FIRA Mirosot Robot Soccer System Using Fuzzy Logic Algorithms

*Elmer A. Maravillas, PhD and **Elmer P. Dadios, PhD
* *Cebu Institute Of Technology (CIT), N. Bacalso Avenue, 6000 Cebu City, Philippines*
***De La Salle University-Manila, Taft Avenue, 1000 Manila, Philippines*

1. Introduction

In November of 1996, the first Micro-Robot World Cup Soccer Tournament (MIROSOT) was held in Korea participated in by several countries. Three robots per team play soccer on a 130 cm x 150 cm soccer field. There were more than 25 research papers submitted on the proceedings and technical sessions dealing with various aspects of robot soccer system development and game control strategies (1996 Micro-Robot World Cup Soccer Tournament Proceedings, November 9-12, 1996, KAIST, Taejon, KOREA). From then on several robot soccer tournaments were held in many places of the world.
A robot soccer system is a multi-agent intelligent control system composed of two or more robots, vision system, communication equipment, and a personal computer. Each robot in the system has its own movement mechanism and therefore can move independently as well as cooperate with other robots.
Robot soccer game is an ideal venue for finding solutions to the many challenging problems facing the multi-agent robotic system. These problems include coordination between robots, motion planning of robots, recognition of objects visually, obstacle avoidance, and so on [24]. The robots must be taught different behaviors so that they will be able to react appropriately in any given situation. These behaviors shall be put into action in coordination with other robots so that a specific game plan shall be accomplished. It is therefore imperative that better soccer players (robots) are made to follow a deliberate plan of action from a module that provides very swift decisions especially in critical situations, which in a real-time game can mean a difference of many goals [25].
Since 1996 there are already lots of developments in multi-agent robotic system as a result of the MIROSOT undertaking. The behaviors of robots were improved dramatically as major breakthroughs in the control system are piling one after another.
This chapter will focus on the development of intelligent behaviors for the Mirosot wheeled robots to generate human interest in the framework of entertainment using fuzzy logic algorithms. Algorithms for scorer, goalie, obstacle detection and avoidance will be developed into the robot soccer system. In addition, a game strategy is also implemented in real-time using fuzzy logic algorithms.

2. Fuzzy Logic FIRA Robot Soccer Game On-line Scorer

Since 1996 when the first FIRA robot soccer world cup tournament was held in South Korea, scoring for robot soccer games were always done by a human scorer. Because of the human's susceptibility to biases and assumptions many researchers have proven that the reliability of most decisions made by humans is greatly undermined. Therefore faulty decision-making tends to be present in almost all systems managed by humans employing only instincts as their tool. In several competitions such as horse racing, dog racing, track and field, swimming, etc., video camera is used on the finish line to determine the true winner of hotly contested competition.

The existing FIRA robot soccer system uses a camera also but as a component of its vision system hardware for locating the positions of the robots and the ball. The information gathered through this camera is fed to the computer for use in the game strategy adopted by the competing teams. The decision-making, as far as scoring is concerned, is still made by a human game official.

This section presents an artificial scoring system for FIRA robot soccer games given the Cartesian coordinates of the ball. It aims to resolve very contentious scoring situations that a human scorer will most likely fail to recognize. The system incorporates cheering sounds and physical coordinated/synchronized movements of robots after a score is made. Fuzzy logic system is a formal methodology for representing, manipulating, and implementing a human's heuristic knowledge about how to make decisions [1,2,3]. Fuzzy logic scorer for FIRA robot soccer game is an artificial decision maker that operates in real-time. Figure 1 shows the block diagram of the fuzzy logic scorer. The system has four main components: (1) the "rule-base" holds the knowledge in the form of rules, of how best to score a FIRA robot soccer game; (2) the inference mechanism evaluates which control rules are relevant at the current time and then decides what the input to the scoring process should be; (3) the fuzzification interface simply modifies the inputs so that they can be interpreted and compared to the rules in the rule base; and (4) the defuzzification interface converts the conclusions reached by the inference mechanism into inputs to the scoring process. The fuzzy scorer takes the Cartesian coordinates (x,y) of the ball and output a **score (1)** or **noscore(0)** depending on the *points* computed by the defuzzification interface which ranges from 0 to 100. The scoring process receives the value of *points* and gives output according to the following:

$$\mathbf{output} = \begin{cases} \mathbf{1\ if}\ \mathit{points} >= \mathbf{85} \\ \mathbf{0\ if}\ \mathit{points} < \mathbf{85} \end{cases}$$

Figure 2 shows the Cartesian coordinates of FIRA robot soccer system's soccer field. The shaded portions of the field indicate the goal areas on both sides. The Cartesian coordinates for the LEFT goal are –10cm <= x <= 0cm, 45cm <= y <=85cm, and for the RIGHT goal are 150cm <= x <= 160cm, 45cm <= y <= 85cm.

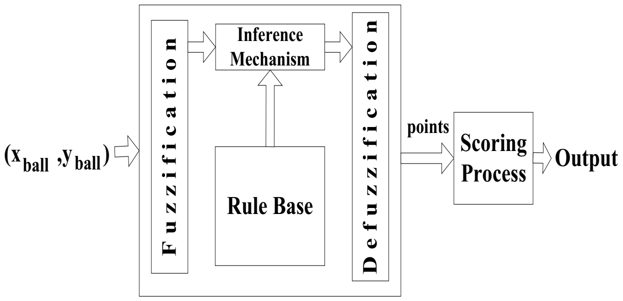


Fig. 1. Fuzzy scorer for FIRA Robot Soccer System.

When the ball's coordinates fall within these ranges, a scoring process will be done to determine whether a score could be awarded or not. The scoring process will be invoked only if the ball arrives within the specified areas of the field considered very close to the goals. Figure 3 shows some of these critical instances on the LEFT goal where the scoring process could be invoked. The same occurrences could happen on the RIGHT goal. In Figure 3, the ball's positions show that a greater portion of its body is beyond the goal line towards the goal area. Since the goal line is imaginary to the human referee, a particular team could be deprived of a score if the human referee fails to recognize the scoring situation. With fuzzy scorer, these things are not possible to happen.

