Robocup Junior Rescue Line 2025

Team Description Paper

Fusion Zero

---

0 | Abstract

FusionZero's fourth-generation robot debuts on the international stage. Measuring just 200 × 145 mm, its compact footprint boosts manoeuvrability and obstacle clearance. A two-stage claw autonomously collects, stores, and sorts two victims, streamlining evacuation-zone tasks. Downward-facing and forward-facing cameras handle line tracking and victim recognition, while a low centre of mass, IMU, and custom whegs keep motion stable over ramps and debris. Algorithms exploit the competition's fixed layout, mirroring human heuristics to cut computation. A front sweeper with synchronised LEDs normalizes vision input; ToF sensors and limit switches map nearby geometry. All subsystems run on a Raspberry Pi 4B powered by two 18650 cells.

1 | Introduction

FusionZero is a two-member team from Auckland, New Zealand, formed organically as older teammates moved on and the international rules demanded sharper, broader skills.

Frederick Sun, team leader, shapes the overall robot architecture and chemical-spill strategy. He models and simulates in Onshape, 3-D-prints custom parts, lays out the core program framework, and brings both vision-based victim detection and GitHub source control into the workflow.

Aidan Zhu entered RoboCup at age 11 and first tackled line-following, chemical-spill, and chassis design. Now partnered with Frederick, he refines the line-tracking algorithms, stress-tests evacuation-zone behaviours, and designs PCBs for the main controller and sensor arrays.

Together as a team, they have achieved:

- [Nationals] 2024, Singapore Open U19 International, 3rd Internationally
- [Nationals] 2024, Australian Open U19 International, 2nd
- [Nationals] 2025, Japan Open U19 International, 4th
- [Nationals] 2025, Singapore Open U19 International, 2nd

2 | Project Planning

A | Overall Project Plan

Our objective is a podium finish in the team's first international run. We decompose the rescue-line challenge into four work-streams: constraints, mechanics, software, and testing, ensuring each constraint is addressed early and iterated quickly.

Ruleset analysis

>The team captain may make further attempts at the course to earn additional points from scoring elements that have not already been earned before reaching the next checkpoint.

We did not understand that we could continue going to points for other scoring elements, not just tile. We must continue trying to get points for other elements.

>The line along the ramps can contain gaps, speed bumps, intersections, obstacles and debris [...] exit [...] entrance [...]

There has been a trend in the past years where the line-following section of the competition has dramatically increased in difficulty. Hence, we need to create a robust system that can handle many different elevations, lighting conditions, and terrain. Thus, a camera was necessary to replace our old colour sensor board.

Mechanical constraints

Many teams fail certain challenges, such as tilted or inclined boards, gets stuck between obstacles and walls, or tips over on sea-saw. We endeavored to create the smallest possible robot whilst all components could still be fitted on.

There can be speed bumps and debris within the evacuation-zone as well, and thus our old claw design, where it slid along the floor, would not work. We need to transition to a top down claw design.

Debris on tiles that perfectly covered the line, rendered our feedback system (colour sensors, camera) useless as the line was blocked. We need to create a design that maintains a consistent feedback loop without false information.

The problem of going uphill and maintaining traction, especially during the uphill-tilted transition and speed bumps uphill, would always cause wheel hop or loss of traction. We need to switch our wheel approach to something with higher CoF and can get past speed bumps.

Software constraints

We want to be able to run two cameras and perform AI inference. Thus, our hardware limits our opportunities to explore various software solutions. We must use a Raspberry Pi 4B or better, alongside a Google Coral TPU.

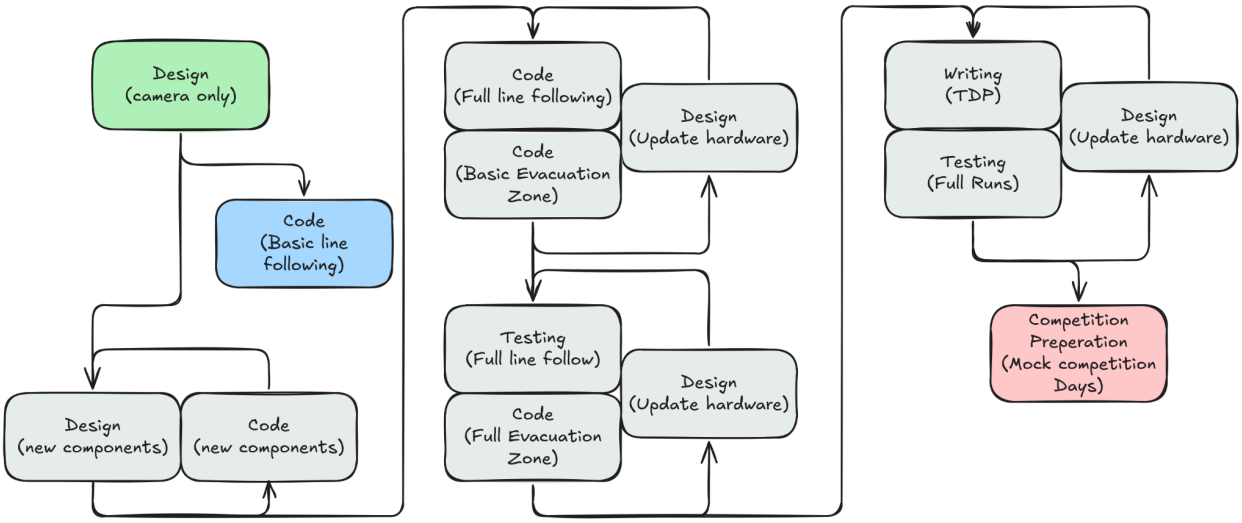
Depending on the OS we run, there are certain python libraries that are incompatible with the python version, such as the inferencing SDK working on python <3.9. Thus, we are bound to run Raspberry Pi Bullseye.

Time constraints

There is little time between Robocup Singapore 2025 and Robocup International 2025, thus making large changes impossible due to the lack of security as we have no practice. This prohibits us from changing to DC servos with encoders despite their superior precision and control.

B | Timeline

Our milestones leading up to the competition will follow: planning - design - software - testing, in a waterfall style approach. However, within each stage barring the initial designing, we will employ an iterative approach. The waterfall aspect is because we will need a physical robot before we can attempt any software, as simulation does not reflect real life. The iterative aspect is required as there will be problems that cannot be solved by simply software, and requires a hardware modification.



Frederick (**F**)
Aidan (**A**)

13 - 20th March (0th stage): Review and consolidate mistakes

Description

F Gather feedback from real competitions (Robocup Japan, Robocup Singapore, 2025). Draft a fix in pseudocode for code, or component/sketch for hardware.

Result

Document summarising issues and proposed solutions used in reference for the new robot.

