China Open 2011 – Soccer 2D Simulation Team Description Paper
<Riton>

< Team Leader: Masoud Alavi >
< **Avayealvand@yahoo.com**>

Tehran - IRAN

Rajaee Artificial Intelligence Labrotaory (RAIL),

Shahid Rajaee Teacher Training University

< Members : Mohammad Falaki Tarazkouhi , Arezoo Azaran , Alieh Fazeli >

Abstract. This paper is written with the goal of summarizing some processes which are done by Riton Soccer 2D Simulation team affiliated with Shahid Rajaee Teacher Training University in order to take part in Robocup 2011. This paper's main goal consists in our algorithm in Decision Mechanism and our different skills.

1. Introduction

Riton Soccer 2D Simulation team was established 3 years ago. We achieved the 4th place in PNUCup2011 and also took part in AUTCup2010 and RoboCup IranOpen 2011. We started our project by applying fuzzy logic and now our efforts are concentrated on using new ideas for the team to progress and succeed by more advanced artificial intelligence algorithms.

2. Decision Mechanism

Decision making is the most crucial mechanism applied in Riton that controls all actions of strikers. In this part, all received and saved data in World Model are scanned and as a result, the best action is performed.

We consider four general conditions in the game:

1. Defending
2. Defense to Attacking
3. Attacking
4. Attacking to Opponent's Defending System

All actions are introduced in following ream description:

1.2 Defending Strategy

This strategy reduces the opponents speed and doesn't let them to move forward and cuts down all connections between opponents. Also we try to put pressure on opponents in order to reduce the decision making time.

Implement: First, we try to approach to ball, as near as we can. In this situation, opponents have less area to move. We use our blocking skill to abandon opponent's movement and the central teammate will try to take the ball from opponents. In order to cut down the connections between opponents, we use our marking skill too. For better results, we made some changes in formation. In this case, the performance of the agents is the best possible situation for teammates in order to defend the ball from getting into the goal position.
(In order to recognize the nearest central player or the nearest defender, a class named World Mode is coded.)

2.2 Attacking Strategy

Attacking here means to conquer the ball and move forward to opponent's side. Attacking strategy is defined in this way: the continuous circulation of the ball between the teammates until we reach the activation of the shooting function.

Implement: The first step to be done is checking the teammate to see whether he can shoot or not. If not and if he was a central player, priority is through passing to wing attackers for getting near to sides of the field. If not again, dribbling and direct passing will be activated. For wing attackers, priority is dribbling and moving forward with passing. Central attackers also will be checked for trough passing and direct pass. Other teammates will approach attackers as possible. Wing attackers then move to offside line of opponents.

3.2 Defense to Attacking strategy

In this strategy, we get the ball from opponents on our side. We try to kick the ball as far as we can with passing and giving in the ball to attacking line.

Implement: First, it must be check whether a quarantined pass out of penalty area can be done or not. If not, circulating with ball will be checked with a trustable condition. Then ball will be kicked to one of the two central sides of the field.

4.2 "Attacking to Opponent's Defending System" Strategy

This is the position in which we have surrendered the ball to the opponent's side. Here, we must press the opponent which got the ball in order to set our teammate free.

Implement: Pressing is done in one third of opponent's side by the nearest attacker and also in central one third of the field. All other teammates will mark opponents except defenders.

3. Skills Architecture

This architecture is consisted of actions which enable players to act. In general, these actions are performed by using the saved data in World Model and processing them and finally making a comprehensive instruction. The action selection is done by prior architecture.

1.3 Direct Pass

In this type of passing the probability of success is more than its failure. In this action, both sender and receiver should have a correct perception over the field. In passing method, we check several cases and list some of teammates with the least acceptable situation. We chose the best player with exercising our scoring system. Some of the conditions that we consider for scoring are:

1. Least and most distance
2. The non-existence of opponent's player within the 1 distance from the line
3. Last received data from teammate in less than 3 cycles
4. The non-existence of teammate in offside

Scoring System: We draw a line between the agent and chosen teammate and we draw spots on it. Then it will be discovered that in which spot and cycle, opponents and nearest teammate can reach the ball. We distract these two times from each other and name this difference, diffTime.
This is how we calculate the score:

*total = diffTime * dissTimeRate + distGoal * distGoalRate + badDist * badDistRate*
distGoal = PsGoal.dis(PosBall)-PsGoal.dist(placeUs)
badDist = fabs (placeUs.dist(wm.ball().pos()) - 20)
diffTimeRate = 6
distGoalRate =1
if (distGoal <0)
distGoalRate =2.5

Also for providing correct perception for the receiver, Also say will be performed.

