1 Problem

To design a simulation to illustrate multi-agent collaboration for setting up moves in soccer. The problem statement focuses on the scenario of Robocup. We are required to select a sub task from the Robocup competition and simulate the same in multi-Agent environments inferring concepts and ideas from Reinforcement learning.

2 Problem Specification

RoboCup is an annual international robotics competition proposed and founded in 1996 by a group of university professors. The aim of the competition is to promote robotics and AI research by offering a publicly appealing – but formidable – challenge. [Wikipedia]

Sub-Task Chosen : Corner Kick
A corner kick is the method of restarting play in a game of association football when the ball goes out of play over the goal line, without a goal being scored and having last been touched by a member of the defending team. The kick is taken from the corner of the field of play nearest to where it went out.

Our Solution will include the following: Considered task assumes a total of 6 Agents, 3 Agents from each team (Attacking and Defending) named as Team Red and Team Blue, and a ball to operate with. We have taken 2 goalkeepers, out of which 1 goalkeeper of the attacking team thus, is not considered as the chosen task doesn't require any involvement of the attacking team goalkeeper. The other goalkeeper will be inside the penalty area only as it is not allowed for the goalkeeper to leave the goal when a corner kick is taking place. One player will be taking the corner from the attacking team and one player from the defending team will be present in the penalty area.Only 3 agents (initially) will be present in the half goal.
We have considered a half goal instead of a complete view as our task will be focused on one part of the playground.
Each team member is wearing the cloth belonging to their team i.e. blue and red. The ball is of black color.
The task considered will consist of 2 worlds, one is the agent world and the other is the agent playground. Agent world restricts the possible movements of the agents wherein, the agent playground restricts the region for football.

3 Design of Agent

An agent is made up of (20X20) pixels where one block of 10*10 pixels is missing which represents the foot area for the agent from where it can grab the ball and shoot it,the agent takes a set of actions from the state space.

We have taken 2 goalkeeper out of them, 1 goalkeeper of the attacking team is not considered as the task doesn't include any involvement of the attacking team goalkeeper. The other goalkeeper will be inside the penalty area only as it is not allowed for the goalkeeper to leave the penalty area when a corner kick is taking place. One of the players initially from the attacking team will be taking the corner thus, only 3 agents (initially) will be present in the half goal. Agents are made up of 20*20 pixels where one block is missing which represents the foot area for the agent from where it can grab the ball and shoot it.

Consider below images for the agent, goal, ball, playground design. Also, set of actions and possible state tables is also included in this section below. Refer Figure 4, 5 and 6.

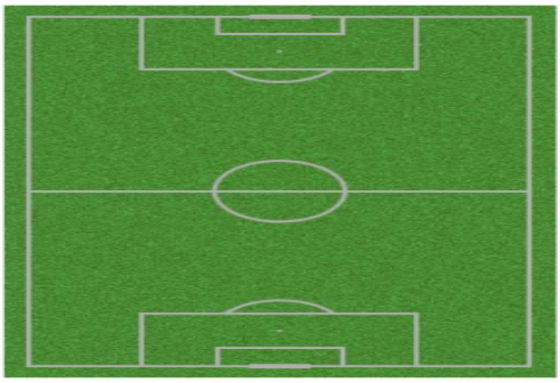

Figure 1: *Environment*

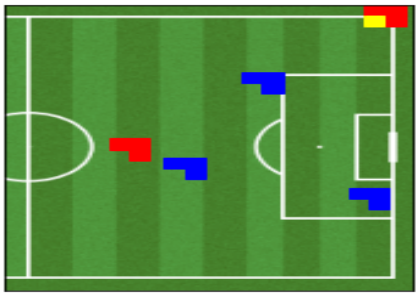

Figure 2: *Considered Case*

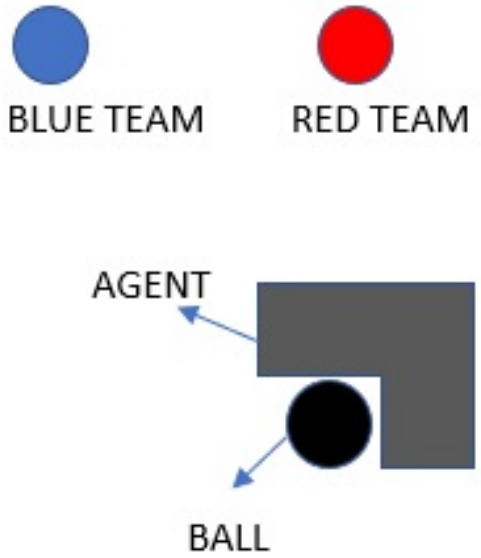

Figure 3: *Explanation of considered case*

Figure 4

3.1 State space with respect to ball

HOLDING
NOT HOLDING
INSIDE THE GOAL
NOT INSIDE THE GOAL
OUTSIDE THE BOUNDARY
NO. OF PLAYERS INSIDE THE PENALTY AREA
GOALKEEPER INSIDE/OUTSIDE THE PANELTY AREA
BALL STUCK BETWEEN THE AGENT
AGENT INSIDE THE GOAL
BALL MOVES INFINTESIMALLY TIMES