20th March - 3 April (1st stage): Design, Camera only robot

Description

F Will quickly create a mock-design for **A** for line-follow testing. **A** gives feedback regarding camera placement and angle.

Gate

Robot must have basic CAD

Result

A robot with only motors, voltage regulator, battery, Raspberry Pi 4, and a downwards facing Raspberry Pi Camera Module 3.

4 April - 2 May (2nd stage): Code, basic line following | Design and code new components

Description

F will finalise robot design including claw, extra sensors, handle, debugging, and stability. Will clean-up the Github repository, and update the readme. **A** will begin the basic line following with the camera-only robot. This allows him to relearn the basics and begin to anticipate hardware improvements.

Gate

All sensors should be validated and working - otherwise our software solutions would be incomplete or not optimized.

Result

A finished V1 of the robot on Onshape, with all components, and a 3d printed model that reflects CAD, verifying that all components work. Robot can follow basic tiles commonly seen throughout other competitions. Intersections are not included.

4 May - 15 June (3rd stage): Code, Full line following | Code, Basic evacuation zone| Design, Update hardware

Description

F will get basic evacuation zone running, as well as working on the overall program flow - incorporating the run on switch, wireless connectivity, and bash setup script. **A** will focus on making line follower reliable, by extensive testing with our custom tiles within full courses with speed bumps, intersections, gaps, obstacles, inclines and tilts.

Result

Victims are successfully identified, routed to, grabbed and lifted. Evacuation points (red/green triangles) are successfully identified, routed to, and victims are dumped according to what is sorted. Full line follow is possible as individual modules, but requires integration.

16 May - 22 June (4th stage): Testing, Full line follow | Code, full evacuation zone | Design, Update hardware

Description

F will get a full evacuation zone running (entry, exit, routing,obstacle detection, IMU integration). **A** will run tests on full courses designed to be varying in difficulty to determine the robots performance and key areas to focus on, being robust and having tested as many possible cases as possible.

Result

Routing around the evacuation zone works (wall-follow) with entry and exit alignment, and obstacle avoidance is implemented. We are able to classify all tiles with modifiers into “easy” and “hard”.

23 June - 30 June (5th stage): Testing, Full Runs | Writing, TDP, Poster, Video| Design, Update hardware

Description

F will be creating TDP, Poster, and Video with **A** assisting. A will be simulating full runs with both line follow and evacuation zone, working on said transition, and manual start.

Gate

TDP, Poster, Video must be complete.

Result

Stuck detection is implemented within the line follower. **A** will become familiar with the starting procedure. TDP, Poster, Video are all complete to a high standard in regards to the marking scheme.

1 July- 8 June (6th stage): Testing, Mock Competition Runs| Packing, Documentation, Datasheets

Description

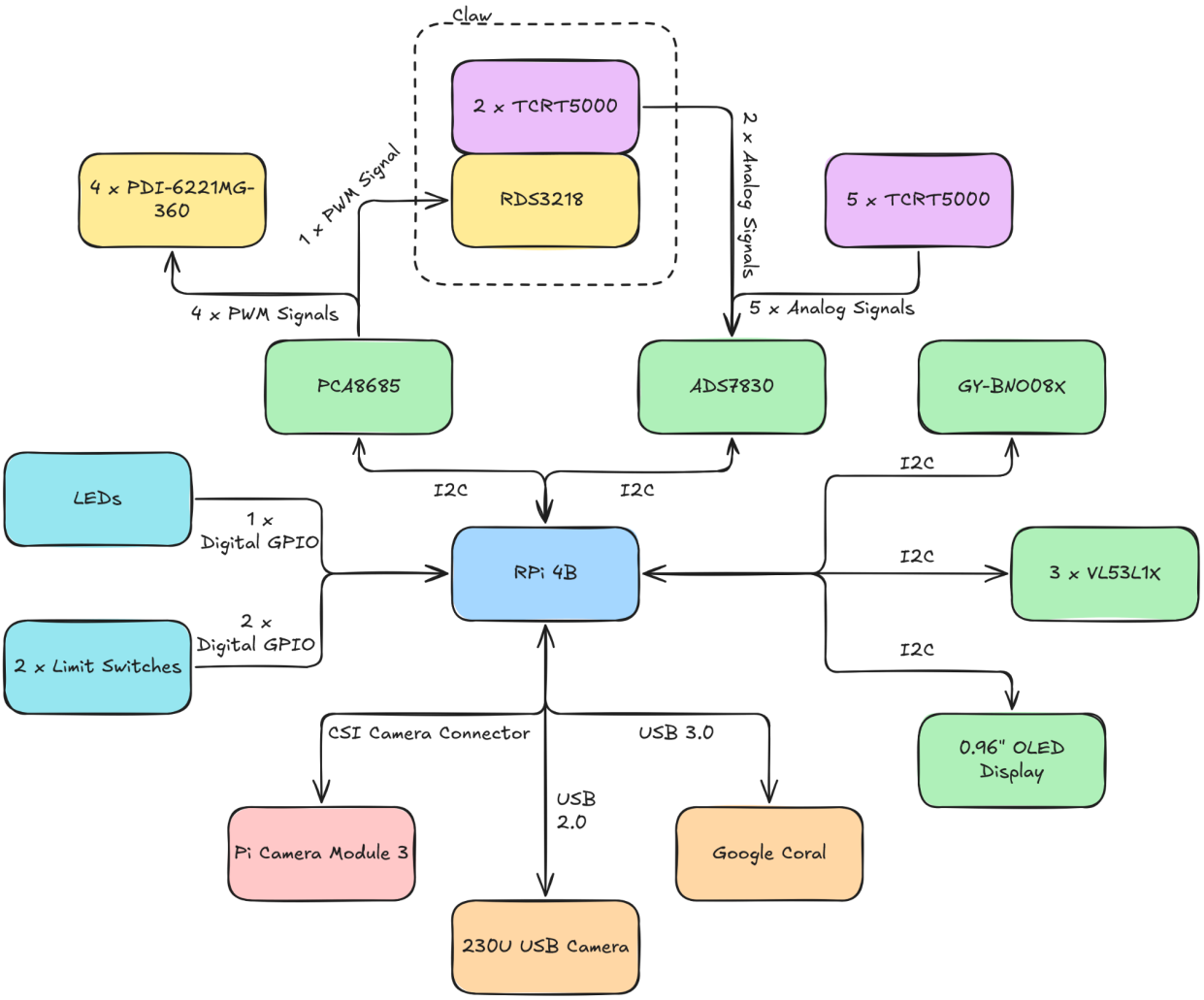
F sets up a course; **A** runs it ONCE. While **F** swaps the course, **A** works on fixes in an isolated environment. We will be tracking the score for each run. **F** will be working on gathering all relevant documents and datasheets.

