RoboCup Nationals 2026
Team Description Paper

League Name:	Rescue Line
Age Group:	Under 19
Team Name:	System Override
Team Website:	https:/G/teamsystemoverride.godaddysites.com/
Participants And Technical Roles:	**Ayaan Suri:** Team Lead, Hardware, Programmer, 3D modelling, Electronics, and Documentation. **Shaurya Mahajan:** Programmer,Data Collector, and Researcher. **Tanash Garg:** Programmer, Electronics, Media, Documentation, and Researcher.
Team Photo:	
Mentor Name:	Ganga Bhuvaneswari Subramanian
Institution:	Shiv Nadar School Noida
Region:	India
Contact Person:	Ganga Subramanian
Contact Email:	ganga@sns.edu.in
Date:	10th January 2026

1. Abstract

Our robot is engineered to efficiently tackle the challenges of the RoboCup Junior Rescue Line competition. It is built around an **open EV3 brick** that has been enhanced with **external buttons** for improved ergonomics and modified by removing excess plastic to reduce weight. The robot employs **tracks** for superior traction on inclined surfaces and integrates **2 time-of-flight** (**TOF**) **laser sensor** for precise obstacle and exit detection. To extend the EV3's capabilities, **two Arduino Nanos** are connected via **I2C**, enabling the use of multiple sensors and actuators. An **OpenMV camera** running a custom **machine-learning model** identifies and differentiates victims using a dataset of over **5,000 self-labelled images, trained with Edge Impulse**, which greatly reduces search time in the rescue area. This multi-controller architecture (EV3, Arduino Nano, and OpenMV) combines the strengths of each system to overcome hardware limitations, resulting in a reliable, versatile robot optimized for speed and accuracy in completing the rescue mission.

1. Introduction

1.1 Team

About our team:

Our robotics journey began in 6th grade through EV3 classes, igniting a passion that drove us to form a team and design autonomous robots. Along the way, we've achieved **2nd place** at RoboCup Nationals India in **2024 and 2025**, secured **1st place** in the primary category at **RCAP 2023** in South Korea, the secondary category at **RCAP 2024** in Qingdao, and 1st place at the **Singapore Open** in April 2025.

Our team has three members:

Ayaan Suri: Ayaan, our **team lead**, has been competing in RoboCup since 2022 and is responsible for the robot's **hardware** and managing its **electronics** with Arduino and an OpenMV camera. He has also done **3D modeling** for making custom parts, EV3, managed team **documentation**, and has soldered all the sensors. He has made the external EV3 buttons.

Shaurya Mahajan: Shaurya has been competing in RoboCup since 2022 and is one of the **programmers**. Beyond his programming expertise, he developed the **line following logic**, created the core programming framework, and programmed two key microcontrollers, EV3 and Arduino. He has also helped in **documentation**.

Tanash Garg: Tanash, joined our team last year, bringing expertise in programming, especially Python. As one of the team's **programmers**, he is responsible for coding the OpenMV camera. In addition, Tanash manages the team's **media** presence, has helped with **documentation**, and leads **research** on new innovations for the robot, including exploring and evaluating new sensors.

2. Project Planning

2.1 Overall Project Plan

2.1.1 Team Objective

Our team's primary goal is to design and operate a **fully autonomous robot** capable of completing the RoboCup Junior Rescue Line challenge with maximum reliability, speed, and accuracy. To achieve this, we aim to meet the following requirements:

- **Compliance with Competition Rules:** Ensure the robot adheres to all RoboCup Junior Rescue Line regulations regarding **dimensions**, **AI rules**, and **allowed components**.
- **Efficient Navigation:** Build a robot able to follow the rescue line, handle intersections, ramps, and debris while maintaining stable movement within time constraints.
- **Victim Detection and Rescue:** Utilise advanced sensors and a machine-learning model to accurately identify, locate, and respond to victims within the evacuation zone, such as the **IR TOF** and **OpenMV Camera**.
- **Hardware Integration:** Combine EV3, Arduino Nano, and OpenMV systems to maximize sensor and actuator functionality while maintaining **strong communication**.
- **Reliability and Speed:** Test extensively to get consistent performance under competition conditions like **lighting**, **different types of fields**, etc.
- **Resource Optimization:** Design within the limits of available materials, budget, and battery capacity while ensuring it's a **compact robot**.

2.1.2 Project Plan

Our team began work on **November 28th** and aims to complete all development and testing by **January 8th**. The project is divided into 8 phases, each with responsibilities and review gates to ensure steady progress.