Figure 5: *State space*

3.2 Set of actions taken by an agent

FRONT
BACK
RIGHT
LEFT
ROTATE CLOCKWISE
KICK
STOP
ROTATE ANTI-CLOCKWISE

Figure 6: *Actions taken by an agent*

4 Design of Collaboration and Behavioral Model

Randomly initialise the locations of agents but 2 agents from each team Red and 2 agents from team Blue are to be located and rest 1-1 on goal. Goalkeeper initially is set at random location and later based on attacker movement it will update.

4.1 the Ball Location

Initially at the corner of the field (any side) it depends on which side the agent is placed. No. of agents collaborating at a time n<=3 in which n_{blue} <=2 ,n_{red} <=1.

4.2 The Agents

The agents and the ball are represented by a single coordinate(x,y) in the agent world (environment) which when given to a function will build its solid structure as mentioned in the previous section. With the help of these points the agents will interact and collaborate with each other.

4.3 Initial Simulation

For the initial case, the agent(attacking team), taking the corner will look if it can directly make a goal provided, there is no one obstructing the path from the defending team, and if the goal happens, the game will restart and the paths are updated for both team Red and team Blue. If there is any obstruction then the agent will try to pass the ball to its team player only if that direction also doesn't have any obstruction by the defending team. If all the scenarios of obstacles are detected, then the agent will itself take a step ahead (20 pixels in appropriate direction) from the set of actions. From the appropriate direction we mean that, it is based on the reward which it is getting. For the reward thing, we have considered Bellman's Equation and updated a little to design it for our problem statement. Its details are mentioned in the implementation part.

5 Design of Dynamic Interactions

As mentioned earlier, agents, balls, goal, penalty area, agent world, agents play area, everything is guided by coordinates or range of coordinates. To detect the obstacle, we have collected all the points which lies in the straight which the ball will follow after a

kick from the agent and then we are checking if any agents coordinates are matching those points, if match is found then agent will decide whether to pass or to avoid based on the team of the agent. Similar steps will be taken by the other agent as well when the bold is in its control.

The attacking agents will try to maximise the objective function which is based on bellman's equation whereas the defending team will try to maximise the objective function. Objective function is given as follows:

$$TeamRed(attacker) : max\left(\left(1 - \frac{R(s,a)}{255.0}\right) * \frac{d^{new}}{d^{max}} + \left(\frac{R(s',a')}{255.0}\right) * \frac{d^{present}}{d^{max}}\right)$$

$$TeamBlue(defender) : min\left(\left(1 - \frac{R(s,a)}{255.0}\right) * \frac{d^{new}}{d^{max}} + \left(\frac{R(s',a')}{255.0}\right) * \frac{d^{present}}{d^{max}}\right)$$

Here,
d = Euclidean Distance
R(s,a) = Reward at next state 's' after action 'a'
R(s',a') = Reward obtained at present state after performing action 'a" on state 's"

5.1 Red Team

Initially the agent 1 is at fixed coordinate (near corner) and agent 2 is at the coordinate which is randomly generated. Agent 1 kicks the ball either to the goal or pass the ball to the subsequent team member and updates its coordinate. Agent 2 either receives the ball or the ball is captured by another team's agent , if the agent2 receives the ball it tries to directly kick into the goal or pass the ball to another agent and change its location based on objective function.Agent 3 remains in the same position as its goalie.

5.2 Blue Team

Initially the agents of blue team are located at the randomly generated coordinates ,and one the agent is definitely in the penalty area. When the agent of team red kicks the ball, the agents of team blue tries to defend the goal and the agent of team blue whose distance from the ball is less tries to capture the ball.

5.3 Ball

Ball is initialised at the same coordinate as the agent of team red taking the corner and gets updated in every episode since its continuously changing its position.