Result

A full scoring sheet with increasing scores as the day progresses. All practice runs from 2024 should have been complete.

C | Integration Plan

The overarching subproblems can be further broken down into each individual component. We first develop a component, then test its validity within the wider system, creating an iterative design loop and eliminate problems.



Microcontroller / Controller

We have decided to use a Raspberry Pi 4B as our main controller for this event. It provides digital I/O, exposes the kernel I2C, with its 40-pin GPIO, CSI camera header, and USB ports, 4GB of RAM, and 1.5GHz clock speed, a clear upgrade from the Raspberry Pi Zero 2W.

Drive chain, Claw (Movement)

We will be using 4 x PDI-6221MG-360, as opposed to the standard DC servos with magnetic encoders, due to the time constraint mentioned earlier; we are familiar with these 360 degree continuous servos as they only require a PWM signal to control. They offer high power (20kg @ 10cm) at 6v, which fits within our power budget. Regarding the claw, we will be using a high-performance 270 degree servo - RDS3218 - that can lift the entire claw mechanism.
Communication: I2C between RPi and PCA8685; generating up to 12 PWM signals

Line follow

A downward pointing Pi Camera Module 3 Wide Angle will be used to filter for the black line, given an image with consistent image. Thus, we have used White COB LED, 5V, 3mm to normalise any shadows. To understand the current terrain the robot is facing, we have included the GY-BNO08X IMU, which accurately relays the current pitch, roll, yaw of the robot. We have kept the colour sensor board, consisting of 5 x TCRT 5000s.

The rules state that speed bumps can have a height of "1 cm or less". When placed on an incline or tilted tile, proving to be a challenge. To combat the slipping and vibrations, we have decided to create alternative wheels, known as "whegs" out of silicone, which has a high coefficient of friction of ~15.

Communication: CSI camera connector between RPi and Pi Camera; Digital Output between RPi and MOSFET; connecting to the LEDs; I2C between RPi and IMU; I2C between RPi and ADS7830; connected to each TCRT5000 analog output

Evacuation Zone

A forward pointing 2303U USB2.0 camera to view the entire field, at 120 degrees horizontal FOV. We use an SG90 as part of our claw to control picking up and releasing victims, combined with 2 x TCRT 5000s to verify victim colour and presence.

As a backup, we have installed a Google Coral TPU that allows us to use AI inferencing if our current search methods fail during the competition.

Communication: USB2.0 between camera and RPi; I2C between RPi and ADS7830; connected to each TCRT5000 analog output; USB3.0 between RPi and Google Coral TPU

Navigation (Evacuation Zone, Obstacle)

To understand our environment, we have included 3 time of flight sensors, specifically the VL53L1X's, facing left, right, and forwards. However, they can be inaccurate at times, which is why we have also included limit sensors at the front, for absolute data.
Communication: I2C between VL52L1X and RPi; Digital Input between limit sensors and RPi

Interfacing (Handle, switches, OLED Display)

As a quality of life improvement, to allow for easier debugging, and in accordance with the rules, we have used binary switches and an OLED display to show information without needing a laptop everywhere we went.

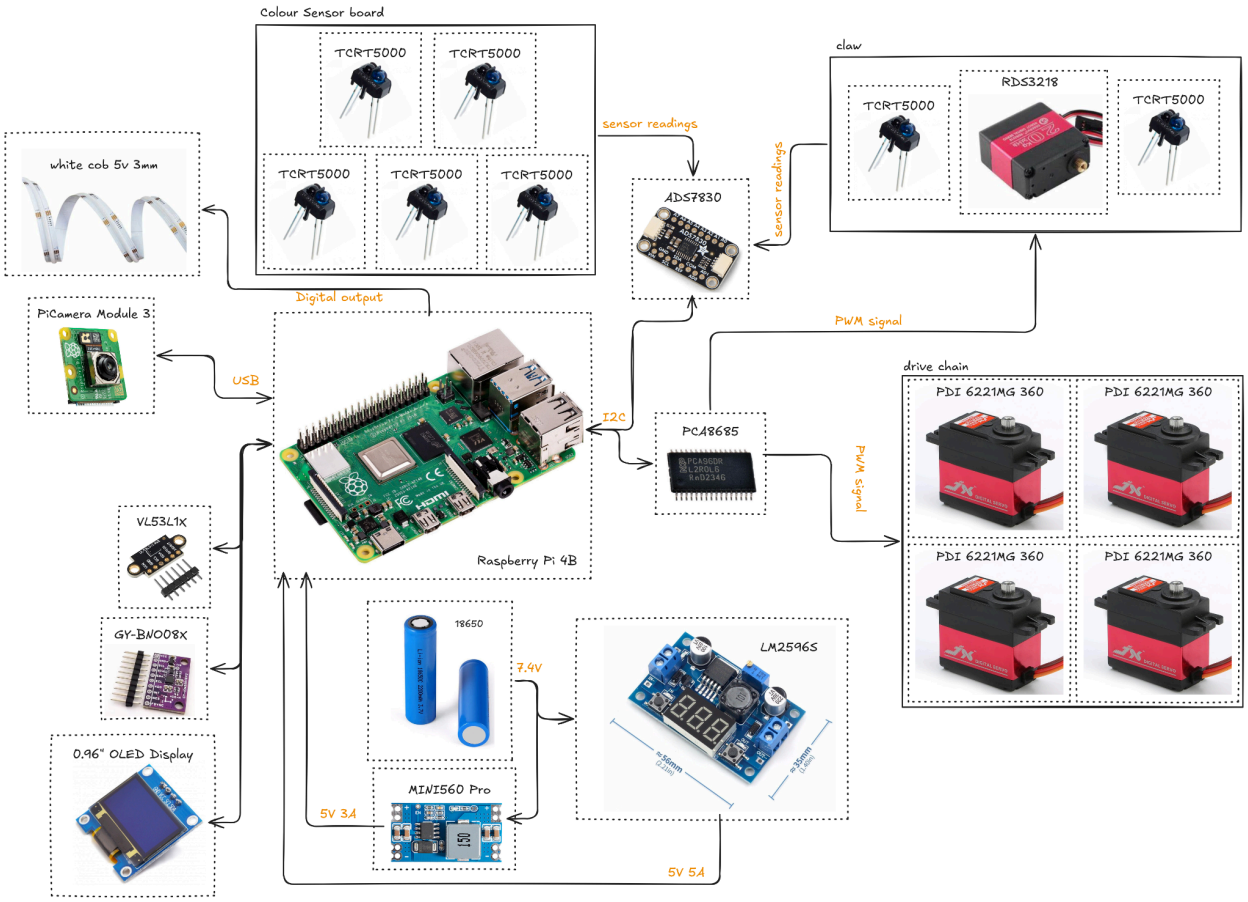
Communication: I2C between the OLED display and RPi; Digital Input between binary switches and RPi

3 | Hardware

Our goal was a small, low centre of gravity, light chassis robot. To accomplish this, we combine CAD modelling with 3D printing with PLA, giving a free-form structure, optimizing the required space for basic structures, as well as allowing for rapid development and iteration, and giving space to grow.

