RoboCupJunior Rescue Line 2023

Technical Description Paper

CATBOT NEO

This version of CatBot NEO's Technical Description Paper is up to date as of ***Monday, 19th of June 2023***
All code, as well as other technical documents (Poster & TDP), can be found on our GitHub Repository

Abstract

- This paper describes CatBot NEO, a robot designed to compete in the International RoboCup Junior Rescue Line competition in 2023. By utilising camera vision with a Raspberry Pi and Open CV, CatBot NEO navigates through a course around intersections, green markers, obstacles, and debris, with the primary goal of reaching the Evacuation Zone, where it employs image processing techniques to detect and rescue ball-shaped victims. This paper presents a comprehensive overview of CatBot NEO's design, hardware, software, and capabilities. The paper also highlights the innovative solutions and approaches implemented by our team throughout the development process.

1. Introduction

a. Team

- The CatBot NEO team, formerly known as "CatBot", was established in Melbourne, Australia in 2016 as a part of the beginning of St Micheal's Grammar School's Robotics Club. The robot has since evolved over the years, having started with Lego Mindstorms, building up through Arduino, and finally to the current version, a Raspberry Pi based robot.
- The team consists of two members, Zac McWilliam, and Luna Verratti. Luna primarily handles the software side of the robot, and Zac contributes to both the hardware and software.
- The team has competed in approximately 11 events in Australia: *(no results data available prior to 2021)*

2016 Melbourne Regional and Victoria State Event		
2017 Melbourne Regional and Victoria State Event		
2018 Melbourne Regional and Victoria State event		
2019 Melbourne Regional event		
2019 Victoria State event		
2019 Australia National event (Melbourne)		
2021 Australia National event (Online)	**2nd Place**	(Scoresheet Link)
2022 Melbourne Regional event	**3rd Place**	(Scoresheet Link)
2022 Victoria State event	**1st Place**	(Scoresheet Link)
2022 Australia National event (Adelaide)	**2nd Place**	(Scoresheet Link)

- Members regularly meet in person and over online platforms such as Discord to work on the robot
- Results history is available on the RCJA Scoring System, of which the development team is led by Zac

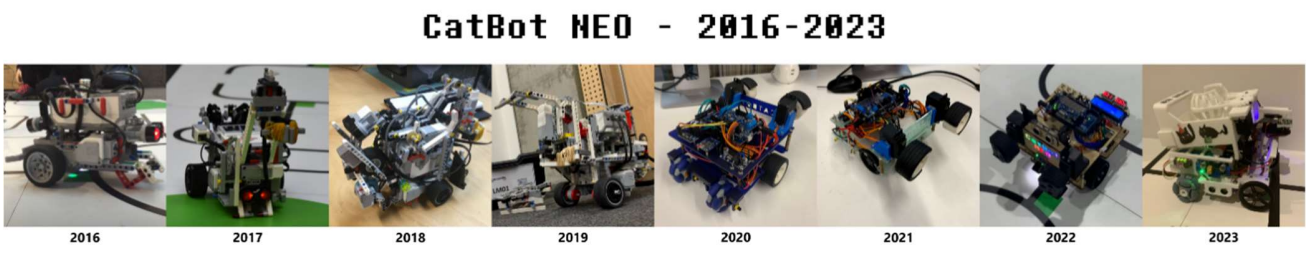

Figure 1 *- A history of CatBot NEO from 2016 to 2023*

2. Project Planning

a. Overall Project Plan

- Our aim for the RoboCup Junior Rescue Line competition is to be able to showcase our team's skills, creativity, and problem-solving abilities in designing and constructing a robot capable of effectively navigating the rescue line course, simulating real-life scenarios where autonomous robots can assist in rescue operations. By participating in the competition, we strive learn along the way by doing the best we can in the short time that we have.

- As outlined in table 1, our project plan includes several milestones and key tasks to ensure a systematic and efficient development process. The major milestones involve designing the robot, ordering parts, assembling a prototype, programming line following, programming the evacuation zone, and finally, completing tests and fixing any problems as they occur.