2.3 Through pass

In this pass, the ball is not directly passed to the receiver but the probability of ball arrival is the same as the direct pass. It is sent in an area that none of opponents can reach it. First we try to draw a line from receiver to the passing point. Then we recognize which spot on the line is the best to catch the ball by using our getRichDistance function. In this function, we find out how many cycles are needed to reach the spot.

cycle = (team->playerTypePtr()->cyclesToReachDistance(team->post().dist(end)))

According to the calculated cycle, we calculate pass power

*Power = (ball_dist_to_end/cycle+1) + (0.05*cycle)*

And with the help of the written function, we check oppCantReach that whether any opponents can intercept the route with the mentioned power or not. Here is how this function works.
Based on the kicking power, it recognizes all the opponents with less than 5 poscount and keeps it in an array. Then it will calculate the next point of ball in coming cycles:

ball_pos += ball_vel;

*ball_vel *= ServerParam::ballDecay();*

And it checks whether all opponents in every point are able to intercept the ball or not. For this sake, we use below mentioned formula:

*Opp_dist_buf=opp_max_speed * std::min(4,opp->posCount())+opp->kickableArea()*

*if (player_next_pos.dist (ball_pos) < player_max_speed * step+player_dist_buf)*

return false;

If none of the opponents were able to intercept the ball, false is returned.
If false is returned, the point and power will be returned. Else, the new point will be made at distance 3 from the last point and oppCanReach levels will be performed for it until we reach a trustable point. If no points were found, false will be returned, And if a point were found, it will be added to the list and scoring system will be activated for it and the best point for passing will be chosen. Also say will be performed.

3.3 With Ball Skill

Dribbling is the skill which depends on agent's decision. In this skill, ball should be moved to a point with a specified direction. If the player is out of opponent's penalty area, the destination will be opponent's penalty area line and if the player is in the opponent's penalty area, destination will be the central spot of the gate. Then thirteen routes will be chosen and rating method will be activated. We consider following points for our rating method:

1. Body angle
2. Nearest angle to the main route
3. Best route

And at last, best route will be selected for dribbling.

*temp (i*15=90);*
path=pos(Agent)-vecPath;
pathDr=getDangerRate(pos(Agent) path);
*angleDiff=abs(i*15-90)*
*totalScore -= pathDr*7+angleDiff/3.0;*
angleDifference=Vector2D::normalizeAngle (temp-wm.self().body().degree())

getDangerRate function will be explained in detail in the following of description pape

4.3 Shoot

Shooting is the last action which leads us to the team's success. If the below mentioned cases are checked successfully, shoot will be acted.

1. In this method, we draw 10 points on opponent's gate and a line from ball to each point to calculate danger rate by getDangerRate function. The point with less danger is chosen. If danger percentage is less than 0.8, shoot will be performed. The above mentioned way, is performed only when the teammate's distance from goalie is less than 10.

2. In this method, we draw spots on the gate from right to left. The ball will be shooted to the first spot that opponents can't intercept. In order to check whether any opponent can intercept it or not, we use CanReachOppShoot function. It somehow works like CanReackOpp with some differences.

5.3 Block

This skill is trying to abandon opponent's forward movement. At the beginning, we specify a point according to the opponent's route. We draw a line from the opponent to the last point. Then we draw points with the distance of 1 on the mentioned line. Then we check whether the agent can reach the point sooner, or the opponent:

```
Current=End_poss_near_opp;
up=nearest_opp_pos.dist(End);
for (int icycle=0; icycle<=up; icycle++){
    Current=pos_near_opp+setVecPolar (cycle, current.th);
    if (current.dist (posAgen)<(cycle-1) * 1.1){
        posBlock=Current;
        break;}}
```

Teammate will move to the point, if any point was found. We have an idea in order to prevent body movement. if the current point's distance with the last point is less than 1, teammate will move backward to his last position more over if the teammate's distance with the line is less than 1, he will move to the opponent.

6.3 Man Marking

Marking is done in order to keep the opponents away from the ball and not to let them reach the ball. We choose which opponents to mark based on following levels:
First we have to recognize the *Near_Opp* from the agent. If the agent was the nearest teammate to *Near_Opp* object too, the man is the best choose to mark. If agent wasn't the nearest teammate, and if an opponent was nearer to the nearest teammate to *Near_Opp* and the agent was the second closest teammate to *Near_Opp*, we chose *Near_Opp* for marking. If none of the cases above were confirmed, we consider *Second_Near_Opp* from agent to mark. (Marking is activated if just one of the conditions above is true) after that, we draw a line from the opponent to the ball and teammate will move there and sticks to opponent and moves in a very short distance of 1 with him. However simple, but is useful!

4. DangerRate Function

This function is for calculating danger percentage of player's route. It will receive the Vector2D of all players according to its algorithm. We will explain how this algorithm works: it will draw a line between a teammate with the ball and the last point of route. Then it will be divided in eight and circles are drawn around eight mentioned points. The presence of opponents in these circles is calculated. The circle with most danger percentage is called Danger Rate. We will explain how it is calculated:

*(1) Current = Vector2D (bordar.x() * (i+1) * 0.125, bordar.y() * (i+1) * 0.125)*
*(2) Radius = 1/n * bordar.getmagnitude*
*(3) DangerRate = 1/dist_opp_to_current * Radius*

5. Future Research Programs

In this paper we introduced Riton team's major ideas. Our main goal is to focus on decision mechanism. As a result, we have planned to study about Q-learning algorithm. Besides, we are trying to improve scoring with Fuzzy Logic, blocking and a better through passing for central attack.

6. Refrences

1. Riedmiller, M., Merke, A., Meier, D., Hoffmann, A., Sinner, A., Thate, O., Kill, C., Ehrmann, R.: Karlsruhe brainstormers - a reinforcement learning way to robotic soccer. In Jennings, A., Stone, P., eds.: RoboCup-2000: Robot Soccer World Cup IV, LNCS. Springer (2000)
2. Stolzenburg, F., Obst, O., Murray, J.: Qualitative Velocity and Ball Interception. In: KI 2002: Advances in Artificial Intelligence, Twentyfifth Annual German Conference on AI, Aachen, Germany (2002) 283–