An Improved Fuzzy Mechanism for 3D Soccer Simulation Agent's Shoot Skill

Mehrdad Mozafari, Amin Milani Fard, *Student Member, IEEE*
Vahid Salmani, *Student Member, IEEE*, and Mahmoud Naghibzadeh

***Abstract*-- RoboSoccer simulation is a complex multi-agent system in which agents play the role of soccer players and the main goal is to have a perfect domain for research and investigation on artificial intelligence. This paper intends to propose a new scoring module to select the best point on the goal line to shoot, considering player's position, catching and shooting time difference, and distance to target. Two different approaches have been implemented for this purpose. The first one is a simple decision making policy and the latter one takes advantage of statistical measurement and fuzzy decision making. Results show the superiority of the proposed method performance in over our previous work implemented on Nexus 3D RoboSuccer team of Ferdowsi University.**

***Index Terms*-- RoboCup Soccers Simulation, Interpolation, Fuzzy Decision Making, Shoot, Multi-gent System**

I. Introduction

THE main goal of robotic soccer is to have a perfect domain for researchers and a standard problem for investigating and examining new artificial intelligence as well as multi-agent approaches and techniques. In all soccer matches scoring a goal is the ultimate purpose. Therefore providing an effective scoring policy is of high importance. Shooting is defined as finding a point within the goal in which there is the highest scoring probability considering the current ball handler position. In RoboCup simulation league, teams of 11 autonomous software agents compete against each other by using RoboCup soccer server simulator software which is available from the official simulator website [1].

The ball controller agent is capable of performing shoot, passes or dribble actions and informs the server of his decision so that the server updates the playing environment. Considering the noise produced by the soccer server and uncertainties which affect all the perceptions and actions of an agent, shooting as the most important action, is influenced by a a random tolorance.as shown in Fig. 1, the ball controller agent is to find the best shoot among a set of feasible shoots. As a result we need an efficient algorithm that guarantees a scoring policy with upmost success probability.

II. Related Works

An intelligent agent uses the recently received information from the server and decides its best possible action. Considering parameters of each of the three possible actions (shoot, dribble, and pass), the information received from the surrounding area and the existing conditions can be divided into two parts: The information that is related to only one specific action and the information that is common among all three actions [2]. As examples of specific parameters we can name *shoot distance* and *target angel* for shoot, *number of opponents around* for pass and dribble, and for common measures action *interception probability* can be mentioned.

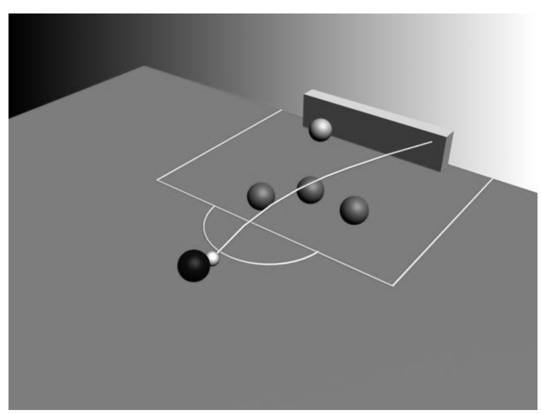


Fig. 1. A feasible shoot for an agent in a simulated environment.

Authors are with the Software Simulation and Modeling Laboratory at Department of Computer Engineering of Ferdowsi University, Mashhad, Iran.

M. Mozafari (e-mail: me_mo72@stu-mail.um.ac.ir).
A. Milani Fard (e-mail: milanifard@stu-mail.um.ac.ir).
V. Salmani (e-mail: salmani@ferdowsi.um.ac.ir).
M. Naghibzadeh (e-mail: naghib@ferdowsi.um.ac.ir).

The particular parameters of each action can be used as measures to evaluate different feasible actions and find out which one is the best. An action is considered to be feasible if, firstly it can be accomplished by the agent and, secondly, the ball can not be overtaken by the opponent during the action execution. To evaluate possible actions, various methods have been suggested [2, 5, 6, and 7]. Given the specific measures as well as the common ones, we showed in [2, 3] that there are two ways to evaluate each possible action: One-phase and two-phase decision-making mechanisms.

