Axiom 2013 Team Description Paper

Mohammad Ghazanfari, S. Omid Shirkhorshidi, Farbod Samsamipour,
Hossein Rahmatizadeh Zagheli, Mohammad Mahdavi, Payam Mohajeri,

S. Abbas Alamolhoda

Robotics Scientific Association and Multi-Agent System Laboratory
Department of Computer Engineering
Iran University of Science and Technology
Narmak, Tehran, Iran, 1684613114

Abstract. Axiom is a 2D soccer simulation team participated in many Robocup competitions such as Robocup 2012 (Mexico City). In our efforts we adopted A.I. techniques in order to enhance agents' performance. In this paper, we briefly described our recent researches about implementing A.I. algorithms in our team.

1 Introduction

Axiom is a team consisting of undergraduate and graduate students of Iran University of Science and Technology (IUST). Axiom is a member of IUST Robotics Scientific Association and has a close cooperation with IUST Multi Agent Systems Laboratory. We participated in many competitions such as Robocup 2012 (Mexico City). Some notable successes of Axiom are 3rd place in IranOpen 2011, 4th place in IranOpen 2012 and 3rd place in SharifCup 2012.

Our team is based on Agent2D base developed by H. Akiyama [1].

This paper is organized as follows. The section 2 describes our new offensive strategy, using Reinforcement Learning (RL) method [2]. In the next section we present our pass strategy that is implemented from scratch and our new safety checker based on Artificial Neural Network (ANN) [3]. The 4th section describes our new dynamic positioning system and in the last section we briefly conclude our work.

2 Adopting an RL-Based Offensive Strategy

As we described in Axiom 2012 team description paper [4], Hierarchal Reinforcement Learning is suitable for learning strategy in complex, dynamic and stochastic environments such as RCSS2D [5]. In first step we focused on offensive with-ball strategy, because it's not only an important part of a team, but also is one of our weaknesses. We chose a Bottom-Up approach to learn this strategy. It means that a successful attack ends with an appropriate shoot that leads to a score. So the agent in the last step of an offensive strategy (the leaves of offence Decision Tree) should pick the best possible shoot. Hence, finding an optimal shoot policy is the first step in hierarchy of our RL-Based Offensive Strategy. Now suppose the agent, that has this optimal shoot policy, can start learning from one step former (depth “d-1” in the Decision Tree). For example agent chooses between targets in goal to shoot, in one depth lower, the agent chooses between possible shoots, passes and directions to dribble. This depth presents situations which are expected to ends with a goal soon.

Our purpose in this approach is that the whole strategy would be learnt layer by layer up to the whole strategy. We implemented the depth “d” in the Decision Tree (shoot to goal) using Q-Learning. Now we are working on learning of the upper layers of strategy (depth “d-1”, such as strategy in penalty area).

