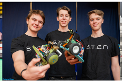
Our team

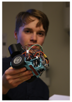
We are STEN Robotics, and we will be representing Sweden in RoboCup Junior Rescue Line 2025! This is our third year in RoboCup, and our second year participating internationally. With a total of three victories at our local district championship, two victories at the Swedish championship, and participation in last year's world championship, where we finished 19th with a victory in a small sub-league, we are thrilled for this year's competition in Salvador!

World Championship 2024



District Championship 2025

Swedish Championship 2025



- Malte Robért

With almost three years being involved in RoboCup Junior Rescue Line and an even longer background of building and coding robotics I am responsible for programming the robot. Ever since I was a kid I have been interested in making autonomous and moving robots that "think". I love to take on the difficult challenges and create my own ways of solving a problem, therefore during the development of the current generation of STEN I have been responsible for the software and algorithms.

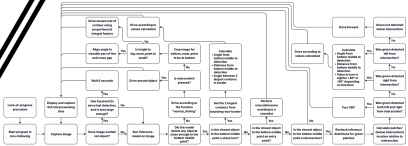
- Hampus Mollgård

Having grown up on a farm, I have always loved making and constructing. Initially this consisted of physical wood/metal fabrication in my own tiny workshop, but the outlet for my interest has evolved throughout my life. My love for inventing has led to a developed interest in 3D-printing, my entrepreneurial personality has contributed to an ability to sink completely into my work, and my relatively newfound love for electronics has been highly developing personally. With this background, I have mostly been responsible for the mechanical construction while simultaneously challenging my electronic competence ever since we started our robot development.

- Oscar Bergh

After getting third place with a randomized group in a local RoboCup competition 2022, me and three others believed that we could take this further and formed a new team called STEN. Before that I had only been working with simple electronic circuits, 3d printers and lego robots. During development ofSTEN I have gained a lot of experience in CAD and programming.
This year I am mainly responsible for the chassis and electrical system.

Software



Software Architecture

The software is structured around a central main() function. This function initializes the system and delegates tasks to specialized sub-functions responsible for different aspects of the robot's behavior, such as line following, intersection handling, and evacuation zone navigation. This modular design ensures that the code is maintainable, scalable, and adaptable across different hardware iterations.

Programming Language and Development Environment

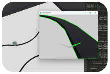
The software for STEN 11 is developed in Python 3.11, chosen for its compatibility with essential libraries such as OpenCV and NumPy, and for its ease of use in rapid prototyping. Development is carried out in Visual Studio Code on personal computers rather than directly on the Raspberry Pi. This approach allows us to benefit from a more efficient development environment, better debugging tools, and cloud-based backups.

The robot's main processing unit is a Raspberry Pi 5, which handles all high-level logic, including image processing and decision-making. The OpenCM9.04 microcontroller is used exclusively for motor control and is programmed using the Arduino IDE in C++. Communication between the Raspberry Pi and the OpenCM9.04 is handled via a serial connection, with the Pi sending either motion parameters or commands to execute predefined sequences of actions.

Navigation and Control Algorithms

For basic navigation, the robot employs a simplified PID controller that uses proportional and integral terms to follow lines on the course. When more complex features are detected—such as intersections or entry points to the evacuation zone—the robot switches to decision-making based on object detection.

The evacuation zone is handled using a checklist-based system, where the robot follows a predefined sequence of objectives. Each object in the zone—such as victims or exit points—is identified using the same detection model, and the robot executes a corresponding action, such as picking up or releasing a victim.

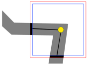


Testing in simulated environments

Testing in simulated environments: The ability to test, practice, and develop our robot in simulated environments is something we've researched and tested multiple times. Our robot has the ability to record and save the camera feed, sensordata, and decision-history from any live run, allowing us to rewatch and identify problems. Previous iterations of STEN have also had the ability to run a simulated run on the recorded data and see how our adjustments affect the robots' decisions. Despite this robot not possessing that ability, it is still built on insights gathered from that simulated environment. Furthermore, the introduction of AI in our robot has led us to investigate the possibility of running and training STEN on a fully simulated track using software like CoppeliaSim or Isaac Sim. We have not fully explored this possibility but did achieve a proof of concept that future teams may be able to build upon.

Custom trained AI models

Being early adopters of the newly released Raspberry Pi AI hat+ capable of 26 TOPS meant we had to explore this board's possibilities ourself. During our testing we experimented both with object detection models capable of identifying interesting features on the track, and image segmentation models capable of distinguishing the line with great accuracy despite glare and debris. Despite this we settled on only using the detection model to save on performance, a decision that allows us to achieve 200 fps running inference on blank images, and about 20 fps in real scenarios, including the rest of our algorithms. Using custom trained AI models makes it easy for us to identify advanced features both in the evacuation zone and during line following. We have also further developed a custom workflow for creating these models. In order to create a new model we use a separate script on the raspberry pi to record while we mode it along the track, split the video into frames, annotate all frames in roboflow, train the model in google collab using YoloV8, and convert it from ".onnx" to ".hef" in a docker container running via anydesk on a custom built PC. This process of creating a whole new model fully accustomed to new lightning conditions takes just over one hour if done correctly with a small dataset.



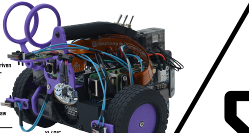
Usage of a "Bounding-box-border"

To calculate the desired angle and direction to turn when on top of a sharp turn we crop and modify the image to only show a thin border around the detected 90º turn. This method gives us two separate contours illustrated in black in the adjacent picture. Using this information we can determine what contour is the entry and exit from the turn, as well as what amount we should turn.

