RoboCup 2021 Worldwide
Team Description Paper
Soccer Simulation Challenge

Team TX - Lee Pak Nin, Marco, Li Ping Yee, Benny
Cheung Sha Wan Catholic Secondary School, Hong Kong SAR, China

Abstract
Three different soccer robot programs were designed with specific roles in the team to create all-rounded attack and defence tactics. Different programming methods were utilized for different programs, as well as some algorithms to allow ball-control.

Introduction

History

The Robotics Team of Cheung Sha Wan Catholic Secondary School (CSWCSS) was established in 2008 and has since actively participated in multi-various local and international RCJ competitions. The present team was formed by 5 senior form students in 2019. Our team has been splitted into two teams, TX and RX, and participated in this competition.

For the first time our team used an online platform to participate in a simulated soccer robot competition. As some of our team members have only received training in C and C++ programming, they were unfamiliar with Python coding and it was a hard task for them to learn from zero and build the programme within a week. Despite that, we were able to obtain the champion in our super-region and will represent the country in this worldwide competition.

Past Achievements in previous RoboCup tournaments

Name	Award	Name	Award
RCJ Taiwan Open 2019	1st Runner-up	RoboCup China Open 2019	2nd Runner-up
RCJ Japan Open 2019	3rd Runner-up	Virtual RCJ Asia-Pacific 2020	2nd Runner-up

Roles of Team Members

Name	Role	Name	Role
Lee Pak Nin, Marco	Attacking role	Li Ping Yee, Benny	Goalkeeper

Links

Instagram : https://instagram.com/cswcss_robotics
Facebook : https://www.facebook.com/hk.csw.roboclub

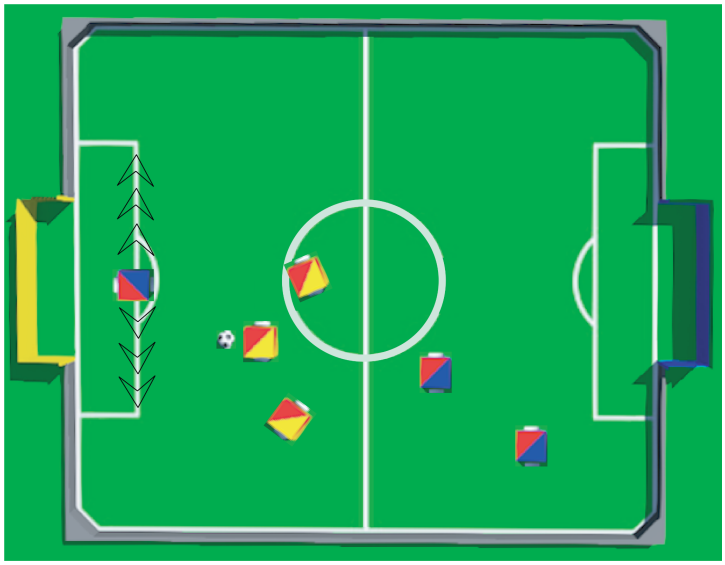
Software

1. Motor Control
    a. Instead of directly controlling the speed of motors, a motor algorithm was developed to calculate the rpm of each motor. With two parameters, the move angle and speed, the robot would move in the designated path. Vector and trigonometry calculations were included in this algorithm.
    b. PID controls were included in the motor control algorithm, a control loop to minimize the errors in automation. PI control, Proportional-Integral control, was used to set the motor current proportional to the error calculated and then change the motor speed when the robot was far from the designated path.
2. Ball tracking
    a. Algorithms were developed to calculate the exact ball angle and distance from the robot. Simple trigonometric functions are included.
    b. For ball tracing, a simple way is by turning itself until the ball is in front of the robot and moving forward. This method is not effective enough. Overshoot caused by turning may happen easily, which causes the robot to oscillate. An advanced way is to calculate the required speed for the robot to turn to a specific angle. The radius of wheels, length of the circular path between each wheel and the designated angle, the difference between the designated angle and the orientation of the robot are included in the calculation.
    c. It was soon discovered that the above method may lead to own-goaling. The ball tracking algorithm is then reconstructed. Instead of directly moving toward the ball, the robot will go to the back of the ball and push it only toward the goal direction. Our robots are able to capture the ball in a “curve” movement. A constant is added or minused to the ball angle according to the quadrant the ball lies in. The constant is determined by the wraparound algorithm. To trace more accurately, the equation is then translated to a simple straight line equation, $y=mx+c$, where c is a parameter proportional to the ball distance.
    d. Wraparound algorithms are also used to calculate the shortest distance between the ball and the robot. In order to capture the ball more accurately, it is divided into different states while tracking the ball. The rpm of motors is setted to be proportional to the ball distance. If the ball is too far away, the speed is the fastest. It is then reduced when being closer to the ball.
3. Ball Prediction
    a. Most of the time, the ball moves at a faster speed than the robots. It is a waste of time to follow the backside of the ball. A ball prediction algorithm is developed to predict the future location of the ball. The current ball data is compared to the previous ball data at a specific time interval. The difference in ball location is then added to the current ball data and resulted in the predicted ball position.
    b. The above algorithm is not suitable when the ball is moving toward corners. The ball will re-bounce to the field which contradicts our prediction. The advanced

ball prediction algorithm is still under development, which will consider the angle of incident and ball speed in order to predict its re-bounce.