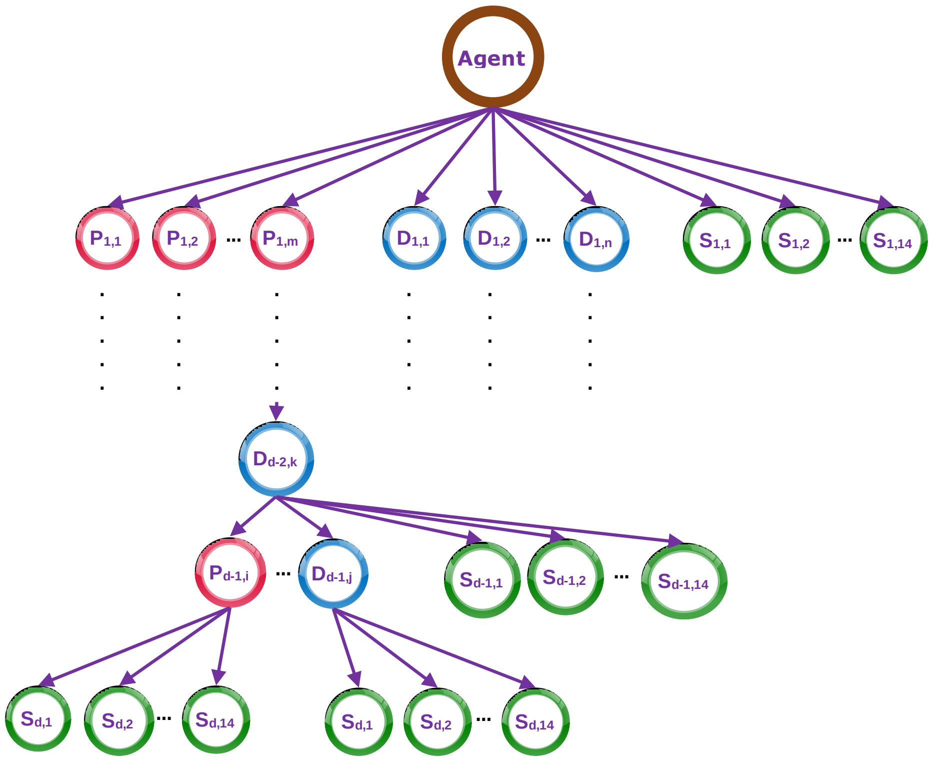

Figure 1. Offensive With-Ball Decision Tree

For learning the shoot, environmental characteristics, actions and options are described in [4] and here we mainly present our method to find the optimal policy.

2.1 Q-Learning Based Shoot

For implementing a Q-Learning method it needs to select the features that define a state, rewards, state space and how a scenario begins and ends [6]. We defined a scenario consist of three agents; a goalie, a kicker and a defender (which is the most dangerous opponent). The size of state-action space in this scenario is about 10^{16} and clearly it almost impossible to converge in such big state space like this. Hence we use abstraction and discretization to solve the problem. For our task, we define the state using a set of variables and features which are listed below in Table 1.

Table 2. Feature Description

Feature	Description
F1	The distance between the ball and the goal line that is divided into 8 none-equal parts.
F2	The angle created by the ball and two goal posts that is divided into 8 none-equal parts.
F3	The position of the goalie that is divided into 15 completely costume parts in front of the goal line.
F4	The direction and velocity of goalie which is considered as one feature that has 5 different cases, one for the case that the goalie is not moving and other 4 cases are defined according to goalie direction and velocity.
F5	The direction and velocity of the defender agent that is defined just like Feature 4.
F6	The distance between defender agent and the ball that is divided into 6 none-equal parts.
F7	The distance between defender agent and the shoot line that is divided into 7 none-equal parts.

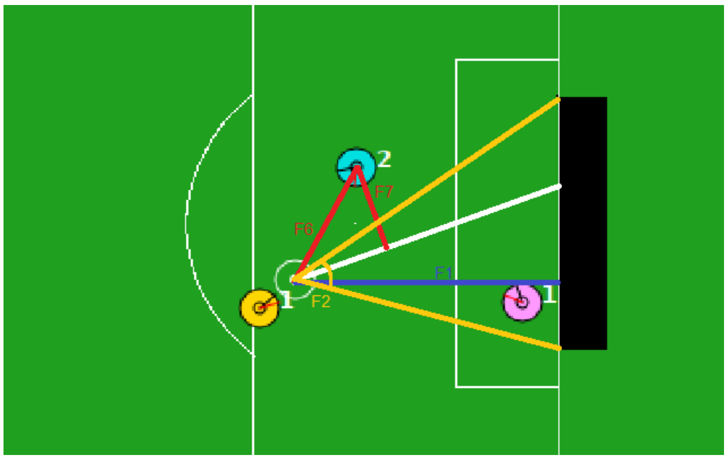

Figure 2. Some Shoot Features

After Abstraction and discretization as described in table1, the size of state-action space reduced to 10^6.

For this task, we defined 15 different actions; 14 shoot with maximum shoot Power to 14 different points in goal line and 1 action for not shooting. The rewards for every action in every state are described in Table 2.