The basis of our design rests on the chassis: separated into layers, we are able to utilize the z-axis much more efficiently, stacking components directly on top of each other. At the very bottom of the stack, we have the colour sensor board and LEDs, then the drive train and line-follow camera, followed by the various small components and sensors, then finally the controller and battery pack. However, the power train and associated voltage regulators are laid at the back for easy access and visibility. Finally, at the front of the robot, the claw reserves space at the top for its movement capabilities, while we place the evacuation-zone camera below it, as well as the various ToF sensors.

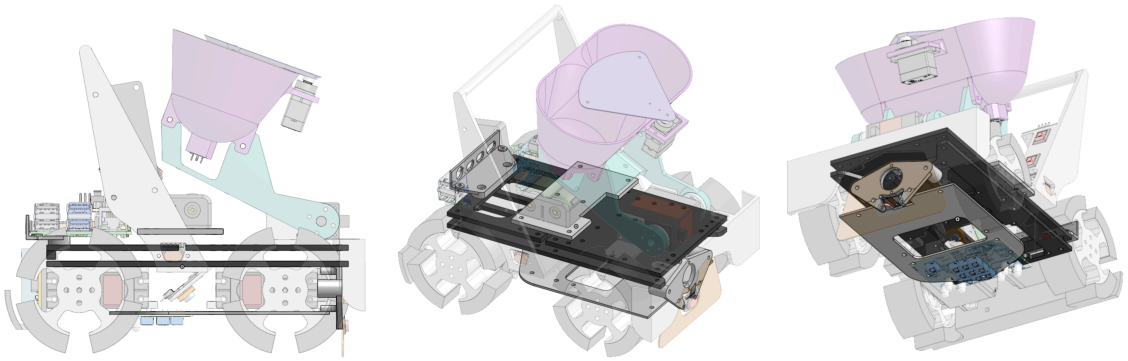
This layout is efficient as there are plenty of options for wires to be routed while keeping the design compact. A key element of this is being able to have two robots available for debugging, due our low cost of production.



A | Mechanical Design and Manufacturing

Chassis

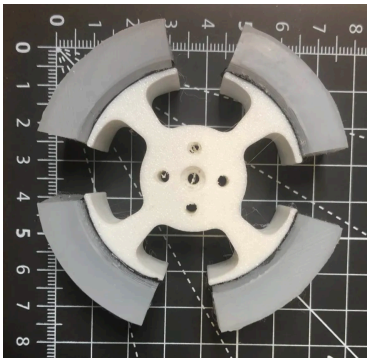
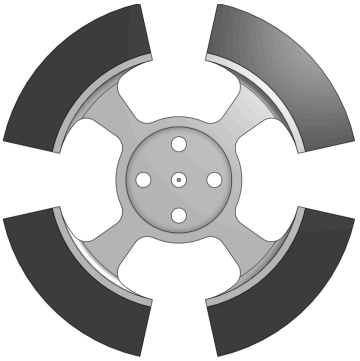
Printed with PLA at 3mm thickness, our chassis provides the rigidity necessary for the small robot and maintaining a small footprint. We have split the robot into separate bases (servo, bottom, back, front, back, battery, and controller) and frames (lower parts, IR sensor, LEDs) that connect and hold all the components together, utilising the z-axis to its fullest. We connect each layer with standoffs of consistent heights, allowing wires to run between each layer. We tested and validated these placements and sizes firstly through visualisation within the CAD, then 3d printing and testing with the real components. This ensured that everything fit together and the entire assembly was viable.



Wheels

This year, we have replaced our standard rubber wheels with custom silicone “whegs” (wheel legs). The choice of silicone as our base material is due to its high CoF of 15, providing the much needed grip whilst going on various inclined or tilted tiles. The term “wheg” is coined from the combination of wheel legs: by removing parts of the circumference, we create a broken circular shape, leaving notches, and allowing the wheels to use leverage in lifting itself past speedbumps.

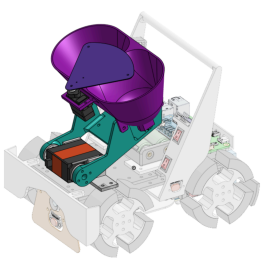
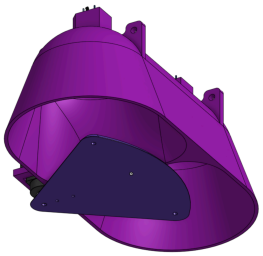
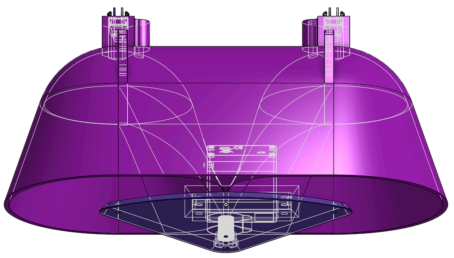
Thus, this hardware solution enables us to navigate even the most difficult terrain, and fulfills our goal of maintaining control in every terrain situation.

There were many designs leading up to this, and we validated each through testing each with the same robot setup, only swapping wheels. Our test configuration was uphill, uphill with speed bumps, downhill, and spinning on an incline, and we visually confirmed each challenge passed and the robot was in control.

Rescue mechanism

Our goal regarding the rescue mechanism is to switch from the present 3-bar-linkage, single actuator claw to a two stage claw as further testing of the old system revealed that “Speed bumps may also be placed anywhere in the evacuation zone” would fail any attempt at grabbing the victim, as it relied on the claw being flat on the ground. Thus we have pivoted and changed to a cupping system, where victims are encapsulated and held, which is a more flexible system as it works regardless of what is on the floor. This fulfills our goal of creating a claw that is effective in every situation.

By holding the ball in a set position, and then closing the bottom, we do not let the victim escape - enclosing it and allowing us to place TCRT5000’s to allow us to read what is currently inside the claw (nothing, dead victim, live victim).

This claw is a straight improvement from our last one, having a capacity of up to two victims. This includes the ability to selectively dump, which could be interpreted as real-time sorting. Compared to other teams, our claw has an advantage of having less mechanical parts. We do not have a separate holding container, fulfilling our size constraint.

