High-level Strategy Design in RoboCup Environment

Zsófia Bodó
Department of Control Engineering and Information Technology
Budapest University of Technology and Economics
Budapest, Hungary

Dr. István Harmati
Department of Control Engineering and Information Technology
Budapest University of Technology and Economics
Budapest, Hungary

Abstract—In the context of robot soccer the agents on the football field interact with each other and the ball in order to win. The team strategy is based on the fact that during the game the agents belonging to each team try to maximize the scoring possibilities for their team and minimize them for the members of the other team. A special variant of the robot soccer is the simulation league which has the benefit that these team strategies and tactical solutions can be tested without the need of building the robots. The paper presents a high level strategy to control the strategy of a RoboCup team based on fuzzy control.

Keywords—robocup, soccer, fuzzy, strategy

I. Introduction

The idea of robot soccer was introduced in 1992 by Alan K. Mackworth in the University of British Columbia. This idea was further developed by Japanese scientists in the tournament called Robot J-League. After joining of American scientists the competition was renamed thus the RoboCup tournament was born.

In the official RoboCup competition four different leagues exist: RoboCupSoccer, RoboCupRescue, RoboCupJunior, RoboCup@Home. The diversity of these leagues allows different research and educational problems to be placed in focus. This article focuses on robot soccer league (RoboCupSoccer) in which the scientists and engineers can compete with robots of different sizes and types from the simulated agents to the humanoid-size robots.

In the simulation league of the RoboCup competition the competing teams join a central server, the so-called Soccer Server during the matches. This application is available during the year between the competitions, it can be used to design, develop and test the team strategy. One of the further advantages of the usage of soccer server is that such an abstraction can be done which lets the developer not to take different robotics problems into consideration – such as object recognition, communication between robots and the solution of several hardware-related problems such as the motion of the robot. Due to this abstraction the developer can concentrate on the high-level concepts, for instance the cooperation and learning of the agents. The Soccer Server simulates the physical environment, but for conducting the match we need a referee.

On the other hand the server ensures an intelligent referee, which perceives the trivial situations, such as when one of the teams score a goal. Nonetheless a number of difficult situations may occur (e.g. a kind of deadlock) where a human referee might be needed to solve the difficulties. One of these deadlock-like situations might be when the agents are concentrated around the ball so densely that it is stuck in the middle of the circle.

Section II presents an overview of different robot soccer strategies and tactics used in the field of the RoboCup competition. Section III provides a general description of the RoboCup framework. The aim of Section IV is to present the fuzzy-based control algorithm used with the different approaches to control the action of a RoboCup team to increase the chance to win the match.

II. Overview of different robot soccer strategies and tactics

While designing a team for the RoboCup competition many important research areas can be highlighted, such as the behavior of autonomous agents, designing reactive behavioral rules, creating models and planning, real-time image processing, sensor fusion, multi-agent systems, implementation of complex learning functions, strategy planning, cognitive modeling, etc. The team strategy can be formed by the accordingly combined use of these strategies.

One of the frequently used strategy is the DPRE – Dynamic Positioning and Role Exchange [1]. The basis of the DPRE used in the Portuguese team is the work of Peter Stone et al [6, 7].

It recommends the usage of agents with flexible roles and a communication protocol that supports this function. The team players regularly communicate each other their current position in the formation and also their currently used player type. The position- and role exchange will be carried out if its usefulness is positive regarding to the whole team.

DPRE can be also combined with Situation Based Strategic Positioning (SBSP) mechanism, which is an effective method to define the positioning of all agents which is the function of the actual ball position and velocity, game situation and player type [3].

Voronoi Based Strategic Positioning (VBSP) [4] is a significantly different method. The field can be divided in regions by a Voronoi tessellation. Each region is assigned a weight due to the rules of a specific strategy. Those weights are the result of the calculation of the optimal robot position and movement path. A team play strategy can be expressed by the choice of the tessellation as well as the choice of the weights.

III. General description of the RoboCup framework

To develop simulated teams and agents for the RoboCup competition it is useful to have a general framework which provides testing opportunities for the different implementations. In the context of the RoboCup competition different applications exist that can be used for this aim.