Checklist in evacuation Zone

For the purpose of simplification, our raspberry pi regards all objects in the evacuation zone in the same way. At the start we declare a checklist of objects to visit and act at. This list simply states the order of all objects to visit in the evacuation zone: ["Live victim", "Green", "Live victim", "Green", "Dead victim", "Red", "Exit"]. Using this list we can drive toward the currently desired object, perform a sequence of commands to either pick up or release a victim, or center itself on the exit. This is possible because we use the same detection model to detect victims as we use to detect the exit from the evacuation zone. For the exit we actually detect the rims around the exit, and therefore need to center our robot before exiting.

Hardware



STEN 11 features a 3D-printed frame and a compact three-wheeled drivetrain—two front-driven wheels and a rear swerve wheel—for enhanced maneuverability and turning precision over previous omni-wheel setups. It uses two wide-angle cameras (downward for line following, front-facing for victim detection), connected to a Raspberry Pi 5 with a 26 TOPS AI accelerator for image processing and inference. Motor control is handled by an OpenCM9.04 board with Dynamixel XL-320 servos, powered by a 3-cell LiPo battery with regulated outputs. Additional features include a debris-clearing fan, LED light, and a claw for rescue tasks.

- RPI Powerhat
Stepdown converter is used to transform the 11.1V from our 3s lipo into 5V for our Raspberry pi

- Rasberry Pi 5 8GB
Used as the main processor, it handles all image processing, and decision-making during runs.

- XL4015
A buck converter used to step down the 11.1V from the LiPo battery to 7.2V, for OpenCN board.

- Mosfet-module
In order to control our 12V fan we make use of a generic mosfet-module

- 3S LiPo Battery
A 3-cell lithium polymer battery supplying 11.1V to power all electrical components in the robot.

- AI Hat+
A 26 TOPS accelerator mounted on the Raspberry Pi, enabling real-time object detection and AI inference for navigation and victim recognition.

- 12V Axial Fan
Mounted centrally, it clears debris from the track while cooling internal components.

- LED light
A compact 12V lamp mounted on the front to ensure consistent lighting for the cameras.

- 6x Dynamixel XL320
Compact Dynamixel servos used for driving, steering, and actuating the claw, chosen for their feedback capabilities, size and ease of integration.

- Raspberry Pi Cooler
An active heatsink mounted on the Pi to prevent thermal throttling and ensure stable performance.

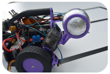
- Open CM9.04
Acts as a dedicated motor controller, receiving commands from the Raspberry Pi and controlling all six Dynamixel servos.

- Motion Sensor
A 6-axis inertial sensor used to detect movement and orientation, aiding in navigation and stability during complex maneuvers.

- 3x Binary Switch
Used to start and stop the robot during both lack of progress and during development

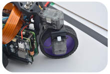
- 2x Raspberry Pi Cameras
One downward-facing for line detection and one forward-facing for victim recognition, both feeding real-time video to the AI system.

- 3x Microswitches
Used for obstacle detection and precise centering when exiting the evacuation zone, providing reliable physical feedback to the control system.



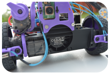
Compact Front-Mounted Claw Mechanism

STEN 11 features a compact yet robust claw mechanism that got carried over from STEN 10. The claw is mounted at the front of the robot and is actuated by two Dynamixel XL-320 servos, allowing for precise and powerful movement despite the mechanisms miniscule size. The claw is both lightweight and durable while holding the victims surprisingly firmly. The mechanism is tightly integrated into the 3D-printed chassis, with careful attention to cable routing and structural reinforcement to ensure consistent performance under repeated use.



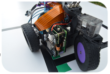
Swerve Wheel for Precision steering

STEN 11 features a rear-mounted swerve wheel that plays a key role in the robot's maneuverability. Unlike traditional four-wheeled systems based on tank steering, the swerve wheel allows our robot to pivot smoothly and maintain a consistent center of rotation. This is especially useful when navigating tight corners, intersections, or aligning in preparation for gaps. The swerve module is powered by a dedicated Dynamixel XL-320 servo, which controls the wheel's rotation angle independently of its drive motor. This setup enables the robot to steer with high precision while maintaining forward motion, giving it a significant advantage in both agility and control.



12V Axial Fan for Debris Removal

A standout feature of STEN 11 is its centrally mounted 12V axial fan, which plays a critical role in maintaining a clear path during runs. Unlike typical fans used for cooling, this component is primarily intended to actively remove debris from the track—such as toothpicks, paper squares, and wooden dowels—that could interfere with the robot's AI-models or image processing. Positioned to blow air forward, the fan helps ensure that the line remains visible and unobstructed, especially on the cluttered tracks we expect during the world championship. This solution was developed in response to real-world challenges encountered during previous competitions, where organizers used strings that tangled into our drivetrain. While the fan does also offer some passive cooling to nearby components, its main function is mechanical—clearing the way for precise navigation and uninterrupted line following.



Compact yet Powerful Raspberry Pi Assembly

At the core of STEN 11's processing is a Raspberry Pi 5, stacked with both an AI Hat and a Power Hat. This compact assembly handles all image processing, AI inference, and decision-making. A heatsink and active cooler are included to ensure stable performance during intensive tasks, all while fitting neatly within the robot's tight internal layout.

S.T.E.N

RoboCup
2025 // SALVADOR

Github

Sponsors

Instagram