MIRG 2020 Team Description Paper

Aatish Rana Vaibhav Kohade Dibya Jyoti Mohapatra
Archana Balmik Anup Nandy
Machine Intelligence and Bio-Motion Research Laboratory
Department of Computer Science and Engineering
National Institute of Technology Rourkela.
http://nitrkl.ac.in,https://mibmnit.in
nitrssg@gmail.com

February 9, 2020

Abstract. The paper illustrates the work of MIRG for the RoboSoccer 3D simulation '20 including the research and the ideas. It describes team approach towards positioning of agents, their role assignment and decision making depending upon various tactical situations.

1 Introduction

MIRG, Machine Intelligence Robotics Group is a recently formed team under the Machine Intelligence and Bio-Motion Research Laboratory of Computer Science Department of NIT Rourkela, Odisha, India. It envisages making the soccer robots play autonomously in the 2020 RoboCup 3D Simulation Soccer. The project is mentored by Dr. Anup Nandy from the same department. Undergraduate students from circuital departments are currently constituting the team behind the project. This is MIRG's first attempt at any 3D simulation challenge worldwide.
In this paper, we describe the approach adopted by MIRG to develop an agent that will participate in game-playing. The work is structured in a given way. Section 2 describes the base architecture on which the agent is built. Section 3 illustrates the positioning of the 11 agents. Section 4 outlines the role assignment module. Section 5 and section 6 describes our work on optimizing the walk and developing new skills. Section 7 illustrates the our future plans.

2 Base code architecture

The agent is built on the base architecture provided by UTAustinVilla's code open-sourced on GitHub. Link : https://github.com/LARG/utaustinvilla3d.

The code provides a platform to develop and deploy our work on the same. The modularity of the code assists in understanding the flow quite quickly. The code consists of a basic walk engine, necessary skills, and behaviors to carry out basic tasks. The double inverted pendulum model is used by walk engine. It also has an environment class known as world model which basically stores the world objects around a particular agent and particle filter for localization. Necessary parsing code for the same is also provided. However, it does not include advanced skills and tactics, including long kicks and goalie dives. The optimized parameters for complex behaviors are missing. Also, the primary game-playing strategy, including agent formations and role assignment, are scraped out.

3 Positioning Module

The positioning of the agents is the primary objective of the game play. The tactics and the role assignments derive it's working from the positioning. MIRG uses Delaunay Triangulation to generate triangles on the field using the strategic points at various parts of the area that decides the strategy of the play. Triangu-

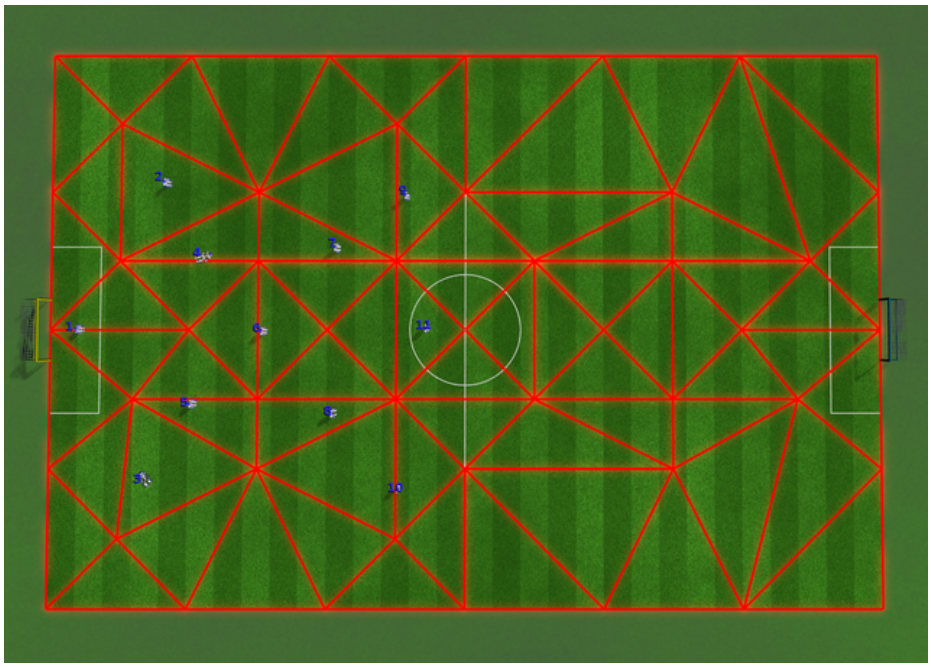

Fig. 1. Delaunay Triangulation on strategic points

lation is the partitioning of any polygon into triangles. One such application of

triangulation is for interpolation. As triangulation is quite useful when applied to polygons, it turns out that they show excellent results with point sets. One such method for triangulation is Delaunay triangulation, which is based on the property that no point should lie inside the circumcircle. A triangulation of a finite set of points is called Delaunay triangulation if, inside the circumcircle of a triangle, there is no other point among the set of points. This property is useful in the removal of skinny triangles, which may arise in scan triangulation[1]. Now we perform Delaunay Triangulation on our field. We first need to find the point set. The point set is the set of points that we have identified as strategically important, and we have mapped those strategic points with the most optimal position of every agent. Now the Delaunay triangulation is applied on these strategic points, and the output is shown in the figure below.

