EE4306
DISTRIBUTED AUTONOMOUS ROBOTIC SYSTEMS

Robot Soccer Project: Arsenal FC

Done by:
Ong Zi Liang (A0111756J)
Pearl Lim Pei Yi (A0112604X)
Dou You Zhe (A0100229X)
Max Schlichting (A0145422N)

Table of Contents

Introduction

In this project, a given number of soccer robots are programmed to complete different tasks at three distinctive stages. At stage 1, a blue soccer robot is assigned to be a goalkeeper and is required to prevent a ball from entering a goalpost during a penalty. This is accomplished by locating the position of the moving ball. During stage 2, three robots are instructed to determine the location of the ball, dribble across obstacles, pass the ball to one another and finally score a goal. To be able to effectively achieve these tasks, the position and orientations of the ball and three robots are essential to note so that a suitable path planning can be decided. Potential walls and fields are also created to avoid collision with any obstacles during dribbling, and to ensure that the robots and ball do not drift away from the intended destination point. The passing of ball requires the speed of the moving robot to be adjusted correctly so that the ball moves according to the desired speed towards another robot without losing sight of it. For stage 3, a main function "RobotMoveToThere" is created. This function moves robot 1, robot 2 and robot 3 to a particular location based on the GPS mapping of Vrep. Therefore, an example would be setting the position of the defender such that it is blocking the opponent's striker to prevent the striker from shooting.

Algorithm and Strategy

Stage 1

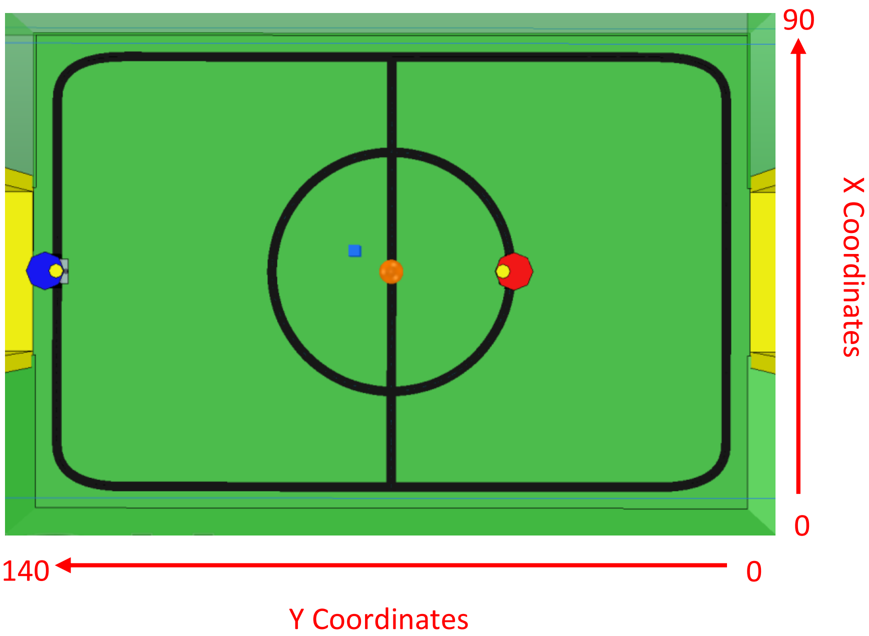

Figure 1: Range of Y and X Coordinates

At the start of the simulation, the robot goalkeeper constantly checks the X coordinates of the ball while remaining at its initial starting position. The X coordinates obtained from Vrep range from -0.45 to 0.45 where -0.45 indicates the ultimate right side of the field while 0.45 indicates the ultimate left side of the field according to the robot's head direction. In order to acquire readings that are much easier for use in programming, an algorithm is used to convert the readings to a range of 0 to 90 so as to eliminate negative numbers and decimals. Only X coordinates are used at this stage while the Y coordinates remain untouched. Nevertheless, the range of -0.7 to 0.7 is converted to a range of 0 to 140 for the Y-coordinates. For the X-coordinates, the right side of the field has been decided to range from 0 to 45 and the left side of the field ranges from 45 to 90. For example, if the reading is -0.15 which is a negative number, -0.15 is multiplied by -1 to become positive. It is then used to be deducted from 0.45, which is supposed to be the centre number within the range, since it is at the right hand side of the field. Subsequently, the result is multiplied by 100 to eliminate decimals. If the reading is 0.15 which is a positive number, it will be added by 0.45 and then multiplied by 100.

