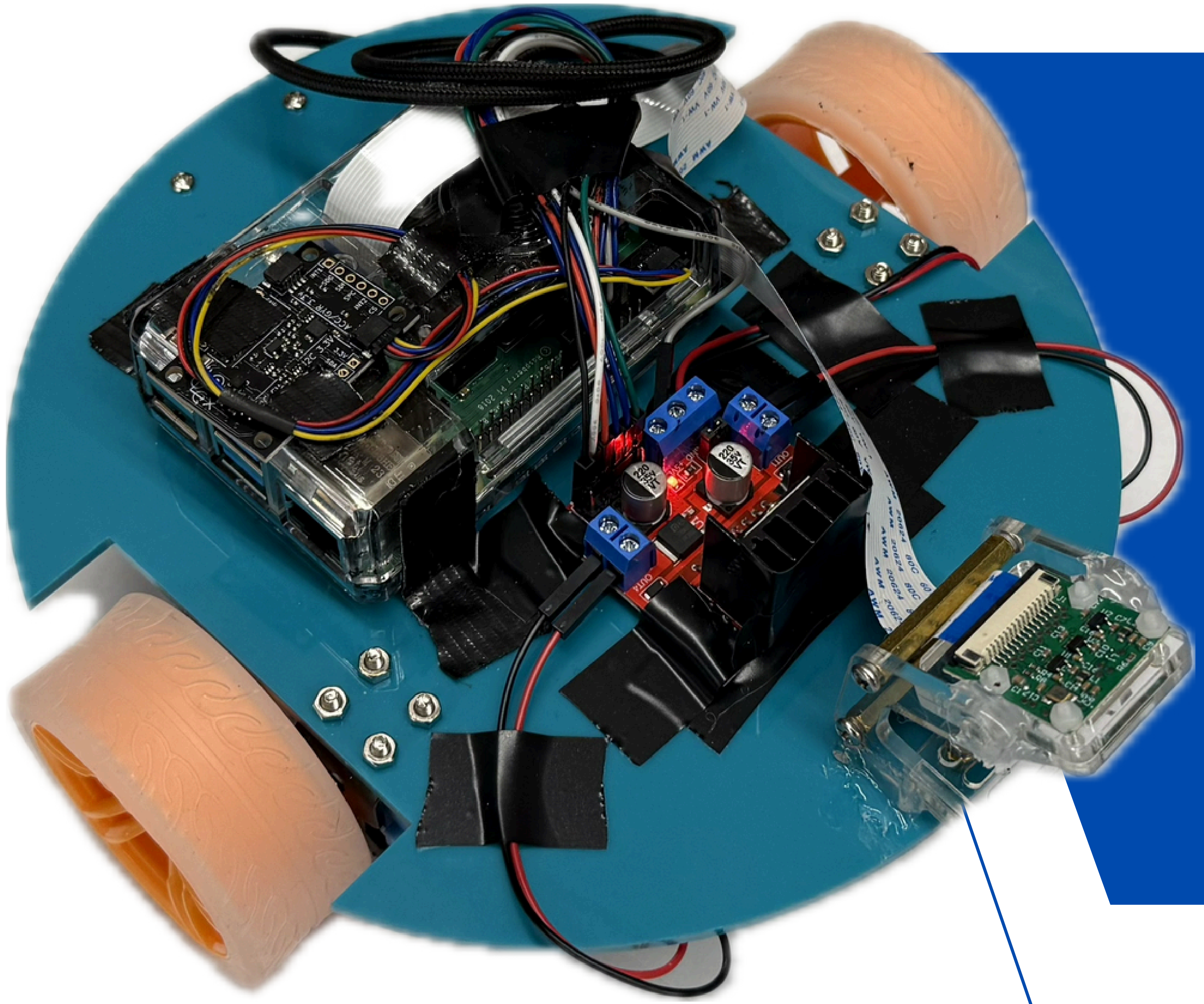


V.R.O.O.M.