Table 2. Rewards

The Action Result in Shoot Scenario	Reward
If the shoot ends with a score	+1
If the selected action is no-shooting	0
If the goalie takes the ball or the ball goes out of the field	-1
If the defender agent gets to the ball and takes it	-1.5

For this task we used around 300,000 train data created against goalie and defender of Agent2D Base (3.1.1). Q-learning method needs more data. For example in our task we need around 30 million train samples, in best case. So some Q-table cells will not be visited and leaved with their initiation value. There are some ways to solve this problem. One of these ways is to decrease the duration for every episode, it could be done by making the server just work on needed parts of the learning scenario and decreasing the simulation-step in server configuration. The other way is using a function approximator such as A.N.N to represent an action-value function or Q-function that maps state and action pair (s,a) [7,8]. We also can let our other offence methods handle these situations (states) that are not visited in the Q-table.

After these 300,000 train samples our agent had a considering improvement in its shoot skill. The average of successful shoots after using these train samples is 78.7% in 1500 test shoots.

3 Pass Algorithm

Pass is one of the most important skills in the soccer simulation 2D league. This year we decided to implement this skill from scratch. At the beginning, every candidate passes are analyzed by an Artificial Neural Network (ANN) Based Safety Checker Module. The output of this module is pass safety value that compared with a threshold. If it is greater than the threshold, will pushed in pass vector, otherwise, it will be discarded. After analyzing all of candidate passes with ANN Module, the best pass is chosen by another module named Pass Decision Maker.

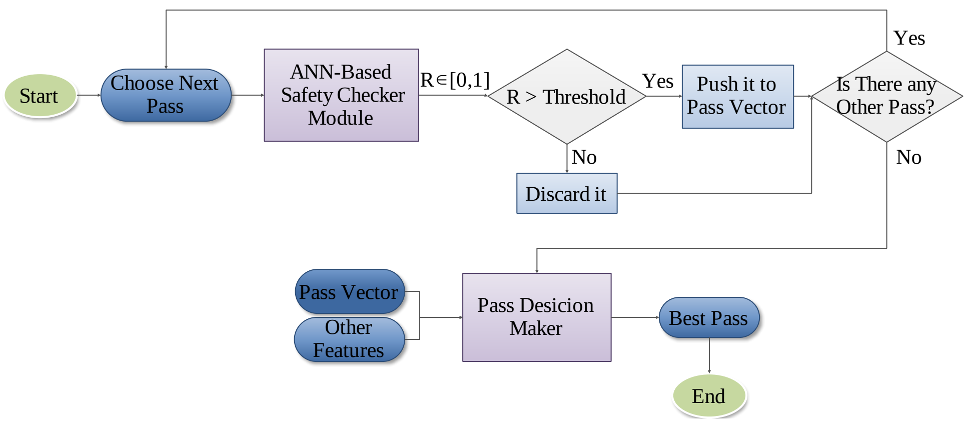

Figure 3. Pass Algorithm

3.1 ANN-Based Safety Checker Module

Our previous input features of ANN Module were completely described in [4]. In new ANN Module we decided to change these features in order to achieving more reliable passes. Now we consider these features as input of ANN:

- Opponents relative body angle to the ball owner (α)
- Receiver relative body angle to the ball owner
- Opponents relative face angle to the ball owner (β)
- Receiver relative face angle to the ball owner
- Receiver distance from ball owner (D_1)
- Opponents distance from ball owner (D_2)
- Opponents distance from the pass line (D_3)
- Effective opponents velocity (V_{eff})

These features are represented in Figure 4.

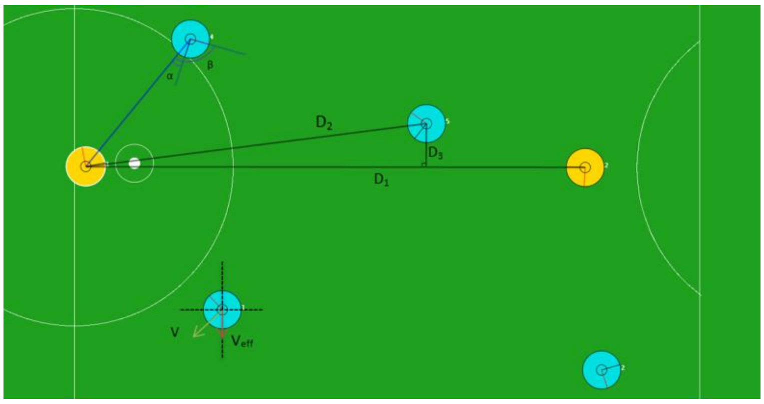

Figure 4. ANN Module Features

3.2 Pass Decision Maker

One of the major changes in pass system is pass evaluation criteria for choosing the best pass. In the beginning, Rate variable is set to a constant value (maximum), and then it reduced to its final value step by step by these formulas, according to some features like receiver area, opponents distance to pass line, number of opponents in area, receiver distance and etc.