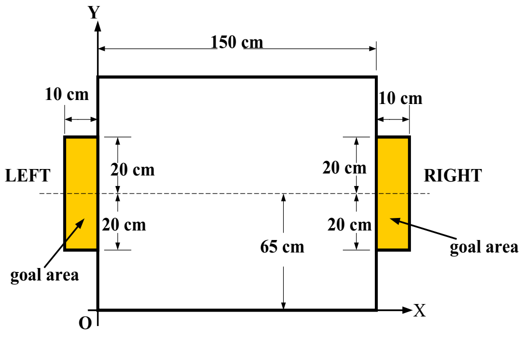


Fig. 2. Cartesian coordinates of FIRA robot soccer system's playing field.

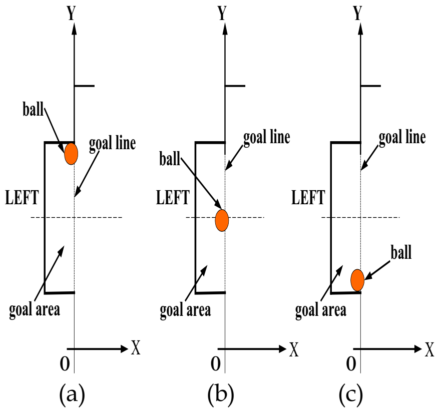


Fig. 3. Examples of critical ball positions where scoring process could be invoked.

Fuzzy sets map each element of the subsets of the "universe of discourse" to a continuous membership value (membership grade) ranging from 0 to 1. Input and output entities of a fuzzy logic system are composed or several maps (also called membership functions) with varying shapes common of which are, gaussian, triangular, and trapezoidal. Each map is assigned a linguistic term that best describes a range of values of the input and output entities. Figures 4 & 5 show the membership functions of the ball's x-coordinate. **CL** functions' maximum value is placed on the goal line (at 0 cm). Figure 6 shows the membership functions of the ball's y-coordinate. **CR's** maximum value is located at 65 cm. The maximum values of **LR** and **UR** are placed at the extremities of the goal area. Figure 7 shows the membership functions of the consequence. Defuzzification of the relevant membership functions of the consequence generates a value that will be assigned to *points* (0-100) as input to the scoring process. The value of *points* determines whether a score is to be awarded or not. It should be noted that the shapes of the membership functions are products of heuristics (trial and error).

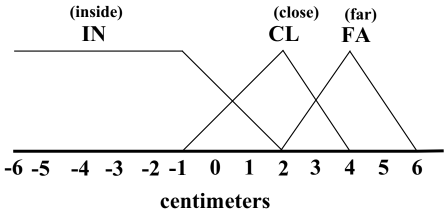


Fig. 4. Membership functions of ball's X-coordinate.

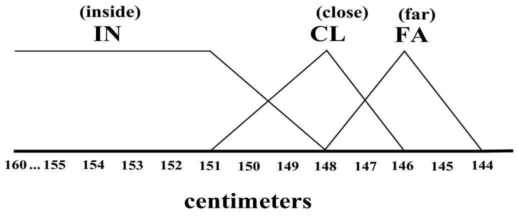


Fig. 5. Membership functions of ball's X-coordinate.

Reliable scoring decisions are made only when there is a clear outline of the set of rules provided by experts. Inputs are associated with each other from which a consequence will be assigned. The table that shows the consequences resulting from all possible associations of the input quantities is called the fuzzy associative memory (FAM) which represents the rule base of the fuzzy logic scorer. Table 1 shows the FAM of the fuzzy logic scorer.
The rules whose antecedents have membership function values greater than zero participate in the defuzzification process using their individual consequences to determine the *points* to be used in the scoring process.

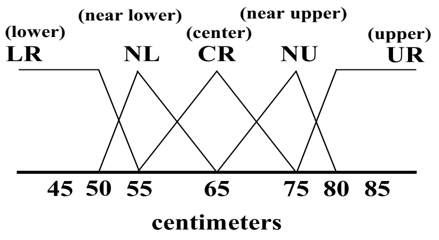


Fig. 6. Membership functions of ball's Y-coordinate.

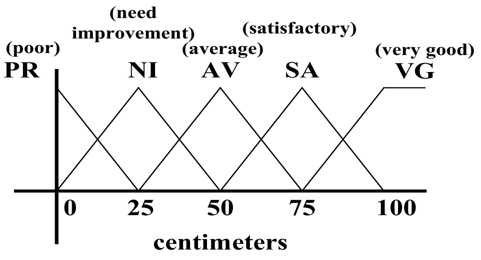


Fig. 7. Membership functions of consequence.

Ball's x-coordinates	Ball's y-coordinates					
		LR	NL	CR	NU	UR
	IN	VG	VG	VG	VG	VG
	CL	NI	AV	SA	AV	NI
	FA	PR	NI	AV	NI	PR

Table 1. Fuzzy logic scorer fuzzy associative memory (FAM).

2.1 Incorporating Sounds

Adding sounds to robot soccer games can make the game more entertaining and brings the game closer to the real world. Sounds can give feedback as an appreciation when a team scores and as criticism when a team commits foul action on the field.

Different recorded sounds can be played as the game progresses. Instances where playing of sounds is appropriate are: (1) when robots are executing field demonstrations simulating a cheering competition music can be played, (2) when a team made a goal a sound of applause can be played, (3) when a team committed foul a booing sound can be played, (4) when a team executes a defense strategy a sound shouting defense! Defense! … may be played, and (5) when a team executes an offense strategy a sound of encouragement may be played. Sometimes a game situation requires the playing of different tunes at the same time.

Support for playing waveform audio is included in Visual C^{++}. The Windows multimedia library *winmm.lib* and the *Wave class* library give a fast and handy way for adding sound effects to any Windows 95, Windows 98, or Windows NT application. A class can be created also to play multiple wave files at the same time. The *Wave class* is only added to any Visual C^{++} project, after linking the *winmm.lib*, it is now ready to add sound to the application. There are only three (3) methods required from the *Wave class* that enables an application to play sound from given medium. First, the recorded sound must be loaded into memory by the Load method. Second, the loaded sound will be played by the Play method. Third, played sound will be stopped by the Stop method [4].