To verify that this hardware works, we tested each version of the claw against different sized victims (40mm and 50mm diameters), different weights and different materials. Furthermore, we created a moments calculation which verifies that the servo does have enough strength:

$$\begin{aligned}\tau_{\text{servo}} &= \vec{F} \cdot d_{\perp} \\ &= 20g \times 1 \cdot 10^{-2} \\ &= 1.962 \text{ Nm}\end{aligned}$$

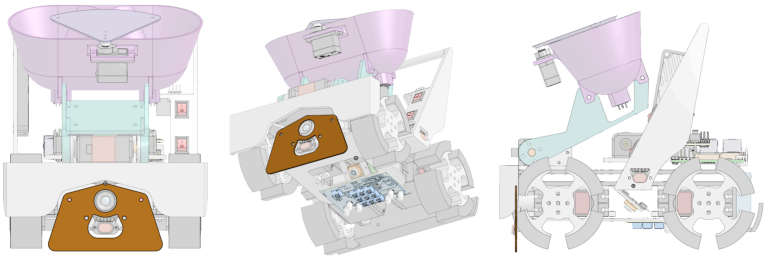
$$\begin{aligned}\tau_{\text{required}} &= \vec{F} \cdot d_{\perp} \\ &= 160 \cdot 10^{-3} \cdot 9.81 \times 76.575 \cdot 10^{-3} \\ &= 0.12019212\text{Nm}\end{aligned}$$

$$\tau_{servo} \gg \tau_{required}$$

Physical sweeper

We have encountered debris that ruined multiple of our runs as it blocks the line from being seen, regardless of camera or colour sensor, making following the line extremely difficult as critical information was lost. To solve this, we create a physical sweeper that drags along the floor and pushes all debris in front of the camera along as the robot moves forwards, ensuring none of the debris moves into the camera image and blocks vision. We accomplish our goal of maintaining consistent information with this sweeper.

We will test the shape and the height of the sweeper by dragging it along the floor with a straight tile, and placing various pieces of debris. We ensure all debris is swept away.



Innovative designs

We consider our rescue mechanism, wheel, and sweeper to be innovative designs, as we haven't seen any other teams employ such tactics, giving us a substantial competitive advantage. Our rescue mechanism that combines both storage and sorting allows our robot to be extremely small, making maneuvering much easier around the course. Furthermore, our wheel's design is suitable across a wide variety of terrains, and thus both line follow and design complexity is reduced with its reliability. Finally, the TPU sweeper is printed in blue, and drags along the floor, removing all debris. It also allows the downwards facing camera to see it when it has been reflected off the silver entrance tape to the evacuation zone, allowing us to detect the entrance.

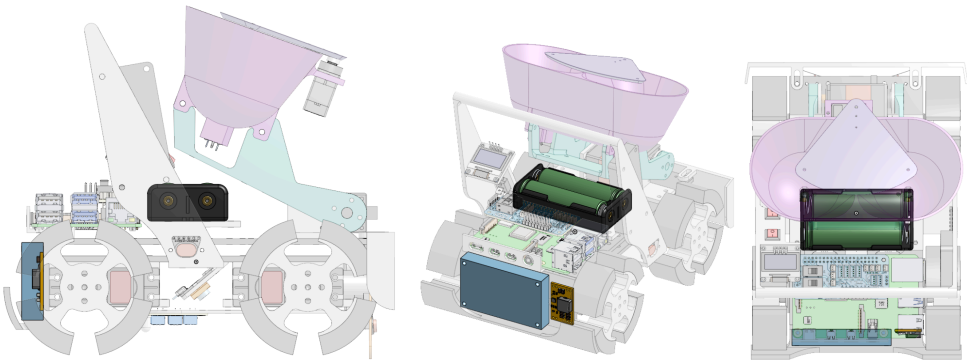
B | Electronic Design and Manufacturing

For all PCB designing, EasyEDA was the software of choice for its ease of use and cloud service. For ordering PCBs and assembling physical boards, JLCPCB was the service of choice, and we manually bought all components.

Power delivery system

To keep in compliance with our small footprint, we had to use 2 x 18650 batteries which could supply, at a nominal voltage, of 7.4V (8.2V peak). There are 3 major systems upon our robot: controller, sensor stack, and actuators. We use a large DC-DC Buck converter (LM2596S) capable of outputting 6V 5A, and a MINI560 Pro 5V regulator, capable of 5V 3A. With this setup, we are able to drive actuators at 6V's, and only reach a max current draw of 4A. The MINI560 Pro powers the Raspberry Pi 4, as it only draws 1A under load, leaving 2A of headroom for the rest of the sensors, operating at either 5V or 3.3V.

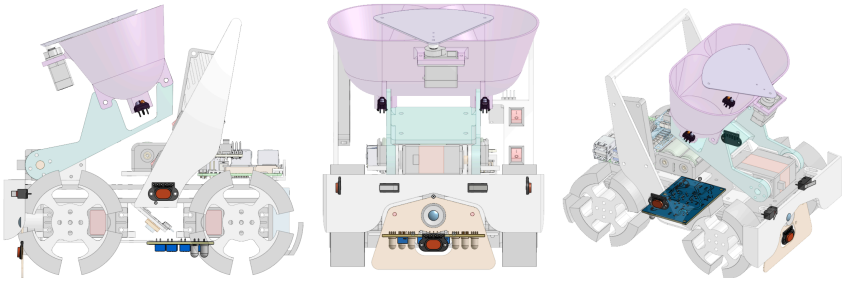
We measured the success of this method by firstly measuring the current draw of each component or finding it within the datasheet, allowing us to find the input power necessary for all the large components.



Sensors

To gather data about the environment for the robot to make confident decisions, we have many on-board sensors. Firstly, ToF VL53L1X sensors handle measuring the distances in the 3 cardinal directions. Second, limit sensors placed at the front provide absolute data regarding collisions at the front of the robot. An IMU allows the robot to track its pitch, roll, and yaw, indicative of inclines, tilts, and vibrations. A colour sensor board containing 5 TCRT5000s aids the downwards facing Pi Camera Module 3 Wide Angle where the lighting or angle is inconsistent. To validate victims grabbed, we use another 2 TCRT 5000s to detect presence, or colour (reflective, black).

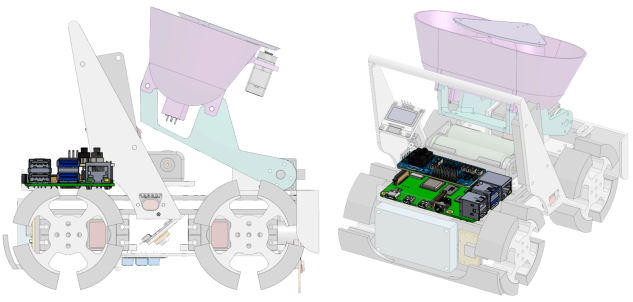
To validate that all these components work, we developed each sensor by itself with isolated hardware and then coding each module with its appropriate library during the 2nd stage of development.