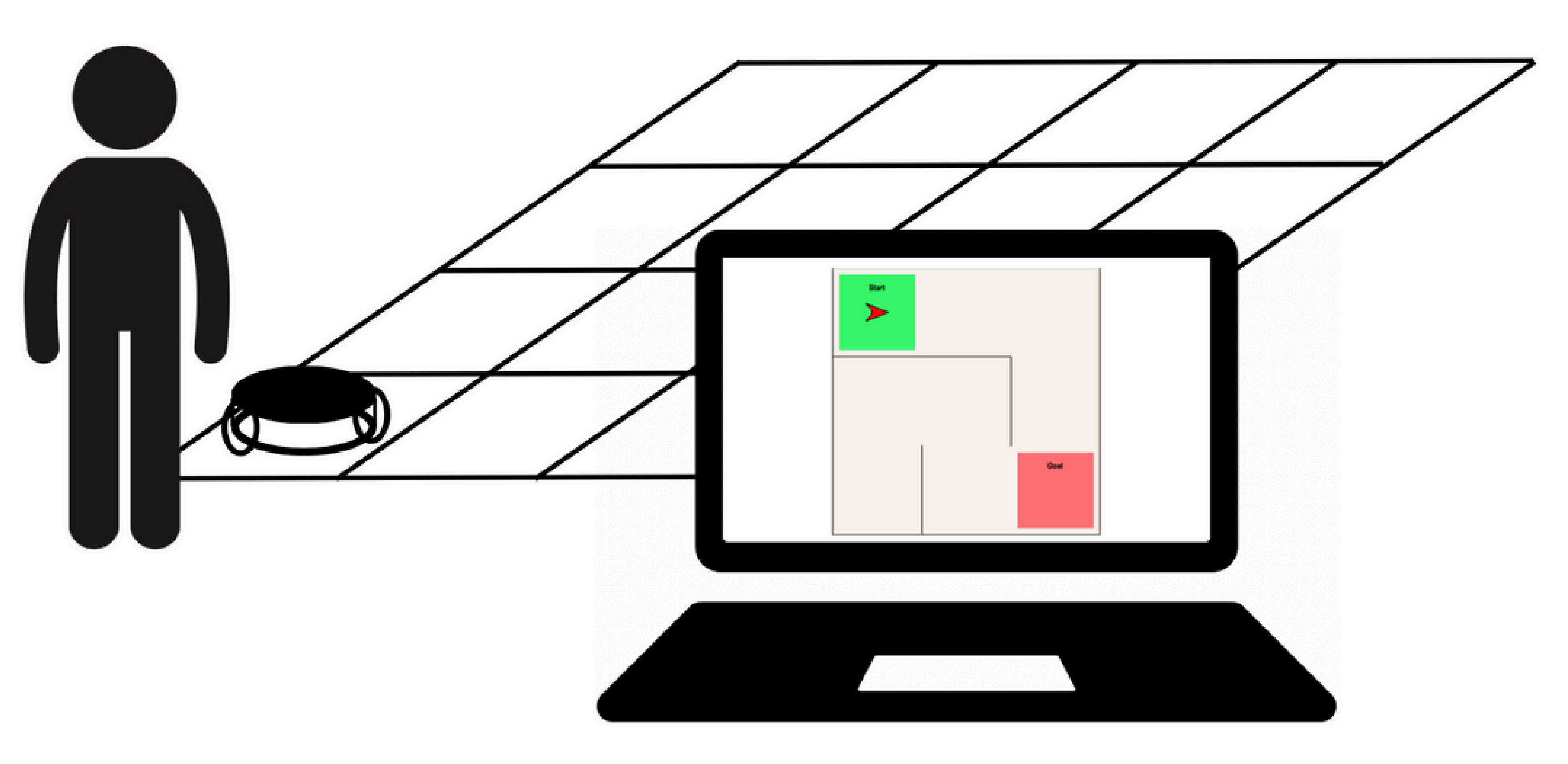
VOICE-RESPONSIVE OBSTACLE OVERCOMING MAZE-NAVIGATOR

