# KgpKubs Team Description Paper
## Robocup 3D Simulation League 2016

Abhinav Agarwal, Shrey Garg, Nishant Nikhil, Vaibhav Agarwal, Buridi Sree Aditya, and Ranjith Kumar

Indian Institute of Technology, Kharagpur
West Bengal, India
`i.nishantnikhil@gmail.com`, `abhinavagarwal1996@gmail.com`

**Abstract.** This paper reports the recent developments by the Kgpkubs team. It describes the work on passing, formation strategies, heuristic role assignment and use of evolutionary algorithms for optimizing low level skills like walking.

## 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. In 2016, we participated in RoboCup (3D Simulation League).

This work is organized as follows. First, we give an overview of our base architecture and strategy in Section 2. Section 3 describes about attacker and goalkeeper tactics. Section 4 describes about the Positioning and 5 about the Role Assignment Algorithm. Section 6 describes the approach we use for deciding and doing pass. Section 7 describes about the various machine learning methods we have used to optimize low level skills.

## 2 Overview

Our base architecture is based on team UT-AustinVilla code available on Github `https://github.com/LARG/utaustinvilla3d/`. The code is divided into appropriate modules 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. Some important roles like attacker, defender, goalie are assigned based on some heuristic methods overriding the Hungarian algorithm.

## 3 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.

The Attacker bot is arguably the most dynamic bot on the field.This role is selected based upon certain heuristics like fallen status,distance from opponent etc. The attacker can quickly dodge, dribble, kick(fast and slow) and drive ball to goal. 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. We have multiple types of dives that goalkeeper can use, depending upon game play.

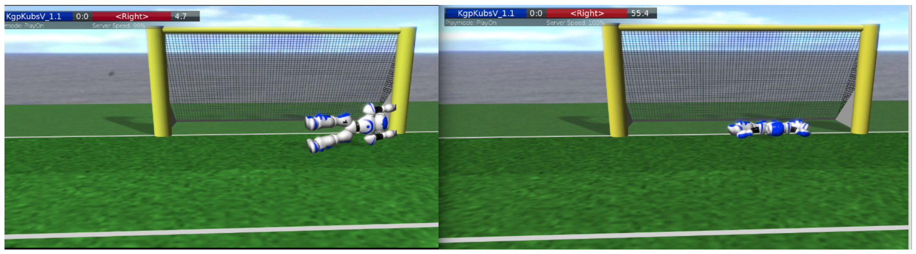

**Fig. 1.** Side Dive - Central Stretch

## 4 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 65 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,
  $m2 = lengthofsegmentP : P1$
  $n2 = lengthofsegmentP : I$

These formation points computed are fixed for every scenario. In order to overcome the troubles caused by real game play we have heuristics to override the formation points.

## 5 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.

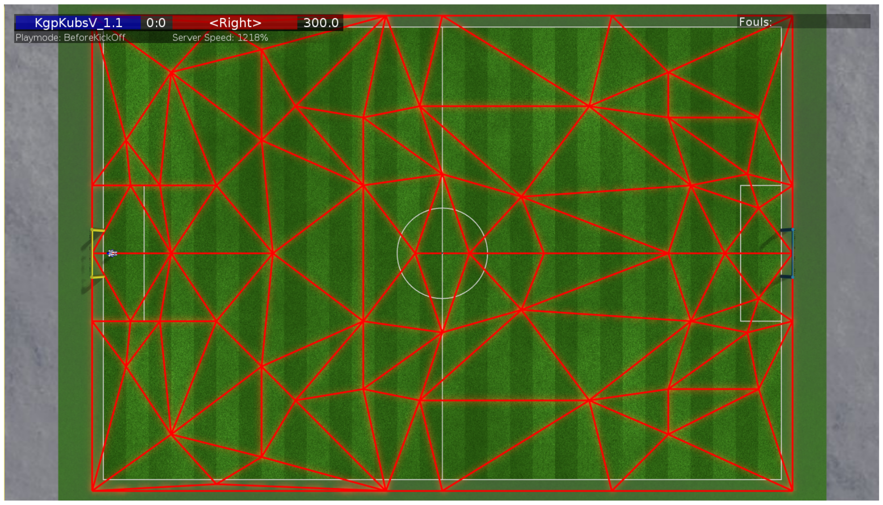

**Fig. 2.** Delaunay Triangles formed

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

We have a superficial layer over the hungarian role mapping. We override the hungarian mapping by using heuristic functions for specific roles for better assignments.

## 6 Passing

Passing is one of the most essential element for a multi-agent football playing system. We implemented a fuzzy logic based passing, which for each team member within a threshold radius takes into account

- target player's distance from source player
- distances of opponent players from source player and target player

- proximity to goal of source player and target player
- angle to rotate for source player to face target player

and based on the above parameters computes a score, if the score is above some threshold we deem ourselves in a favourable position to pass, otherwise we continue as is. Few points worth a notice:

- while calculating distance you can't give the same weightage to an opponent player in front of our player and behind of our player
- you need to consider only those places to pass where you can kick accurately (at time of writing this kgpkubs had not developed a kick which can cover whole of the field)

## 7 CMA-ES

For optimizing low level skills, like walking, it may seems that reinforcement learning is more suitable but on contrary CMA-ES performs at par with the RL algorithms. We treat it as a black-box optimization algorithm, and have not interfered with the algorithm itself. For training we have used different cost function for optimizing different skills. For example, commanding the bot to go straight and using the difference in x-coordinate optimizes walking straight as well as improves its speed. Similarly you can optimize other walk types (like lateral walking) and other skills like kicking. Here we provide a brief about the CMA-Evolutionary Strategy:

CMA-ES is a policy search algorithm that successively evaluates sets of candidates. Each candidate is evaluated with respect to a fitness measure. The next set of candidates is generated by sampling multivariate normal distribution that is biased toward directions of previously successful search steps. Recombination amounts to selecting a new mean value for the distribution. Mutation amounts to adding a random vector, a perturbation with zero mean. Adaptation of the covariance matrix amounts to learning a second order model of the underlying objective function. It is a parallel search algorithm so it can be run on a large server to make the optimization feasible. Some parameters were carefully chosen for optimization keeping others constant to reduce the search space.

## 8 Future Work

Currently we have used CMA-Evolutionary Strategy for improving low level skills, we aim to use deep reinforcement learning for the same and espescially with exciting new research in asynchronous actor-critic algorithms this is the perfect time to implement it. Furthermore we are aiming to use neural networks for automating the positioning module.

## 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."