Table 1 – *Initial Project Plan*

April	May	June	July
Investigate the new competition List the requirements Estimate cost and order all parts and tools Design CAD model	Assemble first prototype Create test code for each subsystem Mostly complete line following Basic obstacle avoidance	Implement intersections/green turns Evacuation Zone Finish off and optimize code Lots of testing and fixing	Travel to Bordeaux Any finishing touches right before the comp

- Analysing the task and its constraints played a crucial role in shaping the project plan, by studying the competition guidelines, rules, and constraints, we could further our understanding to the specific requirements that the competition involved so that efforts could be focused on developing key functionalities such as robust line-following algorithms, obstacle detection, and reliable victim detection and rescue strategies.
    a. One of the major differences between the RCJA and RCJI competitions was the Evacuation Zone. As we have never attempted a similar rescue before, it was crucial to dedicate enough time to plan ahead and investigate possible solutions.
    b. The other key difference was the increase in ramp gradient from 17 to 25 degrees. This puts constraints on the robot design, and we had to consider solutions specifically relating to this
- To further our analysis and enhance our development process, we investigated our past performance in previous competitions, primarily the RCJA 2022 Australian National Event. This self-assessment allowed us to identify areas for improvement and incorporate lessons learned into our project plan. Additionally, we researched solutions and discussed different approaches.

b. Integration Plan

- CatBot NEO's integration plan revolves around effectively connecting and coordinating each major component and the bot's physical structure to help achieve seamless functionality.
  The key components involved are as follows:
  a. **Physical Structure:** CatBot NEO's physical structure, (designed and 3D printed), provides the foundation for assembling the hardware components. It ensures proper positioning and secure mounting of the Raspberry Pi, motor driver board, camera module, and other critical components. *A detailed model can be found in Figure 1.* Techniques to allow for a modular assembly have been utilised, providing an easy way to prototype and make improvements or repairs without needing to completely reconstruct the entire robot.
  b. **Electrical Connections and Power Management:** Establishing reliable electrical connections between the components is critical. As evident in Figure 2, proper wiring techniques and cable management are employed to ensure secure and efficient electrical connections within the physical structure. The power system, which includes the LiPo battery, several voltage regulators, and a distribution structure, is carefully integrated to provide consistent and stable power to each component to ensure continued reliable operation.
  c. **Software and Firmware Integration:** The software running on the Raspberry Pi interacts with the motor driver board, camera module, and other sensors/servos, ensuring coordinated operation and responsiveness. Standard communication protocols such as i2c are integrated to be able to communicate to and receive data from components.
  d. **Calibration:** Scripts are included in order to run calibration procedures; this allows the robot to cope with varying environments and lighting conditions.
  e. **Testing, Debugging, and Documentation:** Extensive testing and debugging is performed to ensure the successful operation of the robot. Systems such as camera functionality and motor control are fine-tuned to ensure optimal performance. Comprehensive documentation is included throughout the source code to aid future maintenance and development efforts, as well as provide an opportunity for others to learn from our findings.

