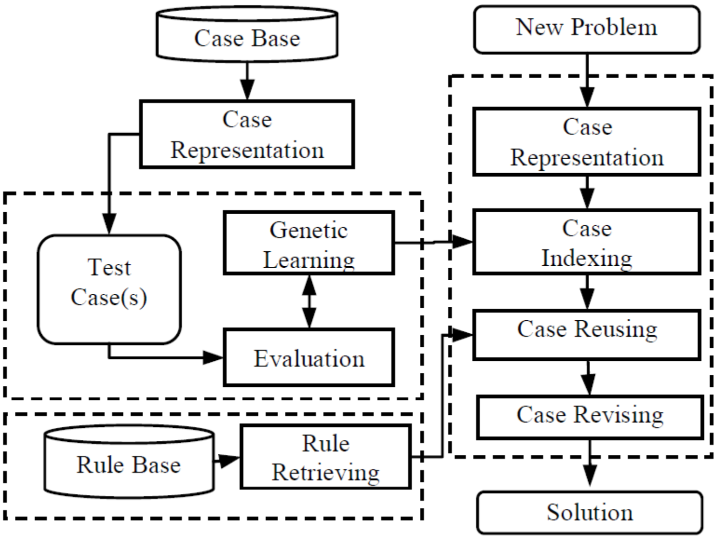


Figure 2.10: The structure of the hybrid approach [20].

[21] used Petri Net Plans (PNP) [22] to design cooperative behaviors for robots, based on Joint Commitments Theory [23]. Joint Commitments Theory says that a set of team members are committed to the execution of a cooperative behavior and they should continue their individual action until one of the these conditions come up: behavior's conclusion with success, behavior becomes irrevelant or behavior's conclusion is impossible. Petri Net Plans are basically a tool for graphical representation of plans, based on Petri Nets [24]. Petri Nets are behaviour presentation frameworks allowing the design of highly expressive plans in dynamic, partially observable and unpredictable environments. They include actions and transitions for representing a plan. To increase modularity and readability, sub-plans can be used in PNPs. A sub-plan is used in the main plan as an ordinary action but inside it there is also a set of actions. So when a transition for a sub-plan is executed, its initial place is reached. With the sub-plan's end transition, a transition to another place in the main place occurs. Multi-robot plans can also be represented in PNPs. Single robot PNPs with synchronization constraints between different robots form a multi-robot PNP. There are two operators for synchronization: hardsync and softsync. Hardsync relies on a single robot sync primitive, and it creates a communication link between two robots to exchange info and synchorinze the execution. Softsync doesn't need a single robot sync primitive, each robot notifies the other one if they end an action and unless the notify comes, robots continue to do other tasks to wait the other one. [21] combined

PNPs with Joint Commitments Theory to create a PNP based teamwork design for passing between robots. Figure 2.11 shows the multi-robot PNP for passing behavior in [21].

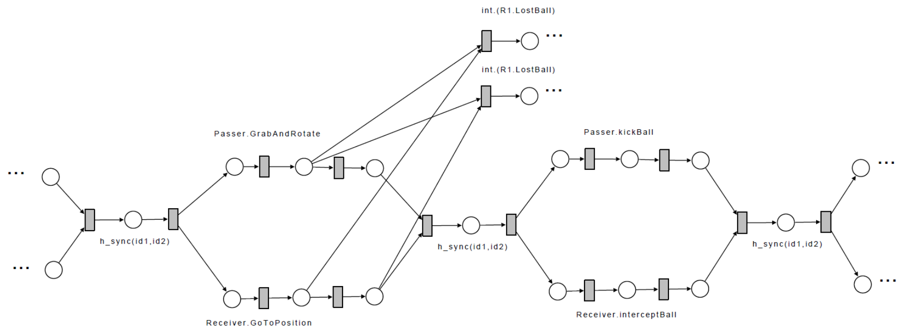


Figure 2.11: Multi-robot PNP for passing behavior. [21]

2.6 Allocating Required Number of Robots for a Task

In [25], gathering with multiple robots in a place problem is investigated. Robots have four different states which they perform different tasks: wait, look, compute and move. "Wait" makes the robots idle and not perform anything. In "look" state, robots observe their environment. In "compute" task they compute with their observation inputs to calculate the destination. In "move", they move to their calculated target. In this approach, robots observe the other robots and decide if they are the most suitable one for a task. If all the tasks can be done faster by the other robots, it starts to observe the environment to find another task. If the robot still can't find a task, it becomes idle and waits for a task but after a given waiting time, they start to observe the environment again. With this approach, required number of robots for the gathering task is determined discretely by the robots.

[26] proposed an auction mechanism for task allocation in a rescue domain. The environment is a disaster area and there are several fires on the map of the area. Fire brigade agents calculate time cost to help the burning buildings they could help. All costs from the agents are send to an auction mechanism. Figure 2.12 shows the auctioning for fire brigades in an example disaster area. The auction mechanism allocates fire brigades to the fires and ensures to reach maximum number of burning areas, all of them, if possible. Second priority of the mechanism is to select the fire

brigades with the least time cost for an area. With the given conditions, task allocation of the agents is planned to be achieved with the optimum number of them.

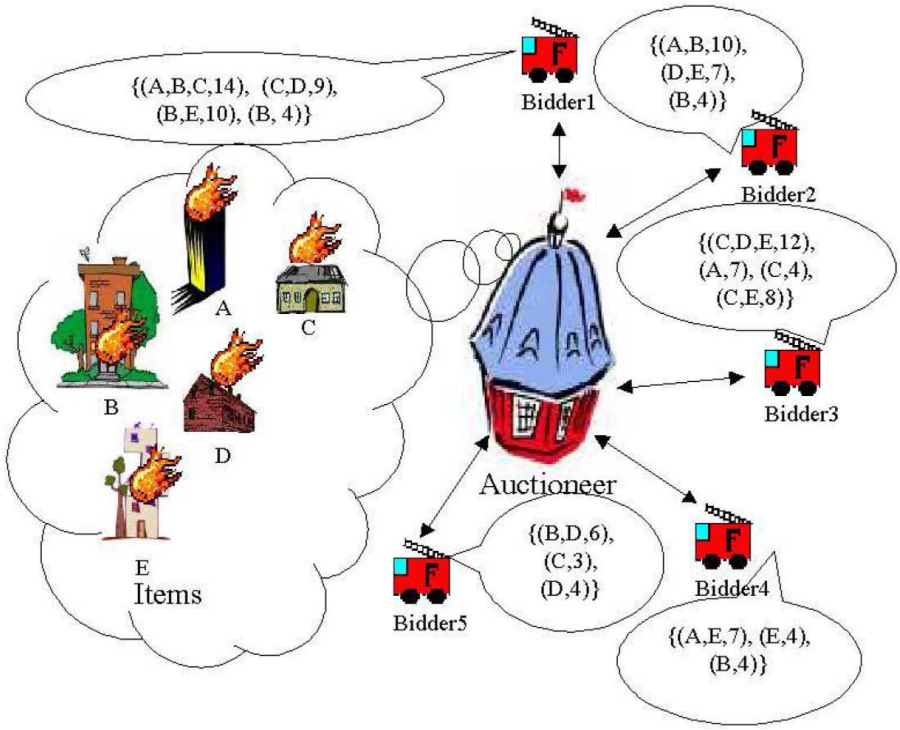


Figure 2.12: Auctioning mechanism of fire brigades in [26]

[27] proposed a probabilistic robot count selection for a fleet of robots. Later they used the algorithm in their work for deployment strategy [28]. Fleet size is determined with energy and time constraints. In some situations robots can have low energy and recharge might be needed. This energy constraints are added into the algorithm with a probabilistic distribution and robots decide on tasks considering the energy level probabilities and the time costs for doing a task.

3. DISTRIBUTED TEAM FORMATION

The proposed distributed team formation strategy involves four sequential processes to determine a target for an agent. Figure 3.1 presents the main modules for the team strategy. Initially two groups, namely attackers and defenders, are formed by using a group formation strategy [29]. The role of each agent is determined based on these groups. The attackers group involves the forward and the midfielder agents while the defenders group involves only the defender agents. The adaptive formation method in this thesis relies on the construction of Voronoi cells, which are generated distinctly by each agent that has the role of midfielder or defender. The centers of these cells form the initial targets for these agents. Target locations are finalized by applying Potential Fields Method [30] for obstacle avoidance and path planning. Because controlling the ball is crucial in soccer, its location is used in cell initialization and forming groups. All agents except goalkeeper continuously send their time costs to control the ball and they decide on the forward agent role according to the incoming cost information and a self-calculation.

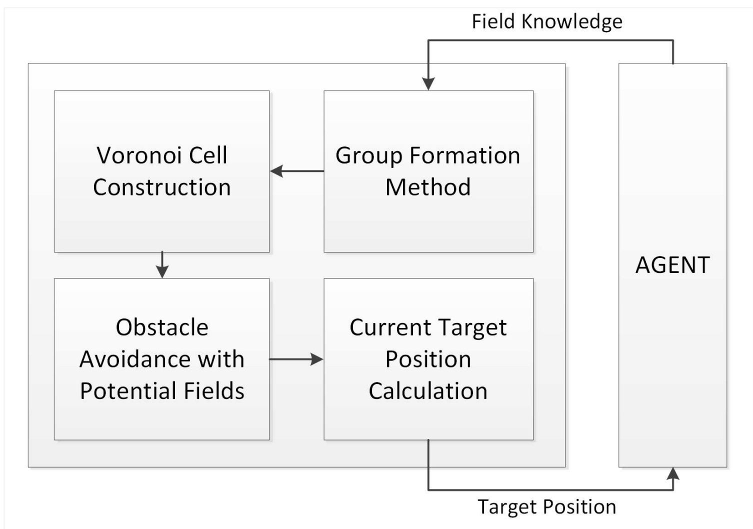


Figure 3.1: General structure of the distributed team formation method.

The Partial Fourier Series (PFS) model is used as the motion model for RoboCup 3D SSL beeStanbul team software [31], which is also used in experiments for the methods proposed in this thesis. Different types of body motions, including straight walks (forward, back, diagonal and side walk), inward turn, outward turn, rotate, kick and stand-up are available for agents. Figure 3.2 shows the overall two-layered software architecture of beeStanbul team [32]. Server layer uses rcssnet library for communication with Simspark server, which is also provided by Simspark. Agent model layer uses the abilities of agents to localize themselves on the field, create strategies and plans, process the information about the environment and move along the field.