- **Phase 1: Planning, Research, and Design** (**Nov 28– Nov 30**)
  - **Planning:** Review competition rules, define objectives, and research suitable sensors and controllers.
  - **Research:** Start researching on **sensors** and **microcontrollers** to be used in the robot.
  - **Design Thinking:** Finalize the robot's design according to the requirements.
- **Phase 2: Lego Construction** (**Dec 1 - Dec 4**)
  - **Construction:** Complete the robot's **LEGO** construction by 18th September, including chassis, tracks, arm, handle, etc.
- **Phase 3: Lego Programming** (**Dec 5 - Dec 6**)
  - **Basic programming:** Complete basic LEGO programming, including line, greens, intersection logic, etc.
- **Phase 4: Arduino and OpenMV Integration** (**Dec 8 - Dec 10**)
  - **Arduino Connection:** Connect two Arduino Nanos via digital communication and connect a I2C Multiplexer for I2C sensors.
- **Phase 5: Data collection and testing Arduino** (**Dec 12 - Dec 21**)
  - **Data Collection:** Collect and **manually label over 5000 victim images**, train a model on Edge Impulse, and integrate it into OpenMV.
  - **Arduino Connection:** Add sensors to the Arduino and connect the I2C sensors to the I2C multiplexer, then verify the setup by running the code on the robot.
- **Phase 6: Full System Integration & Testing** (**Dec 22 - Dec 30**)
  - Combine EV3, Arduino, and OpenMV, run complete rescue line tests with ramps, obstacles, seesaws, debris, speed bumps, and an evacuation zone.
- **Phase 7: Fine-tuning and Readiness** (**Jan 5 - Jan 6**)
  - **Debugging:** Fine-tune Intersections, values, and reliability for competition conditions.
- **Phase 8: Testing and Building Spare Parts**(**Jan 7 - Jan 8**)

- **Testing:** Start combined testing of all 3 microcontrollers with the line and the evacuation zone.
- **Start building spare parts for competition:** 3d print claws and solder spare sensors

2.1.3 Planning and Scheduling Process

- Collaborative Schedule: We began on November 28th, holding team meetings to divide tasks (hardware, software, data collection) and work on Documentation
- Task Dependencies: We first started working on the **chassis** to plan the positions of the motors and color sensors. Then we developed the **basic software**, which included line following, gap, intersections, and ramp.
- Milestone Sequence: Building the base robot first ensured stable hardware before complex coding, reducing rework and improving calibration accuracy.
- Testing at Each Stage:
  - **Mechanical** – traction on ramps and grip, take values of black, green, white, silver, and red for the color sensors.
  - **Sensors** – Check Time-of-flight accuracy, Down Ultrasonic values, Arduino to Arduino connection using **digital communication,** and **I2C** connection with EV3.
  - **OpenMV** – Check for victim detection accuracy and lighting tolerance.
  - **System integration** – Make full rescue line runs with an evacuation zone to validate the performance.
- Adjustments: After each phase, we reviewed results, fixed weaknesses (hardware and software), and reallocated time to stay on track for November 8.

2.2 Integration plan

- **Hardware Approach:** Each microcontroller (EV3, two Arduino Nanos, OpenMV camera) was built and tested individually, then combined step-by-step into **one system**.
- **Actuators and sensors used:**
  - **Large Motors:** Our robot's tracks are driven by two EV3 large motors.
  - **Medium Motors:** We used two EV3 medium motors to control the arm.
  - **Colour Sensors:** Two EV3 color sensors are equipped for proportional line-following.
  - **Mindsensors Gyro:** We equipped the robot with a Mindsensors gyro sensor for precise ramp detection.
  - **Analog IR:** Our robot uses an analog infrared sensor to identify the silver line on the field.
  - **Down Ultrasonic:** A downward-facing ultrasonic sensor helps detect ramps.
  - **IR TOF:** We have utilized two infrared ToF for exit detection in the evacuation zone and obstacle detection..
  - **Digital IR:** Digital IR verifies whether the robot has successfully picked up a live victim.
  - **LDR:** A light-dependent resistor is used to ensure the robot has successfully picked up a dead victim.
  - **OpenMV Camera:** Our robot's OpenMV camera uses a self-labeled image dataset to detect victims and evacuation points.

- **Software & Communication:**
  - The EV3, which is coded in **blocks**. The EV3, which functions as the master controller, communicates with the first Arduino Nano over I2C. Arduino Nano is coded in **C++**. The first Nano exchanges data with the second Nano via Digital communication , while the second Nano manages all I2C sensors through an I2C multiplexer. Communication between the second Nano and the multiplexer also uses I2C. The first Nano connects to the OpenMV camera using UART. OpenMV Camera is coded in **MicroPython**.