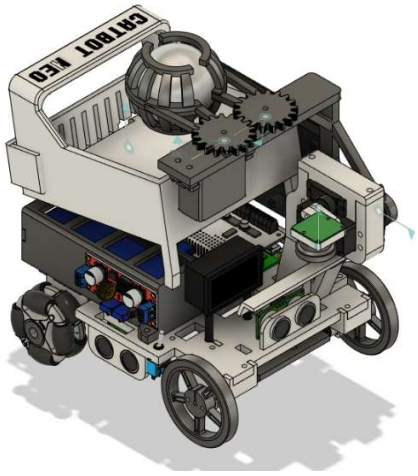

***Figure 1** – Full assembly of CAD model*

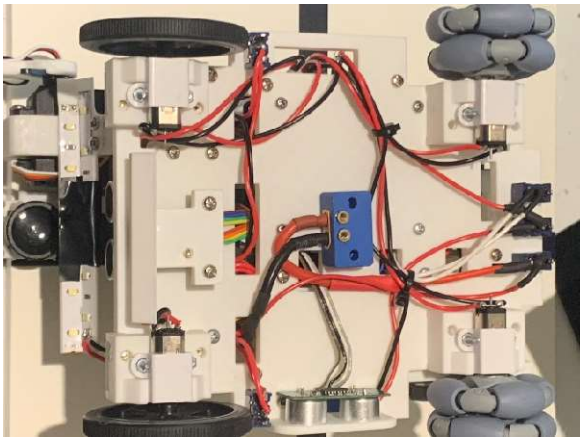

***Figure 2** – Electrical power routing*

3. Hardware

- CatBot NEO features a well-designed hardware system that supports its functionality in the International RoboCup Junior Rescue Line competition. The hardware components work together to achieve precise control of the robot, effective navigation around the course, and efficient rescue operations. The modular design and careful consideration of key mechanisms enhance the robot's versatility and ease of maintenance.

a. Mechanical Design and Manufacturing

- The main structure of CatBot NEO has been designed with modularity in mind (Figure 3). An example of this modularity is the variable-sized wheel brackets (Figure 4), which provide easy adjustment to different wheel sizes by simply allowing a new mount to be printed with a different height and immediately swapped out. This flexibility proved invaluable during the development phase, enabling testing with various wheel assemblies to determine the optimal configuration.
- Additionally, parts such as the cage, claw, and camera are easily removable from the robot base (Figure 5), facilitating easier access for repairs or adjustments. This design consideration streamlines maintenance procedures and ensures efficient modification of specific components when necessary.

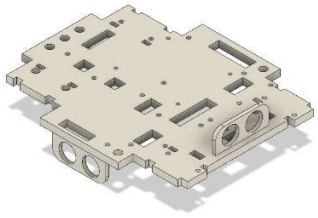

Figure 3 – Modular base provides assembly holes for all components

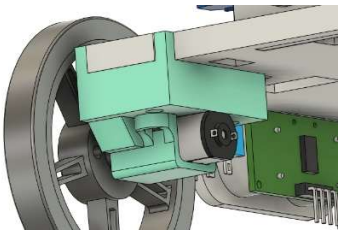

Figure 4 – Modular wheel mount

Figure 5 – Cage and Claw Removed

- **Actuators and Power Train**: CatBot NEO utilises four 380:1 micro-metal-gear motors, controlled by an Adafruit motor driver board which act as the primary actuation system for the robot's movement. These motors were carefully selected for their torque and size, a step up from what we previously used on CatBot in RCJA competitions. They are securely mounted within the robot's structure, ensuring stability and effective transmission of motion.

  In addition to the motors, four servos are incorporated for controlling specific mechanical functions:
  - ***Camera Servo:*** (Figure 6/8) A small SG-90 servo controls the camera's pitch movement, allowing it to switch between line detection and victim location quickly
  - ***Claw Servo:*** (Figure 7) The claw mechanism is also actuated by a large servo motor. This servo controls the opening/closing of the claw, allowing the robot to grip and release victims
  - ***Lift Arm Servo:*** (Figure 8) A large servo motor operates the lift arm mechanism, which is used for raising and lowering the rescue mechanism. This servo enables CatBot NEO to position the claw at the appropriate height for engaging with victims or the rescue kit
  - ***Back Gate Servo:*** The final small servo is featured at the back of the robot to control the gate mechanism; this is responsible for releasing victims as part of the evacuation program

  These servos are securely integrated into CatBot NEO's mechanical structure, allowing for proper alignment and secure attachment. They are controlled by software running on the Raspberry Pi.