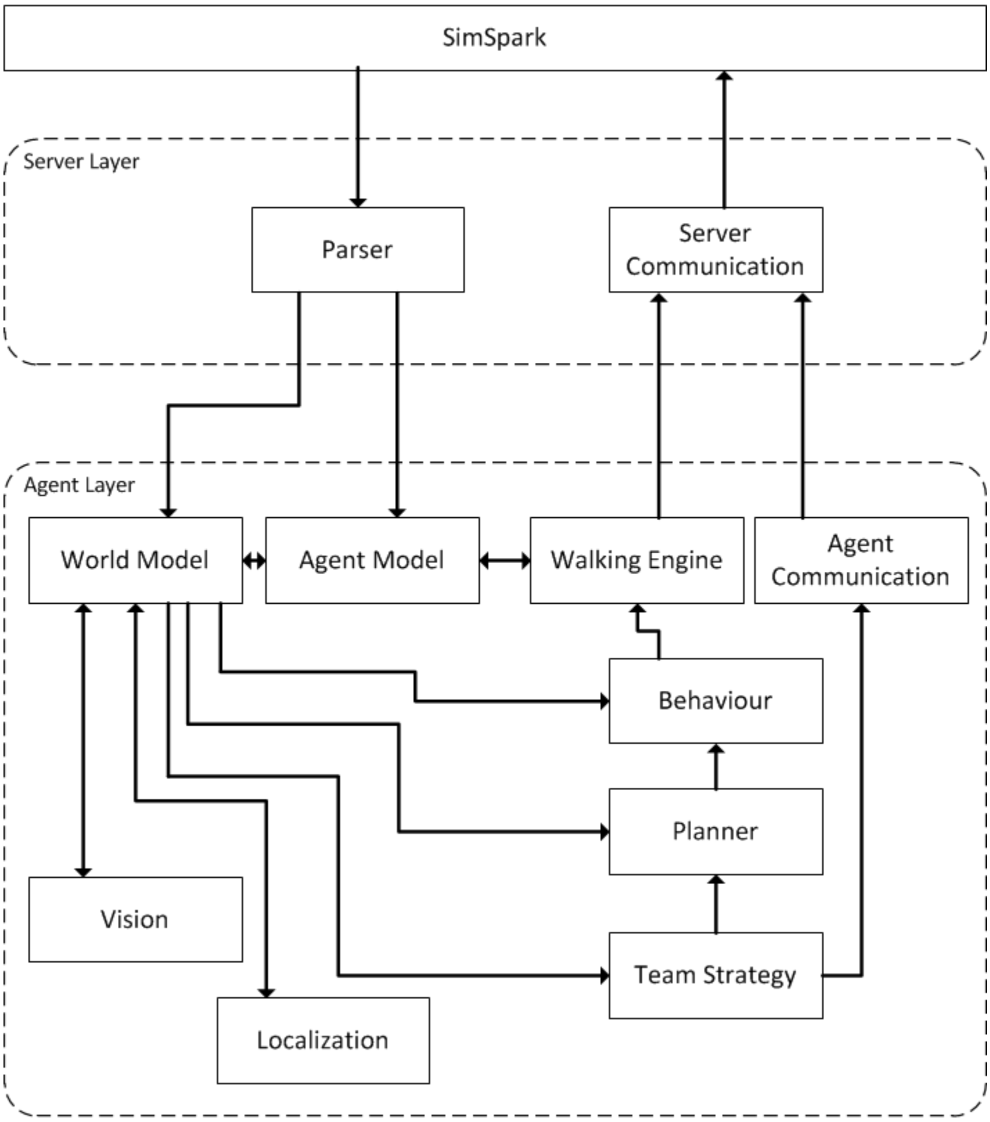


Figure 3.2: Overall software architecture of beeStanbul team [32].

Based on the assigned role of an agent, the corresponding planner is activated. Each plan has a set of behaviors which activate a set of motions. Figure 3.3 shows the decomposition of an example plan (dribble-to-goal) for an agent that has the forward role. In a plan, lower level components are hierarchically activated by selection at a higher level. At the lowest level, primitive actions are selected and the corresponding motion commands are sent to the server. In low level computations, messages received from the server are parsed, refined and passed to the higher levels as useful information.

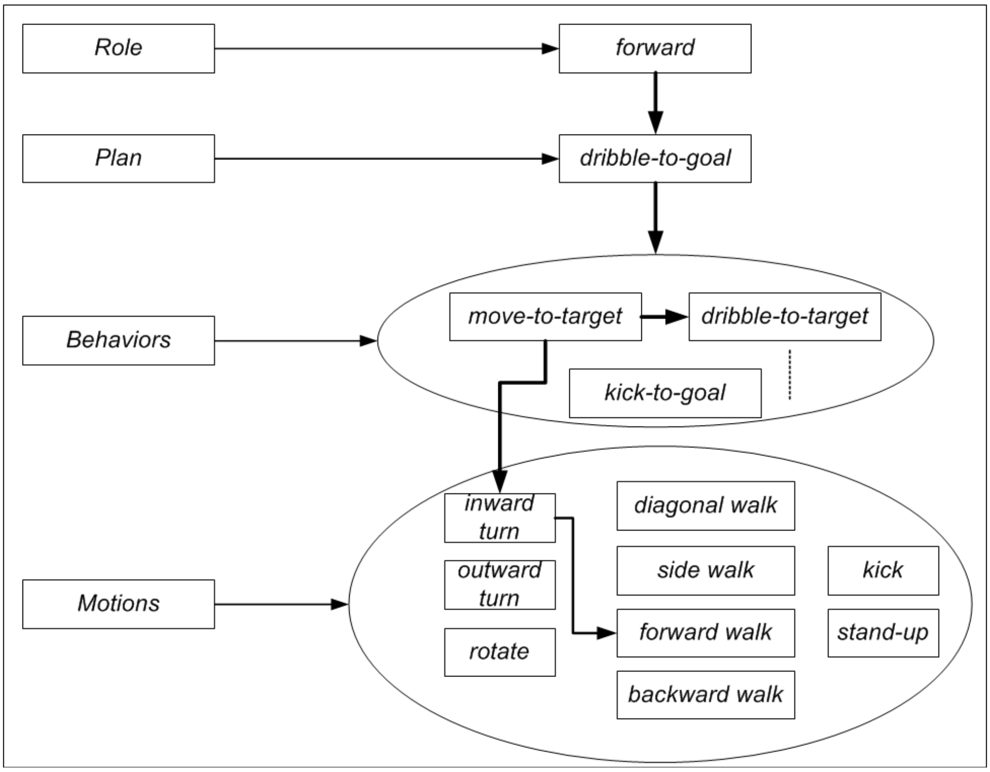


Figure 3.3: The decomposition of an example plan for an agent that has the forward role.

There are three different planners designed for four different roles. Forward planner uses a FSM that moves to the ball regardless of the teammates and chooses an appropriate action from kick and dribble behaviors. Goalkeeper planner always stays in team's penalty area and tries to form an obstacle against opponents to prevent them from scoring. Final planner is a FSM both used by defenders and midfielders. Figure 3.4 shows the FSM for midfielders and defenders. Only difference in the FSM for both roles is the target calculation using Voronoi cells, which will be explained in this chapter.

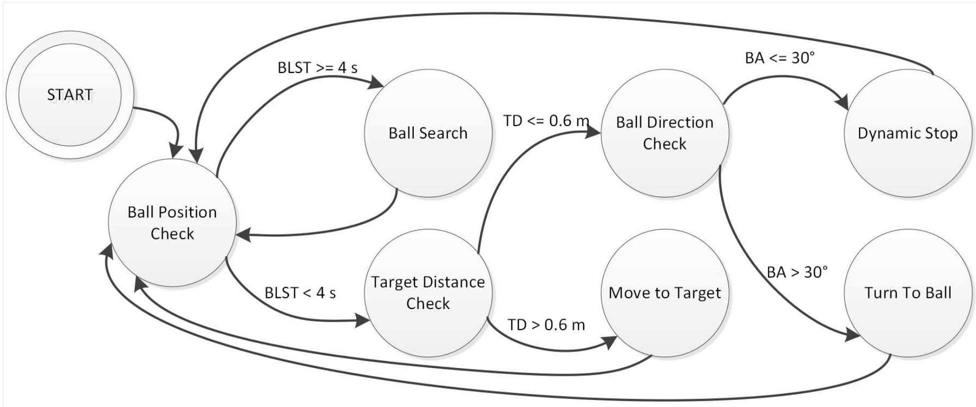


Figure 3.4: Finite State Machine for the agents with midfielder or defender role.

3.1 Group Formation Methods

The current setup of the RoboCup 3D SSL involves nine team players in each team. A single player is assigned to the goalkeeper role. The strategy in this thesis divides the rest of the team into two groups, namely, defenders and attackers, for offensive and defensive strategies. Attackers are formed for constructing an attacking formation around the ball and scoring a goal whenever possible. This group involves the forward agent and the midfielders which usually target to control the ball and score a goal. Defenders are formed for blocking and constructing a defensive obstacle against the opponent team. This strategy prevents the opponent team from scoring. Two different methods are used to achieve group formation, namely static and case-based group formation.

The maximum bandwidth for RoboCup 3D SSL agents is 20 bytes for each cycle which can be used by a single agent. This communication channel can be used by a single agent to send role assignments for group formations. The goalkeeper is selected as the captain of the team because it has the widest line of sight of the field. The time period to communicate is shared effectively by each agent. The goalkeeper is responsible to send group formation messages according to the results of the case-based grouping method while other agents send their costs to reach at the ball position. If agents fail to communicate with each other, they behave according to their field knowledge and observations.

Figure 3.5 shows each agent's role selection strategy and its decision for joining to a group (d_i: i^{th} lowest Euclidean distance between the ball and the agents in the viewpoint, d: Euclidean distance between the agent and the ball, k: the maximum number of attackers). The agent that is closest to the ball assigns itself the forward role and directly looks for ball possession to score against the opponent. The other agents in the attackers group take midfielder role and follow the forward agent in a close proximity for handling passes or failures. The defenders position themselves at a distance behind the ball to defend the goal. Goalkeeper continuously sends the ball position and the numbers of the teammates that are going to be in attackers group according to the case-based grouping method. If a player hears its number in the latest message string, it positions itself as one of the attackers. Otherwise, it takes the defender role. If the goalkeeper fails to send messages to the others due to falling down or any other reason, it sends a failure message to inform them. If the other players don't hear any messages or hear the failure message, they act on their behalf through observation. In this case, five players closest to the ball assign themselves the attackers group and three players the defenders group in a static manner. If a player observes five teammates that are closer to the ball than itself, it acts like a defender and uses a defensive Voronoi cell calculation method. In the opposite situation, it acts as either the forward agent or a midfielder agent. Midfielder agents calculate Voronoi cells to determine their targets while the forward agent directly targets the ball.