Just like real soccer games, the playing of sounds and robot movements must occur simultaneously but independent of each other. When a team scores a goal, the robots of such team celebrates by moving around the robot soccer field while the sound of applause is playing. These tasks cannot be accomplished in ordinary sequential execution of components in a program. Because when the sound instructions grab the control of execution, the robots have to stop or enter in an uncontrollable situation. It is only when the sound finished playing that the robots gain control of execution. It is at this point that a separate line of execution is deemed necessary to allow playing of sounds on different occasions as the game progresses without encroaching on the roles of the robots.

The concept of multithreading is appropriate in this respect. A *thread* is a path of execution through a process' code. A preemptive scheduler inside the operating system divides CPU time among active threads so that they appear to run simultaneously. Secondary threads are ideal for performing tasks such as playing sounds while robots are doing their thing on the robot soccer field.

2.2 Robots Synchronized Movements

When the robots are playing soccer, their movements are not predetermined but depend most on the position of the ball and the game strategy being applied. However, when robots celebrate resulting from a score being made, their movements are predetermined so that proper coordination is attained thus making it more appealing to the audience. Different paths can be generated and loaded in memory at the start of the game so that they are readily available when needed. Geometric curves may be used as patterns for these coordinated movements of robots. Figure 14 shows the robots converging at the center of the playing field after a score has been made.

2.3 Experimental Results Of Scorer

Figures 8 to 10 show the performance of the fuzzy scorer on the left goal of the playing field. Three experiments were conducted on the left goal, namely, on the upper portion of the goal where the ball is made to approach the goal area with $y_{ball} > 75$, another where $y_{ball} = 65 \pm 5$, and another with $y_{ball} < 55$. The figures show the values of *points*, output of the defuzzification interface, as the ball approaches the goal line towards the goal area. Values of *points* equal or greater than 85 means that the scorer have decided it is a score. Experiments show that as the ball crosses the goal line ($x_{ball} = 0$), the fuzzy scorer gives *points* values that translate to a score.

The above experiments were also done on the right side goal of the playing field the results of which are shown on Figures 11 to 13. Consistent with the results of the above experiments, the fuzzy scorer gives *points* values that also translate to score as the ball passes thru the goal line ($x_{ball} = 150$) towards the goal area.

Figure 14 shows the coordinated movements of the robots after their attacker made a score. These actuations mimic the behavior of a human soccer player after making a score.

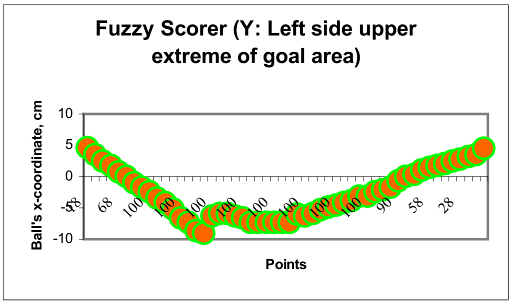


Fig. 8. Fuzzy scorer performance on the left side upper extreme portion of goal area.

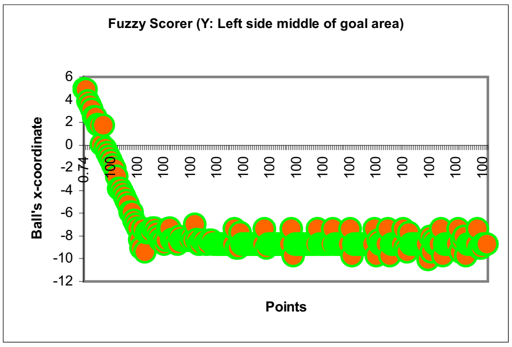


Fig. 9. Fuzzy scorer performance on the left side portion of goal area.

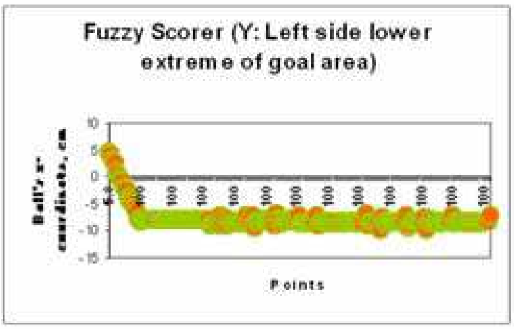


Fig. 10. Fuzzy scorer performance on the left side lower extreme portion of goal area.

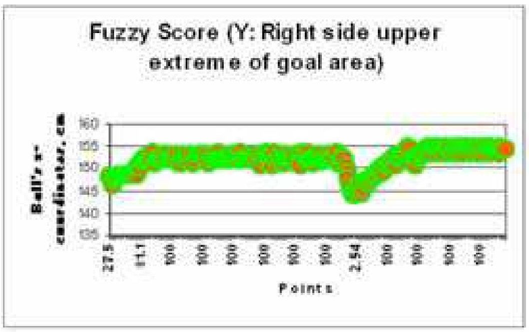


Fig. 11. Fuzzy scorer performance on the right side upper extreme portion of goal area.

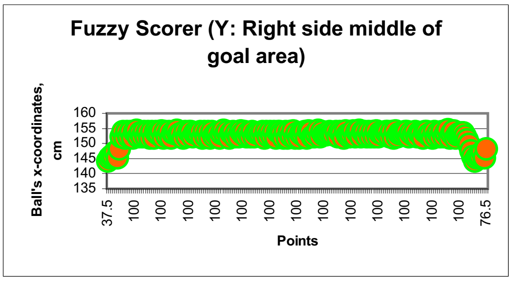


Fig. 12. Fuzzy scorer performance on the right side middle portion of goal area.

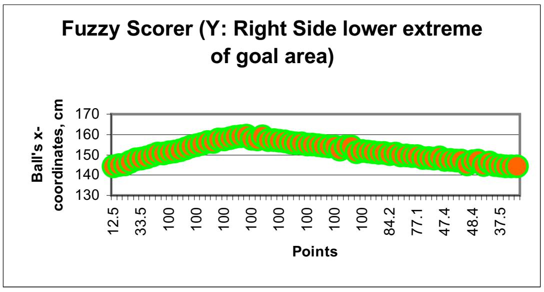


Fig. 13. Fuzzy scorer performance on the right side lower extreme portion of goal area.