  When selecting these servos, an important consideration was to choose metal geared servos instead of plastic geared. These have several advantages, including:
  - **Increased Durability:** Metal gears are known for their enhanced durability and resistance to wear, and tear compared to plastic gears, this is crucial due to the nature of the competition
  - **Improved Precision:** Metal gears provide better accuracy and precision, this ensures that CatBot NEO can perform delicate tasks such as accurately adjusting the camera angle

- **Rescue Mechanism:** CatBot NEO's rescue mechanism includes a ball-shaped claw that is employed to grab and pull in balls during the rescue operation (Figure 7). The claw is carefully designed to ensure effective retrieval of balls, even when they are stuck in a corner. It also provides the ability to pick up the blue rescue kit, if required. The claw mechanism is designed to be completely removable, allowing for easy adjustments or replacement if required. Balls are lifted and then dropped into the cage on top of the robot (Figure 1), ready to be released when required. Precise measurements in CAD were used to ensure the balls would reach the desired location without any parts colliding with each other. A gate at the back of the robot can be opened to facilitate the controlled release of the rescue victims and blue rescue kit.

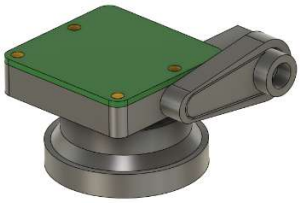

Figure 6 – Camera and Lens Mount

Figure 7 – Claw and Lift Assembly

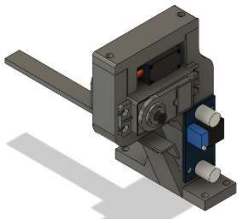

Figure 8 – Lift and camera mount

b. Electronic Design and Manufacturing

- The electronic design of CatBot NEO plays a crucial role in establishing its functions and ensuring reliable performance.

- **Main Controller:** The main controller of CatBot NEO is a Raspberry Pi 4B, it controls and coordinates all aspects of the robot's operation, including motor control, sensor data processing, and decision making. Its processing power and flexibility make it well-suited for this application.

- **Adafruit Motor Driver Board**: Connected to the Raspberry Pi, the Adafruit motor driver board enables precise control of the 380:1 micro-metal gear motors. It provides the necessary power distribution and control signals to manoeuvre the robot.

- **Sensors:** CatBot NEO incorporates various sensors to gather essential data and enable effective navigation and interaction with its environment. These sensors include:
    - **Raspberry Pi Camera Module:** The camera module is connected to the Raspberry Pi and serves as the visual input for line processing and other vision-related tasks. It captures real-time images that are processed to determine the position of the black line, detect markers, and identify victims.
    - **Ultrasonic Sensors**: CatBot NEO incorporates two ultrasonic sensors that enable obstacle detection. These sensors emit sound waves and measure the time it takes for the waves to bounce back after hitting an object. This information is used to detect and avoid obstacles during navigation.

- **White LED strips**: An array of white LEDs is integrated on CatBot NEO to provide sufficient illumination for the camera module. This ensures optimal visibility and accurate image processing even in low-light environments.

- **Power Subsystem:** The power subsystem of CatBot NEO is designed to provide a stable and reliable power supply to all the electronic components. It includes:
    - LiPo Battery: This was selected for its high energy density and capacity to deliver the required power for prolonged operation. Safety measures are implemented to ensure proper handling and charging of the battery.
    - Voltage Regulators: 3 voltage regulators are used to provide power delivery to each major subsystem of the robot, the Raspberry Pi (5.2V), Motors (10V), and Servos (5V). By delivering power to these systems separately, the load on each will be mostly independent and reduce fluctuations in power level due to actions of the robot, preventing crashes.
    - Careful considerations were made, building off previous experience, to ensure optimal performance and safety of the design. This included planning ahead with cable management

- **Voltage Monitoring and Protection:** CatBot NEO incorporates several features for voltage monitoring and protection to ensure safe and reliable operation. These include:
    - A voltmeter and ammeter display: This display provides constant monitoring of the battery usage, allowing the team to keep track of the real-time battery voltage and current draw during operation. This monitoring helps ensure the battery is within safe operating limits and allows the team to make informed decisions about the robot's power consumption
    - Battery Alarm: To prevent excessive discharge of the battery, the robot features a battery alarm system. In the event that the battery voltage drops below a specified threshold (~10.4v), this alarm, which is installed on the side of the robot, produces a loud and distinct sound to alert the team that the battery is nearing a dangerously low level. This system has proven to be extremely effective in preventing battery damage and ensuring the team promptly turns off the robot to avoid further discharge. This can be seen in Figure 9

Figure 9 – Battery alarm

Circuit Diagrams: To aid in the design and documentation of the electronic system, a circuit diagram was created to provide a visual representation of the connections and components on CatBot NEO.