A. One-Phase Decision Making Mechanism

In our one-phase evaluation method, we use a specific weight for each parameter that affects an action. Each weight can be either a reward or a punishment whose summation for each one of the possible actions can result in a computed priority that recommends the most reasonable action. To obtain the weights, we start with an initial value for each weight. Afterward, the agent is made to contest several times and after each contest, the weights are readjusted. For example, in evaluation of the two actions A1 and A2, assuming A1 is better than A2, if evaluation module computes a higher priority for A2, the weights are adjusted by increasing the weights of those parameters which have more positive effect on A1 and decreasing those which have more positive effect on A2. This process is similar to the supervised learning [4], but it is performed offline. The weights will gradually adjust to a stable value. To evaluate the priority for each one of the possible actions, both specific and common measures are used. The highest calculated priority determines the preferred action. The action with the highest priority is then recognized.

In our experiments, we realized that if the decision-making process is broken into two phases, the number of parameters to deal with is reduced and the process is better managed. This lesson is what we learned by monitoring and analysis of numerous test runs.

B. Two-Phase Decision Making Mechanism

To determine the best action from amongst all possible ones for a given situation, we first recognize the best of each action, i.e., the best shoot, the best dribble, and the best pass, independently. It is clear that, when the best possible shoot is sought the parameters that affect the shooting action are considered, only. For dribble and pass actions a similar process is followed. In the next phase, we select the best of the bests, i.e., the best action from amongst the three best actions shoot, dribble, and pass. In this work we focused on proposing an improved mechanism to find best shoot action as follows.

III. Best Shoot Selection Mechanism

To find the best point on the goal line to shoot, it is necessary to evaluate all points and obtain the one with the maximum calculated priority. Consequently we designed an algorithm which firstly eliminates the points at which ball can not reach due to opponent interception (fig. 3) and evaluate the remaining points using the algorithm depicted in fig. 2.

input: CandidateTargetPositions
output: BestTargetPosition
Description: Evaluate candidate positions to find the best one

```
1 For all candidates ShootInfo
2  if ShootInfo.point is not interceptable
      then ShootInfo.Score =
Eval(ShootInfo)
3 Return ShootInfo with maximum score
```

Fig. 2. FindBestShoot Algorithm.

input: BallVelocity
output: Ball Interception Probability
Description: this algorithm determines is the ball can be intercepted while shooting using simulation. In each simulation cycle, OpponentMovementRadious (OMR) increases MaxPlayerSpeed. If OMP can intercept the ball moving path before it reaches the target, the algorithm returns a non-zero interception probability as follow:

```
 1 MinVel = 0.1
 2 MaxPlayerSpeed = 0.25 // meter per cycle
 3 while BallVel > MinVel do
 4   if BallVel > 1.5 then
         BallDecay = 0.78
 5   else if BallVel > 1 then
         BallDecay = 0.80
 6   else if BallVel > 0.85 then
         BallDecay = 0.85
 7   else BallDecay = 0.88
 8   BallPos = BallPos + BallVel
 9   BallVel = BallVel * BallDecay
10   for each opponent player do
11     OMR = OMR + MaxPlayerSpeed
12     if OpponentDistToBall < OMR then
         return (OpponentDistToBall-OMR)/20
13   end for
14 end while
15 target = BallPos
16 if OpponentDistToTarget - OMR < 4 then
      return OpponentCanInterceptBall
   //Opponent gets the ball near the target
```

Fig. 3. OpponentCanIntercept Algorithm.

As a rule of thumb, the shoot evaluation module deals with physical aspects of the ball controller agent, opponents, goalie, and the ball. The aim is to find the best point on the goal line that if the ball is kicked based on which information, it will pass the goalkeeper ending inside the goal.

A. *Temporal Difference Measurement*

One of the parameters we need for the evaluation module is the temporal difference between ball and the goalie movement to reach the target. In other words, we calculate if the goalie reaches the target point sooner than the ball. This parameter would be then fed into the next fuzzy phase to estimate the catch probability.

To do so we subtract the time take the agent to shoot considering rotation[2], from the time takes the goalie to reach the point and catch the ball. This subtraction trivially shows whether the ball pass the goalie or being intercepted.