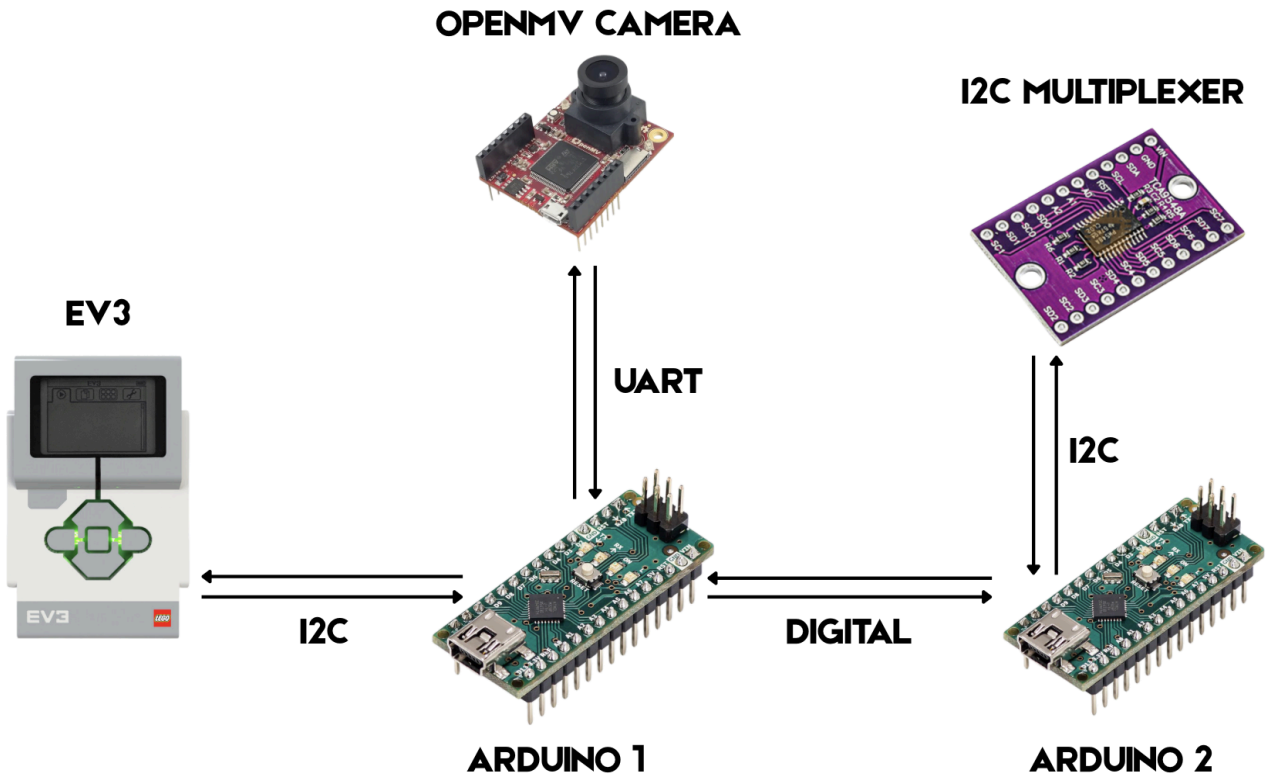

Fig:1 Communication

3. Hardware

3.1 High-Level Overview

The robot's structure is primarily built from LEGO EV3 components, enhanced with custom 3D-printed parts such as claws and an arm stopper for improved functionality. Three microcontrollers—the EV3 brick, 2 Arduino Nanos, and the OpenMV camera—work together to expand sensor capacity and streamline operations, with the Arduino acting as a communication bridge between the EV3 and OpenMV through one of the EV3's sensor ports. The EV3 brick has been modified with external buttons for easier access, and unnecessary plastic, including the battery pack casing and EV3 brick casing, has been removed to reduce weight and size. A custom power cable supplies both the Arduino and the OpenMV camera from **two 3.7 V LiPo batteries** to ensure an efficient and reliable power supply. For actuation, two EV3 large motors drive the tracks while two EV3 medium motors operate the arm, and custom-length EV3 cables connect sensors and motors to minimize clutter and optimize space inside the robot.

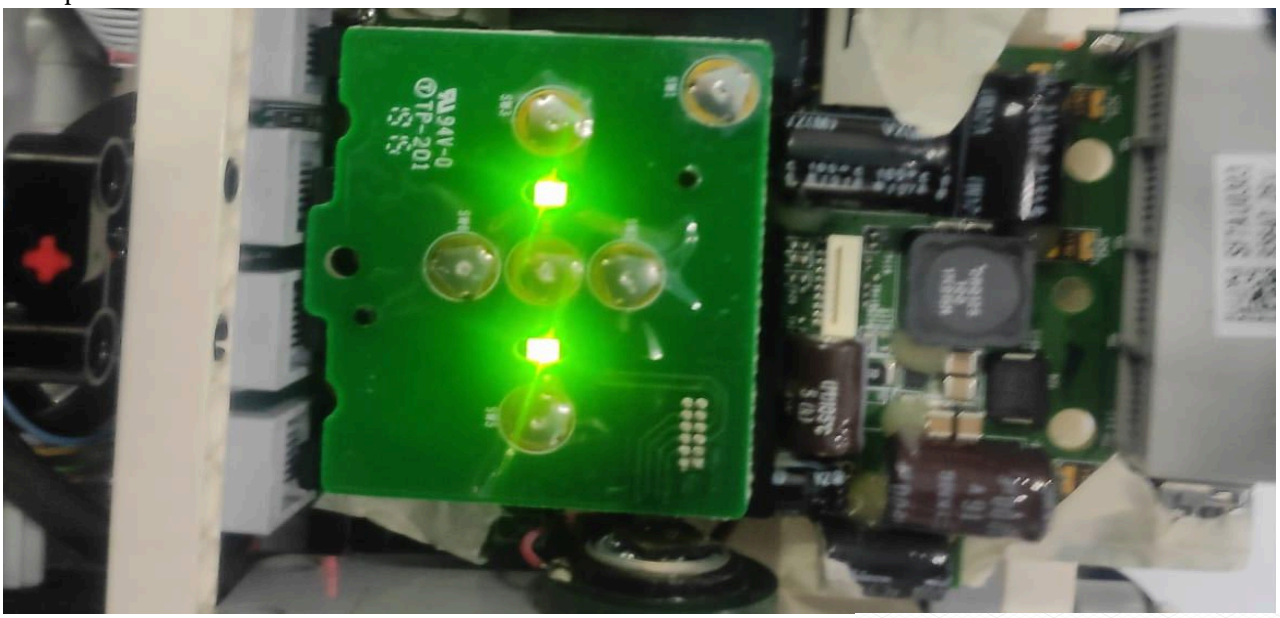

Fig: 2 EV3 open brick

- **AI-assisted Design:** We used **Edge Impulse** to train our own image dataset (**over 5000 self-labeled images**) for the OpenMV camera. Additionally, AI-based CAD and simulation tools such as **Fusion 360** were used to optimize mechanical layouts.