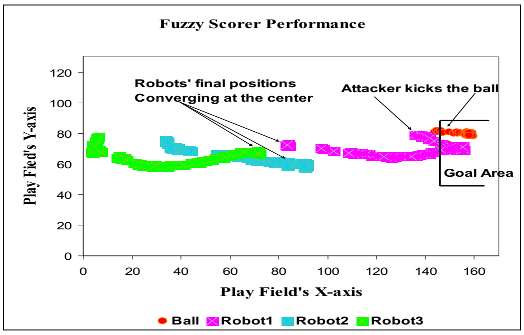


Fig. 14. Fuzzy scorer performance showing coordinated movement of robots by converging at the center after the attacker made a score.

The performance of the fuzzy scorer is greatly dependent on the shapes of the membership functions of x_{ball} and y_{ball}. Inaccurate adjustment of these shapes would result in a score even if the ball has not yet touched the goal line or no score in spite of the ball's being inside the goal area. The speed of the ball can also affect the performance of the fuzzy scorer. If the ball crosses the goal line at high speed, a situation that seldom happens in actual games, the vision system of the robot soccer will not be reliable enough in tracking the ball's position since its frame grabbing speed is constant. This will cause the fuzzy scorer to decide affirmatively a little late, that is, when the ball is already few units from the goal line towards the goal area.

The performance on synchronized movements of the robots after a score is made, depends on the ability of the robots to follow exactly the path design. The robots have difficulty in positioning themselves on the different points that composes the designated path. Actually, the robots cannot be perfectly positioned on the points where we want them to be. But the closeness of the robots to these points can be optimized.

3. Soccer Robot Goalie Strategy Using Fuzzy Logic

This section presents a control strategy for the FIRA robot soccer game goalie using fuzzy logic systems. This strategy adopts the divide and conquer concept by decomposing the goalie's task of defending the goal into four (4) categories. Consequently, each category is characterized by different goalie-ball situations and will be handled by a separate rule-base. The first fuzzy system takes the x-coordinates of the robot and the ball then outputs the category number that determines the rule-base to be used in the second fuzzy system. The second fuzzy system handles the goalie's movements. It takes the current y-coordinate of the goalie-robot and the ball then outputs the y-coordinate of the goalie's next position (destination) which is located along its line-of-action (a line with predefined x-coordinate where the goalie moves back and forth just in front of the goal it is defending). Experiment shows that this strategy is feasible, efficient, and robust.

Robot soccer games are scored by the number of times the ball actually entered the opponent goal. The entity that is tasked to prevent the ball from entering the goal is called the goalie. The goalie plays a very vital role in the game since a weak goalie means a lost game. It moves over an area just in front of the goal, called goal area, to block and clear the ball out from this area. Thus, preventing the opponent team from scoring.

Conventional goalie strategies [5,6,7,8] would use mathematics to predict the next position of the ball and subsequently generates a path to block it. But the data utilized in the mathematical process are imprecise in the sense that they are the result of the image processing procedure of the robot soccer vision system that is laden with varying degrees of imperfections aside from the fact that robot soccer is a very dynamic system. Thus, the results would have far-reaching effects on the actual performance of the goalie robot.

In [6], the goalie predicts the point of intersection between the goalline and the path of the ball and moves to it to block the ball when it threatens to enter the goal. When the ball is far from the goal area, the goalie moves to the center of the goalline. If the ball is blocked and sticks on the goalie, the goalie makes a spin move to drive the ball away from the goal area. This move is a bit dangerous because the possibility of having the ball driven to its own goal is present. Four states are assumed by the goalie in [7]. The first state is assumed when the game begins by moving the goalie to the center of the goal. The second state is assumed when the point of entry of the moving ball towards the goal is predicted and so the goalie moves to that point. The third state is assumed when the ball and the center point of the goal are connected that it is necessary for the goalie to move between this connection. The last state is assumed when the ball is in the goal area that the goalie has to kick it out. The goalie strategy described in [8] places the goalie in front of the goal until the ball arrives at the goal area. The goalie would follow the ball location in the y direction when the ball is outside the goal area. The goalie would move towards the ball as soon as the ball enters the goal area.

The goalie strategies discussed above do have some of the necessary qualities of an effective goal defender. What is lacking however is the fact that these strategies do not provide solution when the goalie is trapped and do not prevent the goalie from pushing the ball towards its own goal. This paper offers an encompassing solution to all possible problems that may prevent the goalie from performing its task of defending the goal effectively. The use of fuzzy logic systems in determining the next position of the goalie not only take care of the imprecise data to dampen its negative effects but also provide sufficient measures to counter the countless possibilities that might confront the goalie.

3.1 Description of the Goalie Function

The importance of the goalie's task in defending the goal cannot be overemphasized. Experience tells us that the most effective way to defend the goal is to have the goalie move back-and-forth along a specified line (line-of-action) just in front of the goal without ever changing its orientation on it (the goalie should either be oriented 90^0 or 270^0 with respect to the x-axis). This means that if the goalie happens to be moderately far from its line-of-action, it should return immediately to it but with careful consideration on the ball's position.

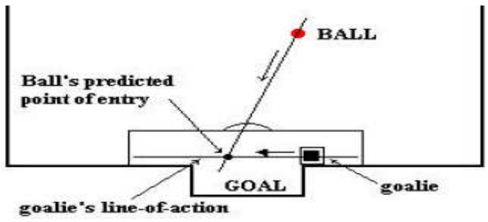


Fig. 15. Category 1 goalie and ball situation.

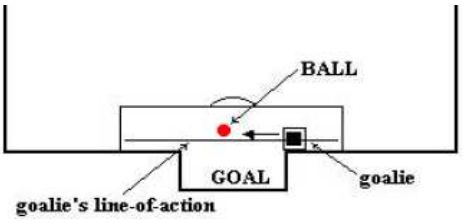


Fig. 16. Category 2 goalie and ball situation.