6 Planning Scheme for achievement of Goal

The complete game will go like a min max algorithm. The attacker will try to minimise the updated Bellman Equation and the defending team will try to maximise the updated Bellman Equation(objective function). The returning value of the Bellman Equation signifies whether the goal is near or far. The detailed discussion is done in the implementation part. The updated Bellman Equation coincides with the heuristic based on path cost. A heuristic matrix is designed which will help the agent to estimate the reward function. Below mentioned is the heuristic matrix representation in an image format. Whiter part denotes more reward and the darkest denotes zero reward.

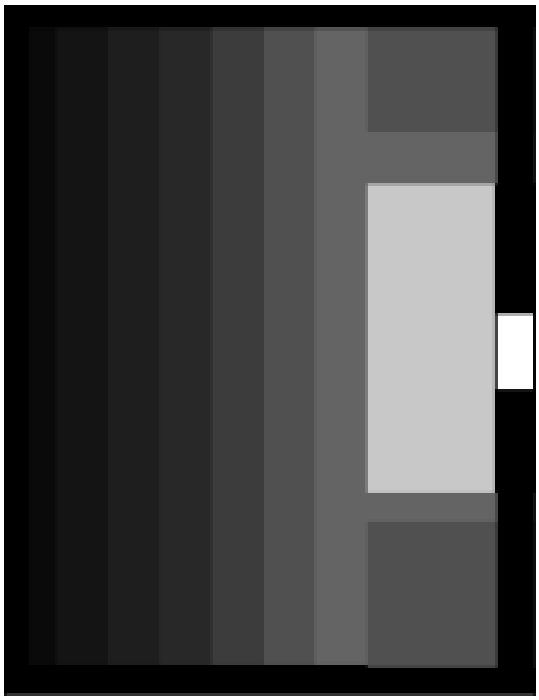

Figure 7: *Heuristic Matrix (Rewards Matrix) : White part denotes the goal with maximum reward i.e. 255 and boundaries denotes least reward i.e. 0*

6.1 Team Red (Attacker Side)

Below is the action plan for Team Red in Decision Tree form.