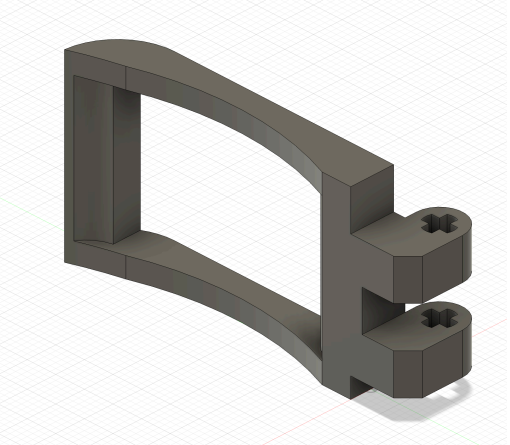

Fig: 3 3D printed claws

3.2 Mechanical Design and Manufacturing

- **Actuators and Power Train:** The robot employs four EV3 motors: two large motors (160 RPM and 40 Ncm stall torque) to drive the tracks for high-traction movement over ramps and speed bumps, and two medium motors (250 RPM and 12 Ncm stall torque) to operate the arm for victim pickup and placement. A servo-driven rescue plate enables precise victim drop-off in the evacuation zone. The tracks provide superior stability and power transfer compared to wheels, ensuring consistent performance on uneven terrain.
- **Modules:** The arm and claw assembly is designed as an independent, removable module for easy maintenance or upgrades. It operates using **four Knob gears** to open and close the claw, while **two bevel gears** and **two sphere gears** transmit motion to rotate the Knob gears and lift the arm. Modular sensor mounts allow quick swapping or repositioning of sensors during testing, while custom-length EV3 cables minimize clutter and optimize internal space. These subassemblies were thoroughly tested for fit, weight distribution, and ease of replacement to ensure reliability under competition constraints.

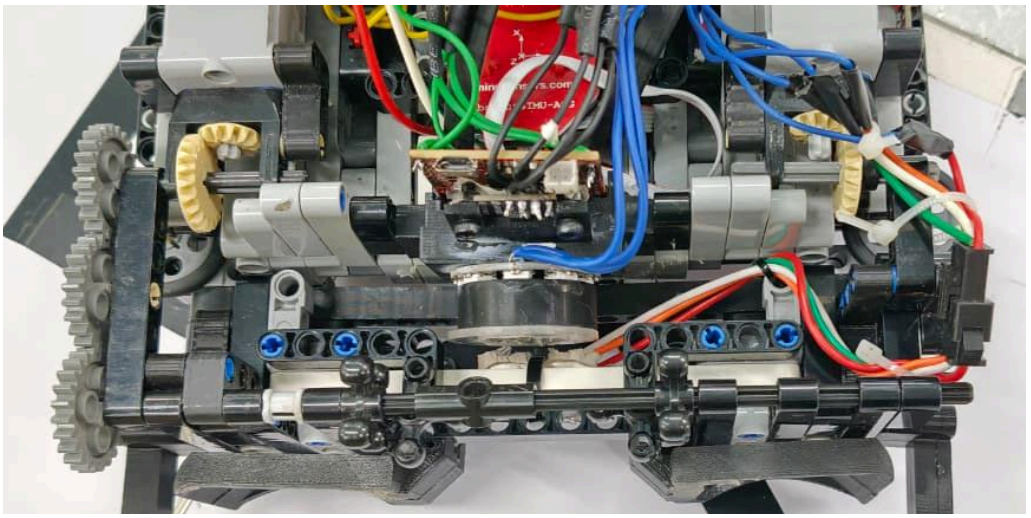

Fig:4 Gears Connection

- **Specialized Modules Deployment Mechanism:** The rescue mechanism combines EV3 beams and 3D-printed elements to form a lightweight but sturdy arm and grabber system. The grabber, designed in CAD and iteratively refined with AI-based simulations, enables the robot to reach victims farther away from its base and securely lift them.

3.3 Electronic Design and Manufacturing

3.3.1 Sensors Used

- **OpenMV Camera:** Detects victims, evacuation points, and goal tiles using self-labeled image datasets.
- **EV3 Colour Sensors** (**x2**): The EV3 colour sensors assist with precise line following, detecting green squares, and **double-checking** silver and red values for improved accuracy.
- **MindSensor Gyro:** Measures **roll and pitch** for ramp detection and stabilization.
- **Arm Ultrasonic:** Detects walls in the evacuation zone, allowing the robot to determine when it is close to an obstacle.
- **IR Time of flight sensor** (**tof**) (**x2**)**:** These detect openings or exits in the evacuation zone for smooth alignment and then exit.
- **Down Ultrasonic Sensor:** The downward-facing ultrasonic sensor assists in detecting ramps and speed bumps. It is activated when both colour sensors get a specific threshold, after which it measures the distance to confirm the ramp or bump.
- **Analog IR Sensor:** Detects silver tape or reflective surfaces to switch between programs, like switching from line following to the rescue area.
- **Digital IR Sensor:** Confirms if the robot has successfully picked up a live victim by giving a quick signal to the EV3.
- **LDR Sensor:** Verifies if the robot has successfully picked up a dead victim by sensing changes in **light intensity.**

