Nexus2D Team Description Paper

Hosein Mohebbi, Reza Etemadi, Seyed Morteza Malekabadi, Messiah Abolfazli Esfahani, Majid Askari

Nexus Simulation and Modeling Lab, Ferdowsi University of Mashhad, Iran

Nexus@um.ac.ir

Abstract. Improving our previous skills was our main concern. Enhancement in making decisions using neural networks and also Advancement in player marking action using Genetic Algorithm are some of these improvements which is explained bellow. A customized log-player tool developed by our team to help visualized debugging for different situations is also explained. Noisy environment of the server encouraged us to develop a fuzzy mechanism for passing skill. This mechanism is also explained in the following.

1. Introduction

2D soccer simulation is one of the Robocup leagues which intends to eliminate areas concerning construction of a robot and its difficulties and instead to focus on cooperation between agents and other high-level issues. This paper is based on 2D soccer simulation server which is a physical simulation system. This server is a multi-agent system and has a dynamic environment [1]. Using simple and static methods is inefficient in a dynamic environment. Therefore, it is a good idea to use dynamic algorithms and methods in such an environment. In the following, these algorithms and methods are explained.

Nexus team was officially started in 2002, participating in different national and worldwide Robocup events. Current team members have also participated in national and international competitions as *Robotoos*, *051*, *Tempux*, and also *Nexus* teams. Our team is based on Agent-2D base code.

2. Enhancement in making Decision

2.1 Optimization the accuracy of shout

In order to optimize the accuracy of our shout we use a function which calculate the probability of the chance of the ball to be goal. In order to implement such this function we measure the chance of ball to be goal by using the information of our past shouts.

In other world, we get deeper of action tree by saying more accurate we traverse the action tree by get deeper in each level. As we are traversing the tree also there is also another job doing we also produce and process actions based on time cycles.

Tree traversal is a form of graph traversal and refers to the process of visiting (checking and/or updating) each node in a tree data structure, exactly once. Such traversals are classified by the order in which the nodes are visited. The following algorithms are described for a binary tree, but they may be generalized to other trees as well.

For completing above sentence I should say in the past we used to deep based on a constant value but now this traverse has optimized to be more accurately. In this way we consider we have time of whole clock cycle time, so we process in this time and then do the task. Comparing this optimized function by previous one although we make decision in more time and perhaps it is better to say it looks in this way we take more time to do the task but it is better to pay attention that although we take more time to take decision but we take better and more accurate decision.

Logically like most of the phenomenon perhaps taking more time to take decision look harmful but we must consider that by taking more time we are hope that we are taking more accurate decision.

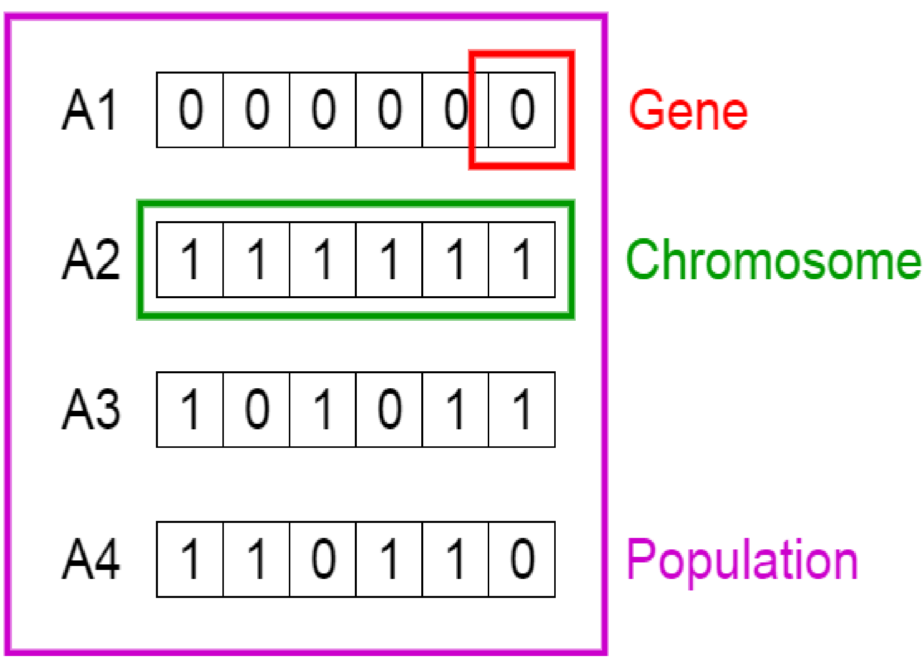
In penalties we have improved our block by using genetic algorithm to have a better block in penalties. We have implemented the genetic code with Matlab.