Let T_b be the time takes ball to meet the target with the maximum speed, and T_r be the rotation time for the ball controller to adjust it's position beside the ball. T_g represents the time takes goalie to catch the ball (Fig. 4). Having calculated the above three parameters we define Δt as:

$$\Delta t = T_g - (T_b + T_r)$$

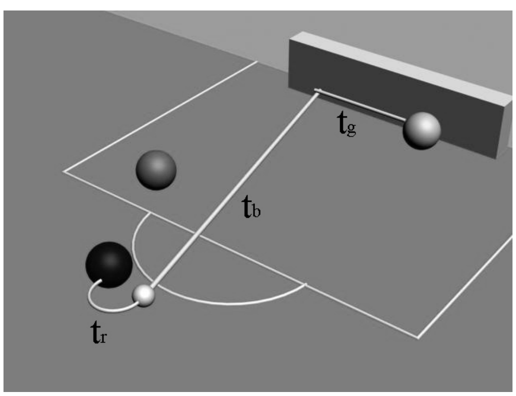


Fig. 4. Temporal Measurements.

If $\Delta t > 0$ then the ball would definitely pass the goalie and if $\Delta t < 0$ the ball would be intercepted. The greater Δt, the higher the probability of scoring goals. All these calculations were done assuming that there are no other agents except the goalkeeper in front of the ball controller to deviate the ball's direction.

In order to approximate the physical features of the environment, 100 of offline training test cases in which an agent shoots the ball from certain point toward goal were done and results saved on a log file. Having saved the above data, we try to formulate T_b, T_g, and T_r by means of interpolation[3]. The Gaussian function $T_b(d)$ calculates the time takes the ball to pass distance d. In which used coefficients are presented in Table I.

$$T_b(d) = a_1 \times e^{-\left(\frac{d-b1}{c1}\right)^2} + a_2 \times e^{-\left(\frac{d-b2}{c2}\right)^2} + a_3 \times e^{-\left(\frac{d-b3}{c3}\right)^2}$$

TABLE I
COEFFICIENTS (WITH 97% CONFIDENCE)

a1 = 359.8	b1 = 19.88	c1 = 1.514
a2 = 15.07	b2 = 23.7	c2 = 6.085
a3 = 4.846	b3 = 23.83	c3 = 19.91

Fig. 5 depicts the interpolated function for ball's movement. We have used a 9[th] degree polynomial equation to model the goalie's motion, in which used coefficients are presented in Table II.

$$T_g(x) = p_1 x^9 + p_2 x^8 + p_3 x^7 + p_4 x^6 + p_5 x^5 + p_6 x^4 + p_7 x^3 + p_8 x^2 + p_9 x^1 + p_{10}$$

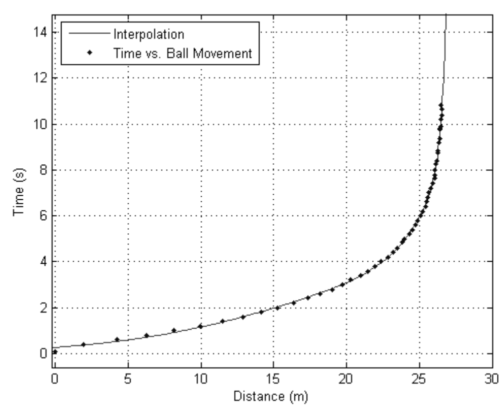


Fig. 5. Interpolation Fit – Ball Movement.

2 In 3D soccer simulation environment, unlike 2D version, agents are to be right behind the ball if they want to kick the ball straightly. In other words agents can only kick the ball in the straight line which passes from the center of the ball and the center of player's body, while there is a kick direction in 2D system.

3 In engineering and science one often has a number of data points, as obtained by sampling or some experiment, and tries to construct a function which closely fits those data points. This is called curve fitting. *Interpolation* is a specific case of curve fitting, in which the function must go exactly through the data points. [*From Wikipedia the free encyclopedia*]

TABLE II
COEFFICIENTS (WITH 95% CONFIDENCE)

p1 = 5.652e-006	p6 = 77.09
p2 = 0.0005268	p7 = 506.3
p3 = 0.02162	p8 = 2114
p4 = 0.5128	p9 = 5092
p5 = 7.74	p10 = 5395

Fig. 6 depicts the interpolated function for ball's movement.

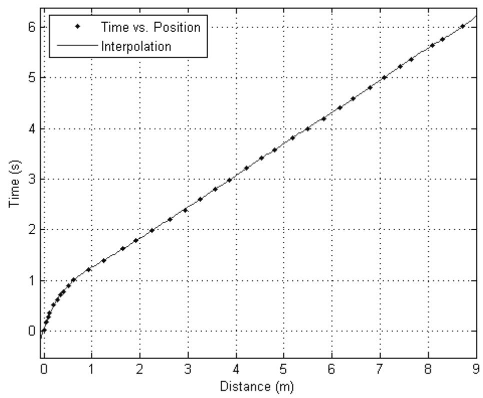


Fig. 6. Interpolation Fit – Goalie Movement.