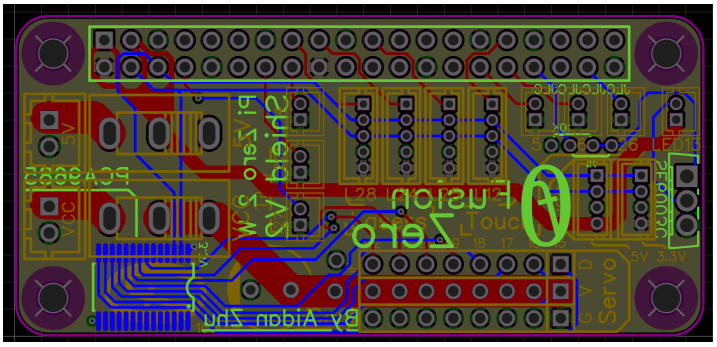
Controller/Microcontroller

For the international build, we eliminated our old microcontroller entirely by upgrading to a Raspberry Pi 4. With significantly higher processing power, the Pi 4 now handles everything - from PWM control (via PCA9685) to analog sensor parsing (via ADS7830) and real-time vision tasks for two cameras. This shift reduced hardware complexity, improved multitasking, and made the entire system easier to debug and maintain, shown through timing each function call in our program and seeing a dramatic reduction in time taken to execute, thus proving that this upgrade was worth it. This fulfills our goal of supporting two cameras while retaining all other functionality. We do not require a secondary controller due to our shield.

To integrate this new controller, we had to increase the size of the robot and test that all our external sensors were compatible first. Because this is a direct upgrade, there were no issues.



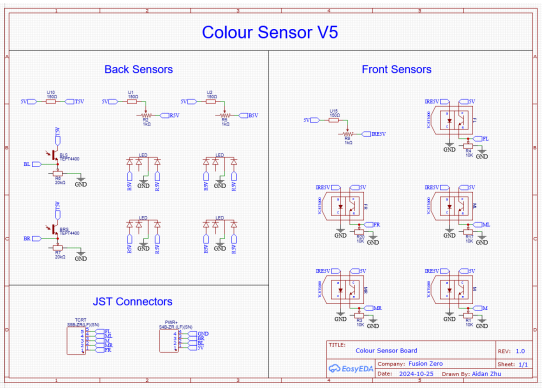
Custom Raspberry Pi Header Shield

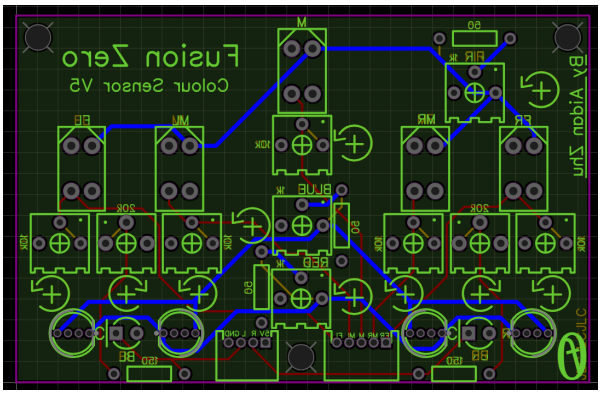


We designed a 4-layer custom shield to support both the Pi Zero 2W and Pi 4, avoiding the need to redesign the board when upgrading. This shield integrates servo control (PCA9685), MOSFET switching for COB LEDs (a voltage divider to handle 3.3V logic), and decoupling capacitors (1000µF and 0.1µF) for servo stability. All peripheral connectors were upgraded to JST headers for secure, vibration-resistant connections. Breakouts for I2C and GPIO are included for clean sensor integration. The result is a compact, modular hub that reduces wire clutter, improves reliability, and simplifies maintenance during development and competition.

This design has been verified in previous competitions (Japan, Singapore) and works as expected. Additionally, as the new controller has the same 40-pin GPIO headers, the integration with the Pi 4 is seamless.

Colour Sensors





The TCRT5000s reliably detect black/white transitions during ramp exits, obstacle avoidance, and stuck recovery. Despite our switch to camera line following, this board remains a valuable fallback system.

To integrate, we simply had to create the appropriate frame at the bottom of the robot, but slightly shifted back so the camera could have an unblocked view of the line as well.

Innovative designs

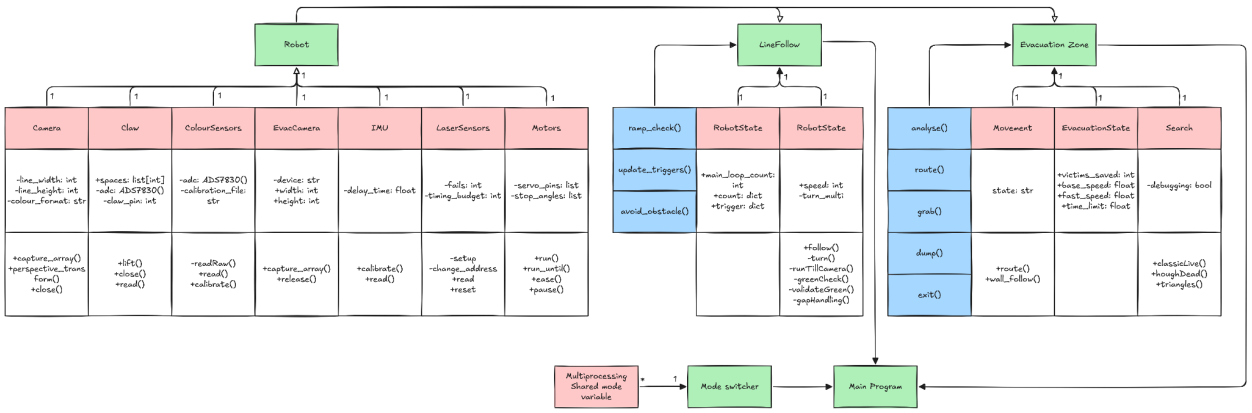
We consider our colour sensor board to be an innovative design as its simplicity is what allows it to be reliable. The high reading frequency combined with their purposeful placement allows it to capture the correct information. Using IR wavelengths makes it immune to changes in the natural light. We also consider our custom Raspberry Pi 40 pin GPIO header to be innovative, as it enables many features that we would otherwise lack without a secondary microcontroller. Finally, our smart choice of the Raspberry Pi 4 enables us to only use 2 x 18650s, allowing us to constrain our footprint as much as possible.