One of these is the Soccer Server (Fig. 1.) which is a framework that makes it possible for different teams to compete with each other in predefined conditions. The communication with the server is made through a well-defined network interface with the help of UDP/IP sockets as the agents join the server as clients.

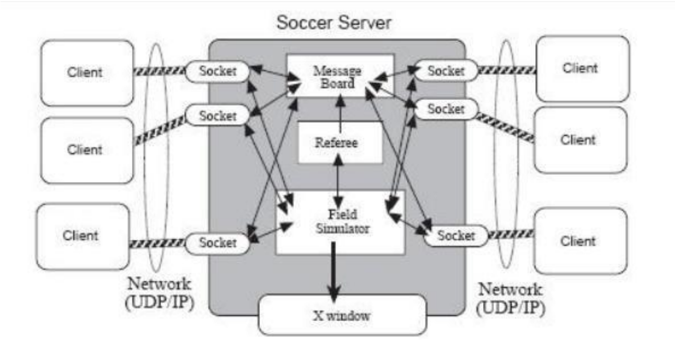

Fig.1. Operation of soccer server.

There are no strict limitations for the internal structure of the clients, the only requirement is that the implementation must support the UDP/IP-based client-server communication. Each player is symbolized by a client and each client uses a separate socket that has to connect to the server through a predefined port. The communication between the server and the clients happens via socket communication – players can communicate with each other only by using the Soccer Server.

The Soccer Monitor is a display tool that enables the team-developers to see what happens inside the server during a match. It displays the name of the teams, the state and result of the game and also the position of the players and the ball. In addition the monitor provides a simple user interface to the server. After the teams joined, the start of the game is indicated by the kick-off which can be performed by this UI. However, the monitor is not mandatory to run the simulation - the server can be operated by pure command line operands.

The log player can be viewed as a video player, with its help the matches can be recorded and replayed. When the server runs a game, the communication between the server and different players can be saved, then later it can be replayed by coupling with the soccer monitor.

The replay can be paused and it can also be rewound and fast-forwarded. This function is really useful because each match can be analyzed and this way it is much easier to find the weak and strong points of the team’s strategy.

IV. Control of the players

The different levels of the multi-agent team games can be separated according to Fig.2. The solution was divided into three different parts. The motion planning (MP) part embodies the lowest level of the problem. The second part is the so-called low level strategy (LLS) which is responsible for communicating with the server. The HLS is on a higher strategic level, it realizes the team coordination and controls the cooperation.

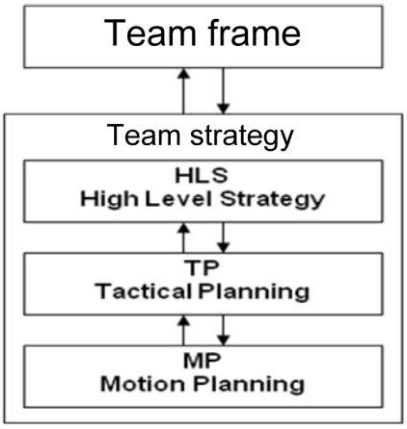

Fig.2. Levels of multi-agent games.

The high-level strategy presented in the paper is based on three different pillars that can be easily separated from each other: formation logic, ball handling and goalkeeper tactic. It is essential for successful operation to cooperate with these three modules successfully.

In the frame of the formation logic the whole team was handled as a separate unit. This logic is responsible for telling the positions to the different agents if the player occupies the ball during the match.

The ball-handling module deals with the individual units, it is responsible for telling the player the actions needed when the ball is in his proximity. Furthermore it turned out to be necessary to deal with the behavior of the goalkeeper in a separate module because his responsibilities and opportunities are sharply distinct from the other player roles. With this in mind was the goalkeeper logic - which defines the behavior of the goalkeeper - created. Separating the goalkeeper logic and the ball handling algorithm turned out to be useful later in further development and testing of the goalkeeper strategy and the strategy of the field players.

For the developing tool of the high level strategy fuzzy control had been chosen. The purpose of this decision was to eliminate the problem that most of the time the agents receive the input information needed to decide the next action imprecisely, ‘not sharp enough’. The aim was to help the players to be able to process this information that is often unequivocal for the players in real life - such as whether the ball is near or far away from them in a given second. During the implementation the input parameters were fuzzyfied. The fuzzy variables were created from the elements of the matrix provided by the low level strategy. The formation logic part of the high level strategy is based on two parts - one of them is called placement rules and the other deals with the ball-handling logic of the team.