A **genetic algorithm** is a search heuristic that is inspired by Charles Darwin's theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation.

Genetic algorithm has 5 phases:

1. Initial population
2. Fitness function
3. Selection
4. Crossover
5. Mutation

Above 5 phases together implement ideal genetic algorithm which we are hope to have better gens and if we take a brief look at above phases we will understand that all of them is important as other phases are and we must pay attention to all of them not one of them.



In penalty blocks there are some parameters which we can use them for measuring the value of fitness function such as were the opposite player is standing and how far is the distance of ball from our goal and the degree between ball and our goal if we consider these parameters we are hope to have a good fitness function and then we could have probably a good genetic algorithm. We use this genetic algorithm In order to block more penalties than having them to be goal.

2.2 Defining the Problem

In 2D soccer simulation, precision of making Decisions is one of the most important factors that has a straight effect in match Result and it can change the process of the Game. Therefor increasing attention in making Decisions is one of challenges of this competitions.

It is clear that one of the factors increasing of attention is correct anticipating opponent. If we know that opponent's players in next cycles are which position, then we can make move suitable Decisions for current cycle.

Since now we have used chainaction structure for enhancement of performance but the problem this structure has is that in the time of anticipating next states, it considers opponent constant and only change the ball's position according to the current parameters.

Now we have progressed one more step and we try to anticipate opponent's movement as a result we consider next states more closely to the reality.

2.3 Solving the problem with neural network

The goal of the neural network is to solve problems in the same way that the human brain would.

Neural networks are a computational approach, which is based on a large collection of neural units, modeling the way a biological brain solves problems with large clusters of biological neurons connected by axons. Each neural unit is connected with many others, and links can be enforcing their effect on the activation state of connected neural units. Each individual neural unit may have a summation function which combines the values of all its inputs together. These systems are self-learning and trained, rather than explicitly programmed, and excel in areas where the solution is difficult to express in a traditional computer program.

One of the problems for opponent's movement in special places.

For example, it is possible one of the opponent's players changes its movement for marking one of our players or in other position it is possible to move for getting the ball.

In order to solve the problem, we modeled the problem then we used appropriate neural network's inputs and outputs then we run many games with different teams and collect so much data.

As a result, by computing this Data by using Matlab, we get to a neural network that can learn way of opponent's Game as time passes it can anticipate opponent's movement with more attentions.

