KgpKubs Team Description Paper

Robocup 3D Simulation League 2016

Akshay Gupta, Shrey Garg, Abhinav Agarwal, Nishant Nikhil, Vaibhav Agarwal, and Ishaan Sang

Indian Institute of Technology, Kharagpur
West Bengal, India
shrey91@gmail.com, akshaysngupta@gmail.com

Abstract. This paper describes the developments by Kgpkubs team in order to compete in Robocup 3D Simulation League 2016. This is the first time we are participating in an International Humanoid related event. This paper describes the working of the High Level Strategy which we have implemented using Delaunay Triangulation and Inter Robot Communication.

1 Introduction

Kgpkubs is a team from IIT Kharagpur, India. It aims to make autonomous soccer playing robots.For this, the team is currently focusing on 3D Simulation and Small Size League Event in Robocup. Students from all departments and years are part of this including undergraduates and post-graduates. The principal investigator for the project is Prof. Jayanta Mukhopadhyay and it is also mentored by Prof. A.K. Deb, Prof. D.K. Pratihar and Prof. Sudeshna Sarkar. We have previously participated in FIRA RoboWorld Cup in the years 2013-2015 in the Mirosot League. In 2015, we secured Bronze position in the same. We aim to learn and emerge as a competent team in the upcoming Robocup 2016.

This work is organized as follows. First, we give an overview of our base architecture and strategy in Section 2. Section 5 describes the design of the communication module. Section 3 describes about the Positioning and 4 about the Role Assignment Algorithm. Section 6 descibes about the Tactics we have developed to efficiently use the above system. Section 7 descibes our work on kicking.

2 Overview

Our base architecture is based on team libbats code available on Github `https://github.com/sgvandijk/libbats`. The code is highly modular and provides us with the flexibility to modify and develop easily.

Our strategy is based on using a mix of Delaunay Triangulation for proper positioning of robots on the field and assigning tactics for carrying out specific

tasks. For Positioning, Every robot performs calculation of positions using Delaunay Triangulation for all robots and then uses Hungarian Algorithm to find the position assignment for each robot. This result is then communicated by each robots taking turns. Upon receiving the results from the other 10 robots, the robot performs a voting to obtain their position and roles.

3 Positioning Module

In soccer, player positioning and role allocation is a very important aspect of the game. Meticulous player positioning affects the general temperament of the game and proper collaboration of various tactics is vital for a team to function efficiently.

Kgpkubs uses Voronoi-Cell Delaunay Triangulation method to generate and co-ordinate player positions with respect to the varying circumstances. Voronoi Cells are the result of a partitioning of the space into small regions based on their distances from their focal point. A point in a plane, say x, is said to lie in the Voronoi cell of a point y , if and only if the point x is more close to point y than any other point in the space.

Delaunay triangulation is the Dual graph of Voronoi cell plane. Hence, Delaunay triangles ensure that no other focal point lie inside the circum-circle of the Delaunay triangle formed. Also, due to this property it tends to avoid skinny triangles. As a result, interpolating any point inside the triangle yields to a smooth-gradient continuous equation in terms of the coordinates of the vertices of the triangle.

The algorithm used to generate player positions uses statistical data (bot and ball positions under different conditions of the game) and generates a data set of agent positions with respect to certain ball positions. In all 32 ball positions in strategic locations were identified and triangulated using the incremental algorithm to generate Delaunay triangles. Once the triangles are generated, the Gouraud Shading algorithm yields the value of bot positions at any given point in terms of the values of bot positions stored at the vertices of the triangle enclosing it.