4 | Software

Our Raspberry Pi runs Python 3.9 with numpy, opencv, and multiprocessing as our main constructors of the program. The entire system runs through a single process, except for the mode switching, which is processed in another thread such that state interrupts such as switch toggles are detected. We use the specified library developed by adafruit for every

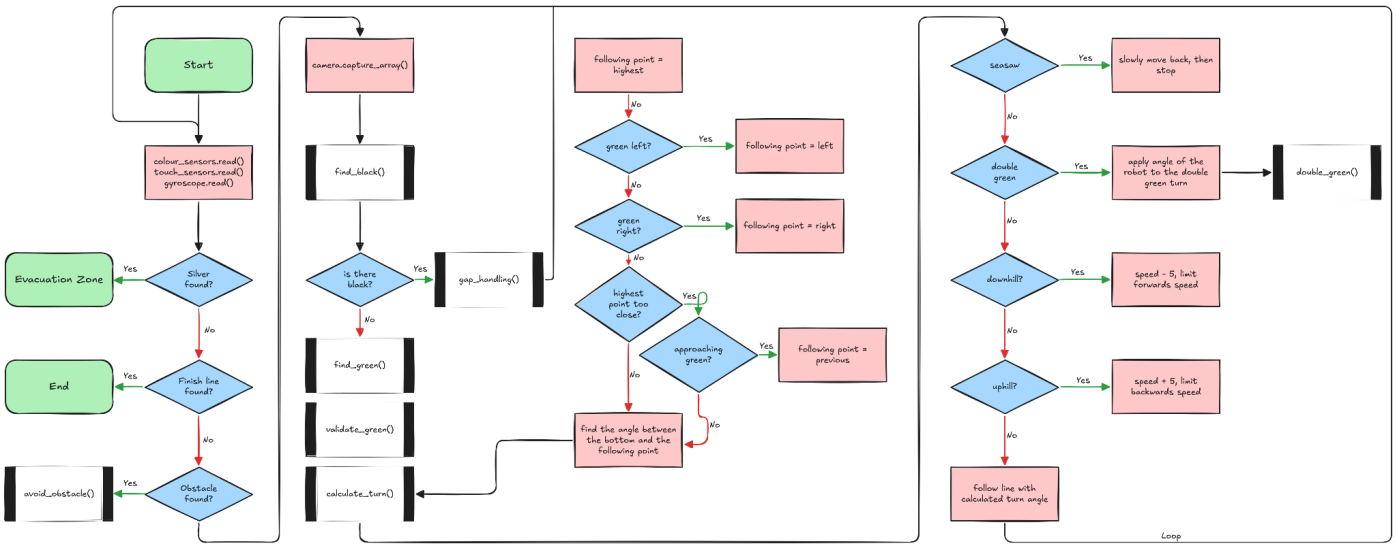
sensor we have, which abstracts the data handling from the hardware level. We use an object oriented programming paradigm as a scaffold, and combine it with an imperative style to yield the expected behaviours. To ensure the robot is predictable, many loops are governed by a closed-loop feedback system. We utilize our OLED screen and the console for debugging.

We use VSCode with the Remote-SSH extension to connect and run code within the robot over a wireless network. As backup, we have also set up the RPi to be an ethernet device that can be connected to over physical USB-C. Tools like ChatGPT, Claude 4.0, and Perplexity allowed us to research and develop our code at a much faster pace. Github acts as our versioning software, allowing each team member to work in parallel. Integration of code is seamless through this versioning, as well as our mode-selector which allows us to test various components before appending to the main program loop.



A | Line Follow

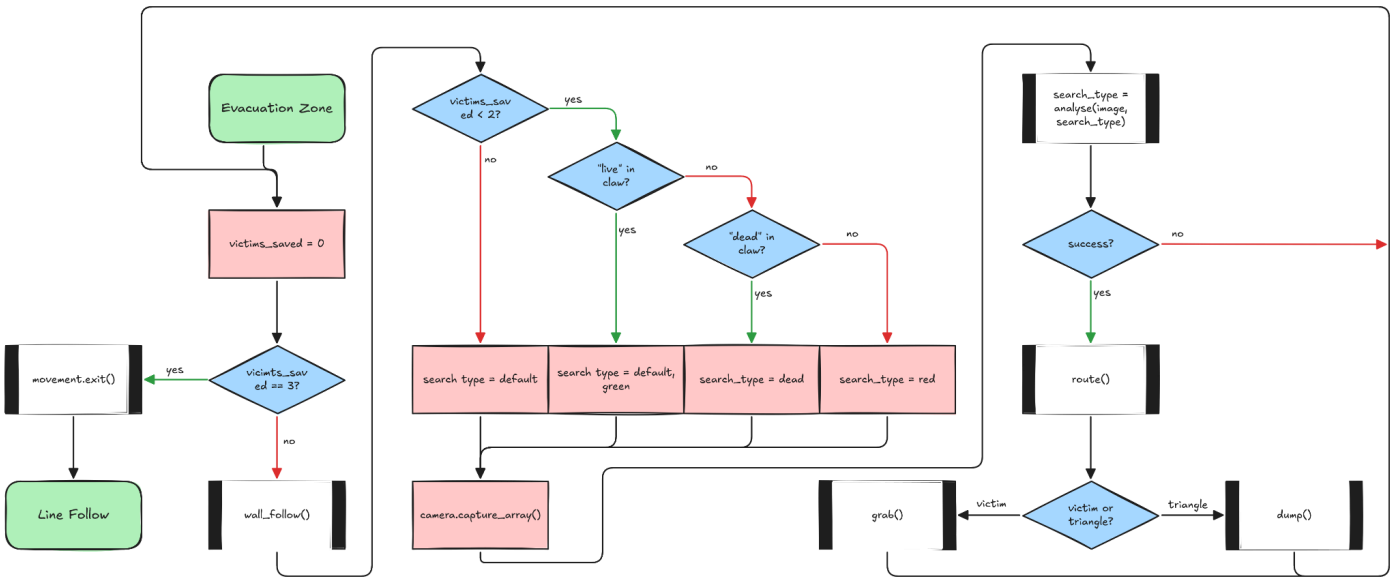
During the line-follow portion, our robot uses cv2 alongside all the sensor libraries. We perform computer visioning, finding black and green contours, and using our custom algorithm to determine the correct direction our robot needs to go. This process is controlled by a closed-loop feedback system with P (proportional gain) only.



We first take in all the sensor information (colour sensor, touch, gyroscope) to determine the best course of action, such as switching to evacuation_zone or avoiding an obstacle. If no external conditions are applied, then we have to be line following. By taking contours of black and green, we are able to calculate the angle between a virtual following point and the bottom of the camera image accounting for intersections. This virtual following point is set in different positions depending on the line and intersection. For testing and quality control, see testing. To integrate, we simply commit into the line_following branch in Github, occasionally creating PRs to get the new hardware updates.

B | Evacuation Zone

During the evacuation zone portion, our robot uses cv2 alongside all the sensor libraries. We perform computer visioning, finding all objects of interest, and routing to and performing the appropriate action with a closed-loop feedback system.