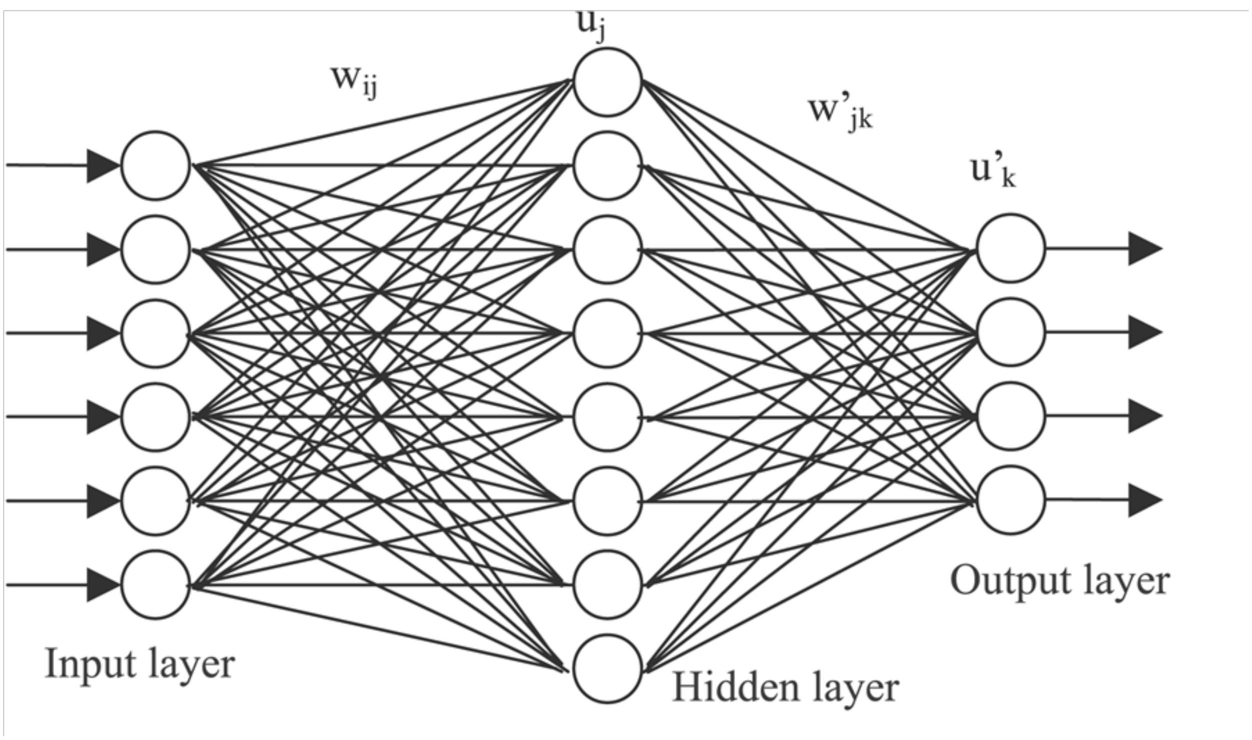


Figure 1- The structure of neural network

3 Genetic algorithm for player marking

3.1 Defining the Problem

Player marking is performed to prevent opponent player from passing the ball to a target player. This action is done by becoming as close to the target as possible. While the target player makes many quick movements and turns, finding a marking position for our player is one important challenge. The other important problem to be solved is the body angle our player is going to have in the marking position. It should be selected in a way that in case of the target player movement, the short distance between marker player and target player could be regained quickly.

These two variables – e.g marking position and body angle - are to be valued based on sixteen different factors like distance between target player and the marker, body directions of both marker and target, velocities, area of field players are located in and etc. All these factors would affect the performance of marking procedure.

3.2 Solving the problem using genetic algorithm

Genetic algorithm (GA) is a useful algorithm for optimizing problems, and our objective is to find sixteen optimized coefficients. Therefore, it sounds a good choice to use GA in this case. [2]

Performing the mark action while the marker has the shortest distance to the target in a duration of time is declared as the best situation. For a period of twenty cycles the mark execution would be scored like:

$$TotalScore = \text{ScoreInCycle}(n,\text{target}) + \text{ScoreInCycle}(n+1,\text{target}) + \ldots + \text{ScoreInCycle}(n+20,\text{target})$$

The lower this *TotalScore* is, the better. So, it's tried to obtain the coefficients in a way we have the least *TotalScore*. This is done by modeling differnet situations where the marker and target have different placements, executing the mark action every time. The goal is to minimize the *TotalScore* using GA.

First, we start with a hundred randomly generated genes. To produce the next generation, fifty best genes are chosen from the previous generation. Then, fifty genes are produced by mating pairs of the fifty best. Thirty of them go into the next generation directly, but we make mutation in twenty remaining and form the next twenty genes for next generation. Figure 2 illustrates the production of the next generation.

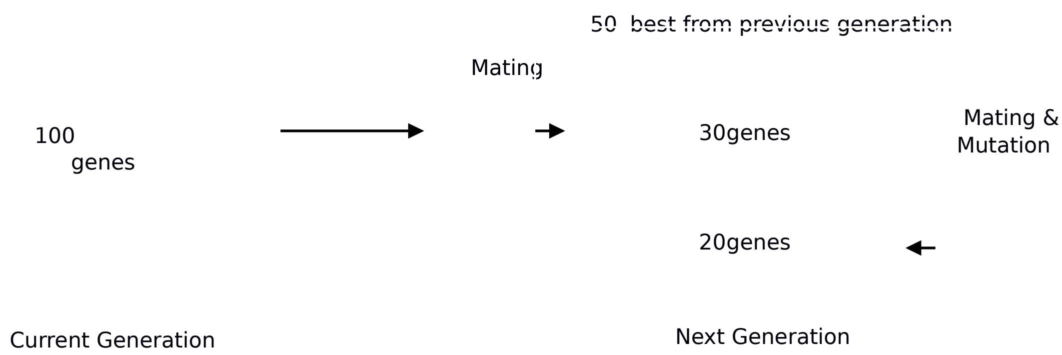


Figure 2 - The production of next generation

4 Customized log player tool

4.1 Defining the problem

Different skills in 2D Soccer Simulation depend on variety of factors and there usually is a score-based engine for both making decisions and performing the actions. Multiplicity of the factors and having many options makes program debugging very hard. Normally it's usual to print the data into text files for watching the efficiency of different parameters and generally for debugging purposes. Finding the problems and improvement based on these text files is sometimes very hard. Besides, having an imagination of some text data like positions and angles is much more difficult than seeing it in real server-like environment.