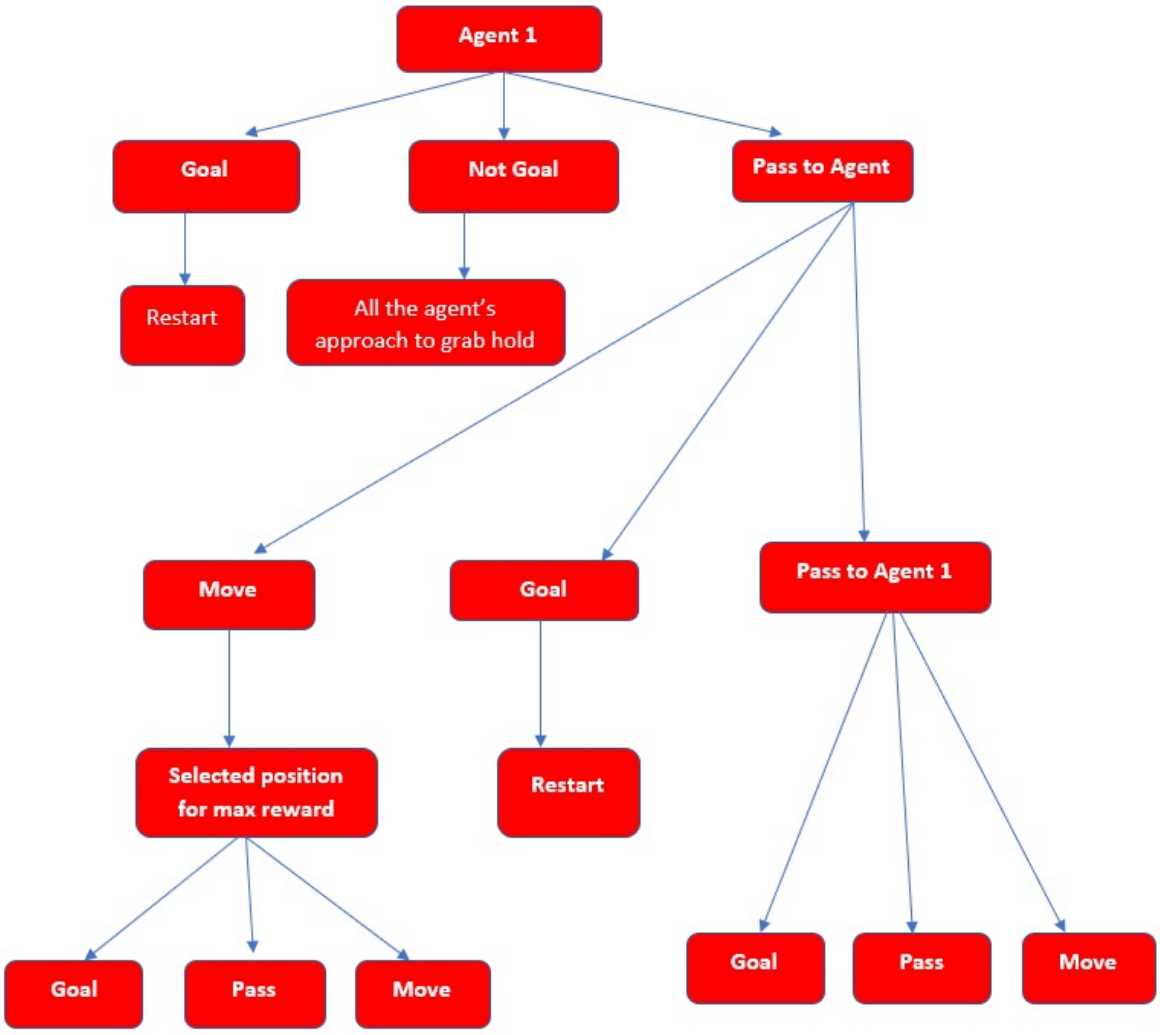

Figure 8: *Action plan for Team Red*

6.2 Team Blue (Defender)

Below is the action plan for Team Blue in Decision Tree form.