When the goalie is on its line-of-action as shown in Figure 15, it can block the moving ball by moving ahead to the position (along the goalie's line-of-action) where the ball is supposed to enter. This is referred to as category 1. However, when the ball is sensed to be within the goal area, as shown Figure 16, the goalie immediately kicks the ball out as it is about to cross the line-of-action and immediately moves back to prevent from being blocked by an opponent robot. This is referred to as category 2.
As shown in Figure 17, the goalie is far from its line of action so it has to be returned back to it but the point where the robot should enter must be carefully chosen so as not to accidentally push the ball into its own goal if it happens to be around. Otherwise, if the ball is not in the goal area, the goalie can go back to its line-of-action on the point nearest it. This is referred to as category 3. When the goalie is trapped in its own goal as shown in Figure 18, it has to move back to its line-of-action immediately to block or kick the ball out. This situation is referred to as category 4.

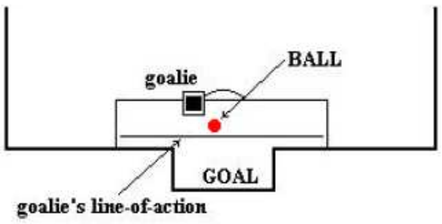


Fig. 17. Category 3 goalie and ball situation.

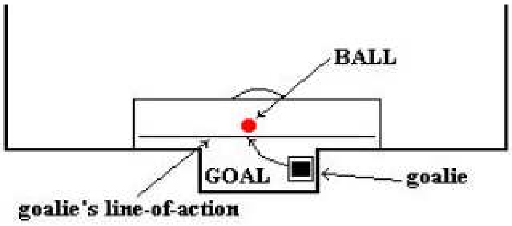


Fig. 18. Category 4 goalie and ball situation.

3.2 Predicting the Position of the Moving Ball

For category 1 goalie-ball situation, the strategy predicts the point where the ball would intersect along the line-of-action of the goalie (see Figure 15). By inspection the path of the moving ball is always a series of straight lines. Being so, it only requires two points to sense the direction of the ball at any given time and that is by knowing its current and previous positions. When the difference between the x-coordinates of the former and the latter is increasing, the goalie doesn't have to worry because the ball is going away from the home goal. Otherwise, the slope of the line connecting these two points can be computed and then the line is extended to the line-of-action of the goalie by finding their point of intersection. If the point of intersection falls in front of the home goal, the goalie has to move ahead to this point to be able to block the ball there. When the slope of the path of the moving ball is infinity, it means that the direction of the ball is vertical and the goalie would have to block the connection of the ball to any point of the goal by following the ball in the y direction.

3.3 Goalie Fuzzy Logic System

The objective of goalie fuzzy logic system is to generate the y-coordinate of the goalie's next position given its current position and the ball's. The line-of-action of the goalie is predefined therefore the point of its next position, where the y-coordinate is output of the fuzzy logic system, lies on this line. The block diagram of the fuzzy logic system used in this research is shown in Figure 19. The goalie moves back-and-forth on its line-of-action so its x-coordinate (X in Figure 19) is predefined and it is fed directly to the goalie. The determination of the y-coordinate for the goalie's next position begins by feeding the x-coordinates of the goalie and ball (Xb and Xr) to the first fuzzy system which outputs the rule-base number that will be used in the second fuzzy system. The y-coordinates of the ball and goalie (Yb and Yr) are inputs to the second fuzzy system that outputs the y-coordinate of the goalie's next position.

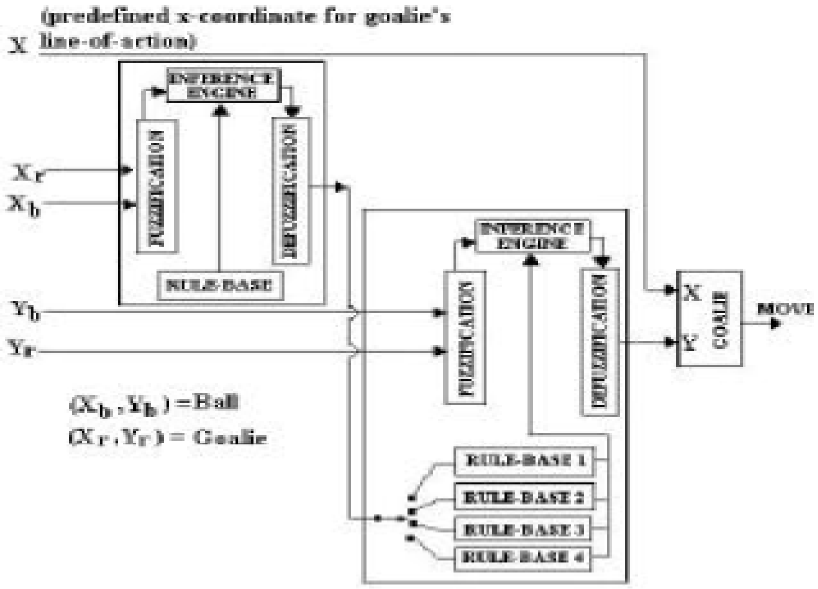


Fig. 19. Goalie's fuzzy logic system block diagram.

3.4 Membership Functions

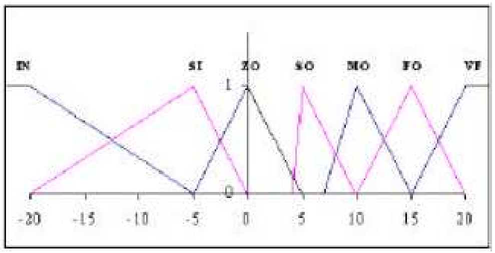
Table 2 shows the variables used to represent the membership functions (fuzzy sets) of the goalie fuzzy logic system with their corresponding linguistic definitions to describe the system's behavior.

Figure 20 shows the membership functions for the x-coordinates of the ball and the goalie used in the first fuzzy logic system of the goalie. Please refer to Table 2 for the variables' descriptions. In Figure 6 the minimum crisp value of **SO** is pegged at 4 cm from the goal-line (at 0 cm) instead at the goal-line itself to ensure that when the goalie goes closer to the goal-line than 4 cm, the first fuzzy logic system declares it as a trap situation (category 4) and the second fuzzy logic system utilizes the rule-base 4 for the goalie's movement. Similarly, the minimum crisp value of **MO** is pegged at 7 cm to ensure that the goalie kicks the ball out when the ball is in the goal area.