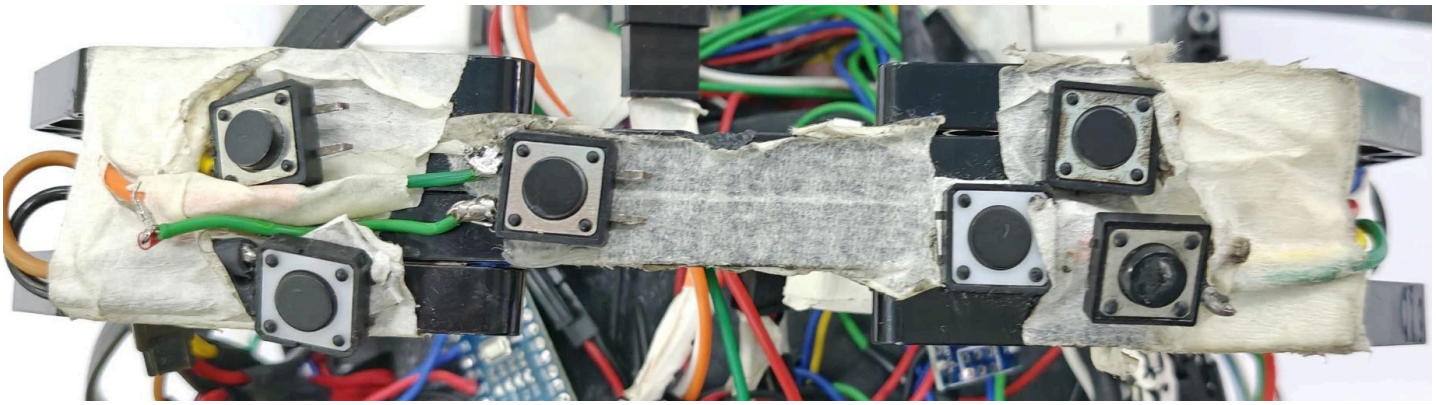

Fig:5 External Brick Buttons and Digital IR

4. Software

4.1 Overview

The bulk of the programming is done in the LEGO Mindstorm EV3 software, whereas we use C++ to program the Arduino and MicroPython for the OpenMV camera. The program is split into two parts: line following and evacuation zone. The EV3 communicates with the Arduino (Master) using I2C communication, then it communicates with the slave Arduino as well as the OpenMV through Digital communication and UART communication, respectively. The Master Arduino sends a certain number through the Digital communication .wire function to the Slave Arduino, where it does a certain if-else condition according to the number received and sends the information back to the Master Arduino. The EV3 communicates with the OpenMV camera via a byte, allowing the Arduino to print a task letter to the OpenMV. The robot detects goal tiles using the camera, avoiding the need for EV3 color sensors for line following. The program detects ramps when the robot is outside a specific range on the y-axis. The robot checks the byte number sent by the Arduino to determine if the time-of-flight sensor detected an obstacle, if the camera identified the goal tile, or if the goal tile was reached and the robot has to do RGB line following till the goal tile is sensed by the colour sensor, or if the analog IR sensed the silver tape.

4.2 General Software Architecture

The program consists of a single infinite loop that contains two additional loops, each with specific conditions for ending the loops. When the EV3 color sensor detects the silver tape, the line-following loop terminates. After that, the program checks which sensor's condition has been met. If the Analog IR sensor is triggered, the program executes the evacuation zone algorithm and then restarts the loop. If the above condition is false, it checks if the OpenMV senses the goal tile, then it plays the RGB line follower and ends the main loop, and the program stops.

Line Following and Intersection:

The robot uses proportional line following with two EV3 colour sensors for precise navigation. An advanced algorithm calculates and tracks the robot’s position to ensure smooth movement. At intersections, the robot checks for declined ramps using the downward ultrasonic sensor. If no ramp is detected, it evaluates double green squares or adjusts for a single green square by rotating a set number of times. The Pathfinder algorithm further helps determine ramp positions at intersections. Additionally, a parallel program monitors roll and pitch using the MindSensor gyro, detecting downward slopes and enhancing line-following accuracy.

Rescue Area

The robot enters the evacuation zone, detects walls using the Arm Ultrasonic, and identifies rescue victims with the OpenMV camera. It aligns the victim to the center of the camera's view and moves forward until either the EV3 colour sensor detects a silver or black line or the Arm Ultrasonic confirms the victim's presence before picking up the victim. If the victim is not detected, the robot releases and rescans. Once a victim is successfully picked up, the robot deposits them at the designated evacuation point. After completing the rescue, it aligns with the wall using the front arm and uses the side IR TOF sensor to detect an exit, seamlessly exiting the zone while maintaining alignment with the line.

