Aria 2006 3D Soccer Simulation Team Description

Hamidreza Baghi, Mojtaba Solgi, Hamid Izadinia,
Sayyed Mohammad Hossein Sayyed Razizadeh,
Sayyed Mostafa Arefian Khalilabad

Amirkabir University of Technology
P.O. Box 15875-4413, Tehran, Iran
{baghi, solgi, razizadeh, izadinia, arefian}@ce.aut.ac.ir

Abstract. Aria3D is participating in the 3D Simulated Soccer competitions for the 3rd time. In this paper we are going to focus on two major parts of our work. First we describe the methods and algorithms we have used in low level skills and environment prediction of our agent in detail. After that an abstract description of our teamworking mechanism, *Scenario-based Teamworking*, is mentioned.

1 Introduction

Aria3d Soccer Simulation team is developed by a group of B.S. students of Amirkabir University of Technology. As a result of our hard effort, the team awarded first place in both Robocup 2004 and Robocup 2005 3D Soccer competitions[1].

We designed a layered architecture for the agent. The agent consists of different parts in different layers. There are three main layers in Aria agent: Communication layer, low-level skills, decision. in low-level skills layer we developed a number of basic skills like *Goto Point Skill* and *Kick Skill*. On top of low-level skills we developed high-level skills like *Pass* which are used in decision making. Each high-level skill also decides on choosing best possible action to do. Decision layer has a planning part that activates a scenario when applicable.

In the following sections we are going to describe three parts of Aria agent. In section 2 we describe the way our agent predicts its position and velocity. Prediction is mainly used in developing low-level skills. In section 3 we describe *Goto Point Skill* which is one of the main low-level skills. Finally in section 4 we describe the scenario-based decision making which is a part of the decision layer.

2 Prediction of Agent Position and Velocity

To implement different low level skills we need prediction of agent movement and agent velocity. Accurate prediction of instantaneous velocity of agent in future cycles is required for implementing *GotoPointSkill*. Similarly, a prediction of

agent movement is needed for eliminating the effect of noise of localization on computing the velocity of ball in current and future cycles. A velocity for agent can be calculated using *vision perception* which is an average speed for agent during 20 cycles. This average speed has 20 cycles delay considering the 10-cycle delay of sensors and 10-cycle delay of effectors which is simulated by SPADES [2]. In addition, prediction of agent position is needed to develop an efficient *Interception Skill*.

We acquired an equation for computing the Final Position ($\vec{X_f}$) and Final Velocity ($\vec{V_f}$) of agent when it moves from initial position $\vec{X_o}$ with initial velocity $\vec{V_0}$ and Drive Force $\vec{F_d}$ for t cycles. The abstract form of these formulas are shown in (1) and (2).

$$\vec{X}_f = f_1(\vec{X}_0, \vec{V}_0, \vec{F}_d, t) \tag{1}$$

$$\vec{V}_f = f_2(\vec{V}_0, \vec{F}_d, t) \tag{2}$$

In the environment of 3D Soccer Simulator the only force that opposites the agent's movement is linear drag. This force is proportional to instantaneous velocity of agent. This is shown in (3).

$$\vec{F}_k(t) = k'\vec{V}(t) \tag{3}$$

Replacing $\vec{F_k}$ with $m \times \vec{a}_k$ we derive (4) which can simply be written as (5).

$$\vec{a}_k(t) = \frac{k'}{m}\vec{V}(t) \tag{4}$$

$$\vec{a}_k(t) = k\vec{V}(t) \tag{5}$$

The drive force ($\vec{F_d}$) is also applied to agent. This force results in an acceleration which is proportional to the drive force ($\vec{a}_d = b \times \vec{F}_d$). We add $\vec{a}_d$ to sides of (5) which results in (6). We write the (6) in differential equation form which is shown in (7).

$$\vec{a}_k(t) + \vec{a}_k = \vec{V}(t) + \vec{a}_d \tag{6}$$

$$\frac{d_{\vec{v}}}{d_t}(t) = k\vec{V}(t) + \vec{a}_d \tag{7}$$

Solving (7) we obtain (8) which is an equation to compute the final velocity of agent after t time units with initial velocity of $\vec{V}_\circ$. Equation (9) is obtained by computing the integral of (8). Using this function the final position of agent after t time units can be computed.