3.1.1 Static group formation

This group formation method is used to have a fixed number for both attacker and defender groups. In this method, there are five attackers and three defenders regardless of the game situation. These numbers are determined according to the previous experiments and don't change in runtime. Goalkeeper sends messages of five teammates that are closest to the ball. Closest one of the five becomes the forward and rest of them are assigned to midfielder roles to construct their Voronoi cells. Three teammates that aren't on the attacker message are assigned as defenders and calculate their defensive Voronoi cells.

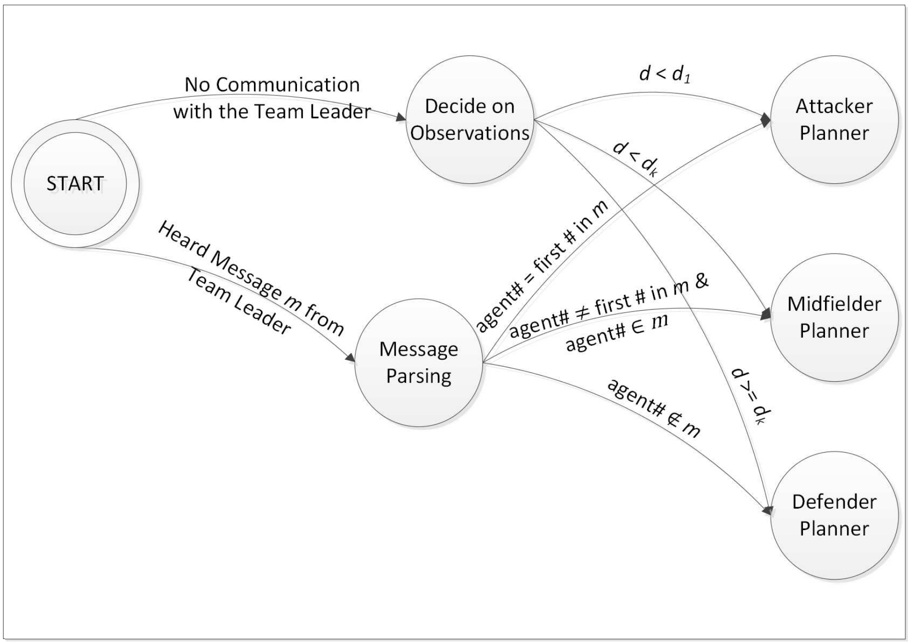


Figure 3.5: FSM for agents' group formation behavior according to the team captain messages or observations.

3.1.2 Case-based group formation

A case-based group formation method [29] is used to determine the number of defender agents and midfielder agents dynamically. Since two agents are assigned to the goalkeeper and the forward roles, the remaining seven agents are to be assigned to these roles. Instead of using a predetermined number for these roles, a case-based method is applied to determine the best separation.

The current game score and the positions of agents and the ball are considered in the problem description of cases. The general structure of cases is shown in Equation **(3.1)**. Each case corresponds to a certain number of agents for defenders and midfielders. For example, if the team is losing in the middle of the game, more players could be assigned as midfielders to tie the game with more attacker agents while taking the risk of conceding a goal.

$$\begin{aligned}&\text{Case} = (\text{Ball Position}, \text{Game Score}, \text{Agent Positions},\\&\text{Number of midfielders}, \text{Number of defenders})\end{aligned} \quad \textbf{(3.1)}$$

The case library initially involves 12 predetermined cases which are allowed to be modified in runtime according to the success of applying them.

3.1.3 Probabilistic group formation

Adaptivity against opponents is beneficial in robot soccer. Training the agents with information from previous experiments can provide more suitable decisions for group formation. To achieve a team that adapts itself according to the opponents, a probabilistic learning model is implemented. An iterative probabilistic learning algorithm based on Bayesian reasoning is used for this method. In Bayesian reasoning, for hypothesis h and observation o:

- $P(h)$ is the prior probability of hypothesis h
- $P(h|o)$ is the likelihood probability
- $P(o|h)$ is the posterior probability

In the proposed method, the best hypothesis is the one with the highest probability, given the observations of the domain. Equation **(3.2)** shows the formula of the probability of hypothesis h, given the observation set of o. h is an element of the set of all the hypotheses on the domain, given as H.

$$P(h|o) = \frac{P(h|o)P(h)}{P(o)} \tag{3.2}$$

Since $P(o)$ value is the same for all the hypothesis because it is the current observation of the environment, hypothesis with the maximum formula can be found with Equation **(3.3)**. This is called maximum a posteriori hypothesis and shown in the equation below as h_{MAP}.

$$h_{MAP} = argmax(P(o|h_i)P(h_i)) \, , h_i \in H \tag{3.3}$$

For the group selection, there are eight different hypothesis because one agent is defined as goalkeeper and one is the forward, so the remaining seven agents can be assigned in eight ways (seven defender - none midfielder to none defender - seven midfielder). $P(h_i)$ is called the prior probability for hypothesis i. These values are given to the algorithm at the start of the game and changes online with success or failure of the hypothesis. Prior values for the hypothesis are ($P(XdYm)$ means the prior probability for X defenders and Y midfielders): $P(7d0m)$ = 0.05, $P(6d1m)$ = 0.1, $P(5d2m)$ = 0.15, $P(4d3m)$ = 0.2, $P(3d4m)$ = 0.2, $P(2d5m)$ = 0.15, $P(1d6m)$ = 0.1, $P(0d7m)$ = 0.05. These values ensure that extreme situations like assigning all the agents for midfielder or defender has a lower probability and has less chance to be selected. α parameter is used to train prior probability values of hypotheses and defined as 0.007. $P(o|h_i)$ is the likelihood of the formula and these values are obtained by training of the team against the opponents.

An initial training phase is executed for learning. Ten training matches were run to gather likelihood values. Likelihoods are always updated within the algorithm, thus algorithm is still trained while being executed in matches. Observations for the algorithm include the ball position and the positions of both teammate and opponent agents. Algorithm 1 explains the probabilistic group selection method that uses an iterative learning, which is implemented to the beeStanbul team software. The method is called with 2 s periods. In each period a new hypothesis is decided and prior and likelihood probabilities are updated.

Algorithm 1 takes the ball position, the agent positions, ball position from last iteration and seen agent counts from teams. The field is divided into three 7×14 areas and agent positions are defined according to the area they are in. In the algorithm, prior probability value for the previous selected hypothesis is updated first. If the ball is 1 m closer to the opponent goal than the previous iteration, hypothesis is ranked as successful and it's prior probability is increased by α parameter. To keep the total prior probability value at 1, other hypotheses' prior probability values are decreased by $\alpha/7$. If the hypothesis is not ranked as successful prior probability is decreased by α and others are increased by $\alpha/7$. After that likelihood probabilities are updated according to the observations. These values refer to the previous statistics that counts percentage of different observations seen before using the selected hypothesis. After the updates, new hypothesis selection phase starts. If seen teammate count or opponent count is below 5, hypothesis with the highest prior value is selected. If both of seen agent counts are equal to or higher than 5, hypothesis with the highest multiplication of prior and likelihood probabilities is selected. The hypothesis that is selected according to the second condition can also be called as maximum a posteriori hypothesis.

Input:
P_B: the ball's current seen position (b_x, b_y)
P_{B_p}: the ball's position from previous update (b_{px}, b_{py})
P_{T_i}: the current position of teammate i (p_{tix}, p_{tiy})
P_{O_i}: the current position of opponent i (p_{oix}, p_{oiy})
tc: seen teammate count
oc: seen opponent count
o: combined and encoded observations of (P_B, P_{T_i} and P_{O_i})
$P(h_i)$: current prior probability of hypothesis i
$P(o|h_i)$: current likelihood probability of hypothesis i
pk : previous hypothesis
Output:
k : selected hypothesis

// Prior probability update
if $(b_x) > (b_{px})$ + *1 m* **then**
  $P(h_{pk})$ = $P(h_{pk})$ + α
  foreach k *where* $k \neq pk$ **do**
    $P(h_k)$ = $P(h_k)$ - α/ 7
  end
end
else
  $P(h_{pk})$ = $P(h_{pk})$ - α
  foreach k **where** $k \neq pk$ **do**
    $P(h_k)$ = $P(h_k)$ + α/7
  end
end
// Likelihood probability update
update $P(o|h_{pk})$
if $tc < 5$ **or** $oc < 5$ **then**
  return k **where** $P(h_k)$ is $max(P(h_i))$
end
else
  return k **where** $P(h_k)$x$P(o|h_k)$ is $max(P(h_i)$x$P(o|h_i))$
end

Algorithm 1: Iterative probabilistic learning algorithm based on Bayesian reasoning for selecting the maximum a posteriori hypothesis.

3.2 Distributed Target Selection by Adaptive Voronoi Cell Construction

The midfielder and defender agents need to position themselves for maintaining close proximity to the forward agent and defending the goal respectively. This is accomplished by a distributed Voronoi cell construction approach in which each agent calculates its own cell independently from that of the others. Therefore, every agent has a differently shaped cell and these can overlap.

In conventional Voronoi diagram computation, Fortune Algorithm (FA) [33] is used. The approach in this thesis differs from FA in the construction of the final cell. The initial cell is constructed by considering the ball location and then, iteratively narrowed down to get the final cell for the agent. In FA, the lines that construct the cells are perpendicular bisectors of the line segment between teammate locations. In the approach of this thesis, a line from the corresponding teammate position parallel to the perpendicular bisector is used. The main procedure for the distributed cell construction approach for each agent is given in Algorithm 2.