Once the Delaunay triangles are recognized, we tend to find that triangle that contains the ball. The identified triangle is now used to interpolate the region inside the triangle and find the position of each agent according to the ball position inside the triangle. For interpolation of the agent coordinates we are using the following formula:

$$O(PointBall) = \sum_{j=1}^{3} O(Pointj) * A_{4-j} * Wj)/(\sum_{j=1}^{3} A_{4-j} * Wj) \qquad (1)$$

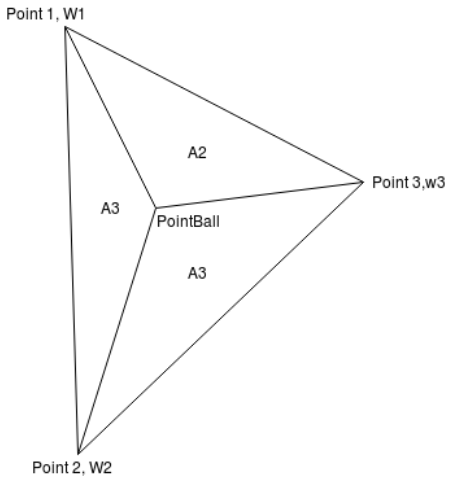

Fig. 2. Delaunay Triangulation with weighted vertices

P is the ball position. P1, P2, and P3 are the vertices of the enclosing triangle, and their weights are W1, W2, and W3, respectively. A1 is Area of triangle PointBall, Point1, Point2 A2 is Area of triangle PointBall, Point1, Point3 A3 is Area

of triangle PointBall, Point2, Point3 O(Point1), O(Point2), and O(Point3) are the agent positions according to vertex Point1, Point2, and Point3, respectively.

Now W1, W2 and W3 are the weights of the strategic points and O(PointBall) denotes the agent positions according to the ball position PointBall which is calculated as

Weights have been assigned to a every strategic point and initially they are assigned in such a way that it is proportional to the x- coordinate of the point. Once the game begins the weight of these strategic points become dynamic such that they depend on the ball's position in a particular Delaunay triangle at any instant of time.

Once the target positions for each agent have been identified, they are being sent to the role assignment algorithm for the proper execution of the movement of agents from the current position to the target position.

4 Role Assignment

When it comes to coordination between multiple agents, efficiently dividing the tasks is a necessity. Distance plays a vital role in shaping the coordination in this multi-robot system. The goal of the module is to find a role assignment function $f(x)$, between n-n elemental sets.

S1 refers to the 11 agents with start and target locations. S2 refers to the same 11 agents with the same start locations but different target locations(or roles). We have 11 roles in total. The Goalie is a bit static and does not change its position because it's the final defence mechanism. The OnBall is the primary player, and the role is transferred to different bots according to the situation on the field. Different role assignment functions map various robots to their target locations. The target locations are given by Delaunay Triangulation. Static assignments randomly map the agents to randomly selected target locations in a way that the roles remain constant once assigned. But the issue with this method is it cannot avoid collisions. The implemented role assignment module maps various agents to their roles in such a way that the longest distance from a player to a target is minimized taking into consideration every mapping. It avoids collisions happening underway while moving towards their assigned positions. It is dynamically consistent such that if f, the role assignment function, gives a mapping m of players to their target locations at time T, then at time $t > T$ it outputs the same m. In that way, we ensure that the role assignment gives minimal makespan and ensures that it remains unchanged as long as the target positions are not reached. The role assignment module uses the Hungarian algorithm to map the roles to various agents[3].

5 Decision Making

Once the ball is in possession of the agent, it has to decide whether to pass,dribble or try to score a goal. This decision is made on the basis of various fuzzy variables which change according to the position of the agent[5].

5.1 Passing

For passing, we define a source player who is in possession of the ball and a target player to whom the ball has to be passed. The fuzzy variables considered here are[6]

- Target's distance from the source player.
- The locations of opponent agents near the target player.
- The angle of rotation for the source player to the target player.

These variables are evaluated and certain conditions are defined which evaluate whether the pass is possible or not.

5.2 Goal Scoring

For scoring, we use three fuzzy variables

- The distance of the agent to the goal line.
- The opponent goalie's position.
- The opponent's position nearest to the ball.

For goal scoring, we check if the distance of the agent is in close proximity of the goal post, then we check whether the goalie or any opponent is in any position to stop the ball. If not, the agent attempts to score a goal.

5.3 Dribbling

The kick skills have been modified which allows an agent to dribble which can be used to move the ball throughout the field.

5.4 Clearing The Ball

When the agent decides that it can neither pass,score nor dribble the ball,the clear conditions are defined which cause it to kick the ball outside of the field. This is a defence strategy to prevent enemy assaults.