Our main evacuation loop simply follows the boundary of the evacuation zone. This strategy allows us to cover the entire playing field, as well as find different, more suitable angles for grabbing. Following, we select which objects are of interest based on the current number of victims saved and what the claw has. If an object is found, we attempt to route to it using a closed loop feedback system using the object's x position within the camera frame. Finally, when all victims have been found, we follow the wall and find the exit again.

To find these objects of interest, there are several key components necessary. For live victims, our entire algorithm exposes the repetitive setup and environment in any competition and at practice - silver coloured victims are often made from metal, and thus contains a reflective property, truly reflecting light. These create "spectral highlights" which are always apparent and visible to our camera. Instead of searching for victims, we can simply search for these reflections that only exist on live victims. For dead victims, there is great contrast between the black sphere and white background, and thus we can mask the original image for the dark pixels and the area surrounding it. Next, we are able to run a HoughCircle algorithm which allows us to locate the victim while checking for the circularity of any object. Finally, for the triangles, we can simply mask for vivid green and red colours with an HSV colour space, and then find the relevant contours and centre of mass. For testing and quality control, see Evacuation Zone. To integrate, we simply commit into the evacuation_zone branch in Github.

C | Innovative Solutions

Our line following algorithm also allows us to account for tilted or inclined boards with full line following capabilities. This is done through limiting the range our motors can run at, allowing us to maintain control at various rolls and pitches, while understanding how to turn without losing the line. Furthermore, the adaptive following point allows us to anticipate intersections and handle a greater variety of cases, which fulfills our overall goal of becoming more robust.

Due to previous hardware issues, we were forced to develop alternative ways to identify victims. We think our approach is much more efficient as we do not require active cooling and the Google TPU that would increase the robot's footprint and power draw. We exploit the predictable nature of the competition venue to our advantage, and thus we constrain to our small footprint goal. However we still acknowledge the reliability of the AI technique and endeavored to create a suitable model in case our approach does not work.

Finally, to identify the silver tape that marks the entrance to the evacuation zone, our clever sweeper design printed in blue TPU is reflected off the tape. What usually would not be in the camera frame is now visible, and thus we can perform a simple blue mask and contour check to identify if there is an entrance. Furthermore, as we still have retained our colour sensor board, we have extra verification. We reduce the complexity of this problem that previous teams required AI by relying on colours that are not present anywhere else within the competition.

5 | Performance Evaluation

A | Testing procedure

Remember that all development and evaluation is part of the iterative cycle: each problem identified always leads to a hardware or software solution. Our team has been able to develop two robots such that these two parts of the competition can be developed simultaneously, alongside hardware improvements.

Line Following

When testing line following, and other assorted components, it is clear that we needed variation in our board. Thus, using PVC foam boards, we created all our boards by hand. Using electrical tape and a craft knife allowed us to create black lines of varying widths. Using green electrical tape from different brands got different shades of green, cut using the craft knife. This meant that our tiles replicated the conditions found in competition. We modelled our own ramps, seesaw, obstacles in Onshape, and 3d printed them. Using some wood standoffs and white tape, we were able to create boards at higher elevations. We will evaluate performance based on a percentage success for each tile in various configurations, such that we could slowly categorise it into “easy” and “hard” tiles.

Stage 2 would focus on the basic line follow algorithm. This entails a simple, flat line following with what we classify as common tiles: easy tiles with little turns, but no intersections, resulting in a consistent line follower, verifying our ideas regarding camera line following. We used this stage to iterate on the appropriate camera angle, as well as the LED placement, quantity, and diffusion level. **Stage 3** introduced all variations (gaps, speed bumps, intersections, obstacles, tilts, inclines), replicating the expected level of difficulty. **Stage 4** introduced the international difficulty, as many rounds are full of debris and speed bumps placed at awkward angles, as well as freeform boards that we had not seen before in competition. This added difficulty allowed us to test various, new conditions, simulating a competition.

A problem that has plagued our team is our inability to overcome inclines and tilts due to poor configuration. Leading to the competition, we have tested 3 materials and 4 different wheel designs. This led to the best hardware change which reduced the complexity of line follow tenfold. As previously explained, we finally settled for silicone with the “wheg” wheel design with silicone. Another problem could be the transition between uphill - flat ground. The LEDs cannot counteract and light up an empty void, thus many pixels are considered “black”. Our solution to this was simply limiting the vision of our robot to not include this by manually cropping the image.

Evacuation Zone

By developing a jigsaw pattern, we were able to laser cut pieces that fit together and provided a transportable evacuation zone. The walls of the evacuation zone are fully 3d printed, and can be slid out, allowing for customizable entrance/exit combinations. The evacuation points are created with the PVC foam boards glued together. We have bought silver, reflective spheres off Aliexpress at diameters 50mm and 40mm. Furthermore, using Onshape and rice, we have created 45mm weighted spheres wrapped in tinfoil or electrical tape.

Stage 3 includes simply being able to identify, route, and grab or dump the object of interest. This forms the core evacuation loop, so it is important that this is done properly. By testing at different times of day, and with various ball weights, sizes, and triangle locations, we had simulated many common positions that could occur during competition. **Stage 4** introduces the complete routine, including entry and exiting into and out of the evacuation zone, as well as handling obscure cases such as obstacles being placed. Furthermore, the wall-follow algorithm is refined with sensor fusion with the IMU.

A problem we faced was verifying that we picked up a victim. Our claw would temporarily block the front camera's vision, and when it was lifted again, there would be no guarantee that the victim had not moved. Thus, we integrated TCRT5000s into the claw, reading into the remaining analog inputs on the ADS7830. This simple solution provided great value, as it also saved computational power as now we had confirmation on what tasks were remaining within the evacuation zone. Another problem we've consistently faced is the issue of wall-following. Due to our limited laser sensors on the side, trying to run a feedback loop with no true information, only a set value, leads to difficult navigation. This competition, we refined the algorithm by introducing a virtual pivot point which delayed the turning effect.

B | Full mock runs

Stage 5 tests the transitions between line following and evacuation zone. **Stage 6** will run full mock competitions, where we would build the previous years courses, run them a single time, and score them. As one team member works on assembling the next course, the other will practice our competition routine of creating a new branch and modifying the relevant code that the robot failed on. This trains our ability to work under pressure, generate new ideas with little to no testing environment. Furthermore, this allows us to accurately track how scoring and points work, and when we need to skip checkpoints depending on the time left.

6 | Conclusion

Fusion Zero is a team that prides itself in its extensive testing, small robot, and innovative hardware and software decisions. From custom wheels to exploiting predictable environments, our robot performs consistently and reliably, avoiding touches and deduction of points by playing slow and maximising our time to the 8 time limit.