3.2.1 Cell formation

Algorithm 2 takes ball position, current position of the agent that runs the algorithm and the agent's teammate positions as input. It also takes three parameters that are given before for calculations. Those parameters are the midpoint of team's current goal line for defensive Voronoi cell calculation, and distance limits for cell initialization and the crowding rule. Cell initialization distance limit, which is represented with l in the algorithm, is defined as 4 meters and used to define the length of side edges of the initial cell and drawn as 2 m to the left and 2 m to the right of the agent. Length of the other two edges change according to the distance between the agent and the ball. The reason that l is defined as 4 meters is to avoid unnecessary calculations for the teammates that are far away and also suit to the crowding rule parameter, which is 2 m. With 2 meters to the left and the right of the agent, there is no need for crowding precautions unless an agent is closer than 2 m. Crowding rule distance limit is represented as m in the algorithm and given the value of 2 m, because agent that is closer to another agent than 1 m might crowd the area and positioned out of the field as a penalty. With the current motion model, the agents can't change direction instantly. To suit the response time, calculations for crowding rule precautions start when an agent comes closer than 2 m, which defines the m parameter. After getting the inputs and the parameters, initial cell is calculated. There are two different types of initial cells according to the role of the agent. Defenders draw a line between their position and the midpoint of the team goal line center and the ball position, while midfielders draw it between their position and the ball position. This line is used for the initial cell's edge calculations. Two edges are drawn parallel to this line with the same length, one is $l/2$ m to the left and one is $l/2$ m to the right of it. One of the other edge connects the two other, with being perpendicular bisector to the others and including agent's position. Last edge connects the remaining free corners two each other and complete the initial cell. The cell structure for the initial cell is stored in the memory as set of edges for calculations, like the latter calculated cells. After the initialization of the cell, a loop for each teammate in the view point of the agent begins and alters the cell with the calculations within the loop. If there is no teammate in the view point, initial cell becomes the final cell. In each cycle of the loop, if the seen agent's distance is bigger than 2 m., there is no need for a crowding rule precaution and a line is drawn form the seen agent's position where the line is perpendicular bisector of a line between the agent and the seen one. If the seen agent is closer than 1 m and still not positioned out with a penalty, same line is drawn parallel to the previous one, but 1 m away from the seen agent to make the agent back off from the other. If the seen agent's distance is between 1 m and 2 m, the line still is drawn parallel to the first one, but it is drawn from where the midpoint of the line between the agent and this line is 1 m away from the seen agent. After the line is drawn with one of these three rules, intersections of the line with the current cell is examined. If there is no intersection, the cell remains the same for the next cycle. If it intersects with the cell, it divides the cell into two different cell, because shape of the cell is always a convex polygon. In this situation, the cell that contains the agent is selected for the next cycle. After all the iterations in this loop, shape of the final cell is determined. Center of this cell is calculated with the average of it's vertice coordinates. This simple calculation works for the convex polygons so it can be used in this algorithm. It has a low computational cost thus helps the algorithm's simplicity. Finally the center is altered with Potential Fields method and this makes a minor change to the cell center for avoiding collisions

Input:
P_B: the ball's last seen position (b_x, b_y)
P_i: the current position of a_i (p_{ix}, p_{iy})
P_G: the midpoint of the team's goal line
P_S: the initial cell start point
l: the distance limit for cell initialization (4 m)
m: the distance limit for the crowding rule (2 m)
Output:
$cell_k$: the Voronoi cell for a_k
c: the center of $cell_k$
t_k: target destination of a_k

L_i: Line between L_{i1} and L_{i2}
m_{L_i}: Slope of L_i
if $agent = midfielder$ **then**
    $P_S = P_B$
end
if $agent = defender$ **then**
    $P_S = (P_B + P_G)/2$
end
L_0: Line between P_S and P_k ($L_{01}P_S$, $L_{02} = P_k$)
L_1: Line between L_{11} and L_{12} **where** ($L_1 \perp L_0$), $P_S \in L_1$,
    $dist(L_{11}, L_{12}) = l$ $dist(L_{11}, P_S) = dist(L_{12}, P_S) = l/2$
L_2: Line between L_{21} and L_{22} **where** ($L_2 \perp L_0$) , $P_k \in L_2$,
    $dist(L_{21}, L_{22}) = l$ $dist(L_{21}, P_S) = dist(L_{22}, P_S) = l/2$
L_3: Line between L_{11} and L_{21}, **where** m_{L_3} = m_{L_0}
L_4: Line between L_{12} and L_{22}, **where** m_{L_4} = m_{L_0}
create $cell_k$ (enclosed area between L_1 , L_2 , L_3 and L_4)
foreach *Teammate* $(a_i \neq a_k)$ *in point of view* **do**
    p: Coordinate to draw line according to a_i
    L_p: Line between P_k and P_i
    if $dist(P_k, P_i) > m$ **then**
        $p = p_i$
    end
    else if $dist(P_k, P_i) \leq m/2$ **then**
        $p = x$ **where** $x \in L_p$, $dist(x, P_i) = m/2$
    end
    else
        $p = x$ **where** $x \in L_p$, $dist(x, P_i) = dist(P_k, P_i) - m/2$
    end
    create line L **where** $p \in L$, $L \perp L_p$
    if L *intersects* $cell_k$ **then**
        $cell_k$ = $cell_j$ ($j \in 1, 2$ and $P_k \in cell_j$)
    end
end
calculate center coordinate of $cell_k$ (c)
calculate t_k by altering c according to obstacles using Potential Fields

Algorithm 2: Voronoi cell construction for agent a_k

with opponent players. After this alterations, the altered center cell becomes the target for the agent and this target is given as output to the agent's planning algorithm.

The time complexity of the algorithm is $O(n^2)$ where n is the number of agents in the team. Figure 3.6 shows the iterations for calculating the final cell and the corresponding target as the center of this cell for agent #2 (a_2), which is a midfielder and draws its initial cell according to the ball position. As mentioned before, only teammates in the viewpoint of the agent are considered. The area that is out of a_2's point of view is shown as the shaded area. Figure 3.6 (a) shows the initial cell construction by considering the ball position (P_B). In Figure 3.6 (b), (c), and (d), the cell is modified according to the locations of a_5, a_6, a_8 and a_9, respectively. The line for a_9 doesn't have any intersection points with the current cell, so it doesn't make any changes in the cell. The final Voronoi cell of a_2 is shown with the red frame and the center of that cell is marked with a red point in Figure 3.6 (d).

Algorithm 2 is used for both midfielders and defenders. Defenders create their cells with the same algorithm, but their initial cell is calculated according to the midpoint of the line connecting the ball position and the center of the team's goal position while midfielders use the ball location. Figure 3.7 shows the initial (Figure 3.7 (a)) and final (Figure 3.7 (b)) cells of a defender (a_7) in defensive Voronoi cell calculation. Grey point shows the midpoint of the line between the ball and the team goal's center. Red point indicates the center of the Voronoi cell. Only the line for a_2 intersects with the cell of a_7 so the only modification in the cell is caused by it.

3.2.2 Crowding rule

After constructing the cell for itself, each agent determines the center of the cell as its new target. Agents become closer to each other by using this strategy, which is more beneficial for attacking in soccer. However, RoboCup 3D SSL league have some rules to prevent crowding an area with multiple agents. According to these rules, a player is repositioned out of the field if it is in a circle that has a radius of 1 meter with two other teammate players. In order to overcome the situation where there is a teammate closer than 2 m, the cell is adjusted to keep at least 1 m distance from that teammate. Applying these alterations on the construction of a cell, the distance to any teammate is

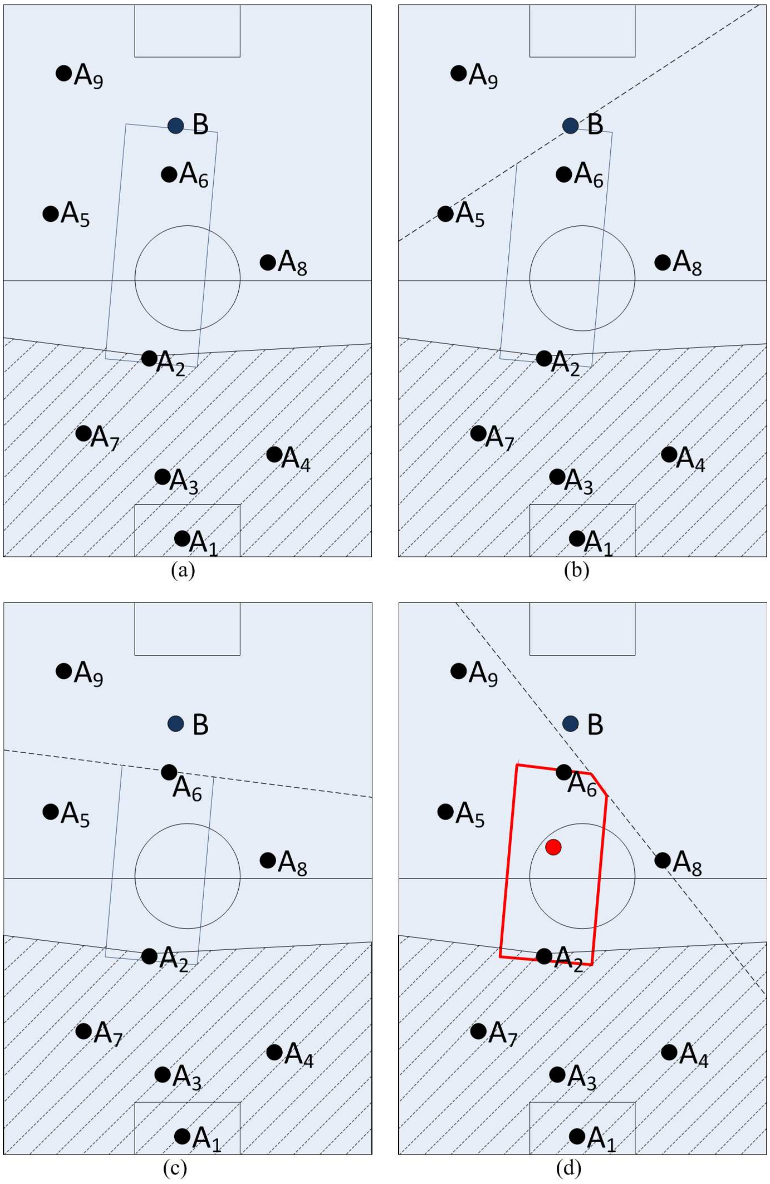
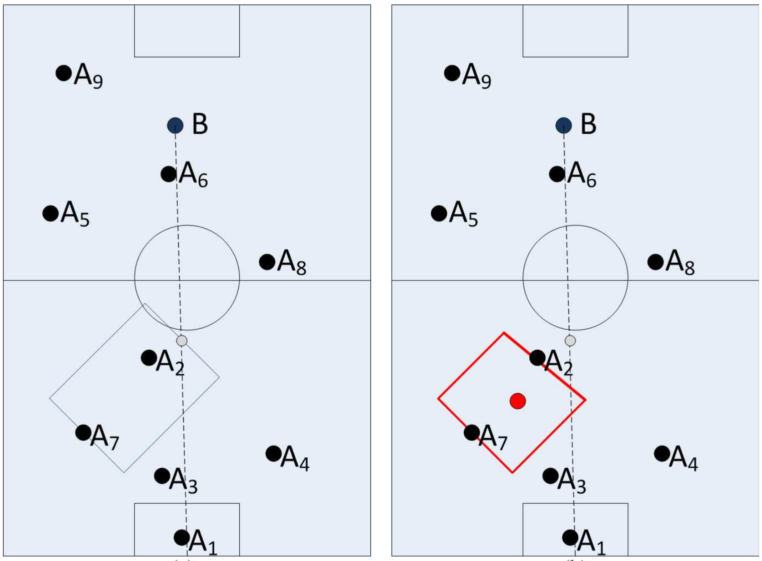


Figure 3.6: Step-by-step calculation of the Voronoi cell for a_2.

guaranteed to be greater than 1 m. In RoboCup 3D SSL, each agent has a 120 degrees angle of view. Therefore, agents only consider the positions of teammates they can see and the ball's last seen position to construct their Voronoi cells. Euclidean distance is used for distance calculations. Equation **(3.4)** shows the distance formula for two coordinates $(A(x_1, y_1), B(x_2, y_2))$ that is used in Algorithm 2.

$$dist(A, B) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \qquad \textbf{(3.4)}$$



(a) (b)

Figure 3.7: Initial and final cell for a defender agent (a_7).