The estimation function for agent's rotation around ball is interpolated as:

$$T_r(x) = p_1x^7 + p_2x^6 + p_3x^5 + p_4x^4 + p_5x^3 + p_6x^2 + p_7x^1 + p_8$$

That used coefficients are presented in Table III.

TABLE III
COEFFICIENTS (WITH 94% CONFIDENCE)

p1 = -8.247e-016	p5 = -1.814e-005
p2 = 1.057e-012	p6 = 0.001153
p3 = -5.362e-010	p7 = -0.01283
p4 = 1.366e-007	p8 = 2.001

Fig. 7 depicts the interpolated function for rotating around ball.

Candidate shooting targets is a set of 25 points distributed along the goal line4 with 30cm interval. Fig. 8 shows temporal difference measurement (Δt) through the goal line.

4 Goal width in 3D soccer simulation is 7.32 m

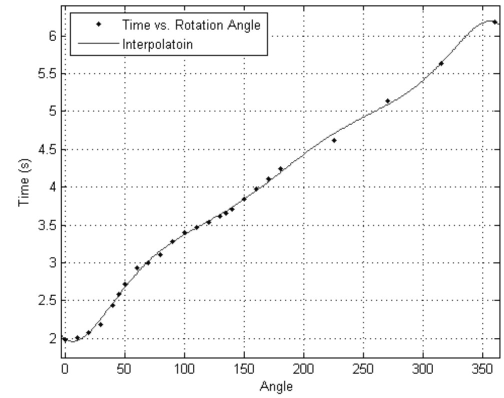


Fig. 7. Interpolation Fit – Turning Around Ball.

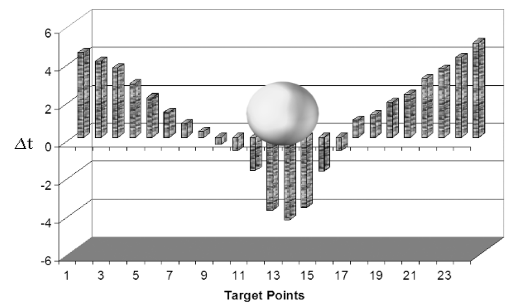


Fig. 8. Temporal Difference Measurement (Δt) Through The Goal Line.

B. Fuzzy Decision making

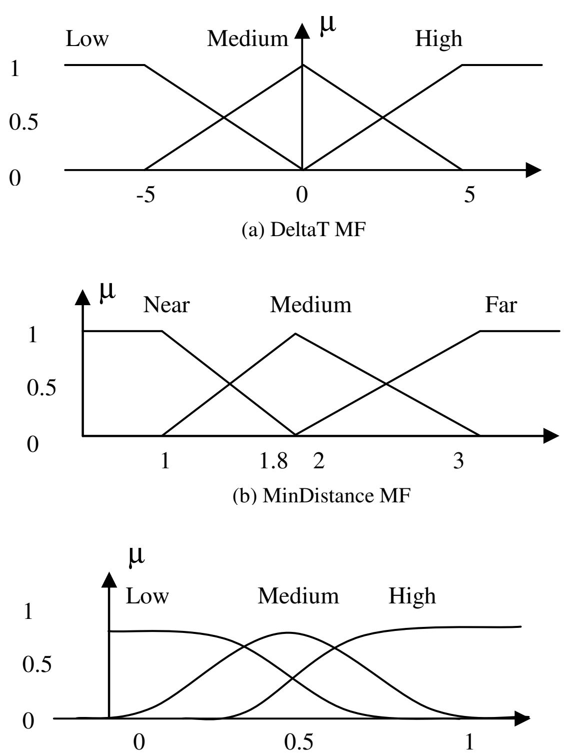
In this work, we proposed a fuzzy approach to select best shoot decision. It has shown that fuzzy systems provide a simple, efficient, and fast way of decision-making in comparison with the cumbersome and tedious process of applying many different rules for achieving the same results. Fuzzy sets were first introduced by Zadeh as a means of representing and manipulating data that is not precise, but rather fuzzy [8]. We expected the fuzzy system to be appropriate for shoot evaluation process in the soccer simulation environment, considering the noise produced by the soccer server and uncertainties which affect all the perceptions and actions of the agents. To solve this problem, we have used some concepts and techniques from fuzzy logic theory [9,10] as opposed to crisp set theory and the following steps are to design any fuzzy system [10,11].