The above flowchart explains the basic decision making process. Each agent checks whether he has the ball or not. If he does, he tries to determine if he can score, if not, he tries to pass the ball, if he can't pass, he tries to dribble the ball to the strategic regions where he can score.

If dribbling is not possible, he tries to kick the ball either in a particular region where there are less players or out of the field as a defensive strategy.

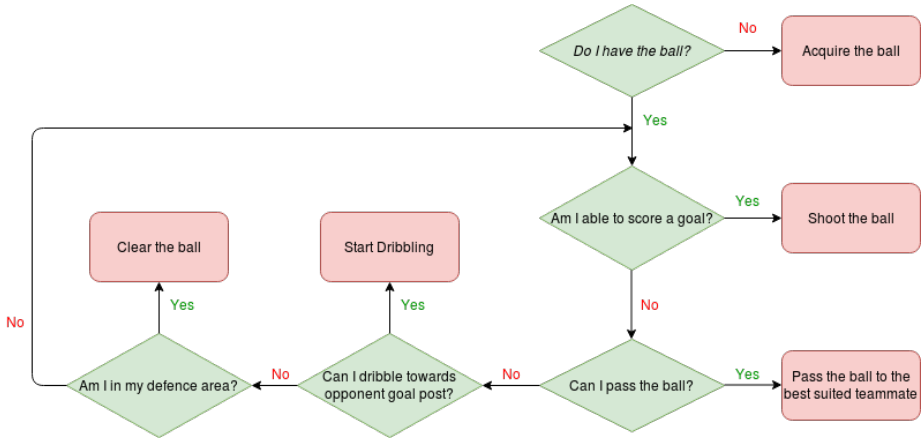

Fig. 3. Flowchart describing the decision making process

6 Skills

6.1 Walking

Currently, the base code contains the omnidirectional walk engine for agents which uses the double inverted pendulum model. But the walk speeds which can be achieved by the available walk engine are quite slow and the stability of the bot is an issue when speed of the walk is increased.The are several parameters which govern the walk and the stability of the agent can be optimised using optimisation algorithms.Currently we are using CMA-ES algorithm for improvisation of the basic and low level skills[4].

6.2 Kicking

The basic skill for kicking is based on implementing multiple frames as a periodic state machine where every frame is the static posture of a fixed joint position[2]. To separate one frame from another, a delay called waiting time is executed which ensures that the target joint angles for each frame are reached. These frames are written in a special language called Skill description language and which get parsed using a skill parser.

6.3 Goalie Actions

The goalie is the most static member in the field as it never changes its role with another agent. Goalie aligns itself to the ball also follows the ball using the distance between the ball and goalie as the parameter. We have created special actions for goalie which prevents the opposition's attack.

7 Future Work

Our positioning module is static but we would like to make it dynamic by changing the strategic points during the game play itself and hence Delaunay triangles formed will no longer be static and hence will be the agent positions w.r.t the ball. In role assignment we will be implementing prioritised role assignment with man-to-man marking. This will help in maximising the possession of the ball to ourselves. Our Decision making module uses fuzzy logic to decide whether a player will pass the ball, dribble or shoot it. In the future, we plan to divide the field into a set of regions for each playing agent. Each player will have a dominant region, where he can reach at the least possible time, a passable region which will list all possible locations where he can pass. The intersection of these regions will help us execute the best possible passes which will help us optimize our game pan. Walk engine parameters need to be optimised for more stability and higher speeds.We tend to use reinforcement learning to do the same. Similarly the kicking skill can be optimised for longer kicks and we would like to use genetic algorithm to optimise the same. We also aim to use overlapped layer reinforcement learning to learn high level skills from the low level skills.

Acknowledgements

The team also acknowledges the mentorship and guidance of our guide Dr.Anup Nandy. We also thank KgpKubs team from IIT Kharagpur for supporting us in our journey so far. We would also like to express our gratitude for Rahul Singh for helping us in formulating the strategy.

References

[1] Hidehisa Akiyama and Itsuki Noda. "Multi-agent positioning mechanism in the dynamic environment". In: *Robot Soccer World Cup*. Springer. 2007, pp. 377–384.

[2] Sven Behnke et al. *RoboCup 2016: Robot World Cup XX*. Vol. 9776. Springer, 2017.

[3] Patrick MacAlpine, Eric Price, and Peter Stone. "SCRAM: Scalable collision-avoiding role assignment with minimal-makespan for formational positioning". In: *Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2015.

[4] Patrick MacAlpine et al. "Design and optimization of an omnidirectional humanoid walk: A winning approach at the RoboCup 2011 3D simulation competition". In: *Twenty-Sixth AAAI Conference on Artificial Intelligence*. 2012.

[5] Khashayar Niki Maleki et al. "A simple method for decision making in robocup soccer simulation 3d environment". In: *arXiv preprint arXiv:1212.1570* (2012).

[6] Xu Yuan and Tan Yingzi. “Rational Passing Decision Based on Region for the Robotic Soccer”. In: *Robot Soccer World Cup*. Springer. 2007, pp. 238–245.