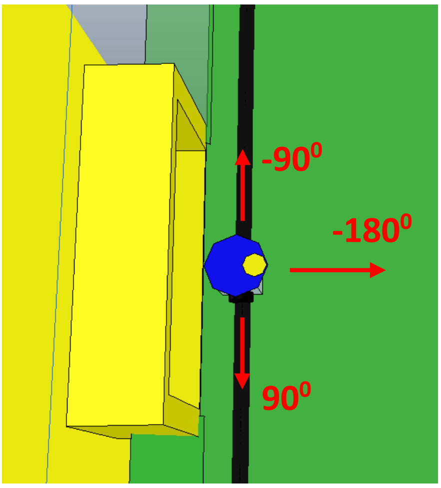

Figure 2: Orientation Angle of Robot

As seen on figure 2, the robot has an orientation angle of -180^0 at the beginning of the simulation. Once the ball moves, the robot tracks whether the ball is heading towards the left or right hand side of the goalpost. If it is heading towards the right, the robot turns in a clockwise direction until it reaches an angle of 90^0. If the ball is heading towards the left, the robot turns in an anti-clockwise direction until it reaches an angle of -90^0. If the ball remains stationary at an X-coordinate of either 44 or 45, the robot stays at its initial position. When the direction has been decided, the robot will move according to where the X-coordinate of the ball is at. Both the left and right hand side of the goalpost are divided into three sections each which means that the goalpost is divided into six sections in total. When the ball is at an X-coordinate between 45 and 53 on the left hand side of the goalpost, the programme checks if the robot is at an X-coordinate lesser or more than the X-coordinate of the ball. If it is lesser, the robot moves forward at a velocity of 400. If not, the robot reverses at a velocity of -400. Similarly, when the ball is at an X-coordinate between 52 and 57, the velocity would be either 500 or -500. As for the X-coordinate that is above 56, the velocity would be either 600 or -600. The same method is applied to the right hand side of the goalpost.

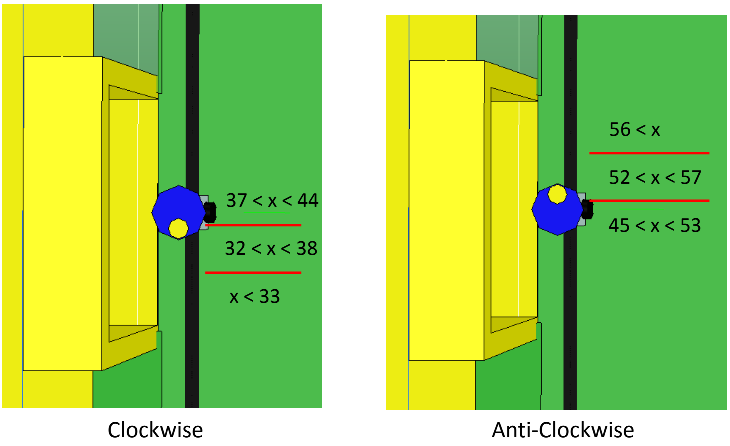

Figure 3: Left and Right Hand side of Goalpost divided into 3 sections each

Stage 2

For this stage, the dribbling and passing of ball are carried out. The programme starts by dribbling the ball with the help of one robot, and then followed by the passing of ball among the robot players in the field. The dribbling is achieved through the use of image processing which aids in creating potential fields and path planning for the robots and the ball. The rectangular image obtained has a width and length of 128 and 256 cells respectively. Through image processing, each individual cell are assigned with a value. There are a total of 2 potential fields created for this stage and the first field having four layers while the second field having 2 layers.