X-coordinates (X_b and X_r)	Y-coordinates (Y_b and Y_r)	Category (1,2,3,or 4)
IN: Inside the goal	DN: Down the origin y=0	ONE: Category number 1
SI: Slightly inside the goal	MD: Moderately down the origin	TWO: Category number 2
ZO: At x=0	SD: Slightly down the origin	TRE: Category number 3
SO: Slightly outside the goal	GD: Going down the origin	FOR: Category number 4
MO: Moderately outside the goal	CR: At y-coordinate's center line	
FO: Far outside the goal	GU: Going up	
VF: Very far from the goal	SU: Slightly up	
	MU: Moderately up	
	TP: Top or at maximum value of y-coordinate	

Table 2. Variables used in the membership functions.

Figure 20 shows the membership functions for the x-coordinates of the ball and the goalie used in the first fuzzy logic system of the goalie. Please refer to Table 2 for the variables' descriptions. In Figure 20 the minimum crisp value of **SO** is pegged at 4 cm from the goalline (at 0 cm) instead at the goalline itself to ensure that when the goalie goes closer to the goalline than 4 cm, the first fuzzy logic system declares it as a trap situation (category 4) and the second fuzzy logic system utilizes the rule-base 4 for the goalie's movement. Similarly, the minimum crisp value of **MO** is pegged at 7 cm to ensure that the goalie kicks the ball out when the ball is in the goal area. Figure 21 shows the membership functions of the y-coordinates used in the second fuzzy logic system of the goalie. The distance from **MD** to **MU** is the length of the goal area and the distance from **SD** to **SU** is the length of the goal. **CR** is the location of the horizontal centerline of the robot soccer playing field.



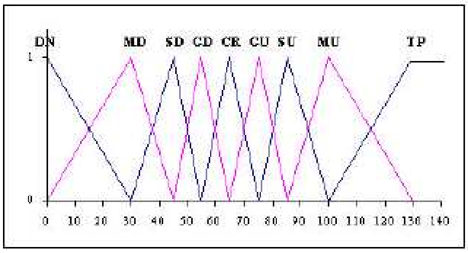


Fig. 20. Membership functions of x- coordinates. Fig. 21. Membership functions of y-coordinates.

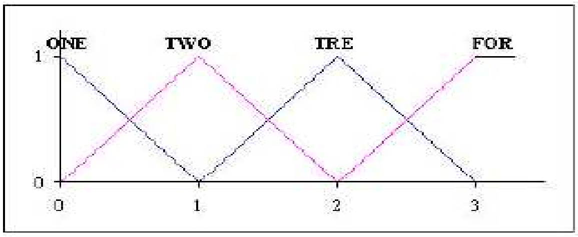


Fig. 22. Membership functions category.

3.5 Fuzzy Associative Memory (FAM) For The Goalie

Tables 3, 4, 5, 6, and 7 show the FAMs (fuzzy associative memory or rule-base) of the first and second fuzzy logic systems of the goalie. They contain the sets of rules provided by experts in associating the antecedents in order to generate the consequence. Logical implications are utilized in formulating these rules like as follows:

(1) IF **Yr** is SD(slightly down) *AND* **Yb** is GU(going up) *THEN* **Y** is GU(going up) or **SD Ù GU** or **GU**
(2) IF **Yr** is GU(going up) *AND* **Yb** is TP(top) *THEN* **Y** is SU(slightly up) or **GU Ù TP** or **SU** and so on…

The rules whose antecedents have membership function values greater than zero participate in the defuzzification process using their individual consequences. Looking at Table 5, there are cells that are empty. This is because Table 5 is a rule-base used by the second fuzzy system for category 4 which is a trap situation for the goalie. That is, the goalie is inside the goal and is being restraint by the goal's boundary walls. This means that the goalie's **Yr** cannot go beyond **SD** and **SU.**

Ball's y-coordinates, Y_b

Goalie's y-coordinates, Y_r

	DN	MD	SD	GD	CR	GU	SU	MU	TP
DN	SD	SD	SD	GD	CR	GU	SU	SU	SU
MD	SD	SD	SD	GD	CR	GU	SU	SU	SU
SD	SD	SD	SD	GD	CR	GU	SU	SU	SU
GD	SD	SD	SD	GD	CR	GU	SU	SU	SU
CR	SD	SD	SD	GD	CR	GU	SU	SU	SU
GU	SD	SD	SD	GD	CR	GU	SU	SU	SU
SU	SD	SD	SD	GD	CR	GU	SU	SU	SU
MU	SD	SD	SD	GD	CR	GU	SU	SU	SU
TP	SD	SD	SD	GD	CR	GU	SU	SU	SU

Table 3. Fuzzy Associative Memory (FAM) category 1.

Ball's y-coordinates, Y_b

Goalie's y-coordinates, Y_r

	DN	MD	SD	GD	CR	GU	SU	MU	TP
DN	SD	SU	SU	SU	SU	SU	SU	SU	SU
MD	SD	SD	SU	SU	SU	SU	SU	SU	SU
SD	SD	SD	SD	SU	SU	SU	SU	SU	SU
GD	SD	SD	SD	GD	SU	SU	SU	SU	SU
CR	SD	SD	SD	SD	CR	SU	SU	SU	SU
GU	SD	SD	SD	SD	SD	GU	SU	SU	SU
SU	SD	SD	SD	SD	SD	SD	SU	SU	SU
MU	SD	SD	SD	SD	SD	SD	SD	SU	SU
TP	SD	SD	SD	SD	SD	SD	SD	SD	SU

Table 4. Fuzzy Associative Memory (FAM) for for category 2 of the second fuzzy system.

Ball's y-coordinates, Y_b

Goalie's y-coordinates, Y_r	DN	MD	SD	GD	CR	GU	SU	MU	TP
DN	SD	SD	MD	SD	GD	CR	GU	SU	SU
MD	SD	SD	MD	SD	GD	CR	GU	SU	SU
SD	SD	SD	GD	SD	GD	CR	GU	SU	SU
GD	SD	SD	GD	CR	GD	CR	GU	SU	SU
CR	SD	SD	GD	CR	GD	CR	GU	SU	SU
GU	SD	SD	GD	CR	GU	CR	GU	SU	SU
SU	SD	SD	GD	CR	GU	SU	GU	SU	SU
MU	SD	SD	GD	CR	GU	SU	GU	SU	SU
TP	SD	SD	GD	CR	GU	SU	MU	SU	SU

Table 5. Fuzzy Associative Memory (FAM) category 3 of the second fuzzy system.