Definition of membership functions (MFs) and linguistic variables are the first steps in fuzzy system designing. Each linguistic variable contains terms which are interpretation of technical figures. The experimental MFs we used are depicted in fig. 10.

The second step in designing a fuzzy system is the creation of a fuzzy logic rule base which supplies the knowledge of the system. [12]. A fuzzy logic rule is an *if-then* rule. The *if* part is a fuzzy predicate which is defined in terms of linguistic values and fuzzy operators Intersection (t-norm) and Union (s-norm). The *then* part is called the consequent. There are many implementations of fuzzy union and intersection operators. In this paper we have used product t-norm [13] as our aggregate method as follow. Our fuzzy rule base includes 15 rules as fig. 9 shows.

1. **If** Minimum_Distance_To_Goalpost is near **then** Out_Probability is high
2. **If** Minimum_Distance_To_Goalpost is medium **then** Out_Probability is medium
3. **If** Minimum_Distance_To_Goalpost is far **then** Out_Probability is low
4. **If** Δt is high **then** Catch_Probability is low
5. **If** Δt is medium **then** Catch_Probability is medium
6. **If** Δt is low **then** Catch_Probability is high
7. **If** Out_Probability is high **and** Catch_Probability is high **then** Goal_Probability is very low
8. **If** Out_Probability is high **and** Catch_Probability is medium **then** Goal_Probability is low
9. **If** Out_Probability is high **and** Catch_Probability is low **then** Goal_Probability is medium
10. **If** Out_Probability is medium **and** Catch_Probability is high **then** Goal_Probability is low
11. **If** Out_Probability is medium **and** Catch_Probability is medium **then** Goal_Probability is medium
12. **If** Out_Probability is medium **and** Catch_Probability is low **then** Goal_Probability is high
13. **If** Out_Probability is low **and** Catch_Probability is high **then** Goal_Probability is medium
14. **If** Out_Probability is low **and** Catch_Probability is medium **then** Goal_Probability is high
15. **If** Out_Probability is low **and** Catch_Probability is low **then** Goal_Probability is very high

Fig. 9. Fuzzy Rule Base.



(a) DeltaT MF

(b) MinDistance MF

(c) Out / Goal/ Catch Probability MF

Fig. 10. Membership Functions.

IV. Simulations and Results

The proposed algorithm was implemented in C++ on Nexus soccer simulation team [2, 3]. Experiments were done under Linux SUSe 10 distribution operating system on a desktop computer with Pentium 4 CPU 2.5GHz and 1GB RAM. To measure the shoot performance, *precision* measure was used as the ratio of the number of goal retrieved to the number of shoots through the goal expressed as a percentage.

$$\text{Precision} = \frac{\text{Number of Goals}}{\text{Number of Shoots}}$$

As table IV shows, the results of 50 shoots comparing fuzzy approach and the non-fuzzy one, confirm the proposed method's superiority.

TABLE IV
THE RESULTS OF 100 SHOOTS

Number of Shoots	Simple Shoot Evaluation	Proposed Shoot Evaluation
10	6	7
20	11	13
30	13	18
40	19	23
50	22	28
Avg Precesion	42%	51%

Fig. 11 indicates that the precision of the fuzzy approach has a better and smoother line in contrast with the non-fuzzy one which it's precision changes time to time.

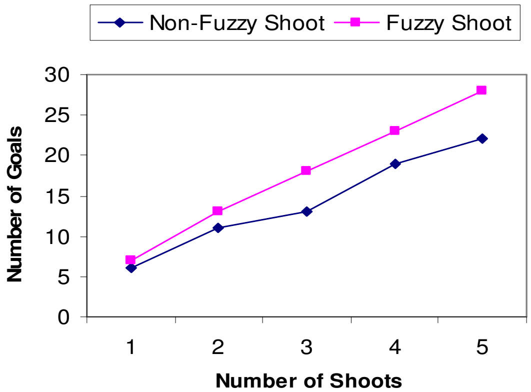


Fig. 11. Number of Goals In Non-Fuzzy and Fuzzy Shoot System.

V. CONCLUSION

To evaluate agent possible actions more precise and specific measures are needed. In this paper we have devised a fuzzy shoot action selection method to be used in our previous fuzzy two-phase evaluation methods. The outcome clearly showed the superiority of the proposed fuzzy method.