A. Formation logic

The formation logic algorithm defines how the team members should be located on the field in the given moment. The aim was to set up a system of rules which is suitable for making a decision for the next action of the agent based on the information which is available for the player about the actual environment. Usually the player has only partial information about the environment and the state of the match by a RoboCup competition due to limited seeing and hearing capability which is sometimes made intentionally. Such a team strategy were designed that can make decisions about the next action in cases when only partial information is available.

The formation logic is based on two main parts. The first one is the placement rule of the players. By the creation of the placement logic and the different configurations the purpose was to use such roles which are widely used in soccer. The other main part is the focal point rule which is detailed in 4.C. point.

B. Placement rules

For the more efficient gameplay the team needed an effective way to deal with the different decision-making situations which the agents may meet during the game. With the help of the placement rules the dimension of the decision tree can be reduced, thus quicker decisions can be achieved. Each player can occupy different roles. The team contains ten midfield players and a goalkeeper. The goalkeeper is a special role, he can catch and keep the ball by himself. A winger takes place between the goalkeeper and the midfield players. Their job is mainly to cancel the scoring chances of the opponent team. The midfield players are usually positioned between the wingers and forwards. They have to bring the ball to the forwards. The forwards are the players who are positioned the nearest to the opponent team’s goal line. Their base job is to score a goal. A team can have minimum one and maximum three forwards.

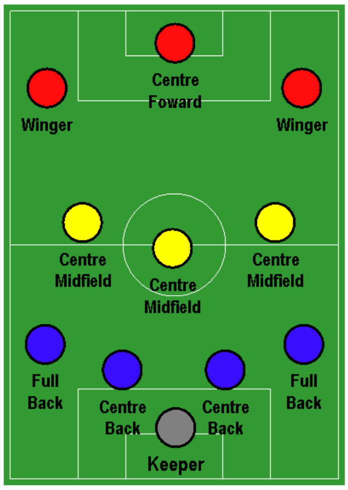

Fig.3. Simulation of a 4-3-3 team formation.

The formation means the role distribution between the players: the number of wingers, midfield and forward players. The most usual formation is 4-4-2 which means 4 wingers, 4 midfield and 2 forward players. There are many other useable formations like 3-5, 4-2-3-1, etc. In this example the 4-3-3 formation (Fig.3) were chosen because in this case the distribution of the players is more homogenous on the field and each player can be assigned to a defined position or role. Therefore it is not needed to implement a special formation that is highly dependent of the varying condition.

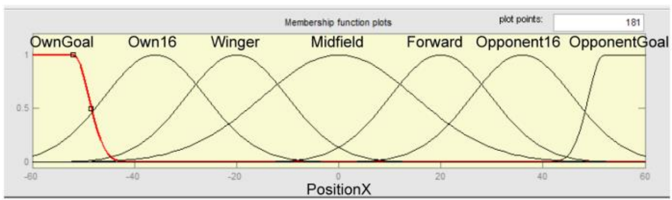

Fig.4. Membership functions of the PositionX variable.

The numbering of the players was defined by their role. The goalkeeper is 1, 2-5 are the wingers, 6-8 are the midfield players and 9-11 are the forwards.

The players can be assigned to a position by X and Y directional fuzzy variables. PositionX (Fig.4) has the following membership functions: OwnGoal, Own16, Winger, Midfield, Forward, Opponent16. PositionY (Fig.5) informs the player about his latitudinal (Y directional) position by the following membership functions:

Left, MiddleLeft, Center, MiddleRight, Right.

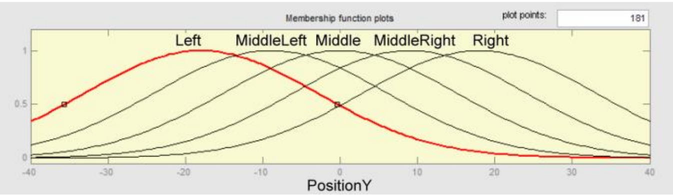

Fig.5. Membership functions of the PositionY variable.