Figure 10 showcases an example circuit diagram of CatBot's electronic system, illustrating the connections between the Raspberry Pi, sensors, motor driver board, and other electronic components. The circuit diagram serves as a guide for the assembly process and provides a reference for troubleshooting and maintenance.

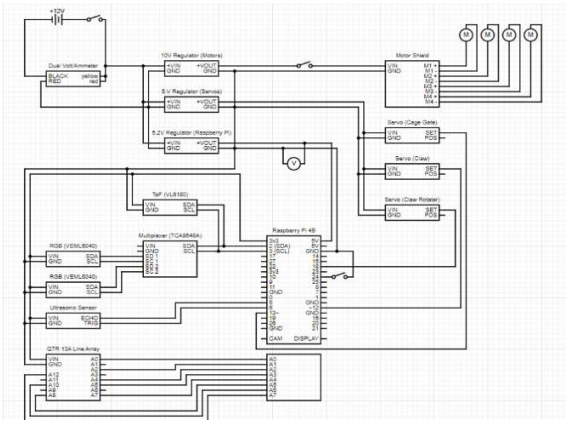

Figure 10 – Full circuit diagram of CatBot as of 24/4/2023

- The manufacturing process of CatBot NEO involved the assembly and integration of each electronic component into the robot's structure. This includes careful wiring and connection of each sensor, the Raspberry Pi, motor driver board, and other peripherals. Attention is given to proper cable management, secure mounting of components, and ensuring electrical connections are reliable.

- The manufacturing process also included testing of each sensor and module to verify the functionality of the electronic components, such as by using sensor readings, or a multimeter. These procedures helped to identify and rectify any issues or inconsistencies to ensure the reliability and performance of CatBot NEO.

4. Software

- CatBot NEO's software is designed to ensure efficient line following, precise control, and effective victim detection. The software architecture is based on a primary Python program, **follower.py**, which serves as the main entry point and orchestrates the entire operation of the robot.

a. General software architecture

- The software architecture consists of several key components and helper files, working together to achieve the desired functionality. Here's an overview of the main components:
    - **runner.py:** This file enables easy start and stop of the main program by watching a switch on the robot. It eliminates the need for a direct connection to a laptop, providing convenience during operation. This file is automatically started on boot.
    - **follower.py**: This primary program launches the entire software stack and initiates the line following process. It serves as the central control hub for CatBot NEO's operations.
        - At the top of this file, various configuration settings for sensor ports and default thresholds are included. These provide easy customisation and adjustment of parameters as needed.
        - **Servo Control:** By utilising GPIOZero's AngularServo class to control the servos, we can achieve very precise control of each servo.
            1. Custom min_pulse_width, max_pulse_width, and initial_angle values are defined based on manual calibration to achieve desired behaviour
        - **Sensor Monitoring**: A custom monitoring class is implemented to monitor sensors in a separate process, the main loop can continue while data is collected concurrently.
    - **Helper Files:** Additional helper files, such as **helper_motorkit.py** and **helper_camera.py,** aid in streamlined code execution.
        - **helper_motorkit.py** provides wrapper functions around Adafruit's MotorKit library, simplifying motor control and increasing code readability.
        - **Helper_camera.py** launches a camera monitoring instance, creating a video stream that provides constantly updated frames for the main program to access. It also includes pre-processing steps to generate useful versions of the image, such as grayscaled, binary, and HSV versions, based on calibrated parameters.
    - **Calibration Files:** When calibration is required, files such as **calibrate_white.py** and **calibrate_cam.py** can be used to fine-tune calibration data for the robot. These files allow adjustments to be made when lighting conditions change to ensure optimal performance.
        - Calibration data is stored in JSON files and accessed by the main program
- The camera processing in CatBot NEO follows a series of steps to extract relevant information efficiently for line following. The main stages include:
    1. **Image Processing:** The captured camera image is processed to generate masks for the line and green marks. Contours for white and black regions are also identified. This step prepares the image for subsequent processing stages.
    2. **Green Marker Detection:** This part of the software checks for the presence of green markers in the image, this image is used to make decisions on how to change the image
    3. **Intersection Handling:** Handling intersections with just black lines is an added challenge for camera-vision following, this involves converting the image to optimise the line handling after encountering an intersection. For example, a four-way intersection can be converted into what is perceived as a straight line after the intersection code is executed.
    4. **Line Following:** The processed image is further converted into a steering and error value, which can then be used as an input for the PID algorithm for precise line following.
- The software code is extensively commented, and docstrings are provided for most functions when useful. This promotes code readability, ease of maintenance, and efficient collaboration and sharing

