[HARDWARE]

[GY-BN008X] (Missing CAD)
Axis IMU unit that accurately tracts roll, pitch, and yaw.

[ADS7830] (Missing CAD)
Analog signal to I2C converter that allows us to interface with analog sensors, namely the TCRT5000s for both the colour sensor board and for the claw.

[SG90]
Has a 180 degree range of motion, which allows us to control which victim we're releasing.

2 x [18650 4.2V Batteries]
SBC handling all coding, sensors, communication, camera, and logic flow.

[Coral Edge TPU]
Allows the Pi to inference quicker, enabling the option to use object detection models

[CUSTOM Raspberry Pi shield]
3st connections, PCA8685 for PWM. Exposes I2C ports, V+ and VCC.

[Raspberry Pi 4B]
4 Core, 4GB, SCB that holds the entire environement for development and for runs.

[DSSERVO RDS3218]
High performance 270 degree servo that lifts the entire claw mechanism

[TCRT5000]
Uses IR reflections to determine confirm which victim type we grabbed.

[LM2596] [MINI560 Pro]
Voltage regulators converting from 8.4V to 6V and 5V respectively for diffrerent components.

[Touch Plate, Limit Switches]
Physical digital actuators capture precise movement. Extremely responsive and clicky. Used to find obstacles

[4 x PDI-6221MG-360]
360 degree continuous servos with high torque allows for easy movement.

[Physical Sweeper]
Removes all debris from blocking the camera view. Made from TPU, and always touches the ground

[CUSTOM Colour Sensor Board]
A combination of TCRT5000s and visible light photo-resistors, carefully spaced to capture line information.

[VL53L1X]
Laser Time of Flight sensors, taking information from the front and both sides. Used in evacuation routing and obstacle.

[2303U USB2.0 Camera]
Pointing directly ahead for information regarding the evacuation state inside the evacuation zone.

[PiCamera Module 3 Wide Angle]
Angled downwards to capture slightly ahead of the robots center of rotation .

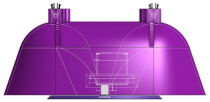
[CUSTOM Wheg wheel designs]
Wheels with cuts in them creates physical leverage, allowing for easier navigation of speed bumps.

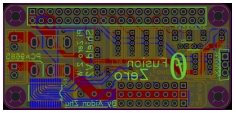
EasyEDA onshape

[CUSTOM claw]
Comprised of two large "cups" that entrap victims in specific places. There are left and right compartments, allow storage for two victims without increased robot size. However, a downside is that grabbing victims is no longer head on, but at an angle to the robot.

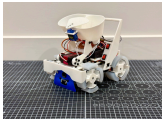
The cups are designed to lead the victim into a pre-determined location, and thus making the TCRT5000 sensor readings consistent for validation.

We can individually sort victims as well, as the compartments are not connected, allowing us to pick up both live and dead victims at the same time.



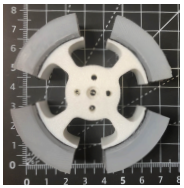


[Raspberry Pi Zero 2W Shield]
The Raspberry Pi Zero 2W only provides digital and I2C connections. Thus, we have a PCA8695 and ADS7830 handling communication for all components that do not have I2C communication. Furthermore, utilising XH connectors allow for much more stable connections compared to dupont.



[CUSTOM Wheg wheel designs]
Navigation around the entire course could be littered with speed bumps or debris. At a maximum height of 10mm, a large radius is necessary. However, placing speed bumps on inclines creates vibrations, hindering the grip the wheel has, and thus the friction it generates.

Thus, cuts along the circumference allows for "levers" to form, and hence physically grip past speed bumps, and eliminating vibrations.