$$\vec{V}_f = f_2(\vec{V}_\circ, \vec{F}_d, t) = \frac{e^{kt}(k\vec{V}_\circ + \vec{a}_d)}{k} \tag{8}$$

$$\vec{X}_f = f_1(\vec{X}_\circ, \vec{V}_\circ, \vec{F}_d, t) = \frac{1}{k^2}(e^k t - 1)(\vec{a}_d + k\vec{V}_\circ) - (\frac{t \times \vec{a}_d}{k}) + \vec{X}_\circ \qquad (9)$$

In (8) and (9) variable $\vec{a}_d$ is linearly dependent on $\vec{F}_d$ with coefficient b ($\vec{a}_d = b\vec{F}_d$).

We need coefficients k and b to complete the equations. To find these coefficients we put values in equations (8) and (9) for which the result is known. For example, with $t = +\infty$ in (8) and applying maximum drive force ($\vec{F}_d = 100$) and considering the fact that the k coefficient is negative[1], we have (10).

$$b = -\frac{\vec{V}_{max}}{100} \times k \qquad (10)$$

To acquire magnitude of maximum velocity of agent, maximum drive force is applied to the agent in one direction until the agent reaches a constant speed. Computing the average velocity of agent in this situation, we have a good approximation for $|V_{max}|$. Now we eliminate b coefficient from the equations. To compute k we can follow two approaches which are described below. We use the second method to obtain an initial approximation for k and then we use the first method to increase the accuracy of obtained value.

1. We start with a known state (e.g. $V_\circ = 0$ or $|V_\circ| = 0$) and apply a specific drive force to agent. Then we measure the distance the agent traveled. Using 9 we compute the coefficient k.
2. We draw a graph of agent positions in different time stamps. A sample graph for agent movement is illustrated in Fig.1. We use mathematical tools (like MATLAB) to find coefficients that fit the 9 on the graph.

3 Goto Point Skill

GotoPointSkill is an important skill and needed in most other skills and high-level behavior. The goal is to generate a series of drive actions to reach a target point with a specific velocity efficiently. The GotoPointSkill uses current velocity and position of agent and prediction functions described in previous section to generate the sequence of drive actions.

For this purpose we assume that the agent starts moving with drive force $\vec{F}_1$ which has maximum magnitude for duration of t_1 time units and then apply drive force $\vec{F}_2$ which is also maximum in magnitude for t_2 time units to reduce its velocity. The $\vec{F}_1$, $\vec{F}_2$, t_1 and t_2 must be computed so that the agent reaches the target with the specified velocity. These variables should satisfy equations (8) and (9).

$$\vec{V}_f = f_2(\vec{V}_1, \vec{F}_2, t_2) = f_2(f_2(\vec{V}_\circ, \vec{F}_1, t_1), \vec{F}_2, t_2) \qquad (11)$$

[1] When t approaches infinity the velocity approaches a constant value not infinity

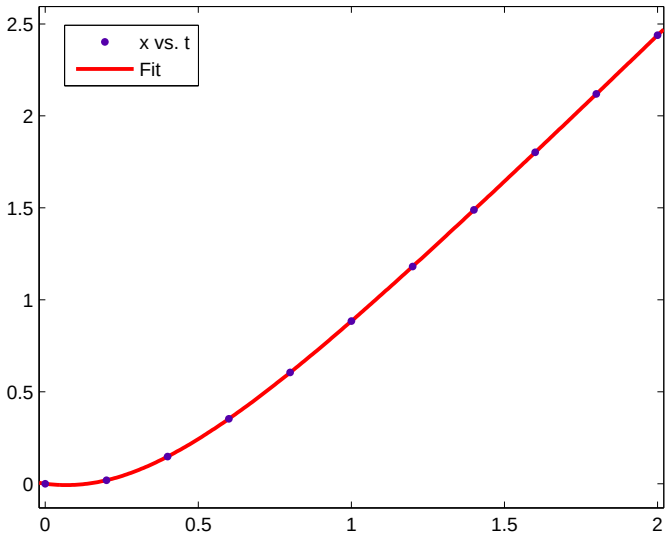

Fig. 1. Position-Time graph for agent and a fitting function for it

$$\vec{X}_f = f_1(\vec{X}_1, \vec{V}_1, \vec{F}_2, t_2) = f_1(f_1(\vec{X}_\circ, \vec{V}_\circ, \vec{F}_1, t_1), f_2(\vec{V}_\circ, \vec{F}_1, t_1), \vec{F}_2, t_2) \qquad (12)$$