5. Performance evaluation

- CatBot NEO's performance is to be thoroughly evaluated through a series of tests to assess its line following capabilities, obstacle avoidance, and victim detection, as well as many other operational aspects. The evaluation aims to measure the robot's efficiency, accuracy, and reliability in completing the rescue mission tasks.

- To evaluate the performance of CatBot, the following testing procedures can be implemented:
    - **Line Following Test:** CatBot should be tested on various line configurations, including straight lines, curves, intersections, and challenging scenarios such as ramps. The robot's ability to accurately follow the line and navigate through intersections should be assessed, and results used to further improve each algorithm.
    - **Obstacle Avoidance Test:** By continuously placing an obstacle in front of CatBot, with many varying shapes and sizes, the performance and obstacle avoidance ability can be tested as well as its ability to reconnect to the line effectively.
    - **Victim Detection Test:** By placing the robot inside the evacuation zone, several tests can be conducted to evaluate how well CatBot is able to find the victims, approach, grab, and successfully complete the rescue challenge.
    - **Competition Performance:** After competing in the competition, CatBot's final performance can be evaluated based on its ranking and scores received from different tasks. The competition will provide valuable opportunities to evaluate the robot's performance in a competitive environment and benchmark it against other teams. Results from this can then be used to provide further evaluation on what could be done differently if this challenge is to be attempted again.
- By completing these tests, specific areas for optimization can be identified, such as fine-tuning the line detection algorithm, improving object detection thresholds, and improving victim recognition algorithms. These evaluations would also influence the team's decision-making process for prioritizing enhancements and focusing on areas that would yield the greatest improvements.

6. Conclusion

- In conclusion, this version of CatBot NEO represents a significant advancement in our team's participation in the RoboCup Junior Rescue Line competition. Through the integration of camera vision, advanced algorithms, and a well-designed hardware system, CatBot NEO demonstrates its ability to navigate complex rescue line courses, detect victims in an evacuation zone, and successfully rescue them. The development process has involved iterative design, testing, and optimisation to meet the requirements of the competition and achieve optimal performance.
- The hardware design of CatBot NEO incorporates a modular structure that allows for easy assembly, maintenance, and adjustment. The integration of high-torque micro-metal gear motors, servos for precise control, and a robust power subsystem ensures reliable and efficient operation of the robot. Furthermore, the inclusion of sensors such as ultrasonic sensors and a Raspberry Pi camera module enables effective obstacle detection, line following, and victim identification.
- The software architecture of CatBot NEO is built around a primary Python program, which orchestrates the robot's operation and coordinates its various components. Through careful image processing, line detection, and intersection handling algorithms, CatBot NEO can achieve accurate line following and effective navigation. The software also includes calibration procedures and comprehensive monitoring to ensure optimal performance and responsiveness.
- CatBot NEO represents the culmination of our team's dedication, hard work, and passion for robotics. We are proud of our achievements and look forward to the exciting challenges and opportunities that lie ahead.

Appendix

- CatBot NEO's project repository is publicly available on GitHub, providing access to the complete source code, documentation, and some additional resources related to the robot's design. The repository serves as a comprehensive reference for anyone interested in exploring and understanding the inner workings of CatBot NEO. Each file includes internal documentation where possible.

  The repository can be accessed at https://github.com/ZMcWilliam/catbot-rcji/