Results

Strategy

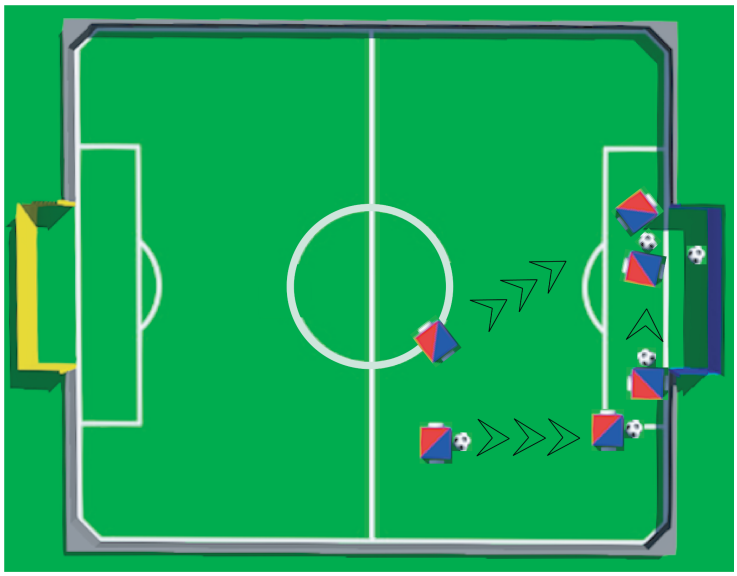


Defence Strategy 1

One robot acts as a goalie to prevent the opposing team from scoring. It receives the angle of the ball relative to the direction the robot is facing, then moves forward and backward to block the ball.

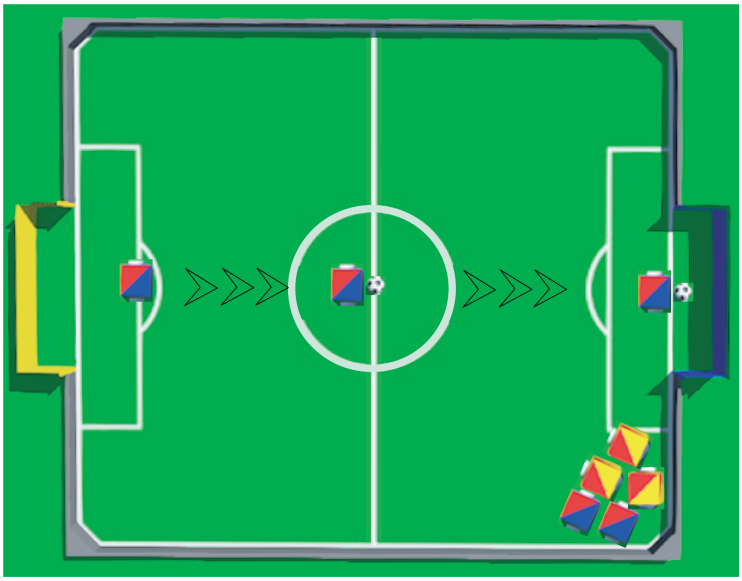
Defence Strategy 2

If the ball is on the flanks, each robot does a different action. One robot moves towards the flank to intercept the ball. It pushes the ball into the wall or flicks it upfield to clear the danger. Another robot moves into the goal. In case the ball gets past the first robot, once it reaches the goal the other robot will push it away from the goal, switching from defense to attack immediately. The last robot moves into the middle of the field to wait for the ball to respawn in case of lack of progress.



Attack Strategy 1

Two robots will be the main attacking force in the team. One robot will push the ball towards the side and try to shoot it towards the middle, while the other robot will go to the other side of the goal and wait for the ball.



Attack Strategy 2

If the ball is unable to be shot towards the middle and is stuck in the corner, the goalie will move to the middle and wait for the ball to be spawned. This can almost guarantee a clear shot at goal and at most the robot will move back to defend if the ball is not spawned in the middle.

Conclusion

We were able to create an all-rounded team combining different attack and defensive strategies to deal with the ever-changing environment in the simulation. It was a great experience for everyone involved as we were able to learn python coding and earned a spot in the world championship. Still, there are some improvements that can be made to our programs and it is expected that they will be implemented in the future.

Future work

1. Teammates communications

   In the future, it is planned to develop internal communication between robots. Self position, ball distance of each robot will be shared with other robots. Roles switching can then be implemented. For example, if one robot is closer to the ball, it will act as the attacker while one will back to the defensive position and one will goto the corner of the goal.
2. All-Attack-Tactics

   In order to increase the chances of goal, we planned to adopt all attacking tactics. The defence robot will give up defending while the attacking, i.e. when the ball is not in our side. With three robots attacking, it is possible to overload the opponent's defensive line. When the ball tends to move to our goal, two of our robots will go back to the defensive position immediately to prevent counter-attack.
3. Two-sided ball tracking

   Our current ball tracing algorithm is one-sided, in which it will keep the front side facing the opponent's goal anytime. In some situations, for example, when the opponent's goal and ball is behind the robot, it is absolutely time wasting to self-turn and trace the ball. The two-sided ball tracking tactic, enables our robot to trace the ball with its backside.