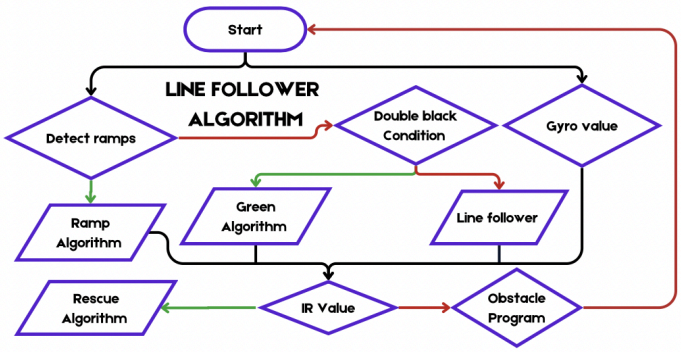

Fig:6 Line Following Flowchart

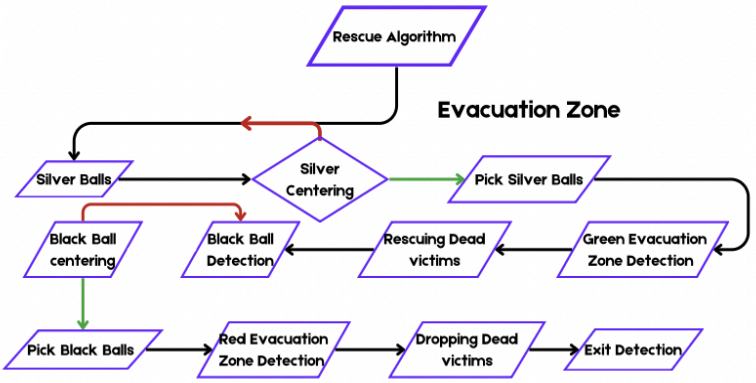

Fig:7 Evacuation Zone Flowchart

4.3 **Innovative Solutions**

During development, we faced several challenges that required innovative solutions. These included difficulties with the robot correcting itself on sharp turns or when it strayed off the line, as well as frequent access needed to the EV3 brick buttons. Various creative strategies were implemented to overcome these issues and improve overall performance,

- **Digital communication :** We implemented Digital communication communication between the Master and Slave Arduino to ensure fast and efficient data transfer. Additionally, an I^2C multiplexer on the Slave Arduino allows multiple I^2C sensors to be connected simultaneously, overcoming the limitation of only two native I^2C ports. This setup ensures continuous, reliable data transfer from all connected sensors.
- **Pathfinder:** We developed a Pathfinder algorithm to improve the robot's navigation. It allows the robot to autonomously correct its course when it strays from the line or faces sharp turns. Using data from the EV3 colour sensors, the algorithm detects whether the line is lost during a turn; if so, the robot rotates in the opposite direction for a set number of rotations. By analyzing this information in real time, the algorithm accurately determines the robot's position and adjusts its path, ensuring smooth and reliable navigation on complex tracks.

5. Performance Evaluation

5.1 Evaluate the Robot's Performance

The robot completed all tasks under various scenarios, handling line-following and rescue area challenges without losing progress. Its strengths include a dual-movement arm for independent motor control, a compact yet sturdy chassis, and efficient differentiation between live and dead victims. The main limitation is the exposed EV3 brick, which can lead to issues such as button short-circuiting, component corruption, and difficulty tracking victims during approach.

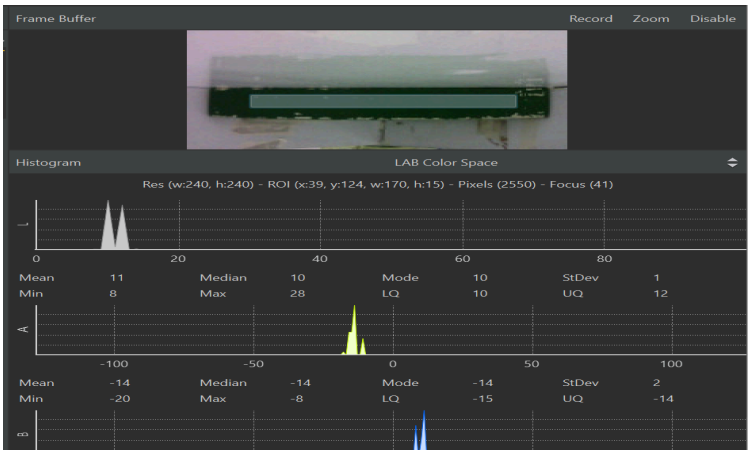

Fig:8 Measuring Lab Values for Green Wall

5.2 Testing Procedures:

The methods and procedures that were used to test the robot's performance involved using different types of obstacles, like a transparent bottle, a wooden block, and a brick. We have used different sizes of the rescue victim to ensure that the robot can successfully pick and deposit all sizes of the rescue victims that may be used for the competition. We had used different objects and settings to differentiate between alive and dead victims for the digital IR LDR and OpenMV individually.

5.3 Analysis and Impact:

EV3 RGB Mode:

The EV3 colour sensors' RGB mode is crucial for our robot's algorithm, enabling accurate identification of green squares. Although it requires more computation during line following than IR mode, it performs more reliably when the sensor and ground distances vary. Strategically interspersing Infrared with short-duration RGB line following at specific points improves our robot's line following capabilities.

	VALUES	FULL GREEN	FULL WHITE	FULL BLACK	WHITE/ BLACK	WHITE/ GREEN	GREEN/ BLACK
3	RED	14	117	11	88	42	12
	GREEN	47	117	15	107	66	22
	BLUE	34	158	17	103	75	21
2	RED	15	132	16	51	76	13
	GREEN	73	167	28	63	124	43
	BLUE	29	137	17	48	88	18

Fig:9 RGB Values of Colour Sensors

OpenMV Evacuation Point:

To identify the green and red evacuation sites inside the evacuation zone, we used the OpenMV camera. After successfully retrieving the rescue victim from any distance, we employ this technique to save time when looking for an evacuation spot. Initially, we have set up three parameters: the camera's scanning region of interest (ROI), contrast, and brightness. We construct three threshold ranges by analysing three graphs set in the LAB Colour Space mode and by manually drawing a rectangle to take values around the evacuation point.