$$|\vec{F}_1| = |\vec{F}_2| = 100 \qquad (13)$$

Having equations (11) , (12) and (13) we compute $\vec{F}_1$, $\vec{F}_2$, t_1 and t_2. Since first two equations don't have explicit reverse functions we solve the equation by using numeric methods.

4 Scenario-based Teamworking

In Aria2005 team we have developed a Scenario-based Teamworking (SBT) system for high level decision making. We get the main idea from [3] and did some adaptations to port these ideas to 3D environment. Using SBT agents can perform pre-designed or learned plans (scenarios) in cooperation with each other. It is a multi-state procedure and agents should be able to keep track of the scenario steps (states).

We used SBT as an alternative to our normal decision making. A general view of decision algorithm is shown in Fig.2.

An important consideration for applying SBT to 3D Soccer Agent is that there is no communication between agents in 3D environment. In such an environment, agents cannot inform each other about their decisions; As a result,

```
If there is an active scenario in progress Then:
   If STOP conditions of the active scenario is valid Then
      Stop the scenario and go to the normal agent decision making
   Else
      Each agent checks if it is involved in the activated scenario or not.
         If so, it determines in which step the scenario
         is and plays its role in that step.

Else
   For each learned or saved scenario
      If START conditions of scenario is valid Then
          Add the scenario to the potentially
          selectable scenarios list
   If there is no selectable scenario Then:
      Go to the normal agent decision making procedures
   Else
      Select the best scenario from potentially selectable scenarios
      considering the point each scenario has due to the various
      factors of the current situation.
```

Fig. 2. General view of decision algorithm

they may loose coordination when they are involved in team working tasks like scenario-based decision making. In order to implement the agent coordination in this environment, all the agents must have a same set of identical scenario patterns. Also we should develop a mechanism for scenario triggering and step evolving so that all the agents detect the start of the scenario patterns simultaneously and have same evaluation of the step that the active scenario is currently at.

According to our experience, Scenario-based Teamworking is an effective method for high level decision making and agent coordination in the communication-less environment of 3D Soccer Server [4]. We used this method for some pre-designed patterns in our 2005 team. One successfully implemented pattern is Counter-Attack Scenario which is triggered in situations when the number of opponent defenders is low and there is a high chance of scoring.

5 Conclusion

In this paper, we described a few aspects of our agent. Because the 3D Soccer Simulation is at its early stages of development, the main focus of teams is on low-level skills. In Aria we have tried to develop a set of accurate low-level skills which ease the implementation of complex decision making layer. We have also spent a lot of effort on implementing the 2D ideas of decision making in 3D. This resulted in scenario-based decision making layer. One of our future plans is to apply machine learning methods to SBT.

References

1. Hesam Montazeri, Ahmad Nickabadi, Sayyed Ali Rokni Dezfouli,Mojtaba Solgi, Hamid Reza Baghi, Omid Mola, Sajjad Moradi. Aria 2005 3D Soccer Simulation Team Description. *Robocup 2005 Championship*
2. Patrick Riley. SPADES System for Parallel Agent Discrete Event Simulation User's Guide and Refrence Manual. *2003*
3. Lev Stankevich, Alexei Kritchoun, Anton Ivanov, Sergey Serebryakov. Zenit-NewERA Team Description. *Proceeding of Robocup 2004*
4. Oliver Obst and Markus Rollmann. SPARK – A Generic Simulator for Physical Multiagent Simulations. In Gabriela Lindemann, Jörg Denzinger, Ingo J. Timm, and Rainer Unland, editors, Multiagent System Technologies – *Proceedings of the MATES 2004* , pp. 243-257, Lecture Notes in Artificial Intelligence 3187, Springer, September 2004.