The first potential field aids in the path planning for the ball to reach the end point. The second potential field ensures that the robot is able to reach an ideal spot surrounding the ball which allows it to kick the ball towards the intended destination point. At the start of stage 2 simulation, the first and second layers of the first potential field are established.

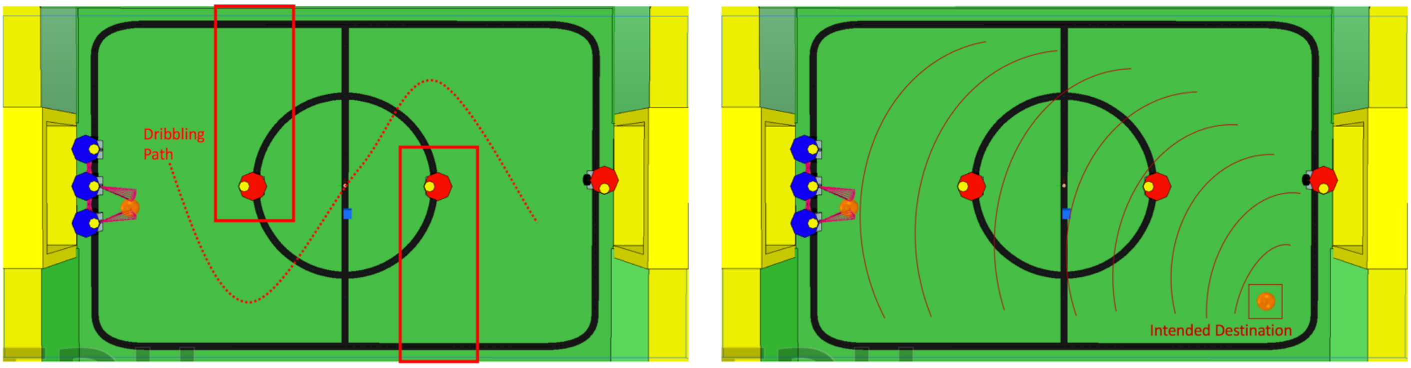

Figure 4: First and Second Layers of First Potential Field

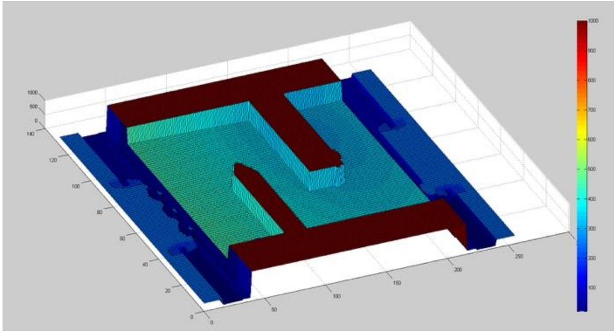

Figure 5: 3D Plot of First and Second Layers of First Potential Field

The first layer of the first potential field, as seen in the left image of Figure 4 is used to determine the path of the ball for dribbling while ensuring that it does not collide with the stationary obstacles. The boundaries surrounding the obstacles are assigned a value of 1000 in each cell. As for the second layer shown in the right image of Figure 4, it indicates the intended destination that the ball should head to at the end of the dribbling path. The destination point has a value of 300 in its cell. All the other values of the cells in the field would be calculated using the following formula that involves Pythagoras theorem:

$$300 + \sqrt{x^2 + y^2}$$

The further a cell is away from the intended destination point, the larger the value of that cell. Hence, the ball plan its path by selecting a neighbouring cell that has the smallest among the rest of the cells surrounding it until it reaches its destination which has a value of 300 in its cell. Once the first and second layers of the first potential field are formed, the position of the ball in terms of cell position is identified. If the ball is found to be at the top half of the field on the left, the arrangements for the first and second layers of the first potential field will remain as it is. However, if the ball is found to be at the bottom half of the field on the left, the first and second layers of this potential field will be altered to become the mirror image of the original arrangements. This means that the dribbling path begins at the bottom half of the field on the left, the boundaries surrounding the obstacles would switch to its opposite position and the intended destination point would be at the top half of the field on the right. The values in each cell would change accordingly as well.