3.2.3 Target selection

Agents continually form their Voronoi cells and move toward their targets. Due to the distributed calculation of cells, a complete diagram is not formed. Cells of different agents may overlap in some situations, but the relevant precautions taken to overcome the crowding rule and the Potential Fields Method ensures that the targets are not too close to each other. This approach also protects agents from collisions.

4. EXPERIMENTAL RESULTS

Two sets of experiments are set to analyze the performance of the proposed team strategy. Experiments are set in the Simspark environment to analyze the overall performance of the approach as compared to existing methods, and particulary its robustness to communication failures.*rcssserver3d* is used for the simulation, which is the official server software for RoboCup 3D SSL competitions and RoboViz [3] as a visualization tool.

4.1 Algorithm Comparisons in Terms of Ball Possession and Position

In soccer game, keeping possession of the ball is one of the key factors for scoring a goal. The first experiment targets to analyze this issue and the average position of the ball in the field. Ball position fields are determined by dividing the 14×21 m field horizontally into 3 equal areas (defense, midfield and forward) each 14×7 m. The area next to the team's goal is called the defense area, the area next to the opponent's goal is called the forward area and the area between these two fields is the midfield area. The proposed method that uses group formation with probabilistic learning is compared to case-based group formation, the earlier method Situation Based Strategic Positioning (SBSP) that is used in RoboCup German Open 2011 competitions by beeStanbul team, the Voronoi cell based method which uses a static grouping strategy instead of case-based grouping and Dynamic Positioning based on Voronoi Cells (DPVC) method [11]. In DPVC, Voronoi cells are used to scatter the agents throughout the field. In SBSP, each agent has a predetermined role and they shape formations according to predefined positions around the ball based on their roles. All the approaches are applied on the latest motion model of the beeStanbul team software.

RoboCup 2011 binaries of Nao Team Humboldt [34] and FC Portugal teams are used as an opponent because the motion model of these teams are close in speed to that of beeStanbul's PFS model. Also they have successful defensive team formation strategies which blocks the opponent. Ten games are runned for each method against Nao Team Humboldt and FC Portugal. A snapshot is shown in Figure 4.1 from an instance during these games where the blue agents are from beeStanbul team and the red agents from Nao Team Humboldt. This figure also illustrates the Voronoi cell of each midfielder agent in beeStanbul. In Figure 4.1 blue polygons indicate the cells of the agents and red circles indicate their centers. The agent closest to the ball assigns itself the forward role while the rest of them are assigned to the midfielder role in the attackers group. As can be seen from the figure, the Voronoi cells of midfielders may overlap as it is allowed. However, target positions as the centers of these cells are always different if agents see each other.

Figure 4.1: An instance from a game using rcssserver3d of Simspark for simulation and RoboViz for the visualization.

Table 4.1 and 4.2 show the overall results of methods with combination of group formation methods and Voronoi cells against Nao Team Humboldt and FC Portugal respectively. Table 4.3 and 4.4 show the results of SBSP and DPVC methods against the same teams with the same order. These results illustrate that, the approach that used probabilistic formation method outperforms the previous approaches and DPVC in terms of ball possession, keeping control of the ball and carrying the ball to the opponent's area. The approach that uses Voronoi cells combined with a static grouping method also gives good results but using a case-based method for grouping further improves the overall performance. According to these results, the key factors for the success of the proposed team strategy can be listed as the distributed online construction of Voronoi cells and dynamic positioning to the centers of these cells. Even when the forward agent falls over during an attack, by this approach, midfielders maintain close proximity with the forward agent and regain control of the ball. Communication is also used to dynamically form attacker and defender groups. Another advantage of the new approach is the unpredictability of the team strategy as a competitive strategy. There are not fixed formations that can be learned and predicted by the opponent during a game.

Table 4.1: Comparison among the Voronoi cell based methods used in beeStanbul team software against Nao Team Humboldt in terms of ball possession and ball position.

	Distributed Voronoi Approach with probabilistic learning	Distributed Voronoi Approach with case-based grouping	Distributed Voronoi Approach with static grouping
Ball Possession Ratio	55.35712% (σ = 0.04172)	53.23177% (σ = 0.04441)	52.92768% (σ = 0.08248)
Ball in Own Area	17.65291% (σ = 0.09266)	15.95018% (σ = 0.04423)	17.11815% (σ = 0.11013)
Ball in Midfield	30.30576% (σ = 0.09335)	37.48712% (σ = 0.08912)	38.37606% (σ = 0.14637)
Ball in Opponent Area	52.04133% (σ = 0.09490)	46.56270% (σ = 0.11047)	44.50579% (σ = 0.18259)

Table 4.2: Comparison among the Voronoi cell based methods used in beeStanbul team software against FC Portugal in terms of ball possession and ball position.

	Group formation approach with probabilistic learning	Group formation approach with case-based method	Group formation approach with static method
Ball Possession Ratio	54.69275% (σ = 0.04059)	53.69744% (σ = 0.04118)	51.41621% (σ = 0.03747)
Ball in Own Area	30.03812% (σ = 0.07485)	33.03172% (σ = 0.07557)	36.63184% (σ = 0.18176)
Ball in Midfield	22.64490% (σ = 0.05850)	24.56132% (σ = 0.09621)	31.85806% (σ = 0.15092)
Ball in Opponent Area	47.31698% (σ = 0.12993)	42.40696% (σ = 0.14406)	31.51010% (σ = 0.17559)

As expected, the performance of DPVC is better than that of SBSP in terms of carrying the ball to the opponent's area due to the dynamism. In SBSP, on the other hand, predetermined formations are easy to be predicted by the opponent in a later time step during the game. However, ball possession performance of DPVC is worse than that of SBSP because it scatters the agents throughout the field. In that case, if the motion model of the agents is not fast enough, they may not responsively regain the control of the ball when it is lost.

Table 4.3: Comparison among DPVC and SBSP methods used in beeStanbul team software against Nao Team Humboldt in terms of ball possession and ball position.

	DPVC	SBSP
Ball Possession Ratio	47.83140% (σ = 0.07796)	50.45958% (σ = 0.06877)
Ball in Own Area	27.76341% (σ = 0.13851)	37.04670% (σ = 0.18277)
Ball in Midfield	31.74228% (σ = 0.11037)	33.22326% (σ = 0.10827)
Ball in Opponent Area	40.49431% (σ = 0.13937)	29.73004% (σ = 0.24710)

Table 4.4: Comparison among DPVC and SBSP methods used in beeStanbul team software against FC Portugal in terms of ball possession and ball position.

	DPVC	SBSP
Ball Possession Ratio	46.91492% (σ = 0.03807)	48.85773% (σ = 0.05240)
Ball in Own Area	52.05741% (σ = 0.11720)	45.50916% (σ = 0.14566)
Ball in Midfield	34.85669% (σ = 0.04263)	28.68214% (σ = 0.08804)
Ball in Opponent Area	13.08590% (σ = 0.08210)	25.80870% (σ = 0.06980)

4.2 Probabilistic Learning Experiments

The method that uses iterative probabilistic learning algorithm based on Bayesian reasoning, combined with Voronoi cells is tested for group selection decisions. As mentioned in Chapter 3, the method always trains itself with new observations online. Before these, each defender-midfielder ratio is trained with 10 matches to give more appropriate results on selection. Three sets of prior probabilities are used for the tests. For each set, 10 matches are made to analyse which hypotheses are tend to be selected by the team. As explained in the third section, prior values are given as: $P(7d0m)$ = 0.05, $P(6d1m)$ = 0.1, $P(5d2m)$ = 0.15, $P(4d3m)$ = 0.2, $P(3d4m)$ = 0.2, $P(2d5m)$ = 0.15, $P(1d6m)$ = 0.1, $P(0d7m)$ = 0.05 for the first set. Figure 4.2 shows the defender-midfielder selection percentage statistics for ten matches against Nao Team Humboldt for the first set. "D" represents defender and "M" represents midfielder on the figure (3D4M means 3 defenders and 4 midfielders are selected for that instance). Selection is made with 2 second periods in a match. Results show that 3D4M selection is made mostly and 4D3M follows it closely. These two hypothesis has the highest prior probabilities and selection of these two is reasonable. The hypotheses with less defenders are selected slightly more than the hypotheses with less midfielders. This means the team mostly stayed on the opponent field, thus needed less defenders.

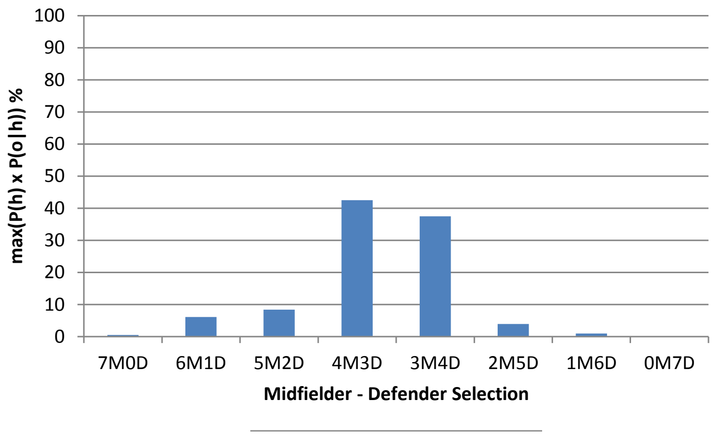


Figure 4.2: Defender-midfielder selection statistics for probabilistic learning method for the first set of prior probabilities.

In the second set of prior probabilities, each hypothesis is given the same value, which equals to 0.125. Results of this set is shown in Figure 4.3. Results show that there is a tendency to select the hypothesis with 3 defenders and 4 midfielders or 4 defenders and 3 midfielders.