4.2 Tool development

We have developed a tool that allows us to visualize information and data. It's done by making some changes in the source of "rcsslogplayer"[7] and customizing it.

Agents save considered data of every cycle in a "XML" file. At the time of playing the game logs, the program mixes data files created by different agents for different cycles. As the result, the desired information would be shown in the logplayer at suitable cycles.

4.3 Usage example

This customized logplayer helps us in improvement of many skills. As an example, the usage of this tool in improving the passing skill is explained. One good way for passing is generating different passes, evaluating them and choosing the best one (Our passing algorithm will be described in next part by details).
For the evaluation part, the agent draws a small circle for every created pass in the appropriate point of field and attaches other desired data to the point. So, different passing options can easily be seen alongside each other and be compared by details. This tool is released on 15 February 2014[8].

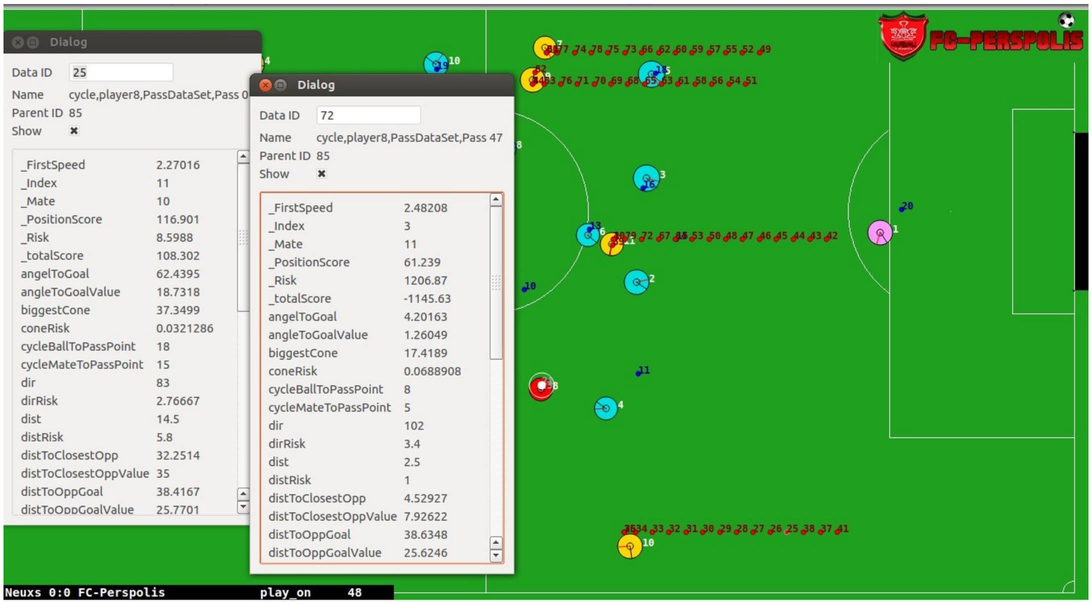


Figure 3. Customized log player tool

5 Passing [9]

Passing is a skill that its success is dependent on both passer and receiver agent. Both of them should have a clear understanding of the environment. Passer must guarantee that the ball will be received to the teammate without any threat. On the other hand, receiver must be aware of that the player who must receive the ball is itself, and it must have the ability to receive it. Since there is noise in the environment, our goal is not setting the ball in a specific point. Our main goal is to move the ball in a direction which the receiver has the potential to receive it in the best position and with the less probability of ball interception by the opponents.

5.1 Choosing the best pass mechanism

For finding the best pass, we have considered a score for all teammates. If the kickable agent passes the ball to a teammate, the corresponding score shows the pass success probability. The teammate with the highest score will be chosen for passing the ball to it.