In order for the ball to move towards the intended destination point, the first layer of the first potential field is required. As stated earlier, when $\sqrt{x^2 + y^2}$ gets smaller, the ball gets nearer to the end point. The programme will first check the values of the cells that are 10 cells away from the ball and are surrounding the ball. The cell that is discovered to have the lowest value compared to the rest of the cells surrounding the ball will be the next position that the ball should be heading to. Once direction of the ball to be dribbled to is decided, the second potential field takes effect. The first layer of the second potential field creates a radius around the ball with a high value of 150. This is to prevent the robot from colliding with the ball while getting to the ideal position to kick the ball. The ideal position has to be diagonally opposite the ball's next position. In the example shown in figure 6, the robot has to move towards the ideal position at the cell on the top left hand corner of the ball so as to kick the ball towards the next ball's position.

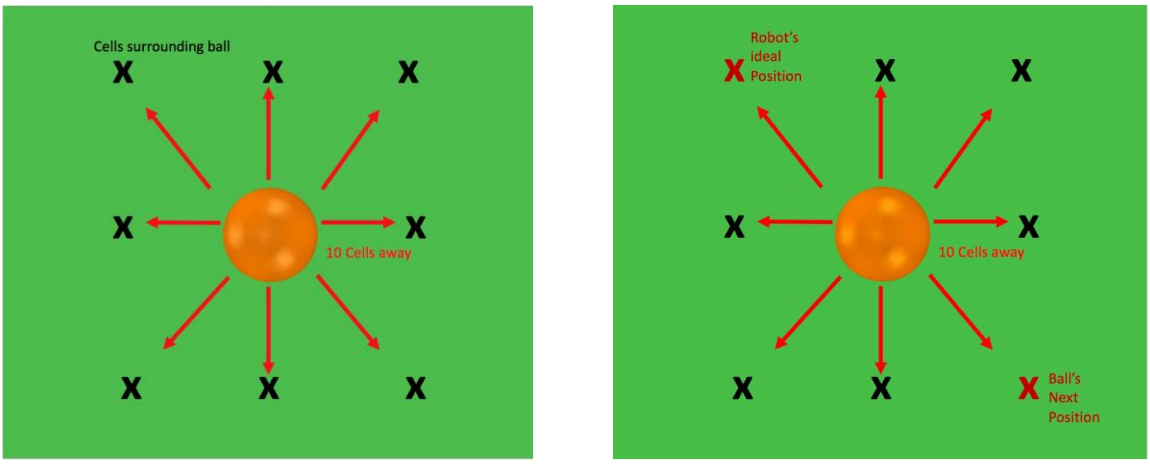

Figure 6: Identifying the cell that will be the ball's next position and an Example of where a robot's ideal position should be when the ball's next position has been decided