6. Discussion and Conclusion

Our project successfully created a compact, autonomous robot for the RoboCup Junior Rescue Line competition. By integrating the EV3 brick, two Arduino Nanos, and an OpenMV camera, we overcame hardware limits and achieved reliable line following, victim detection, and rescue operations. Tracks, a dual-movement arm, and modular 3D-printed parts provided strong traction, precise victim handling, and easy maintenance.

Key lessons learned include the importance of modular design, extensive sensor calibration, and robust communication between controllers. External EV3 buttons and custom wiring improved usability and reduced downtime.

Overall, the team met its main goals of speed, accuracy, and reliability. The robot consistently handled ramps, intersections, and the evacuation zone while distinguishing live and dead victims accurately.

For future improvements, we plan to develop an open-source robot featuring camera-based line following and a wheeled drive system. We also aim to improve victim tracking during approach and explore faster processing capabilities. Beyond its technical achievements, this project has strengthened our teamwork, problem-solving, and project management skills.

During the practice phase, the most challenging tasks were opening a brick and soldering buttons onto its PCB, as well as removing the casing of the EV3 battery and rewiring it.

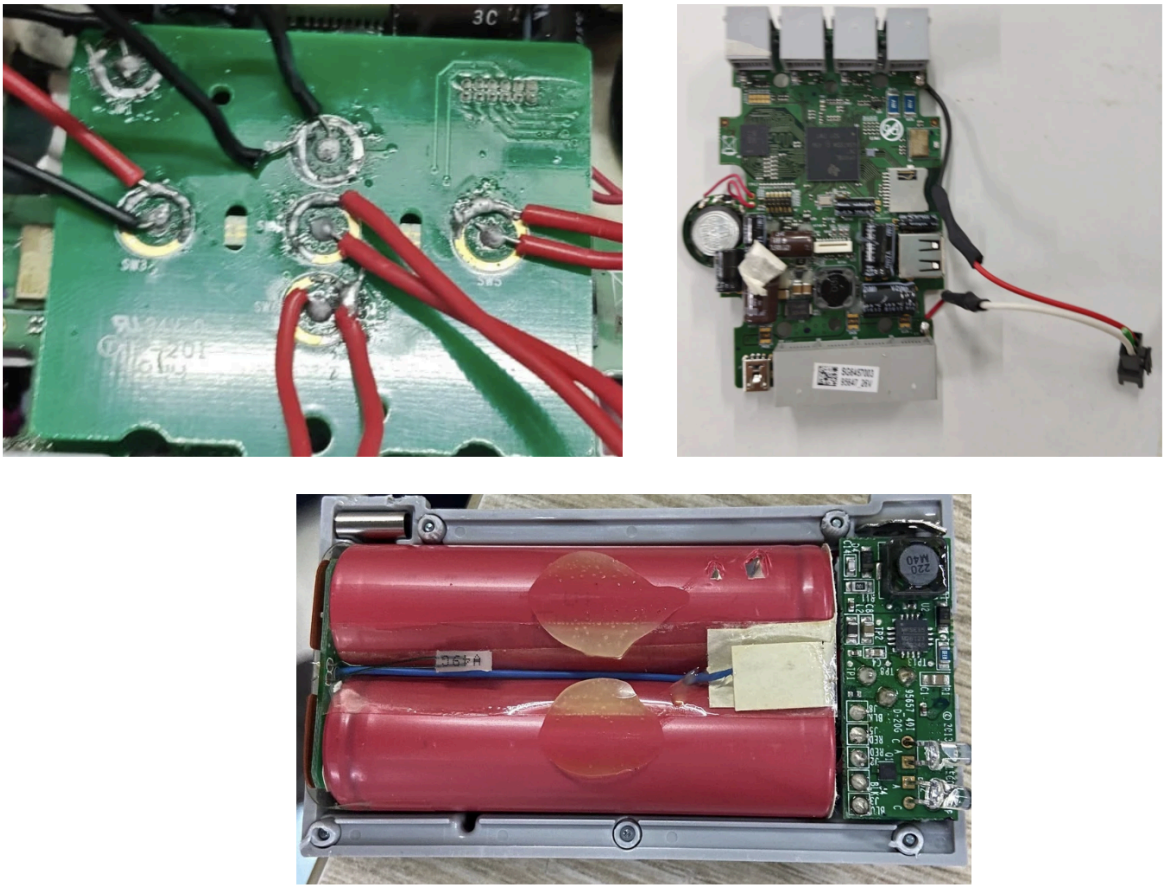

Fig:10 Buttons PCB, Open EV3 Brick and EV3 battery

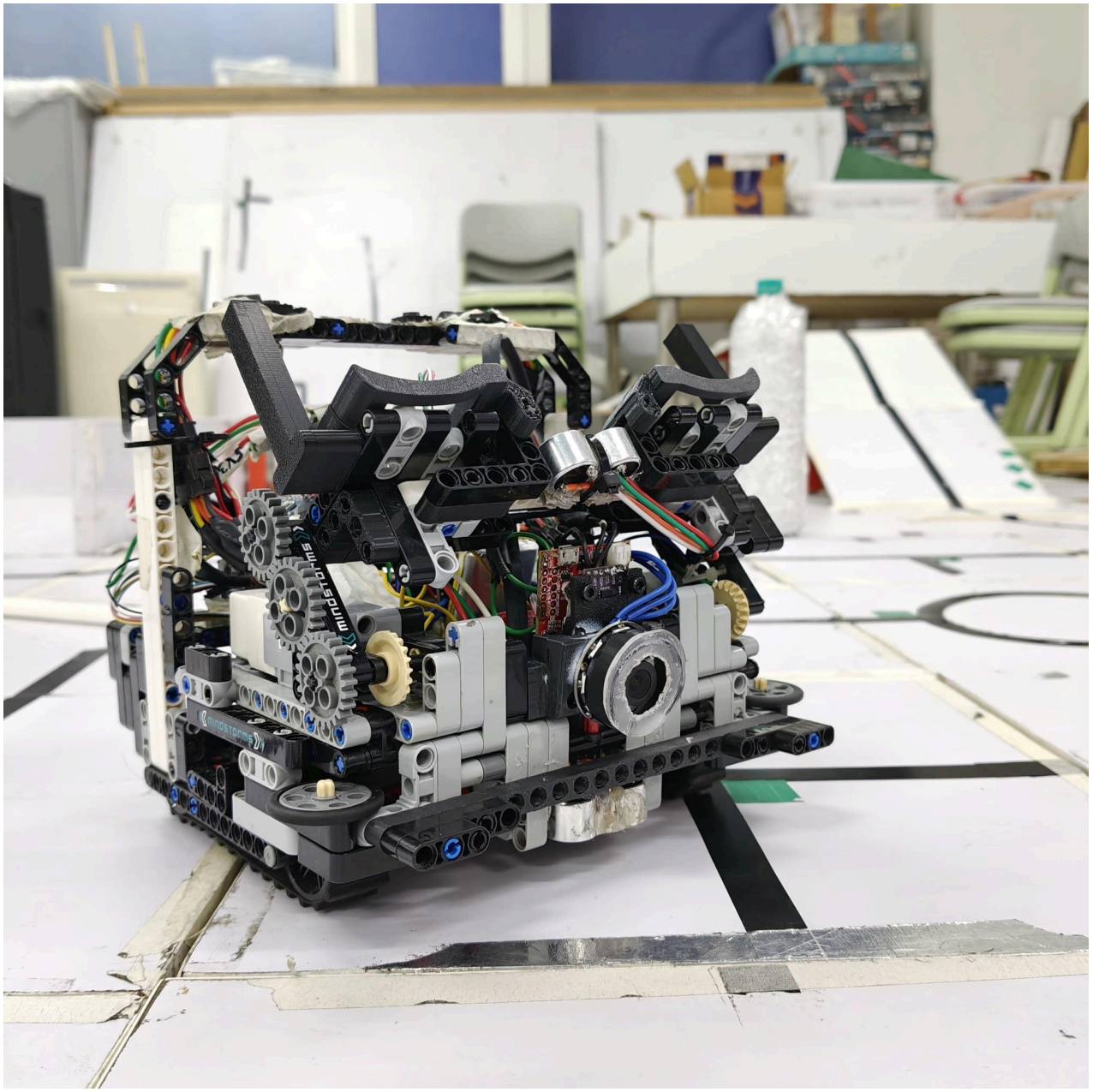

Fig:11 Final Lego robot

7. Acknowledgments