Ball's y-coordinates, Y_b

Goalie's y-coordinates, Y_r	DN	MD	SD	GD	CR	GU	SU	MU	TP
DN	-	-	-	-	-	-	-	-	-
MD	-	-	-	-	-	-	-	-	-
SD	GD	GD	GD	GD	CR	GU	SU	SU	SU
GD	SD	SD	SD	GD	CR	GU	SU	SU	SU
CR	SD	SD	SD	GD	CR	GU	SU	SU	SU
GU	SD	SD	SD	GD	CR	GU	SU	SU	SU
SU	SD	SD	SD	GD	CR	GU	SU	SU	SU
MU	-	-	-	-	-	-	-	-	-
TP	-	-	-	-	-	-	-	-	-

Table 6. Fuzzy Associative Memory (FAM) for for category 4 of the second fuzzy system.

3.6 Experimental Results Of Goalie Strategy

To test the performance of the goalie using the above strategies, a series of real-time experiments were run. Tests were conducted for each of the goalie-ball situation categories. Figures 23, 24, 25, 26, 27, and 28 show the results of such test runs.

Ball's X-coordinate, X_b

Goalie's X-coordinate, X_r	IN	SI	ZE	SO	MO	FO	VF
IN	1	1	1	1	1	1	1
SI	1	1	1	1	1	1	1
ZE	1	1	1	1	1	1	1
SO	3	3	3	3	4	4	4
MO	2	2	2	2	4	4	4
FO	2	2	2	2	2	4	4
VF	2	2	2	2	2	2	2

Table 7. Fuzzy Associative Memory (FAM) first fuzzy system.

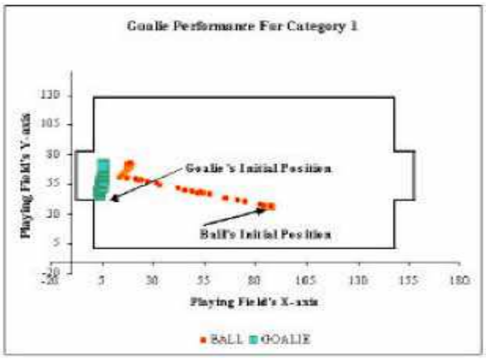


Fig. 23. Category 1 test performance of for the goalie.

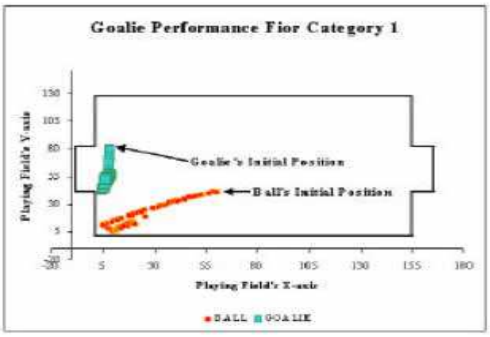


Fig. 24. Category 1 test performance of goalie.

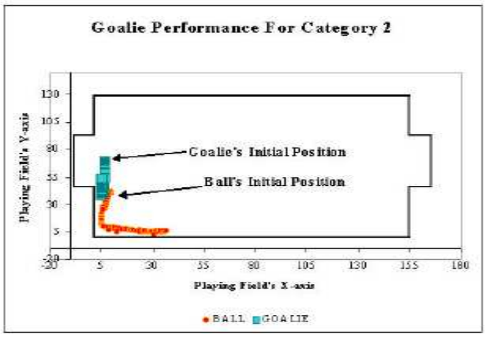
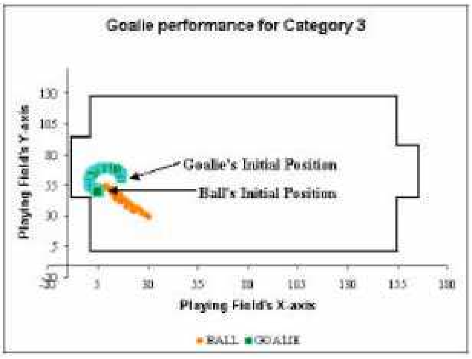


Fig. 25. Category 2 test performance of goalie.



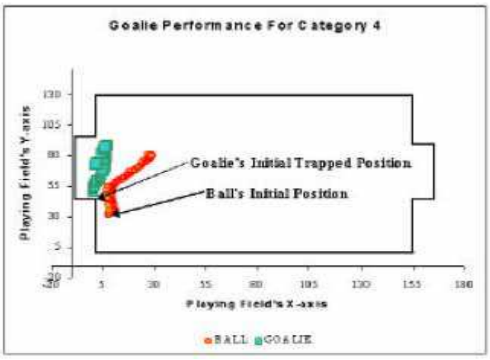


Fig. 26. Category 3 test performance of goalie. Fig. 27. Category 4 test performance of goalie.

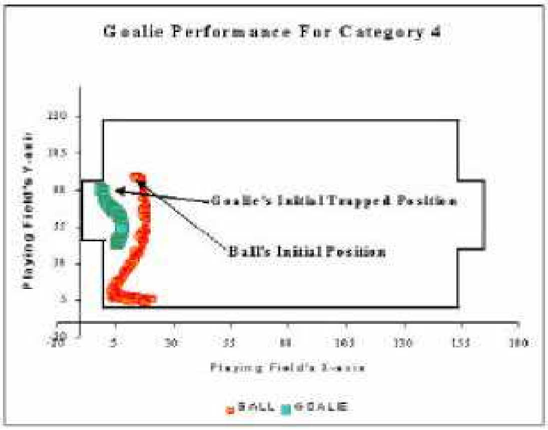


Fig. 28. Category 4 test performance of goalie.