5.2 Fuzzy system

There is noise along with all actions and perceptions of the agents in the simulation environment. On the other hand, fuzzy systems are not sensitive to noise. Therefore, fuzzy systems are a good choice as a decision making system in soccer simulation environment.

The main structure of a fuzzy system is illustrated in Figure 3. Some sets enter the fuzzy system as input sets, and some exit as output sets. In this section, we will study these sets [4].

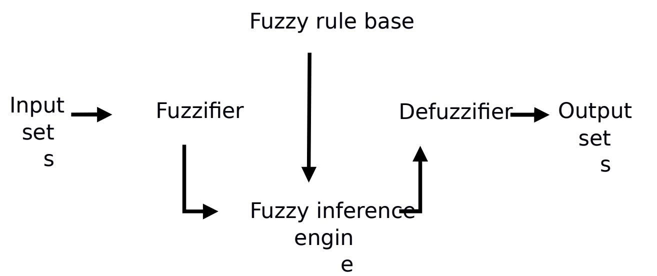


Figure 4 The main structure of the fuzzy systems

5.2.1 Input and output sets of the fuzzy system

One of the inputs of the system is the probability that opponents can interception the ball before it gets to its target. This is one of the important factors, and has a great effect on the performance of our system. If the ball is intercepted by opponent agents before reaching to its target, the pass is completely unsuccessful. Computing this probability is an important issue at this point. Figure 4 shows an algorithm to compute this probability. The algorithm determines the probability by which the ball can be intercepted while moving along its path. In each simulation cycle, opponent movement radius (OMR) is increased by the amount of maximum player speed. If the circle with OMR radius and with opponent agent center includes the ball, this algorithm will return a nonzero value.

```
MinVel = 0.1
MaxPlayerSpeed = 0.25 // meter per cycle
while BallVel > MinVel do
      if BallVel > 1.5 then
         BallDecay = 0.78
      else if BallVel > 1 then
         BallDecay = 0.80
      else if BallVel > 0.85 then
         BallDecay = 0.85
      else BallDecay = 0.88
      BallPos = BallPos + BallVel
      BallVel = BallVel * BallDecay
       for each opponent player do
       OMR = OMR + MaxPlayerSpeed
       if OpponentDistToBall < OMR then
         return (OpponentDistToBall-OMR)/20
       end for
```

```
end while

target = BallPos

if OpponentDistToTarget - OMR < 4 then
        return OpponentCanInterceptBall
        //Opponent gets the ball near the target
```

Figure 5- The algorithm for computing ball interception probability

Another input parameter is the distance between the target and opponent goal. This parameter has an inverse relation with the target score- the less this distance, the more score assigned to that target. In this case, the ball is passed to a teammate which is closer to the opponent goal. Therefore, the chance of obtaining a goal will be increased. The distance between the passer and target is considered as another input parameter of the system. The more this distance, the probability of reaching the ball to the target is decreased. It has reverse relation the pass success or in other words, the more this distance, the less pass success. The last input is the number of opponent agents around the target. The more number of opponents causes the pass success to decrease. Table 1 shows a list of inputs and output of the system.

Table 1- Inputs and output of the fuzzy system

Parameter	Input / Output
Ball Interception Probability	Input
Distance Between Target and Opponent Goal	Input
Distance Between Passer and Target	Input
Number of Opponents Around Target	Input
Pass Success	Output

5.2.2 Knowledge base

Fuzzy systems are based on knowledge and rules. The heart of a fuzzy system is its knowledge base which consists of a set of if-then rules. An if-then rule is a simple if-then statement that some of its words are described by membership functions [4].

The starting point of making a fuzzy system is finding a set of if-then rules from the knowledge of expert people or science of that special field. The next step is to combine these rules in a unique system. Different fuzzy systems use different methods and principles for combining these rules. Figure 5 shows the knowledge base of the system.

1. **If** OpponentCanIntercept is high **then** PassSuccess is low
2. **If** OpponentCanIntercept is medium **then** PassSuccess is medium
3. **If** OpponentCanIntercept is low **then** PassSuccess is high
4. **If** DistanceToGoal is far away **then** PassSuccess is low
5. **If** DistanceToGoal is far **then** PassSuccess is medium
6. **If** DistanceToGoal is near **then** PassSuccess is high

7. **If** DistanceToTarget is far away **then** PassSuccess is low
8. **If** DistanceToTarget is far **then** PassSuccess is medium
9. **If** DistanceToTarget is near **then** PassSuccess is high
10. **If** OpponentsNumber is many **then** PassSuccess is low
11. **If** OpponentsNumber is medium **then** PassSuccess is medium
12. **If** OpponentsNumber is few **then** PassSuccess is high

Figure 6- Our fuzzy knowledge base

In Figure 5, the words high, medium, low, and … are described by the membership functions of Figure 6, 7, and 8. Membership functions of *DistanceToGoal* and *DistanceToTarget* are the same, because both of them are distance.