Third set is the reverse of the first set, so prior probability values are: $P(7d0m)$ = 0.2, $P(6d1m)$ = 0.15, $P(5d2m)$ = 0.1, $P(4d3m)$ = 0.05, $P(3d4m)$ = 0.05, $P(2d5m)$ = 0.1, $P(1d6m)$ = 0.15, $P(0d7m)$ = 0.2 . Results of the third set is given in Figure 4.4. Prior values are changed according to their success in runtime as mentioned before. But likelihood values are obtained entirely from the previous observations and the observations can lead to wrong selections with unreasonable prior probability values

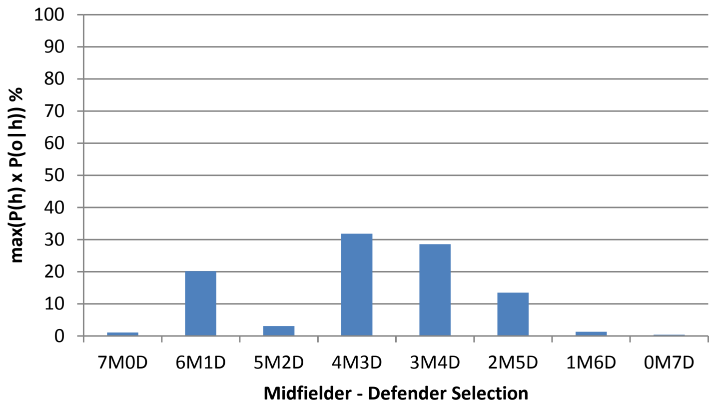


Figure 4.3: Defender-midfielder selection statistics for probabilistic learning method for the second set of prior probabilities.

like this set. To improve the performance with this set or the others, likelihood values can be altered by human experts.

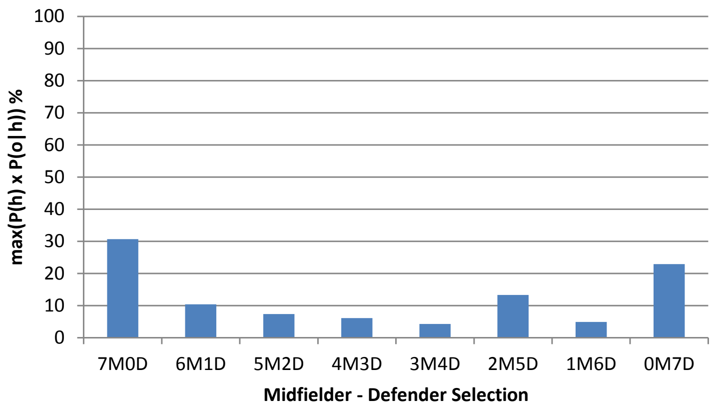
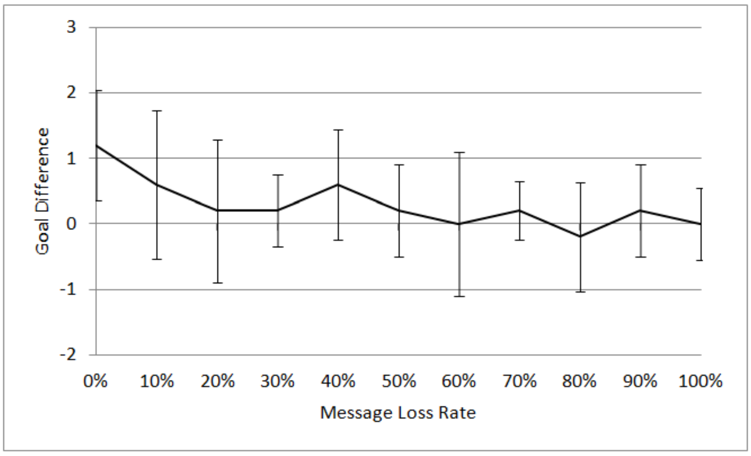


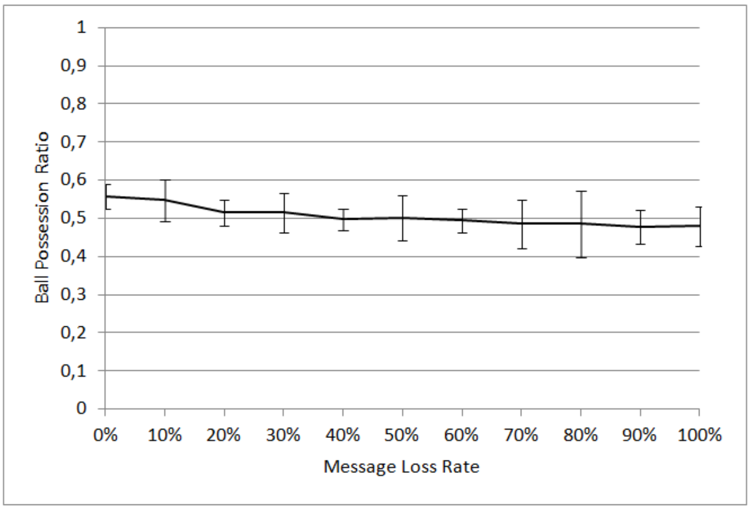
Figure 4.4: Defender-midfielder selection statistics for probabilistic learning method for the third set of prior probabilities.

4.3 Robustness to Communication Failures

In the third set of experiments, we measure the performance of our method for different message loss rates (Figure 4.5). Figure 4.5 (a) shows average goal difference (positive values show the scores in favor of beeStanbul team) and Figure 4.5 (b) indicates average ball possession ratio of beeStanbul team. In rcssserver3d, a team is allowed to send a message periodically in 0.06 seconds. In the current implementation of this thesis, all the available messaging periods are used in order to perform better. In this experiment, communication is manually switched off based on the message loss rate to simulate communication failure. The reported results indicate that the method is robust to communication failures for most of the instances. Even for no communication cases, agents can still make decisions and calculate their Voronoi cells based on observations and they position themselves to appropriate target locations for maintaining an efficient formation. This is achieved by the distributed implementation of Voronoi cell construction. However, as expected, ball possession performance is degraded gradually with the worst value as 47.7%.



(a)



(b)

Figure 4.5: Test results for each message loss rate presented as the averages of 5 games against Nao Team Humboldt.

5. CONCLUSIONS AND FUTURE WORK

Adaptive team formation methods for RoboCup 3D Soccer Simulation League is presented in this thesis. Previous works on the research area are surveyed and the distributed team formation approach is explained. Main approach in this thesis uses a variation of Voronoi cells to position the agents. Voronoi cells have been used for team formation in a previous approach within RoboCup competitions but in this approach, computations are made by a supervisor thus the method is not distributed. There are some other approaches like Dominant Region diagrams or Case-Based Reasoning for positioning the robots but they aren't completely distributed and fully observation or a high-bandwidth communication is needed for these methods. Since RoboCup 3D SSL only provides partially observation and limited communication for the agents, these methods can't be used effectively for the given domain. There are some simpler methods like Situation Based Strategic Positioning and playbook based positioning and planning but these methods need human contribution for deciding the plays or roles thus they are not adaptive. RoboCup 3D SSL is a non-deterministic, noisy environment with the presence of an opponent team and adaptivity provides a better performance for teams.

Voronoi Cell decomposition method in this thesis is combined with a grouping algorithm to make the team strategy more efficient. Adaptive Voronoi Diagrams are combined with different group formation methods controlled by an agent (i.e., goalkeeper) through explicit communication. The agents are divided into defender and attacker groups according to this agent's messages. To select the number of agents in both groups, different algorithms are suggested. First one is the static grouping which the number of agents for the groups are pre-determined and doesn't change while playing. Second method uses a case-based approach which provides more adaptivity for grouping. Cases are pre-determined by the user and updated according to the agent and ball positions. A case determines the number of agents for defenders and attackers for an instance in the match. Last suggested approach uses Bayesian Learning method for group selection, so human contribution isn't necessary with this approach. With this method, probability to select the number of agents for the groups are updated through observations. The agent count hypothesis that gives the highest probability value is selected according to the observations.

The proposed Voronoi cell-based formation generation method requires less computational cost than the standard Voronoi Diagram generation. The ball position is also taken into account during these calculations. Agents calculate their Voronoi cells in a distributed manner. If they are attackers, they aim to be crowded around the ball to possess it. Agents try to position themselves between team's goal and the ball to prevent opponent attacks, if they are defenders. To calculate Voronoi cells, agents create an initial cell according to the ball position and their position. After the initialization, the cell is altered according to teammate positions. With these changes, agents try to be at a similar distance with every teammates they see. While keeping the distance, their cells continuously change and they aim a position near ball if they are midfielders, and a position between the ball and their goal if they are defenders. Algorithm also provides precautions for the crowding rule of competitions which penalties the agents that are too crowded in an area. After the target is selected, it is altered with Potential Fields method for obstacle avoidance.

The proposed methods are compared among themselves and some earlier methods ("Situation Based Strategic Positioning" and "Dynamic Positioning Based on Voronoi Cells"). The results illustrate that the proposed approach outperforms the previous approaches in terms of ball possession and ball position. Further experiments showed that the Voronoi cell based method is also robust to the communication failures and it can still be efficient when no communication available. The robustness of the algorithm is provided with making the agents decide on their own observations in failure situations. Group selection statistics are also investigated for the method that uses Bayesian Learning for grouping the agents.

In the future work, the team leader is planned to be changed to determine the group behaviors dynamically in run time. In order to provide a better formation, the captain role is planned to be assigned to the agent that has the best angle of view on the field dynamically. Agent positions could be added into the messages to improve the partial observation ability of the agents. Experiments are intended to be expanded with comparisons with other RoboCup teams. Motion model and speed's effect on the method is also planned to be examined.

REFERENCES

[1] **Url-1**, `http://www.robocup.org`, retrieve date: 15.12.2011.

[2] **Url-2**, `http://simspark.sourceforge.net`, retrieve date: 12.12.2011.

[3] **Stoecker, J. and Visser, U.**, (2011). RoboViz : Programmable Visualization for Simulated Soccer, *RoboCup 2011 Robot Soccer World CupXV*.

[4] **Url-3**, `http://www.ode.org`, retrieve date: 15.12.2011.

[5] **Url-4**, `http://www.aldebaran-robotics.com`, retrieve date: 12.12.2011.

[6] **Balch, T. and Arkin, R.C.**, (1999). Behavior-based Formation Control for Multi-robot Teams, *IEEE Transactions on Robotics and Automation*, **14**, 926–939.

[7] **Stone, P. and Veloso, M.**, (1999). Task Decomposition and Dynamic Role Assignment for Real-Time Strategic Teamwork, Proceedings of the 5th International Workshop on Intelligent Agents V, Agent Theories, Architectures, and Languages, ATAL '98, Springer-Verlag, London, UK, pp.293–308.

[8] **Candea, C.**, **Hu, H.**, **Iocchi, L.**, **Nardi, D. and Piaggio, M.**, (2001). Coordination in multi-agent RoboCup teams., *Robotics and Autonomous Systems*, **36(2-3)**, 67–86.