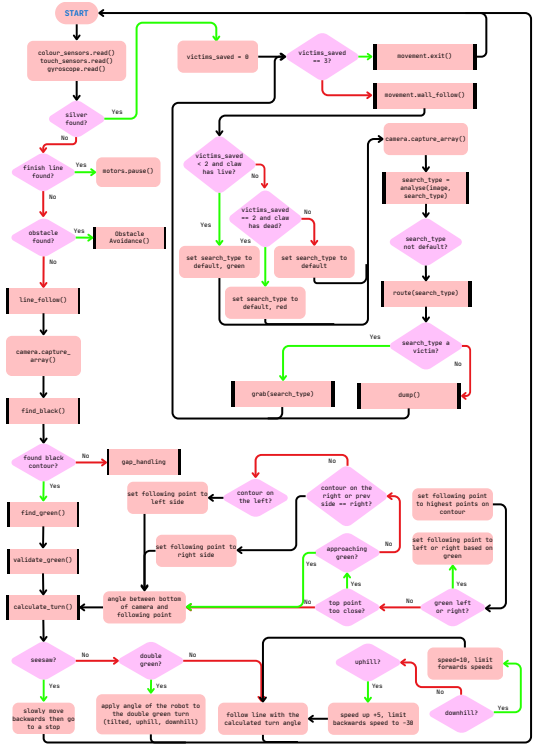
[TEAM]

League: Robocup Junior Rescue Line Region: New Zealand

[Frederick] Team Captain
Lead CAD designer
• 3D printing and assembly
Admin work & Github versioning
• Version control, document creation and submission
Programming:
• Victim identification and rescue
• Program control loop
• Sensor modules

[Aidan]
PCB Designer
• Custom shield and sensors boards
Lead programmer:
• Line follow + intersections
• Obstacle avoidance
• Gap detection
• Evacuation zone detection
• Exit detection
• Incline/Tilt detection

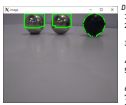
[Results]
[Regionals] 2018 Auckland Secondary
[Regionals] 2018 Auckland Premier
[Nationals] 2018 New Zealand Secondary
[Nationals] 2018 New Zealand Premier
[Nationals] 2019 Australia Open Tertiary
[Nationals] 2021 New Zealand Self Driving Car Senior
[Regionals] 2022 Auckland Premier
[Regionals] 2023 New Zealand Premier
[Nationals] 2023 Australia Open Tertiary
[Nationals] 2024 Singapore U19 International
[Nationals] 2024 Australia U19 International
[Nationals] 2025 Japan U19 International
[Nationals] 2025 Singapore U19 International



[Live Victim Identification]
We take advantage of the consistency of the playing field: there are always spectral highlights due to bright overhead lights, that always appear on the live victims. Thus, filtering for bright values from the camera quickly identifies live victims.

Live victim psuedocode
1.capture image
2.convert to grey-scale
3.filter for bright pixels
4.find contours
5.verify size, position
6.return x

Dead victim psuedocode
1.capture image
2.normalise white and green to grey
3.crop and dilate on black mask
4.find black contours
5.verify size, position, number
6.run hough circle transform
7.verify location
8.return x



[Dead Victim Identification]
In the past, filtering for the black victim was inconsistent at best: the noise from outside the course, especially considering the whiteness of the evacuation zone, created many dark spots as the robot faced the outside.

[Titled boards and Incline]
Our old robots used mercury-tilt switches that were nothing more than on-off contacts. They chattered on every bump, needed separate sensors for nose-up and nose-down. By replacing them with a single BNO080 IMU we now read yaw, pitch, and roll angles, giving us more information.

With continuous pitch data we set a simple threshold–about ±11 degrees–to decide when we are climbing or descending. The moment pitch rises past that limit the code increases base speed to clear speed bumps; when pitch drops below –11 degrees it softens the steering gains and slows the drive so the robot doesn't skid downhill. Crossing from positive to negative pitch within a second flags a seesaw, prompting a short pause until the board settles before moving on.

```
pitch = imu.get_pitch()
if pitch > +11:   speed = 30   # uphill
elif pitch < -11: speed = 15   # downhill
else:             speed = 25   # flat
if sign_flip(pitch,1):
    stop_motors()
    sleep(1)
```

Visual Studio Code
python

GitHub CAD Files Source Code Documents TDR

[SOFTWARE]