CATBOT JET

Rescue Line

Team Members

Zac McWilliam
Hardware: CAD, construction
Software: Camera Vision, version control, optimizations, other elements

Luna Verratti
Hardware: Research
Software: Main program structure, non-vision areas

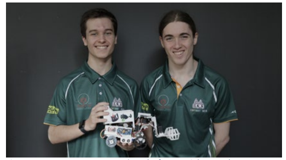
Figure 1 – CatBot JET team members (**Left:** Zac, **Right:** Luna)

Past Experiences

2016:	Melbourne Regional and Vic State Event	
2017:	Melbourne Regional and Vic State Event	
2018:	Melbourne Regional and Vic State Event	
2019:	Melbourne Regional Event	
-	Victoria State Event	
-	Australian National Event (Melbourne)	
2021:	Australian National Event (Online)	2nd
2022:	Melbourne Regional Event	3rd
-	Victoria State Event	1st
-	Australian National Event (Adelaide)	2nd
2023:	**International Event** (Bordeaux)	10th
-	**RoboCup Asia-Pacific** (South Korea)	2nd
2024:	**RoboCup Singapore Open**	???

Full AUS scoresheets and detailed results: https://rcja.app/rcj_cms/
Results prior to 2021 are unfortunately unavailable.


Figure 2 – 2022 Victoria State Event

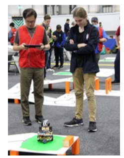
Figure 3 – 2022 Australian National Event – Prelim 5


Figure 4 – 2022 Australian National Event awards

CATBOT JET - 2016-2024

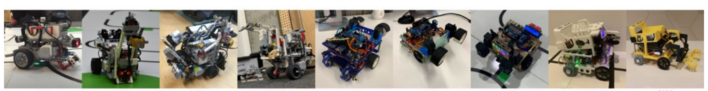

Figure 5 – History of CatBot designs from 2016 to 2024

Hardware

Carrying Handle
Provides convenient carrying ability and reduces damage to components

Gate System
Attached to an SG-90 servo, responsible for releasing victims in the evac zone

Includes side rails to help guide victims

NVIDIA Jetson Orin Nano + Adafruit Motor Shield
The main controllers, coordinates all aspects of the robot's operation, including motor control, sensor data processing, and decision making

The NVIDIA Jetson provides a platform with a GPU which allows the program to operate much faster and integrate AI Vision Processing for the evacuation zone

We found devices such as a Raspberry Pi not powerful enough to perform these tasks at an effective speed

CMPS14 Compass Module
We have included a CMPS14 compass module to provide us bearing data, this primarily assists with obstacle avoidance

Powered Omniwheels
Able to move freely side-to-side, allows 4-wheel drive with steering axis close to the front, which is crucial for vision following

Swappable LiPo Battery Holder
LiPo batteries are easily swappable by design, allowing us to work on the bot continuously whilst another battery charges.

Raspberry Pi Camera Module (Fig. 7)
Provides vision access for the Raspberry Pi, is mounted on a servo for adjusting the pitch when switching between following and victim finding

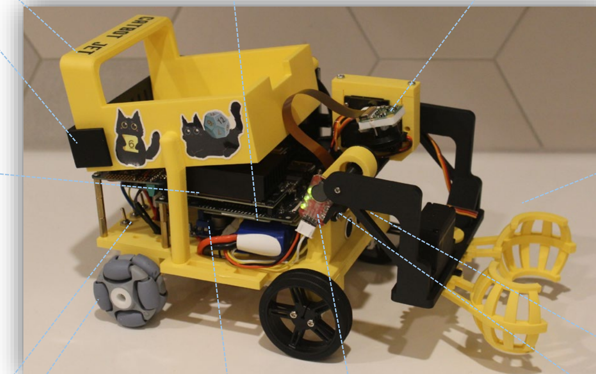
Figure 6 – Final assembled robot, right side view

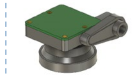
Figure 7 – Camera and lens mount

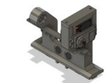
Figure 8 – Lift and cam mount

Rescue Mechanism (Fig. 8, 10)
CatBot JET's rescue mechanism includes a ball-shaped claw that is employed to grab and pull in balls during the rescue operation. The claw is carefully designed to ensure effective retrieval of balls, even when they are stuck in a corner.