[9] **Nair, R.**, **Tambe, M. and Marsella, S.**, (2002). Team Formation for Reformation in Multiagent Domains like RoboCupRescue, In Proceedings of the International Symposium on RoboCup, Springer Verlag, pp.150–161.

[10] **Röfer, T.**, (2003). An Architecture for a National RoboCup Team, RoboCup 2002: Robot Soccer World Cup VI, Springer, pp.417–425.

[11] **Dashti, H.**, **Aghaeepour, N.**, **Asadi, S.**, **Bastani, M.**, **Delafkar, Z.**, **Disfani, F.M.**, **Ghaderi, S.**, **Kamali, S.**, **Pashami, S. and Siahpirani, A.**, (2005). Dynamic Positioning Based on Voronoi Cells, RoboCup, volume4020 of *Lecture Notes in Computer Science*, Springer, pp.219–229.

[12] **Nakanishi, R.**, **Murakami, K. and Naruse, T.**, (2008). Dynamic Positioning Method Based on Dominant Region Diagram to Realize Successful Cooperative Play, RoboCup 2007: Robot Soccer World Cup XI, volume5001 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp.488–495.

[13] **Reis, L.P.**, **Lau, N. and Oliveira, E.C.**, (2001). Situation Based Strategic Positioning for Coordinating a Team of Homogeneous Agents, Balancing Reactivity and Social Deliberation in Multi-Agent Systems, LNAI 2103, Springer Verlag, pp.175–197.

[14] **Ros, R.**, **Arcos, J.L.**, **Lopez de Mantaras, R. and Veloso, M.**, (2009). A case-based approach for coordinated action selection in robot soccer, *Artif. Intell.*, **173**, 1014–1039.

[15] **Mcmillen, C. and Veloso, M.**, (2006). Distributed, Play-Based Role Assignment for Robot Teams in Dynamic Environments, In Proceedings of DARS-2006.

[16] **Ayanian, N.**, **Kumar, V. and Koditschek, D.E.**, (2009). Synthesis of Controllers to Create, Maintain, and Reconfigure Robot Formations with Communication Constraints., ISRR, volume 70 of *Springer Tracts in Advanced Robotics*, Springer, pp.625–642.

[17] **Karol, A.**, **Nebel, B.**, **Stanton, C. and Williams, M.A.**, (2003). Case Based Game Play in the RoboCup Four-Legged League Part I The Theoretical Model, In RoboCup, Springer.

[18] **Floyd, M.W.**, **Esfandiari, B. and Lam, K.**, (2008). A Case-based Reasoning Approach to Imitating RoboCup Players, In: Proceedings of FLAIRS-2008, Florida AI Research Symposium.

[19] **Marling, C.**, **Tomko, M.**, **Gillen, M.**, **Alex, D. and Chelberg, D.**, (2003). Case-based reasoning for planning and world modeling in the robocup small sized league, In IJCAI Workshop on Issues in.

[20] **Kuo, J.Y. and Lin, H.Z.**, (2008). Cooperative RoboCup agents using genetic case-based reasoning, IEEE International Conference on Systems, Man and Cybernetics, 2008. SMC 2008.

[21] **Palamara, P.F.**, **Ziparo, V.A.**, **Iocchi, L.**, **Nardi, D. and Lima, P.**, (2009). Teamwork Design Based on Petri Net Plans, Springer-Verlag, Berlin, Heidelberg, pp.200–211.

[22] **Ziparo, V.A.**, **Iocchi, L.**, **Nardi, D.**, **Palamara, P.F. and Costelha, H.**, (2008). Petri net plans: a formal model for representation and execution of multi-robot plans, Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 1, AAMAS '08, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp.79–86.

[23] **Cohen, P.R. and Levesque, H.**, (1991). Teamwork.

[24] **Murata, T.**, (1989). Petri nets: Properties, analysis and applications, Proceedings of the IEEE Vol. 77 Issue 4.

[25] **Flocchini, P.**, **Santoro, N. and Widmayer, P.**, (2005). Gathering of asynchronous robots with limited visibility, *Theoretical Computer Science*, **337**, 147–168.

[26] **Nair, R.**, **Ito, T.**, **Tambe, M. and Marsella, S.**, (2001). Task Allocation in the RoboCup Rescue Simulation Domain: A Short Note, RoboCup 2001: Robot Soccer World Cup V, Springer-Verlag, London, UK, pp.751–754.

[27] **Mei, Y.**, **Lu, Y.H.**, **Hu, Y. and Lee, C.**, (2004). Determining the fleet size of mobile robots with energy constraints, Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004. (IROS 2004).

[28] **Mei, Y.**, **Lu, Y.H.**, **Hu, Y. and Lee, C.**, (2005). Deployment Strategy for Mobile Robots with Energy and Timing Constraints, Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005. ICRA 2005.

[29] **Aamodt, A. and Plaza, E.**, (1994). Case-based reasoning: foundational issues, methodological variations, and system approaches, *AI Commun.*, **7(1)**, 39–59.

[30] **Arkin, R.C.**, (1998). Behavior-based Robotics, MIT Press, Cambridge, MA, USA, 1st edition.

[31] **Asta, S. and Sariel-Talay, S.**, (2011). Nature-Inspired Optimization for Biped Robot Locomotion and Gait Planning., EvoApplications (2), volume6625 of *Lecture Notes in Computer Science*, Springer, pp.434–443.

[32] **Asta, S.**, **Sonmez, T.**, **Ulusoy, O.**, **Alimoglu, A.**, **Ersen, M.**, **Sozmen, O. and Sariel-Talay, S.**, (2011). beeStanbul RoboCup 3D Simulation League Team Description Paper 2011, *RoboCup 2011*.

[33] **de Berg, M.**, **van Kreveld, M.**, **Overmars, M. and Schwarzkopf, O.**, (2000). Computational Geometry: Algorithms and Applications, Springer-Verlag, second edition.

[34] **Burkhard, H.D.**, **Krause, T.**, **Mellmann, H.**, **Ritter, C.N.**, **Xu, Y.**, **Scheunemann, M.**, **Schneider, M. and Holzhauer, F.**, (2011). NaoTH 2011 The RoboCup Team of Humboldt-Universitat zu Berlin, *RoboCup 2011*.

APPENDICES

APPENDIX A.1 : Simspark Environment

APPENDIX A.1

Simspark is the official simulation server of the RoboCup 3D Soccer Simulation League. It is a physical multi-agent 3D simulation software, based on Spark application framework. Spark is a generic and flexible physical simulation system for different kinds of simulations. Simspark allows users to create new simulation environments with a scene description language. Simspark's predecessor was a two dimensional simulator which modelled the agents and the ball as flat spheres, thus lacks realistic physical environment. In 2004, Simspark was started to be used for official competitions. In first version players were modelled as spheres. With the further developments, agents are currently models of humanoid Nao robots by Aldebaran Robotics.

Simulation process is hosted by the Simspark server, called rcssserver3d. Environment is represented by states and rcssserver3d continuously updates the state within "Simulation Update Loop". Each object in the environment has properties like speed, angular velocity, position etc. which are controlled by a rigid body physical simulation. Properties of the objects are updated according to the effects of collisions, applies drag, gravity etc. that are resolved by the physical simulation. The server also keeps track of agent processes in the environment. In each cycle of the "Simulation Update Loop", server receives and sends sensor information for all the sensors of the agents. Server can render the simulation and monitor it internally. It can also send streaming data to remote monitor processes to be rendered externally.

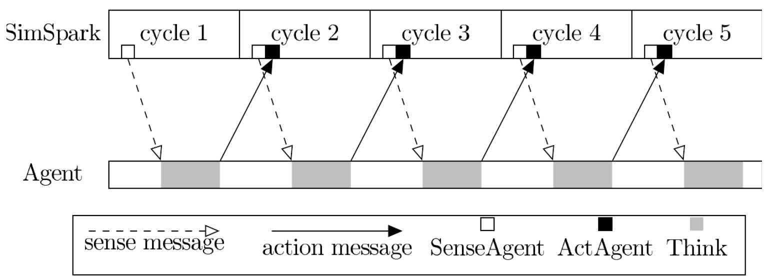


Figure A.1: "Simulation Update Loop" synchronization diagram between Simspark and the agents. [2]

"Simulation Update Loop" has two different modes, namely single-threaded and multi-threaded mode. Figure A.1 shows the synchronization diagram between Simspark and the agents. Action message that is send by the agent in n_{th} cycle will be acknowledged by the server in $(n + 1)_{th}$ cycle. Figure A.2 (a) and (b) shows the diagrams for single-threaded mode and multi-threaded mode, respectively. Single-threaded mode has a main loop which cycles through start cycle, sense agent, act agent and end cycle events continuously. Each cycle duration is 0.02 s, and if simulation runs faster than this duration, server waits until the end of it. If simulation runs slower it can ignore interaction with agents and continue with physics updates.

If it is very slow, server gives a warning and gives up catching the cycle time. Different from single-threaded mode where SimControlNodes perform one by one, multi-threaded mode has SimControlNodes that response for different parts of the simulation and run in parallel. SimControlNode interacts with the physics engine through the active scene.

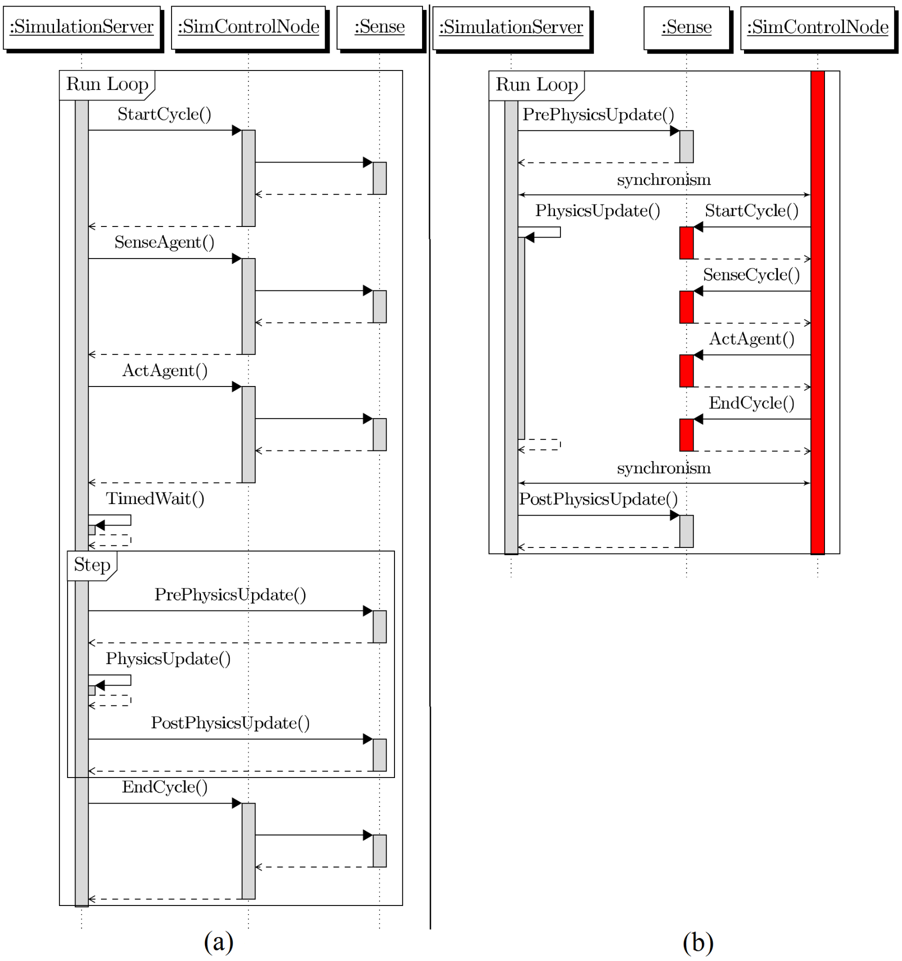


Figure A.2: Single-thread and multi-thread mode diagrams of "Simulation Update Loop". [2]