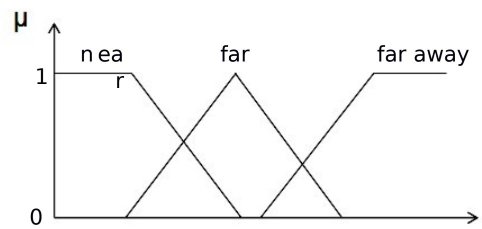


Figure 7- Membership function of *DistanceToGoal* and *DistanceToTarget*

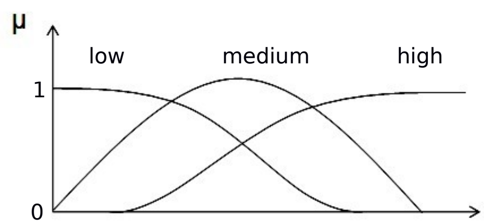


Figure 8- Membership function of *OpponentCanIntercept*

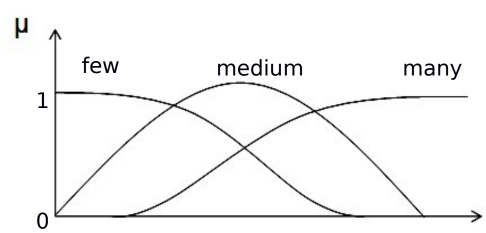


Figure 9- Membership function of *OpponentsNumber*

6 CCS Tool Added To rcssserver2D

One of the most important things in agent simulating like Soccer Robots is to find out our actions effect in physical environments which 2d soccer server makes it. Nowadays this can be done by using rcsslogplayer tool. This gets the offline log files and shows cycle by cycle game. Thus, teams should log their actions to a file and find out what is the actions problem in rcsslogplayer, which needs another tool to install near the server. We developed a tool for solving these problems, Cycle by Cycle Simulator (CCS), this simulator allows users to observe their action effects online in both standard and frameview of rcssmonitor. So, the cycle by cycle simulator tool is added to server and monitors due to the following advantages [6]:

1. Seeing our actions effect on physical environment simulated by soccer server cycle by cycle;
2. Finding suitable fitness function for optimization algorithms;
3. Making precise debugging actions online;
4. Seeing our ranking functions effect;
5. Viewing our online agent logs cycle by cycle;
6. Stopping the game for some instant to do some issues;
7. Changing the mode of game by online coach inside the game to get better results for their neural network and optimal algorithms;
8. Making better low-level executable actions.

7 References

1. Mao Chen, Klaus Dorer, Ehsan Foroughi, Fredrik Heintz, ZhanXiang Huang, Spiros Kapetanakis, Kostas Kostiadis, Johan Kummeneje, Jan Murray, Itsuki Noda, Oliver Obst, Pat Riley, Timo Steffens, Yi Wang, Xiang Yin, *Robocup Soccer Server: User Manual*, 2003
2. Frieder Stolzenburg, OliverObst, Jan Murry, *Qualitative Velocity and ball Interception*, 2002
3. Karen A. Kopecky, *Root-Finding*, 2007.
4. Li-Xin Wang, *A Course In Fuzzy Systems and Control*, New Jersey: Prentice Hall, 1996.
5. Stuart J. Russell, Peter Norving, *Artificial Intelligence: A Modern Approach*, 2nd edition, Practice Hall.
6. http://sourceforge.net/projects/ccsrcssserver2d/files/CCSServer_Manual.pdf/download
7. http://sourceforge.net/projects/sserver/files/rcsslogplayer
8. http://sourceforge.net/projects/logplayerplusdataviewer/files/
9. Nexus2D Soccer Simulation Team Description Paper, Mahdi Abolfazli, Masoud Jamili, Mostafa Sayahi, Saeid Askari, Mehrnaz Ghashgaei, Hadi Kalamat and Sirus Shahini, Robocup2013