Let there be a Delaunay triangle formed from vertices $P1$, $P2$ and $P3$. Output values of bot positions is denoted by $O(P1)$, $O(P2)$ and $O(P3)$. Now suppose we intend to find the value of bot positions (output value) at a certain point P inside the triangle $P1 : P2 : P3$. The Gouraud shading algorithm works as follows:

- Calculates I, the intersection point of the segment $P2 : P3$ and the line $P1 : P$.
- The output value at I, $O(I)$, is calculated as: $O(I) = O(P2) + (O(P3) - O(P2)) * m1/(m1 + n1)$ where,
  $m1 = lengthofsegmentI : P2$
  $n1 = lengthofsegmentI : P3$
- Finally, the output value at point P is given as: $O(P) = O(P1) + (O(I) - O(P1)) * m2/(m2 + n2)$ where,

$m2 = lengthofsegmentP : P1$
$n2 = lengthofsegmentP : I$

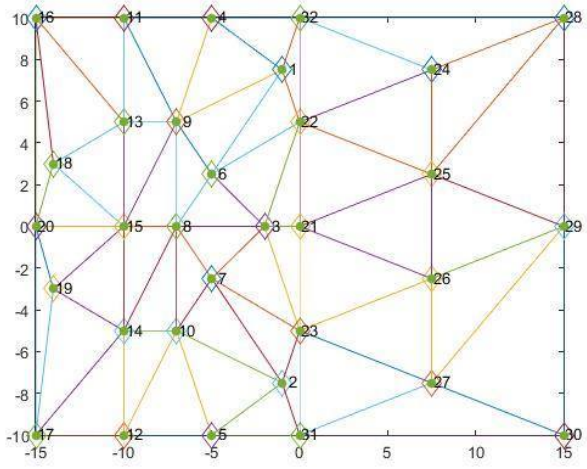


Fig. 1. Delaunay Triangles formed

4 Role Assignment Module

We use hungarian algorithm which solves the role assignment problem in polynomial time. Time complexity of this algorithm is $O(n^3)$. At every one second, as per the ball position, we get a set of target points from voronoi triangulation. We did not run the Voronoi updation after every few cycles so as to prevent associated penalties, like:

- To save time as Voronoi updation is a computationally-expensive action.
- To prevent erratic bot behaviour arising due to sudden change in ball position.

These set of target points are then matched to players on field by hungarian algorithm. The cost function used for hungarian algorithm is euclidean distance between bot's current position and target location. It is also easy to see visualize that using this type of role matching has following properties

(a) The collisions are mostly avoided.
(b) Longest distance is minimized
(c) It is dynamically consistent

5 Communication Module

In Robocup 3d Simulation, the humanoid agents have limited visibility. For proper execution of our strategy, we require every agent to know the position of every other agent in our team. This will enable the agents to have better understanding of the current game situation and hence make better decisions. In Robocup 3d Simulation, each agent can broadcast information to other agents. The server allows broadcasting of 20 bytes of data in every two cycles. This data is received by other agents in the next cycle.

At one point of time, we ensure that only one agent transmits data. Each agent takes turns to send data according to its uniform number(unum).The data sent is encoded so as to reduce the space needed to send it. There are 83 ASCII characters which we use to encode our data. We currently use 16 of the 20 bytes provided to transmit the following:

- **4 bytes:** Ball Position - It may happen that the ball is not visible to an agent. Hence ball position is sent to all the agents.
- **4 bytes:** Agent Position - This is sent because each agent only has the updated positions of the agents in in visibility region.
- **4 bytes:** Role Assignment - Assigning specific roles to agents as and when needed.
- **4 bytes:** Current time - To ensure synchronization and take care of messages that were dropped or not received.
- **2 bytes:** For certain offsets that we use to trasnmit the ball and agent positions.