Figure 23 shows the result of the test made on the goalie with a situation that falls under category 1. The ball was initially placed moderately far from the goal and was pushed towards the center of the goal. The goalie which was initially positioned at the lower part of the goal reacted by moving ahead towards the predicted point of intersection between the ball's path and the goalie's line-of-action and successfully blocked the ball there.
The situation in Figure 24 still falls under category 1 except that the ball was not moving towards the goal but to the lower portion of the playing field. The goalie initially on the upper part of the goal reacted by moving towards the lower part of the goal and positioned itself there to close the connecting line between the goal and the ball.
Figure 25 is a situation where the goalie is on its line-of-action and sensed the ball was entering the goal area. This situation falls under category 2. The goalie's reaction was to kick the ball out from the goal area and moved back to the lower portion of the goal again waiting for the ball's next approach.
Figure 26 shows a situation where the goalie is out from its line-of-action with the ball threatening to enter the goal. This situation falls under category 3. The goalie cannot go

directly to the ball because it would drive the ball to its own goal. Instead, the goalie made a circling move to successfully block the ball from the inside of the goal.
Figures 27 and 28 are situations where the goalie in both cases was initially trapped inside the goal. These are situations that fall under category 4. In both cases the goalie remained on its initial positions until the ball approached the goal then it followed the ball in the y direction until the ball threatens no more.

4. Obstacle Avoidance

Obstacle avoidance uses a fuzzy logic system that has the angle and distance of the robot with respect to the obstacle as its inputs and generates the velocities of the robot's left and right motors that will effectively avoid any perceived obstacle.
As shown in Figure 29, obstacle avoidance starts by detecting obstacles (walls, teammates, and opponents) within a 30 cm circle whose center is 50 cm ahead of the robot along its path towards a given destination. The detected obstacle that is nearest to the robot will be considered as the foremost obstacle and will be the one to be avoided as soon as the robot draws nearer to it by a distance lower than 35 cm. When this happens the input entities to the fuzzy logic system will be computed and the appropriate velocities for the left and right motors are generated to effectively avoid the perceived obstacle.

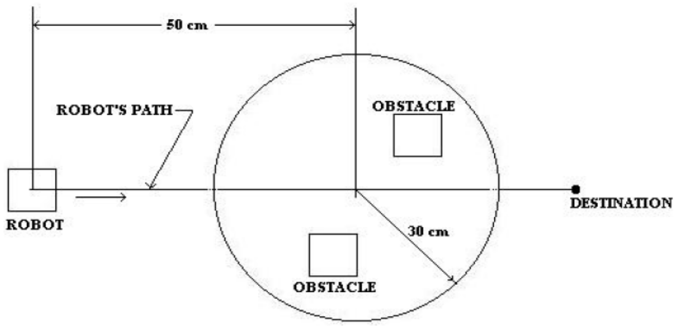


Fig. 29. Detecting obstacles.

Input Distance	Input Angle	Motor Speed
ZE: Zero	NL : Negatively Large	VL: Very Low
VN: Very Near from the obstacle	NM: Negatively Medium	ML: Moderately Low
NE: Near the obstacle	NS: Negatively Small	SL: Slightly Low
SF: Slightly far from the obstacle	ZE: Zero	LO: Low
FA: Far from the obstacle	PS: Positively Small	SH: Slightly High
VF: Very far from the obstacle	PM: Positively Medium	MH: Moderately High
	PL: Positively Large	VH: Very High

Table 8. Variables used in the membership functions of fuzzy logic obstacle avoidance.

The consequences of the fuzzy rules can be designed such that as the robot becomes nearer to the obstacle, the motors' speeds drive the robot away from it but towards the designated destination. Figures 30 to 32 show the membership functions of the inputs and out entities of the fuzzy logic system used for obstacle avoidance. Table 8 shows the description of the variables used. Tables 9 and 10 show the fuzzy associative memory (FAM) for the left and right motors.

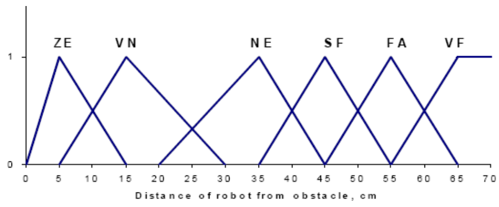


Fig. 30. Membership functions of input distance for obstacle avoidance.

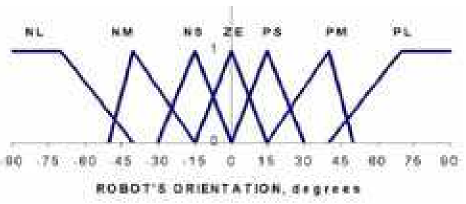


Fig. 31. Membership functions of input angle for obstacle avoidance.

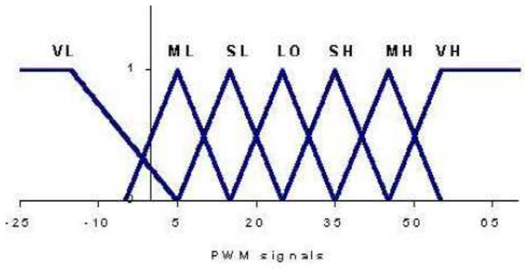


Fig. 32. Membership functions of motor speeds for obstacle avoidance.

	NL	NM	NS	Z	PS	PM	PL
ZE	SL	SL	SL	SL	VL	VL	VL
VN	SL	SL	SL	SH	ML	VL	VL
NE	VH	VH	MH	VH	SH	LO	LO
SF	VH	VH	MH	VH	SH	LO	LO
FA	VH	MH	MH	VH	SH	SH	LO
VF	VH	MH	MH	VH	SH	SH	LO

Table 9. Left Motor FAM for obstacle avoidance.

	NL	NM	NS	Z	PS	PM	PL
ZE	VL	VL	VL	SL	SL	SL	SL
VN	ML	ML	ML	SH	SL	SL	SL
NE	LO	LO	SH	VH	MH	VH	VH
SF	LO	LO	SH	VH	MH	VH	VH
FA	LO	SH	SH	VH	MH	MH	VH
VF	LO	SH	SH	VH	MH	MH	VH

Table 10. Right Motor FAM for obstacle avoidance.

4.1 Obstacle Avoidance Experimental Results

Figure 33 and Table 11 show a home robot prevents collision with other robots by effectively avoiding them before hitting the ball towards the goal.