Each player applies a simple fuzzy rule: checks the goodness factor for his possible positions. The player will take the position where the goodness factor of the membership functions is maximal. If the goodness factor is worse than 0.8 he will change his position and looks for the best he can get due to the nearest two membership functions. For example:

If PositionY ≠ Left or PositionY ≠ Winger then Goto PositionLeftWinger

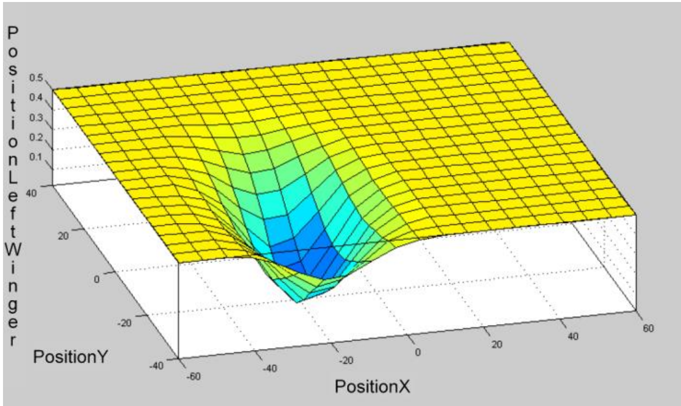

Fig.6. Goodness factor of the GotoPositionLeftWinger variable.

Using this method the players can partition the field between themselves creating quadratic regions. Opportunityal OR connection was applied between the membership functions. Fig.6. shows the goodness factor for GotoPositionLeftWinger in the function of position.

Placement rules for each player are the following:

If PositionY ≠ Right OR PositionY ≠ Winger THEN GotoPositionRightWinger

If PositionY ≠ MiddleRight OR PositionY ≠ Winger THEN GotoPositionRightMiddleWinger

If PositionY ≠ MiddleLeft OR PositionY ≠ Winger THEN GotoPositionMiddleleftWinger

If PositionY ≠ Left OR PositionY ≠ Winger THEN GotoPositionLeftWinger

If PositionY ≠ Right OR PositionY ≠ Middlefield THEN GotoPositionRightMiddlefield

If PositionY ≠ Middle OR PositionY ≠Middlefield THEN GotoPositionCenterMiddlefield

If PositionY ≠ Left OR PositionY ≠ Middlefield THEN GotoPositionLeftMiddlefield

If PositionY ≠ Right OR PositionY ≠ Forward THEN GotoPositionRightForward

If PositionY ≠ Middle OR PositionY ≠ Forward THEN GotoPositionMiddleForward

If PositionY ≠ Left OR PositionY ≠ Forward THEN GotoPositionLeftForward

C. *The focal point rule*

The focal point rule says that the formation can be made more ideal if the focal point of the players is near to the actual position of the ball. In this way the condition of having a player too far away from the ball can be eliminated. If the player knows the position of at least 2/3 of his team-mates he can calculate the focal point of the team and he can check the position of the ball. If the focal point and the position of the ball is different, then he will order the players to get closer to the ball. If the ball is on its own side of the field, the team can be ordered to defend the goal line. If the ball is on the opponent side of the field, the team can be ordered to attack. Compared to the placement rules this function plays a subordinate rule. This rule lets the players to move only in the own regions that are assigned to them.

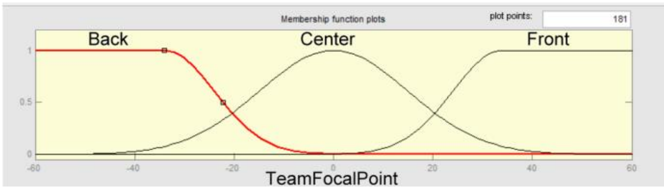

Fig.7. Membership functions for the focal team point rule.

The TeamFocalPoint fuzzy variable and the BallPosition function is defined in the same way, their membership functions are also the same (Fig.7). HLS applies the following rules only if the player knows the 2/3 of the position of his team-mates. This rule can be implemented like this:

If TeamFocalPoint X ≠ Back AND BallPosition ≠ Back THEN PositionPushBack

If TeamFocalPoint X ≠ Center AND BallPosition ≠ Center THEN PositionPushCenter

If TeamFocalPoint X ≠ Front AND BallPosition ≠ Front THEN PositionPushFront

D. *Ball-handling logic*

A simple rule was created for ball-handling by using rules with an ‘If..then’ structure. A fuzzy variable was defined for the distance of the ball. If the ball is close to the player (almost inside the players region) then he goes to the ball. If the ball is very close to the player he tries to get and kick it.