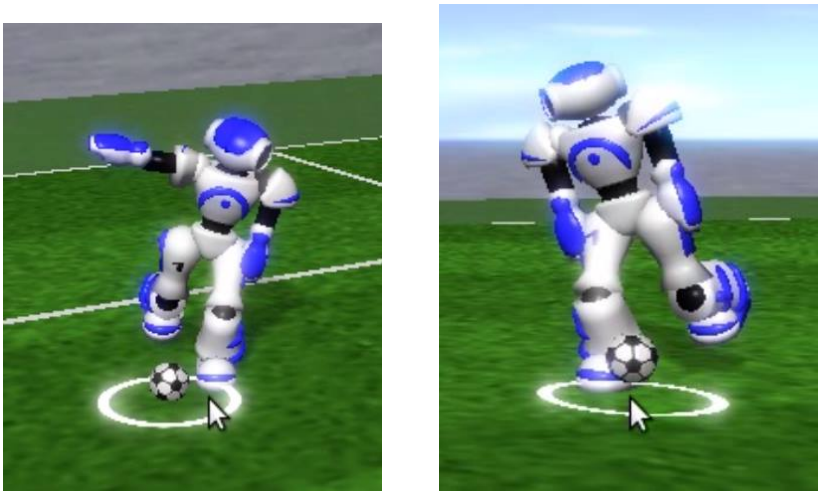
6 Tactics

Although we uses Delaunay triangulation method to generate bot positions at certain instances of the game, it is not always possible to assign a predefined position to all agents. There is a need to obtain a separate tactic for those cases which may not yield desirable results with a predefined tactic data set. These cases are the most dynamic and important of all as they critically affect minor requisites which may arise during the game.

Goalkeeper is a static member of the game. It doesn't switch it functionality with any other bot on the field. It occupy the near-to-goal area of the field and is most sensitive to minor changes in ball position and velocity. The goalkeeper dives when it knows the interpolated ball position is out of its reach in a given time window. Hence, the goalkeeper decides when to dive as an incorrectly timed or useless dive may actually do more bad than good. So, by means of data obtained from ball velocity and position, its expected destination point and arrival time is calculated. The maximum velocity for a bot is previously known and this is used to calculate the time required for the goal-keeper to reach its destination. If this time is more than the arrival time of the ball, the goal-keeper moves to such a distance so that it is within a distance from the destination point equal to it's height and then takes a dive.

The on-ball bot is arguably the most dynamic bot on the field. The bot closest to ball is usually assigned this position. It is responsible to take the ball forward and to block a bot which is coming with the ball.

7 Kicking



(a) Right Kick (b) Left Kick

Fig. 2. Kicking

Currently, we have implemented static kicking. We used the base of libbats kicking sequence(motion sequence controller) and were able to kick the ball after tweaking with parameters and by applying a better lean functionality. Our current design, being a static one, takes care as to not kick the ball out of the field and to implement kick only if the oppponent bots are apart by a certain distance to prevent being dispossesed by them after kicking. We intend to improve on our current design, which exhibits small range of kick presently, by using spline interpolation for better dynamic kicking.

8 Future Work

We aim to implement a walking module based on ZMP reference. We are also implementing spline interpolation for dynamic kicking and aim extend the communication module to provide functionality for announcement-based events like passing and receiving.

Acknowledgements

The team also acknowledges the mentorship and guidance of our professors Prof. Jayanta Mukhopadhyay, Prof. Sudeshna Sarkar, Prof. Alok Kanti Deb and Prof. Dilip Kumar Pratihar. This research is supported by Sponsored Research and Industrial Consultancy(SRIC), IIT Kharagpur. We also thank our former team members who made all of this possible.

References

1. Akiyama, Hidehisa, and Itsuki Noda. "Multi-agent positioning mechanism in the dynamic environment." RoboCup 2007: Robot soccer world cup XI. Springer Berlin Heidelberg, 2007. 377-384.
2. MacAlpine, Patrick, Francisco Barrera, and Peter Stone. "Positioning to win: A dynamic role assignment and formation positioning system." RoboCup 2012: Robot Soccer World Cup XVI. Springer Berlin Heidelberg, 2013. 190-201.
3. Barrett, Samuel, et al. "Austin Villa 2011: Sharing is caring: Better awareness through information sharing." The University of Texas at Austin, Department of Computer Sciences, AI Laboratory, Tech. Rep. UT-AI-TR-12-01 (2012).
4. Erbatur, Kemalettin, and Okan Kurt. "Natural ZMP trajectories for biped robot reference generation." Industrial Electronics, IEEE Transactions on 56.3 (2009): 835-845.
5. Strom, Johannes, George Slavov, and Eric Chown. "Omnidirectional walking using zmp and preview control for the nao humanoid robot." RoboCup 2009: robot soccer world cup XIII. Springer Berlin Heidelberg, 2009. 378-389.
6. Jun, Youngbum, Robert Ellenburg, and Paul Oh. "From concept to realization: designing miniature humanoids for running." J. on Systemics, Cybernetics and Informatics 8.1 (2010): 8-13.
7. Liu, Juan, et al. "Apollo3D Team Discription Paper."