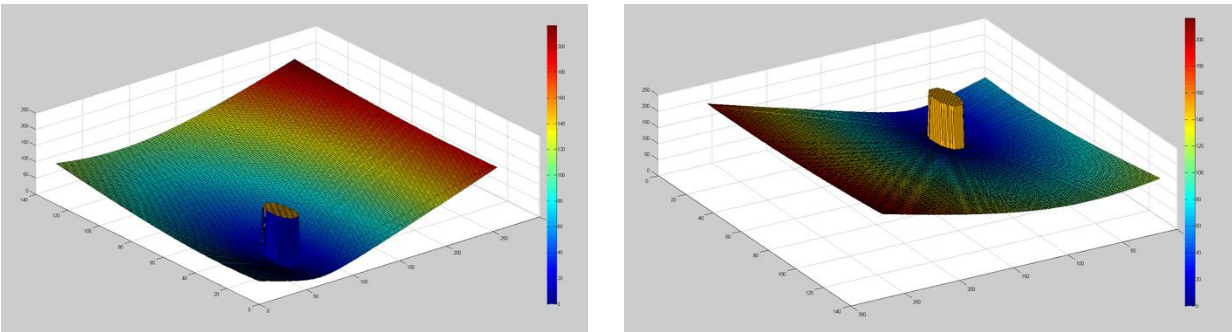

Figure 7: 3D plots of Second Potential Field

A second layer of the second potential field is needed to guide the robot to the ideal position. Instead of checking 10 cells surrounding the robot for the lowest value, it checks 4 neighbouring cells. Hence, the smallest value of a particular cell discovered among the others surrounding the robot will be the next position of the robot, and this process continues until it reaches the ideal position to kick the ball. Once the robot is at that position, the orientation of the robot will be adjusted to ensure that the ball is kicked at the correct angle. In order for the robot to reach its intended cell to kick the robot in the correct direction, the exact cell of the robot must be known. After normalizing the GPS coordinates 100 times:

$$x = \left(\frac{|x_{gps}|}{45} \times 55\right) + 67 \text{ for } x < 0$$
$$x = 62 - \left(\frac{|x_{gps}|}{45} \times 55\right) \text{ for } x > 0$$
$$y = \left(\frac{|y_{gps}|}{70} \times 88\right) + 128 \text{ for } y < 0$$
$$y = 126 - \left(\frac{|y_{gps}|}{70} \times 88\right) \text{ for } y > 0$$

While the first two layers of the first potential field ensure that the ball moves towards the end point based on moving towards cells that are closer to the value ‘300’ in the intended cell position, a problem arises at the centre of the soccer field. As seen on Figure 7, the ball will get stuck at the bottom half of the soccer field at the centre due to Pythagoras formula. For example, if x = 10 and y = 5, the Euclidean distance is equivalent to x = 5 and y = 10. So the ball tends to plan its path within the centre region and it does not get out. To be able to solve this issue, a third and fourth layer of the first potential field must be introduced. The third layer ensures that the values of cells at the bottom half of the centre part of field are bigger, and thus the path planning of the ball prevents it from nearing the wall. It also pushes the ball’s path out of the local minima region. The fourth layer of the first potential field is created to ensure that the values of cells at the orange region, as seen in Figure 8, are bigger than the values of cells in the blue region. Through this, the ball can dribble across obstacles to the intended destination point through the recommended path planning without any collision.

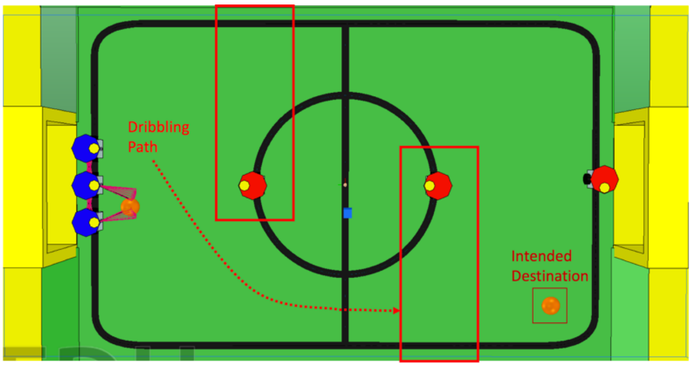

Figure 7: Ball stuck at the centre of the field without the presence of both third and fourth layers of first potential field

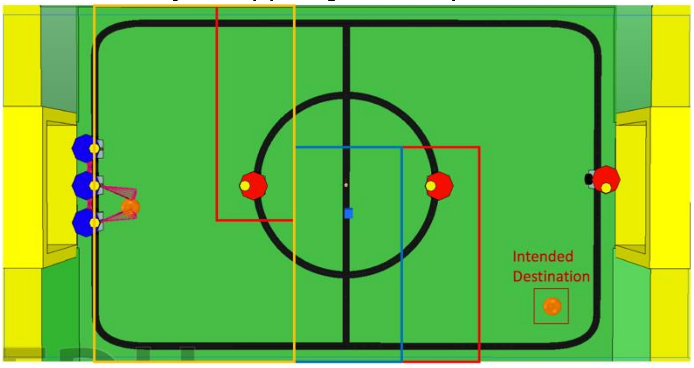

Figure 8: Third and Fourth Layers of First Potential Field indicated by the blue box and orange box respectively