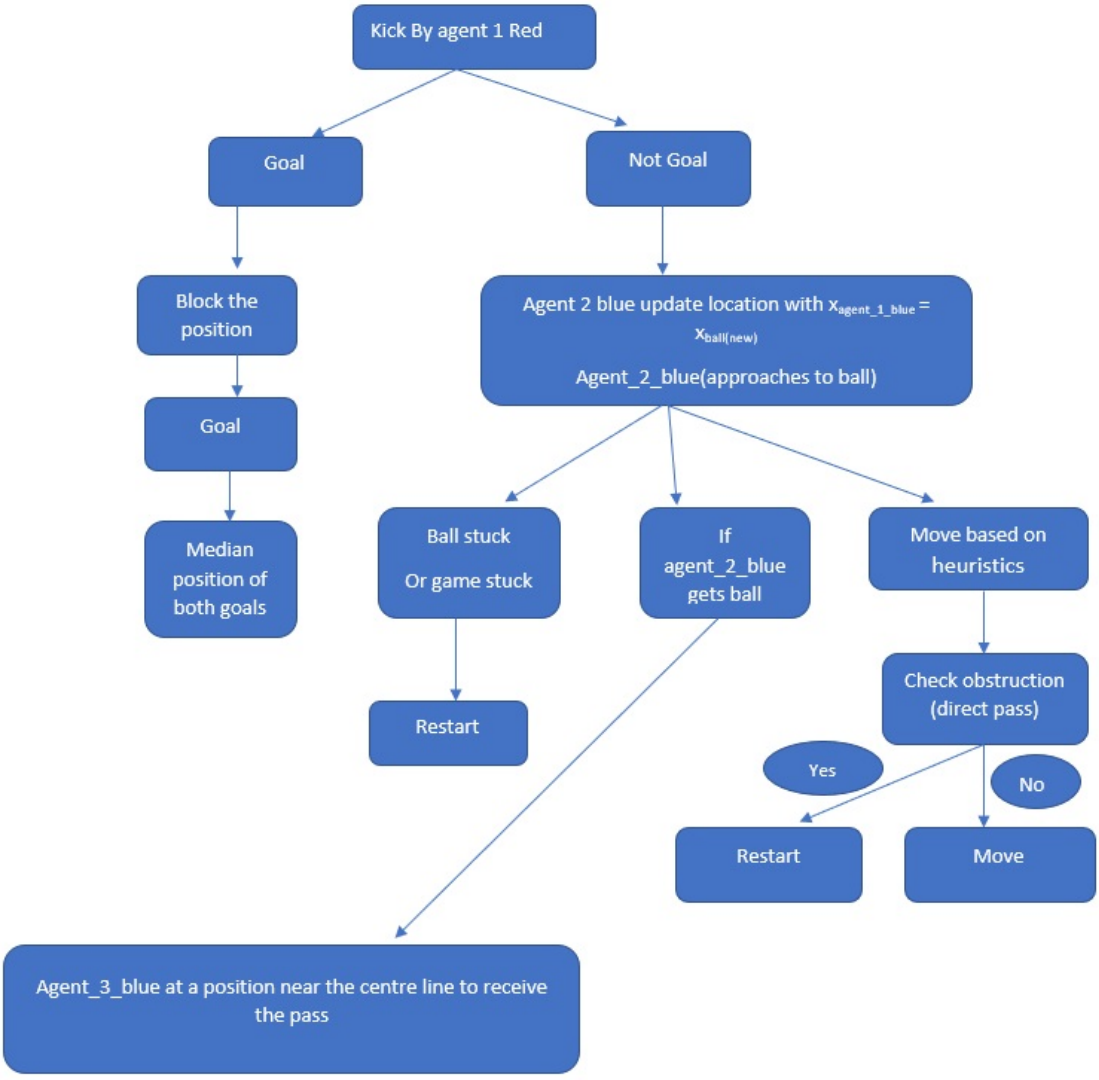

Figure 9: *Action Plan for Team Blue*

7 Implementation

Agent Environment: We have considered half ground as mentioned previously. We have taken Corner Kick scenario from the Robocup competition.
The ground can be divided into 2 parts:
1. Agent World(Environment)
2. Agent Playground

Agent world includes the complete surface area where the agent can possibly be and the agent playground represents the play area for agent and the possible locations for ball. Below image is the real time simulation representation.

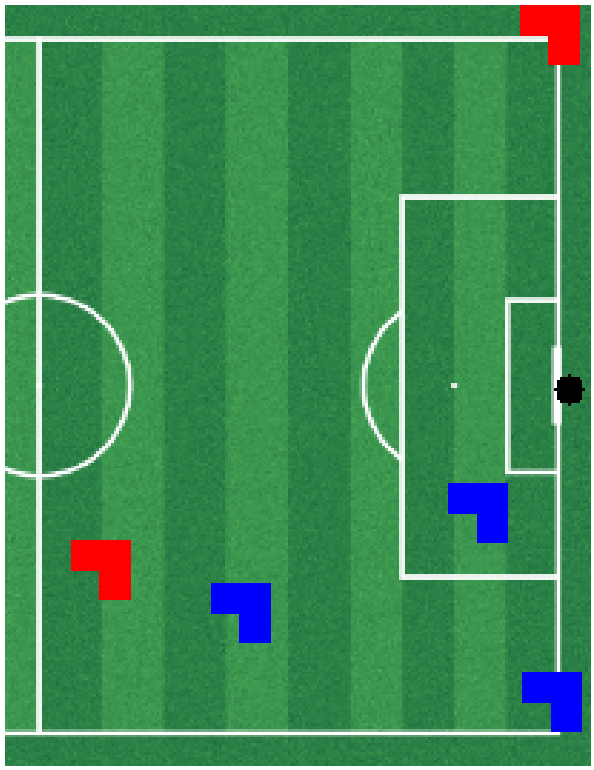

Figure 10: *Action Plan for Team Blue*

Heuristics Matrix: This is nothing but a rewards points allocation according to the area closer to the goal. Brighter the part, the more is the reward.
Objective Function: We have considered an objective function inspired from Bellman's Equation. Based on the reward and the euclidean distance, the moves are decided by the agents. Team Red(attacking team) will be minimising the objective function and Team Blue(defending team) will try to maximise the objective function.

$$TeamRed(attacker) : max\left(\left(1 - \frac{R(s,a)}{255.0}\right) * \frac{d^{new}}{d^{max}} + \left(\frac{R(s',a')}{255.0}\right) * \frac{d^{present}}{d^{max}}\right)$$

$$TeamBlue(defender) : min\left(\left(1 - \frac{R(s,a)}{255.0}\right) * \frac{d^{new}}{d^{max}} + \left(\frac{R(s',a')}{255.0}\right) * \frac{d^{present}}{d^{max}}\right)$$

Software Dependencies:
1. Language used : python
2. Environment : Self Created
3. Python Version : 3.7.6
4. Libraries used and their version:
mplsoccer : 0.0. 20
cv2 : 3.2.0.8
numpy : 1.8.2
matplotlib : 3.3.0

Functions created and their working :
1. move : returns the updated position of the ball and the agent which is holding the ball
2. final_frame :returns the drawn solid objects
3. get_points : returns all the points of a line which is a path followed by the ball or to be followed by the ball
4. agent_move : returns the updated movement of the agents
5. initialisation : To initialise the first case
6. game : performs all logical and other calculative operations.