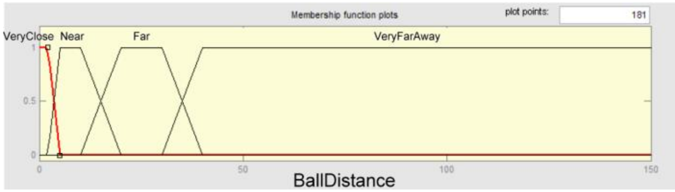

Fig.8. Membership functions of the BallDistance variable.

If kicking is possible, the direction and the power has to be defined (Fig.8). By the actual rule the player tries to kick the ball with full power. Kicking with smaller power would be needed if the player would be able to control the ball. The player determines the direction of the shoot by the PositionX of the ball (Fig.9). If the ball is very close to the own goal line he will make a liberation kick towards the opponent goal line. If the ball is near to the opponent goal line he shoots towards the opponent goal line and tries to score a goal. In any other cases he passes the ball forward.

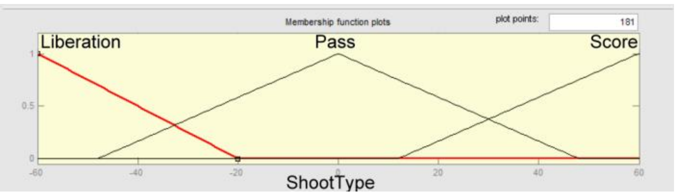

Fig.9. Membership functions of the ShootType variable.

The teammate to pass the ball to can be determined by the following logic. The player tries to pass the ball to a team-mate having a position by one level higher. The order of the passing possibilities can be seen on the passing graph of the game (Fig.10). The player doesn’t pass the ball to a teammate who is on the other longitudinal side of the field. If there are more teammates who can fulfill the rules the player chooses a random player to pass to. Defining the IHaveTheBall variable the rules can be implemented as following:

If BallDistance = Close THEN GoToBall

If BallDistance = VeryClose AND ShootType = Liberation THEN Liberation

If BallDistance = VeryClose AND ShootType = Score THEN Score

If BallDistance = VeryClose AND ShootType = Pass THEN PassToSelectedPlayer.

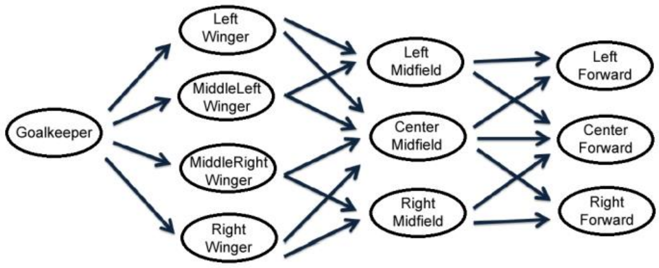

Fig.10. Passing graph of the players.

E. *Goalkeeper logic*

The rules presented so far can primarily be applied to the field players. For describing and control the behavior of the goalkeeper separate logic is needed because of the special location and ball intercepting ability of the player. The goalkeeper moves on a special curve. It aspires to hold the position that is the intersection point of this arc and the section connecting the ball with the center of the soccer gate. Otherwise, it follows the following rules:

- If the ball is very close, he catches it
- If the ball is close, he runs out and catches it
- If the game is in kick-off state, it performs the kick-off

The goalkeeper tactic was refined by the method of marking out two possible curves. In that case when the ball is far from the goalkeeper and all other players are located far from the soccer gate (paying particular attention to the players of the opponent team), then it is more worthy for the goalkeeper to keep more distance from its own gate. Therefore a close and a distant curve positioning was appointed.

The chosen tactic was determined by the distance between the center of the gate and the ball and the minimum distance between the center of the gate and the nearest player. If this distance can be considered close, then the closer positioning arc shall be used, otherwise the further positioning arc shall be used. This positioning tactic was refined, so that the distance of the ball is not really remote and also not really close, then the position used is located in proportion with the distance of the ball between the two ideal positions.