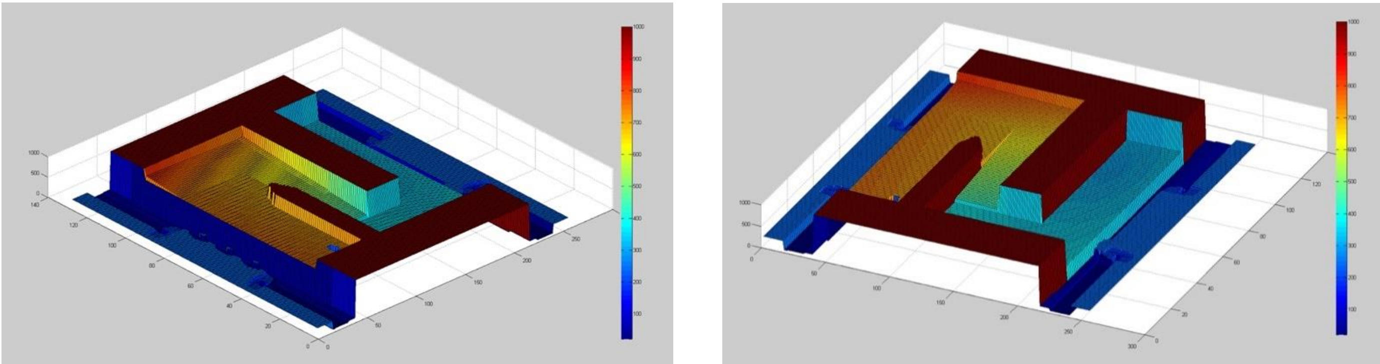

Figure 9: 3D plots of the overall First Potential Field

In the next part of stage 2, there are a total of 5 passes to be completed by the robots, and the passes are only valid if the kicking robot does not perform the pass to the robot that initially passes it the ball. The dribbling algorithm will end with 3 robots positioned properly to receive the ball.

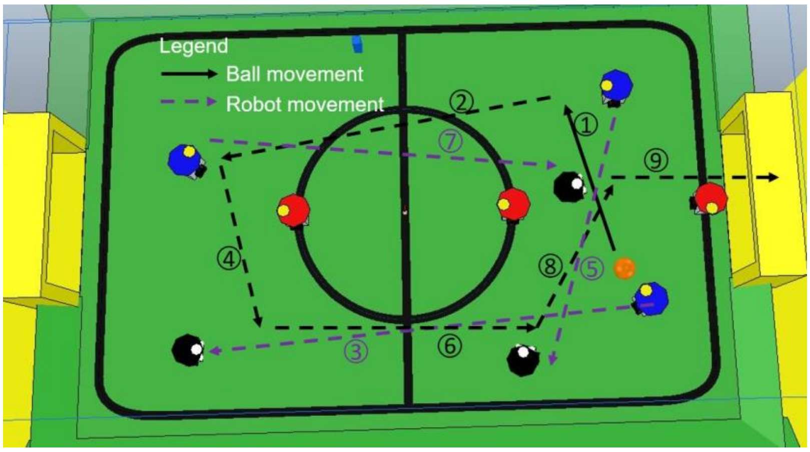

Figure 10: All 9 passes and robot movements

The movements from Figure 10 are in numerical order as the algorithm runs.