Balls are located, grabbed, lifted, and then finally dropped into the cage on top of the robot. Ready to be released to the relevant rescue area when we are ready

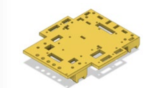
Figure 9 – Base plate (unassembled)


Figure 10 – Claw mechanism

Ultrasonic Sensor
An ultrasonic sensor is incorporated on the front that enables obstacle detection and avoidance.

Voltage Regulators
3 voltage regulators are used for power delivery to each major subsystem of the robot, the Jetson (5.2V), Motors (12V), and Servos (5V). Prevents crashes by reducing fluctuations in power level as each subsystem is independent

Voltage Monitoring/Protection
We have included multiple features to prevent excessive discharge of the battery. A display provides constant monitoring of battery voltage. Additionally, a battery alarm is attached which sounds a loud alert tone when the battery is approaching a dangerously low level. This system has proven highly effective on multiple occasions

Bright White LEDs
An array of LED strips is integrated on CatBot JET to provide sufficient illumination for the camera module. Ensures optimal visibility and accurate image processing in varied environments

Software

Programming Language: Python **Version Control:** GitHub
System: NVIDIA Jetson Orin Nano

__Main Program Order:__
1. Initialize all sensors, camera stream, calibration data, etc
2. Start sensor monitors in a separate thread
3. **Continuous program loop**:
   a) Check front ultrasonic distance for obstacle – **Yes?** Avoid **No?** Continue
   b) **Process camera vision data**
      - Checks every 5th frame for presence of red line to trigger stop program
      - Checks every 2nd frame for presence of silver foil to trigger evac
      - If neither of the above are true, process the image (handle intersections, green) into a line
      - Convert line into a steering value, input into PID program, then move
      - If a ramp is detected, the claw will slowly lower in order to move some mass forwards to assist in moving up the ramp

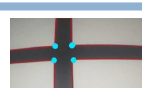
Figure 11 – Optimal 4WC Points

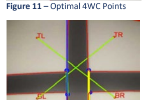

Figure 12 – 4WC intersection debug

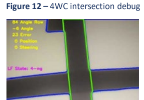
Figure 13 – Processed intersection

Basic 4 Way Intersection Processing Example

When approaching intersections, the goal is to remove all unnecessary parts of the line, so the line follower algorithm can follow it as if it were just a straight line

We need to find two (often vertical) lines on the image so that we can create a mask to get just the line to follow.

To do this, we find the midpoint of each white contour (Intersection on fig. 12), and then find the closest key points of each simplified white contour to get figure 11

The points in figure 11 can then be joined together into two lines, and a mask created to exclude anything on the image that is outside the two lines.

The final result is seen on figure 13, where the green contour is now the new input for the line follower.


Figure 14 – Identified victims in the rescue zone

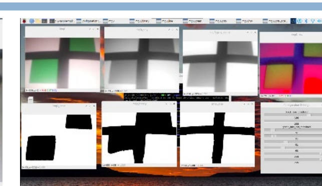
Figure 15 – Initial Image processing examples with calibration interface

Source Code, Journal, & TDP: Available on GitHub
https://github.com/zmcwilliam/catbot-jet/

Thanks everyone who offered support and encouragement along the way.
Especially to Yuma and Seb for their help with getting us started on vision processing

Full Circuit Diagram

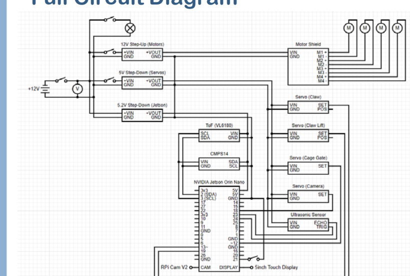
Figure 11 – Full circuit diagram for CatBot NEO, to be updated for JET