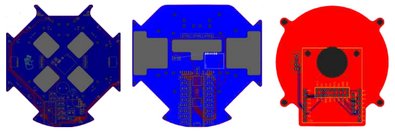
ABSTRACT

In this poster, we describe the overall production process and the strategies we employed - primarily localization, ball control and communication between the robots. We also share the challenges we faced and the improvements made to make our robots more competitive and reliable.

PRODUCTION PROCESS

Hardware Design/Construction

Our PCBs were designed using Fusion360 and EasyEDA, and were fabricated by JLCPCB. The overall robot design was done in Fusion360. Last year, two challenges we faced were that the 18cm size limit made it hard to fit the components well, causing them to block the ball when it was next to the robot, and it was difficult to see the ball when it was far away due to the smaller size. This year, to note how the components affected the camera's view, we drew a line representing a camera ray, and fit the components as best as possible. The robot sees ~50% more of the ball when it is next to it, and ~0.5m further.

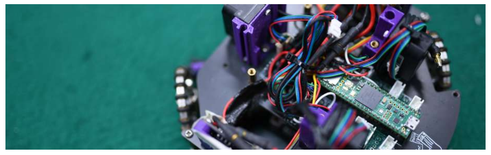


Printed Circuit Boards (PCBs)

Component Positioning in Fusion360

Components

TOTAL COST OF DEVELOPMENT (USD)	1044
1x PCBs	30
1x Teensy 2.0	10
1x Teensy 4.1	30
2x DC-DC Buck Converter 6~14V to 5V/8A	17
1x JF-0826b Solenoid	5
1x CJMCU-7960 Motor Driver	4
1x XL6009 Boost Converter	1
4x JMP BE 3561 Motor	380
4x GTFRobots 50mm Metal Omniwheel	112
2x Elechouse 50A Motor Driver	78
1x TFMini Plus	240
1x HC-05 Bluetooth Module	7
1x OpenMV H7	85
1x GY-951	15
2x Turnigy 1300mAh 3S 45~90C Lipo Pack	30



We previously used the Teensy 3.5 as our MC. However, while sensors updated at ~1ms, our loop time was >1ms. Hence, we switched to Teensy 4.1 for its higher clock rate (600MHz from 120MHz), removing the processing bottleneck and allowing us to run more advanced logic. We also switched from the TFLuna to TFMini Plus for its range (0.1-12m from 0.2-8m) for the 12cm outer boundary.

Software

Both MC (Teensy) are coded in C++ using Arduino, and the OpenMV H7 is coded in Python using the OpenMV IDE. We also rewrote the manufacturer libraries to be non-blocking by creating a state machine and constructing the message packet when data was available, as opposed to waiting for it.

COMMUNICATION BETWEEN ROBOTS

The robots communicate via bluetooth to improve their play.

Components used:
- HC-05 Bluetooth Module

Solution 1: Transitioning Between Roles

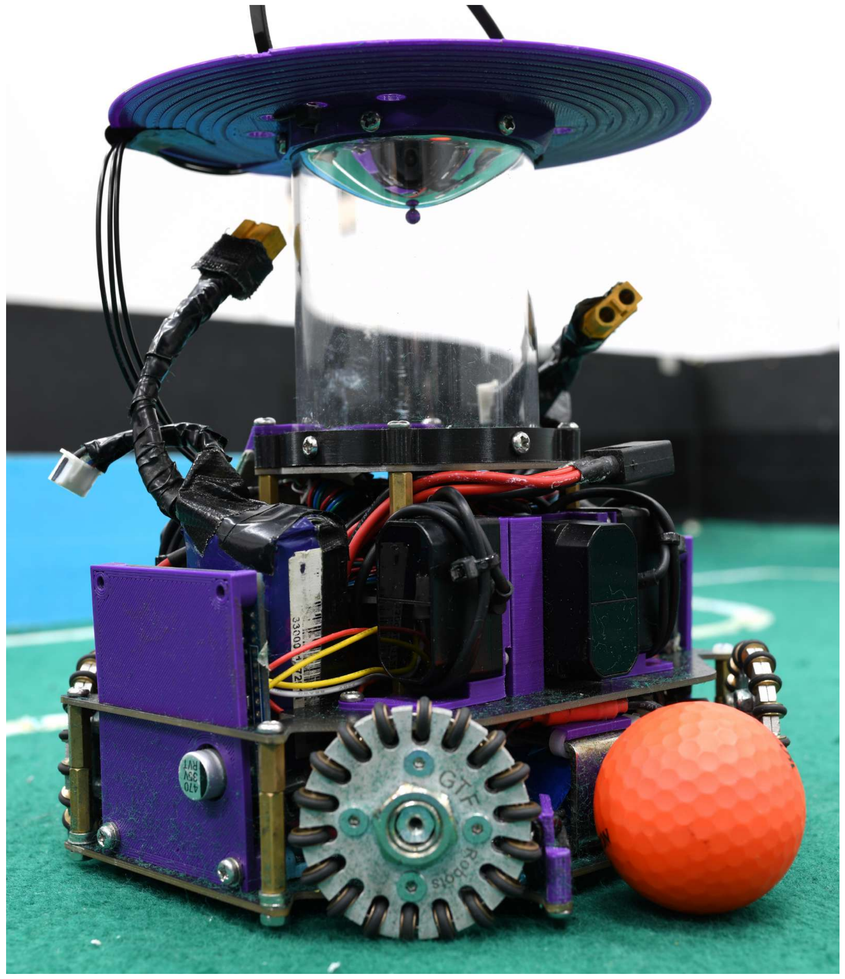
The robots use bluetooth to check if the other robot is in play. Normally, there is one striker and one goalie. To ensure constant defence, the striker will temporarily take on the role of a goalie if the goalie is taken out of play.