$$Rate \mathrel{*}= \left(1 - \left(\frac{TargetArea}{10}\right)\right) \qquad 0 \leq TargetArea \leq 7$$

(Formula 1)

$$Rate \mathrel{*}= min(1.0, \alpha * \sqrt{NearestOppDist * \beta})$$

(Formula 2)

$$Rate \mathrel{*}= \alpha^{-1.0*NumberOfOppInPassSector}$$

(Formula 3)

$$Rate \mathrel{*}= \min\left(1.0, \alpha^{-1.0*\frac{ReceiverDist}{\beta}}\right)$$

(Formula 4)

4 Dynamic Positioning System

One of our previous problems was inappropriate positioning, caused players gathering in some areas and leaving other areas. This problem became crucial when the opponent team implements a good mark skill. Hence, we developed a Dynamic Positioning System.

In this system, receiver uses an algorithm in its without ball procedure, that chooses best position for its next move. This algorithm works as follows; the positioner agent chooses its targets among points that are safe to pass from the ball owner point of view. The safety of a target for pass is evaluated using ANN-Based Safety Checker Module which is represented in section 3.1.

In the next step, the agent chooses the best target for positioning among safe targets according to parameters listed below:

- Positioner agent situation when it receives the ball at the target point
- Opponents density in target point area
- Distance of target point from opponent goal
- Distance of target point from ball owner

5 Conclusion and Future Work

2D soccer simulation is one of the most appropriate domains for developing A.I. techniques because of its complexity and resemblance to real world [4]. This paper describes the current efforts of Axiom 2013 including RL Based Offensive Strategy, new pass algorithm and Dynamic Positioning System.

Our future work consists of two main parts. The first one is concerned with developing RL Based Offensive Strategy in order to learn the whole offensive strategy. There are some problems such as “Curse of Dimensionality” and the time that each train cycle lasts.

Updating of World Model is another important work to do because of its direct effect on every other skill. For this task, there are three relevant parts that should cooperate with each other. These parts includes view method of the agent, using inter agent communication and World Model prediction for every agents. As our future work, we are trying to implement an intelligent algorithm that make this cooperation works well.

6 References

1. Akiyama H., http://rctools.sourceforge.jp.
2. Barto A.G. and Sutton R.S., “Reinforcement Learning: An Introduction”, MIT Press, 1998.
3. Haykin S., “Neural Networks: A Comprehensive Foundation (2 ed.)”, Prentice Hall, 1998.
4. Ghazanfari M., Shirkhorshidi S. O., Beydaghi A., Samsamipour F., Rahmatizade H., Mahdavi M., Zamanipour M., Mohajeri P., Mirhashemi S. M. H., “Axiom 2D Team Description Paper”, Robocup 2012, Mexico City, Mexico, 2012.
5. Barto A. G., Mahadevan S., “Recent advances in hierarchical reinforcement learning”, Journal of Discrete Event Dynamic Systems, Springer, 2003.
6. Dash, Liu, “Feature Selection for Classification”, Journal of Intelligent Data Analysis, volume 1, pp. 131-156, 1997.
7. Kalyanakrishnan S., Liu Y. and Stone P., “Half Field Offense in RoboCup Soccer: A Multiagent Reinforcement Learning Case Study”.
8. Sutton R.S., Maei H.R., Precup D., Bhatnagar S., Silver D., Szepesvári C., Wiewiora E., “Fast gradient-descent methods for temporal-difference learning with linear function approximation”, Proceedings of the 26th Annual International Conference on Machine Learning, ACM, 2009.