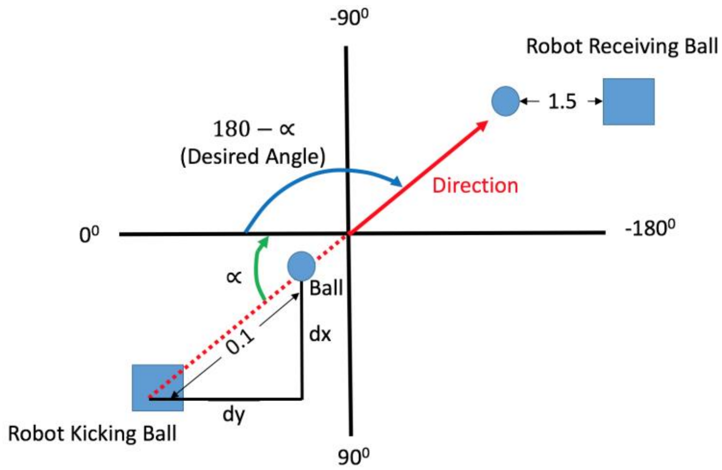

Figure 11: Algorithm and schematic for ball passing

Firstly, the robot positions itself nicely so it creates an angle alpha. From Figure 11, the angle alpha is evaluated to plan the ball's end point such that it does not hit the receiving robot, but instead it should be approximately 0.15 in front of the receiving robot (as seen from Figure 10). In this way, the receiving robot will have more room to adjust itself to create another alpha angle to complete the next pass. In addition, the end point of the ball's position is decided based on the orientation of the receiving robot. If the orientation of the robot is bigger than $-90°$ and smaller than $90°$, the end point of the ball will be planned 0.15 to the left of the robot. Similarly, if the orientation of the robot is smaller than $-90°$ and bigger than $90°$, the end point of the robot will be planned on the right of the robot. In order to complete the passes for the requirements of this project, the robot orientations are coded in a manner that it requires minimal distance for the ball to travel to reach the receiving robot.

Secondly, robot moves and rotate itself to create the angle alpha by utilizing PID control. From Figure 11, regardless of distance d_y or d_x, the Euclidean distance between the center of the ball and the center of the robot is kept at 0.1 and it is done by trigonometry calculations.

Thirdly, the speed at which the robot kicks the ball is proportional to the distance between the ball and its final planned position, the formula is related by:

$$v = 30d - 2$$

where v = robot speed and d = Euclidean distance

From Figure 10, all 9 occurrence happens in numerical sequential order. For the last robot to complete action 9 (to score a goal), it checks if the goalkeeper is at the utmost bottom of the goalpost, so the robot will use PID control to move to the ball such that it aims the shot to the center of the goalpost, increases the chance to score significantly.

Stage 3

The first strategy of stage 3 is to set defensive and goalkeeping roles as priority. One main vital function of the goalkeeper is it keeps itself aligned with the x-coordinates of the ball to prevent it from entering the goal. And the defender keeps itself aligned with an opponent robot nearest to the goalpost to prevent it from shooting. So the Euclidean distance from each opponent robot is constant calculated with respect to the center of our goal, so the nearest distance would be assumed to be the striker robot. The role of the striker robot is to get close to the opponent goalkeeper so that it could possibly obstruct its path and there could be a chance to score. So if the ball is close to our robot striker, it will shoot regardless of the direction. Similarly, identifying if the ball is close proximity to the striker and identifying the opponent's goalkeeper uses Euclidean distance as a determining factor.
The second strategy of stage 3 sends 2 robots to guard the goalpost and send the striker to the opponent's half and kick the ball if it is within close proximity.

Conclusion

Generally, line by line coding is not accurate in instructing the robots actions. For example, if the first line of code instructs the left wheel of the robot to increase its speed 5m/s, the next line instructs the left wheel of the robot to increase its speed to 5m/s, there is a slight delay in turning of the wheel and will cause the robot to be biased to a certain direction. Potential field can be a good algorithm for robot soccer because it is extremely accurate and the path planning has minimal error. However, it is slow because the robot has to constant process the neighbouring cells and this might be impractical for realistic robot soccer match.

One huge disadvantage of this project is the occasional inaccuracy and malfunctioning Vrep API occasionally. It returns wrong GPS value at times and that caused the robots to plan its ball wrongly because the calculation is based on the GPS coordinates.