VI. ACKNOWLEDGMENT

The authors gratefully acknowledge the contributions of members of *Nexus*, the Ferdowsi University 3D RoboCup soccer simulation team, for their modest cooperation.

VII. REFERENCES

[1] M. Chen, E. Foroughi,, Robocup Soccer Server manual 7.07, August, 2002. RoboCup Federation. available: http://sserver.sourceforge.net.

[2] V. Salmani, M. Naghibzadeh, F. Seifi, and A. Taherinia, “A Two-Phase Mechanism for Agent's Action Selection in Soccer Simulation”, The Second World Enformatika Conference, WEC'05, Istanbul, Turkey, pp. 217-220, February 2005.

[3] V. Salmani, A. Milani Fard, M. Naghibzadeh, "*A Fuzzy Two-Phase Decision Making Approach for Simulated Soccer Agent*", IEEE International Conference on Engineering of Intelligent Systems, pp. 134-139, Islamabad, Pakistan, April 22-23 2006

[4] S. J. Russell, P. Norvig, *Artificial Intelligence A modern Approach*, Prentice Hall, 1995.

[5] L. P. Reis, N. Lau, FC Portugal Team Description: RoboCup 2000 Simulation League Champion, FC Portugal 2000 Team Description Paper.

[6] P. Stone, D. McAllester, “An Architecture for Action Selection in Robotic Soccer,” pp. 1-8. AT&T Labs - Research. Florham Park, NJ, 2001.

[7] P. Stone, Layered Learning in Multiagent Systems: A Winning Approach to Robotic Soccer, PhD Thesis, Computer Science Department, Carnegie Mellon University, Pittsburgh, 1998.

[8] L. A. Zadeh, “Outline of a New Approach to the Analysis of Complex Systems and Decision Processes,” IEEE Transaction Systems Man. Cybern, vol SMC-3, no. 1, pp. 28-44, 1973.

[9] L. A. Zadeh, “Fuzzy Sets", Information and Control, 1965. 338-353

[10] L. Wang, A Course in Fuzzy Systems and Control, Prentice-Hall, 1997

[11] C. V. Altrock, “Fuzzy Logic & Neurofuzzy Applications Explained,” 1995.

[12] B. Kosko, Neural Networks and Fuzzy Systems, 1992, Englewood Cliffs: Prentice Hall

[13] J.-S. R Jang, C.-T Sun, E. Mizutani, “*Nuro-Fuzzy and Soft Computing*”. Englewood Cliffs, NJ: Prentice-Hall, 1997.

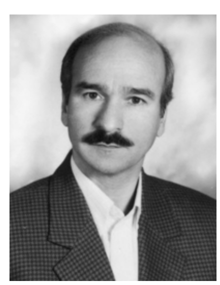
[14] A. Rabiee, N. Gh. Aghaee. “A Scoring Policy for Simulated Soccer Agents Using Reinforcement Learning” 2nd International Conference on Autonomous Robots and Agents, December 13-15, 2004 Palmerston North, New Zealand

VIII. BIOGRAPHIES

Mehrdad Mozafari received his B.S. degree in computer engineering, with concentration in intelligent agents, from Ferdowsi university of Mashhad under supervision of professor Mahmoud Nagibzadeh. He is currently a M.S. student in McMaster university. Mozafari is a member of the Nexus 3D soccer simulation team of Ferdowsi university.

Amin Milani Fard is currently a B.S. student in computer engineering at Ferdowsi university of Mashhad, under supervision of professor Mahmoud Nagibzadeh. He is a member of the Nexus 3D soccer simulation team of Ferdowsi university. Milani Fard is also an IEEE/ACM student member. His research is focused on intelligent data mining, distributed AI, and softcomputing.

Vahid Salmani received his B.S. degree in computer engineering, with concentration in intelligent agents, from Ferdowsi university of Mashhad. Salmani is currently a M.S. student in software engineering, under supervision of professor Mahmoud Nagibzadeh. He is the leader of the Nexus soccer simulation team and the director of software simulation and modeling laboratory. He is also an IEEE student member.



Mahmoud Naghibzadeh received his M.S. in computer science and PhD in electrical engineering from University of Southern California (USC). He is currently a full professor at the department of computer engineering, Ferdowsi university of Mashhad. He also is the supervisor of Nexus soccer simulation team and software simulation and modeling laboratory. His researches include operating system design concepts especially scheduling real-time processes and distributed databases design concepts.