Solution 2: Sending Information

The robots also send the position of the ball to each other. If the robot is unable to see the ball, it can use the position provided by its teammate. This allows the goalie to be better positioned as it is frequently occluded by the striker. This means the goalie does not need to move as quickly and is easier to control.

SOCKS

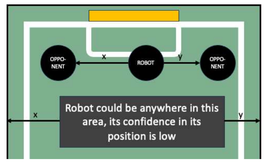
RCJ SOCCER OPEN **SINGAPORE**



LOCALIZATION

Localization ensures the robot remains within bounds and in play at all times.

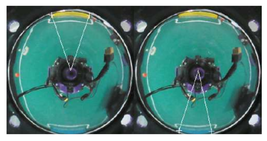
Components used:
- 5x TFMini Plus
- 12x TEMT6000 Light Sensors



Confidence Illustration

Solution: LiDAR-based Localization

5 LiDARs are used to pinpoint the robot's position. If the robot knows where it is in relation to the bounds, it can slow down in time to stop at the boundary. If its position is unknown (if the LiDARs are blocked) and its confidence in its position is low, it reduces its speed such that the TEMT6000s can catch the line and stop the momentum of the robot if it is heading out of bounds.



Camera-based Localization

Improvement: As the robot frequently got blocked by the opponents, we added camera-based localization to increase its confidence. The robot determines the angle to the top left and right corners of the goal, and calculates another possible position. This improves the robot's confidence and it is less affected by other robots on the field.

OBJECT DETECTION

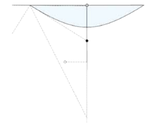
This allows the robot to track objects and score.

Components used:
- OpenMV H7
- Acrylic Tube
- Hyperbolic Mirror

Solution: Hyperbolic Mirror for 360˚ FOV

We used Fusion360 to simulate the FOV of the mirror such that it would not see above the field walls (to prevent false detection of objects).

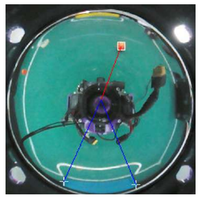
We pressed a heated PVC mirror sheet onto a mould, which was 3D-printed with 0.05mm layer height to ensure layer lines were not visible on the mirror.



Mirror Calculations

Mirror Mould CAD

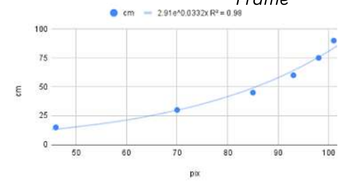
Mirror Forming



OpenMV Frame

The camera detects each object using manually tuned LAB values, sending the angle and distance of each object relative to the robot down to the processor.

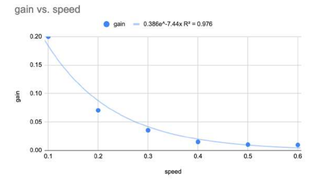
Improvement: Due to human error in the forming process, the theoretical conversion from camera pixel distance to cm distance was inaccurate. We calibrated the conversion by plotting a graph of pixel against cm distance.



Graph of pix distance against cm distance

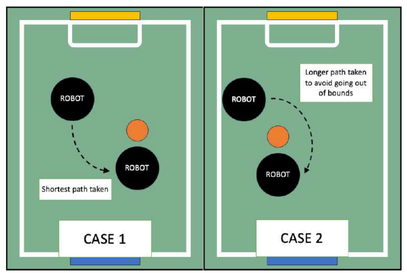
BALL CONTROL

Solution 1: Compass Correction



Components used:
- 4x JMP-BE-3561 JoinMax Motor
- 4x GTF 50mm Metal Omniwheel
- 2x Elechouse 50A Motor Driver
- GY-951 IMU

Using the IMU, the robot obtains its heading and rotates to face the opponent's goal. To avoid overturning and oscillating, it rotates at a speed proportional to its heading error. We tuned the proportional gain at various speeds to obtain the graph above.



Solution 2: Ball Orbiting

To capture the ball in the ball capture zone, the robot orbits around it, taking the shortest path (case 1).

Improvement: The shortest orbit may not be optimal nearing the boundary as it may cause the robot to go out of bounds. As such, the robot takes the longer orbit (case 2).

SCORING

Solution: Localization-based Aiming + Kicking

We used a light gate consisting of an IR transmitter and receiver in conjunction with the camera to detect when the ball enters the ball capture zone. The robot then accelerates towards the goal to maintain possession of the ball and kicks when appropriate.

Components used:
- IR Transmitter
- IR Receiver
- JF-0826b Solenoid
- CJMCU Motor Driver
- XL6009 Boost Module