Message format of the Simspark is shaped by symbolic expressions (S-expressions) that are also used in the Lisp family programming languages. S-expressions are easy to parse and to be read by humans for debugging purposes. Messages use the default ASCII character set and each byte represents one character. The server connects with the agents on TCP port 3100. Agents sends a CreateEffector message and an InitEffector message, respectively to connect to the server. These messages are going to be explained below with the other agent effector types. After connection, the server sends messages that contain the output of the agent perceptors and receives effector messages by the agents.

Agents can be programmed with various ways and on various programming languages. Agents communicate with the server via TCP protocol and if a software manages to use TCP socket communication, it can run on Simspark. Figure A.3 shows the control and data flow diagram of Simspark. Simspark server communicates with the agents via TCP protocol and it also communicates with "Zeitgeist" file manager for object and memory management. Zeitgeist communicates with "Open Dynamics Engine" (ODE), which handles the physics engine issues.

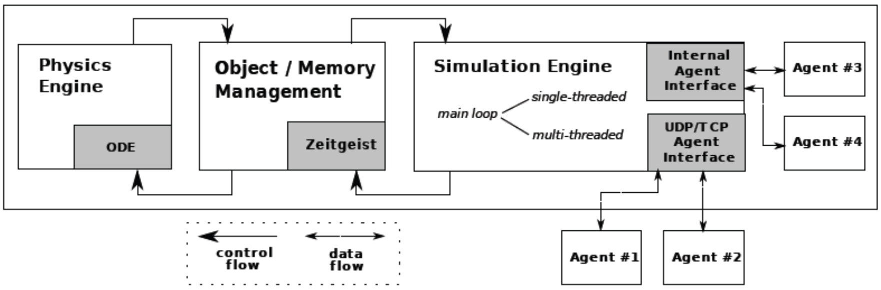


Figure A.3: Control and data flow diagram of Simspark. [2]

As mentioned before, Nao robots by Aldebaran Robotics are currently used as agents in Robocup 3D Soccer Simulation competitions. Nao has a 57 cm height and 4.5 kg weight with 22 degrees of freedom. Figure A.4 shows the anatomy of a Nao robot.

Nao robots have various perceptors, which allows them awareness of their model state and the environment. They also have effectors that allows them to perform actions within the simulation. Perceptors are divided into two categories: general perceptors and soccer perceptors. General perceptors are applied to all simulations while soccer perceptors are specific for the soccer competitions. There are six types of general perceptors, namely GyroRate Perceptor, HingeJoint Perceptor, UniversalJoint Perceptor, Touch Perceptor, ForceResistance Perceptor and Accelerometer. GyroRate Perceptor receives information about the change in orientation of a body. This information is hold with angular velocities along the three axes of freedom of the corresponding body. HingeJoint Perceptor delivers angle information of single-axis hinge joints. Nao robots have 22 hinge joint perceptors. UniversalJoint Perceptor delivers angle information of two axis universal joints. Naos don’t have any universal joints so this perceptor isn’t used currently. Touch Perceptor receive information for collisions with the other simulation objects. This perceptor also isn’t used currently on the server. There are two ForceResistance Perceptors below the feet of robots and delivers information about the force that acts on the body. Accelerometer measures the acceleration of the agent and it considers gravity in the computation. There are four types of soccer perceptors: Vision Perceptor, GameState Perceptor, AgentState Perceptor and Hear Perceptor. Vision perceptor receives information about the seen objects in the environment. These objects are the agents, the ball, field lines and markers on the field. There are currently eight markers. Four of them are on the corners of the field and four of them are on the both goal’s posts. Vision perceptor is restricted to 120 degrees of view and is at the center of Nao’s head. Information about the objects are given as the distance, horizontal and latitudal angle to the center of the object. Figure A.5 shows the polar vision perception of the agents. Vision perception has some noise values to improve realistic simulation. Camera position is changed with a uniformly distributed error between -0.005 m and 0.005 m on each axis and it remains the same for the entire match. There are also dynamic noise values normally distributed around 0.0 for distance, horizontal and latitudal angle values. σ

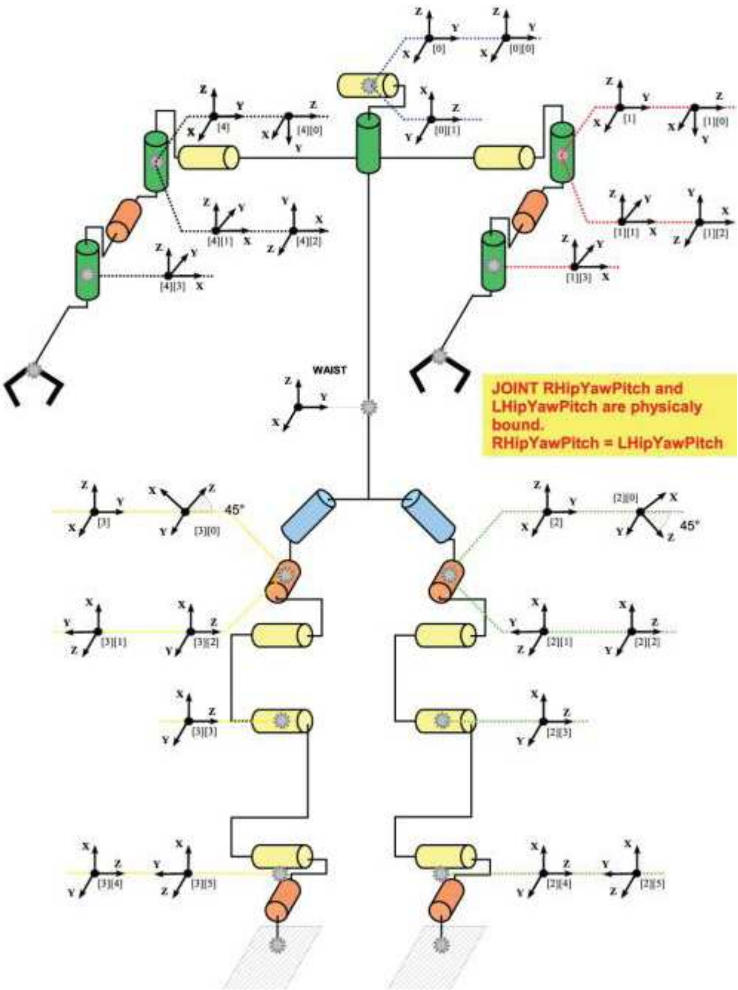


Figure A.4: Nao's anatomy. [2]

values for these dynamic noise factors are 0.0965, 0.1225 and 0.1480,respectively. GameState Perceptor receives information about the current game state. States are the play modes like corner kicks, goal kicks, throw ins etc. There are currently 18 game states for the matches. AgentState Perceptor holds information of agents' internal state like body temperature and battery state. These values are currently constant and doesn't change in the matches. Hear Perceptor receives messages from other players. Currently messages are restricted to a maximum capacity of 20 bytes and agents can only communicate via the server. If more than one agents sends messages on the same cycle to the server only earliest one can be heard. If the player also send a message on the same cycle it only hears itself.

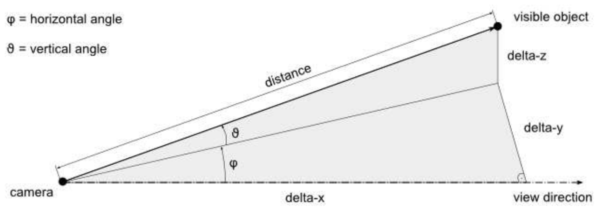


Figure A.5: Polar vision perception of the agents. [2]

Effectors of the agents are also categorized in general and soccer effectors, like the perceptors. There are four general effectors (Create Effector, HingeJoint Effector, UniversalJoint Effector and Synchronize Effector) and three soccer effectors (Init Effector, Beam Effector and Say Effector). Create Effector is used by the agent to advice to the server for creating the physical representation with the other effectors and perceptors. HingeJoint Effector holds the names and the speeds of the hinges with a single axis. There are 22 hinge joint effectors for Naos. UniversalJoint Effector holds the names and the angles of the hinges with two degrees of freedom. Nao doesn't have any universal joint, so this effector is not used currently. Synchronize Effector is used in "agent sync mode". In this mode, server waits for all the agents to connect with a sync message to end a cycle and proceed to the next one. Init Effector initializes a newly created agent on the field. It holds the agent's name and number. Beam Effector is used to position an agent to a coordinate on the field before the start of the game or second half. Say Effector is used to send messages to the other agents. The restrictions that are mentioned with the Hear Perceptor are also applied to this effector.

CURRICULUM VITAE

Name Surname: Onuralp ULUSOY

Place and Date of Birth: Lüleburgaz, 1988

Adress: İTÜ, Fen Bilimleri Enstitüsü, Ayazağa Kampüsü, 34469, Maslak, İstanbul

E-Mail: oulusoy@itu.edu.tr

B.Sc.: Trakya University, Faculty of Architecture and Engineering, Department of Computer Engineering, 2009

Professional Experience and Rewards:
National Scholarship for M.Sc. Students, The Scientific and Technological Research Council of Turkey (TÜBİTAK)

List of Publications and Patents:

PUBLICATIONS/PRESENTATIONS ON THE THESIS

- Ulusoy O. and Sarıel-Talay S., 2012: Distributed Team Formation for Humanoid Robot Soccer, 4^{th} International Conference on Agents and Artificial Intelligence. *Communications in Computer and Information Science*, February 06-08, 2012 Vilamoura, Algarve, Portugal.