After testing, methods led to a change in the keeper's logic regarding on the basis of practical experience. During the test it was found that after a while the goalkeeper can only follow the ideal position with minor delays and its defense mechanism is getting worse also. The problem was that the goalkeeper was constantly trying to keep the ideal position according to the ball movement, and this continuous correction drastically reduced the stamina of that agent. To solve this, a fuzzy variable was interpreted to the distance of the goalkeeper from the actual ideal position, and it is only needed for the goalkeeper to make correction to his position if this distance can be considered big.

F. *Operation without information about the other players*

At the beginning of the common development process of the high level and the low level strategies, the low level strategy was not able to provide data about the other players. In this case a workaround is applicable. If the agent doesn't have suitable information of the ball, it will give command to look for the ball. In this case the agent will rotate until it finds the ball, and after it finds it, tries to follow it.

In this case the focal point rule is not applicable, so it should be replaced with a logic of similar functionality that substitutes the center of gravity with the state of attack of the team. If there are no available information of the position of the ball or none of the teams have the ball, the default positions of the players can be applied. If the team is in attack mode, then each player runs a little ahead, and also if the team is in defense mode everyone can retire a little further back.

The difficulty in testing of the behavior of the fuzzy logic driven team lies in that the competitive teams usually do not release their source code. The operation of the high level strategy was tested against opponent team that used classic strategy. The fuzzy strategy was efficient and it could beat the other team.

The usual classic strategy to implement in a RoboCup soccer team is the "go to ball and shoot" one which was used by the opponent team. The fuzzy control uses an optimized "catch the ball" strategy and also applies a well-defined formation keeping and passing strategy, uses a playing as a team technique. This combined game tactic makes the presented team competitive against many of the teams.

V. SUMMARY

The rules were designed in Matlab by means of a fuzzy interference system (FIS), realizing on one hand the goalkeeper logic and the logic of the field players. After the completion of these logics a Matlab function was implemented realizing the strategy. This is the function that creates the connection point to the lower tactical levels, i.e. this function is called as the callback by the low level strategy - first, when calculating the need for action and, secondly, to carry out the pre-positioning of the fixed game situations.

The implemented team follows the 4-3-3 player line-up. After the kick-off the players successfully take up their positions and they follow the logic ordered by the fuzzy regulator. Overall a high level logic was built up that is able to achieve line-up formation, and with players assigned to different posts in the team it implements a logic similar to a real-life football team. The operation of the team was tested in a simulated environment.

REFERENCES

[1] L.P.Reis,N.Lau,E.C.Oliveira: Situation Based Strategic Positioning for Coordinating Homogeneous Agents, Balancing Reactivity and Social Deliberation in Multi-Agent Systems, 2001, pp 175-197

[2] Mao Cheny, Ehsan Foroughi, Fredrik Heintz, ZhanXiang Huang, Spiros Kapetanakis, Kostas Kostiadis, Johan Kummeneje, Itsuki Noda, Oliver Obst, Pat Riley, Timo Ste_ens, Yi Wang and Xiang Yin, Users Manual, RoboCup Soccer Server for Soccer Server Version 7.07 and later.

[3] Robotic Soccer, Book edited by: Pedro Lima, ISBN 978-3-902613-21-9, pp. 598, December 2007, Itech Education and Publishing, Vienna, Austria

[4] Steffen Kaden, Heinrich Mellmann, Marcus Scheunemann, Hans-Dieter Burkhard: Voronoi Based Strategic Positioning for Robot Soccer, Proceedings of the 22nd International Workshop on Concurrency, Specification and Programming (CS&P), 2013, pp. 271-282.

[5] R. Geetha Ramani, Dr.R.Subramanian and M.Sindurathy, *Strategies of Teams in Soccerbot*.

[6] Jelle Kok, *UvA Trilearn 2003 Users Manual*, 1982.

[7] Mao Chen, Ehsan Foroughi, Fredrik Heintz, ZhanXiang Huang, Spiros Kapetanakis, Kostas Kostiadis, Johan Kummeneje, Itsuki Noda, Oliver Obst, Pat Riley, Timo Steens, Yi Wang, Xiang Yin, *Users Manual RoboCup Soccer Server*