We would like to express our heartfelt gratitude to everyone who has supported us throughout this competition. First and foremost, we extend our sincere appreciation to our teachers and mentors: Ms. Ganga Subramanian, Ms. Kamini Sharma, Ms. Pallavi Saran, Mr. Mudit Adityaja, Mr. Nikhil Sundaresan for their guidance, encouragement, support, and time. Their advice and mentorship have been instrumental in our journey. We are also grateful to our school for its continuous support over the years, enabling us to grow and explore our passion for robotics. Additionally, we would like to thank the organizers of the RoboCup competitions for providing us with this incredible platform to learn, innovate, and challenge ourselves while competing with our peers.

8. References

Here is a list of references our team used while developing our robot. These include technical guides and online discussions that helped us understand design, programming, and problem-solving. Each reference provided useful information that influenced how we built and improved our robot

- We referred to the official Arduino forum for troubleshooting and code snippets.
  - Link: https://forum.arduino.c
- OpenMV documentation is used to implement code in the camera module.
  - Link: https://docs.openmv.io
- Referred to the RCJ forum for Q&A.
  - Link: https://junior.forum.robocup.org/
- We also got our inDigital communication ration from all the Robocup International Teams and their runs, as well as from some websites for hardware aspects
  - https://youtu.be/eP99Jd03Dmo?si=Ag0fRsJof6Hw0rEy
  - https://builderdude35.com
  - https://youtu.be/-4d62VtdYp8?si=P_m_5QFBx3QzH5gw
  - https://youtu.be/xNCCSq1M0o4?si=kB50nhUO4vC6kkYr
  - https://www.youtube.com/watch?v=-EyZ6_YDfWE
- We referred to the RCJ community website for reference to the mats
  - https://rescue.rcj.cloud/events/2024/RoboCup2024/line/practice