FINAL PRESENATION

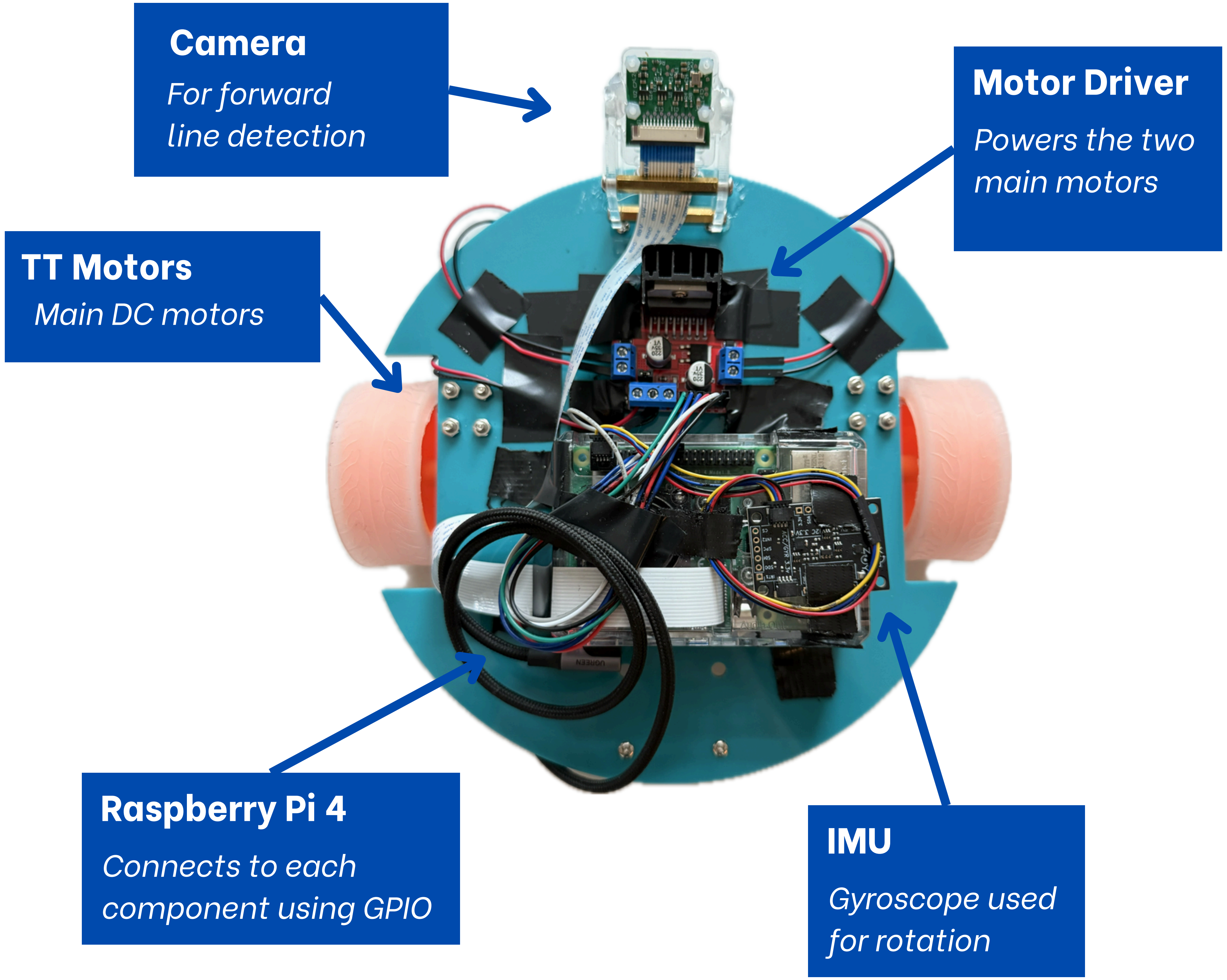
By: Aamina, Emanuel, Steve, and Torin

OVERALL DESIGN

- Real-world maze game with hidden walls
- Player controls car through voice commands & button input
- Player learns each wall location as they navigate through the maze



MAZE NAVIGATOR



Camera
For forward line detection

Motor Driver
Powers the two main motors

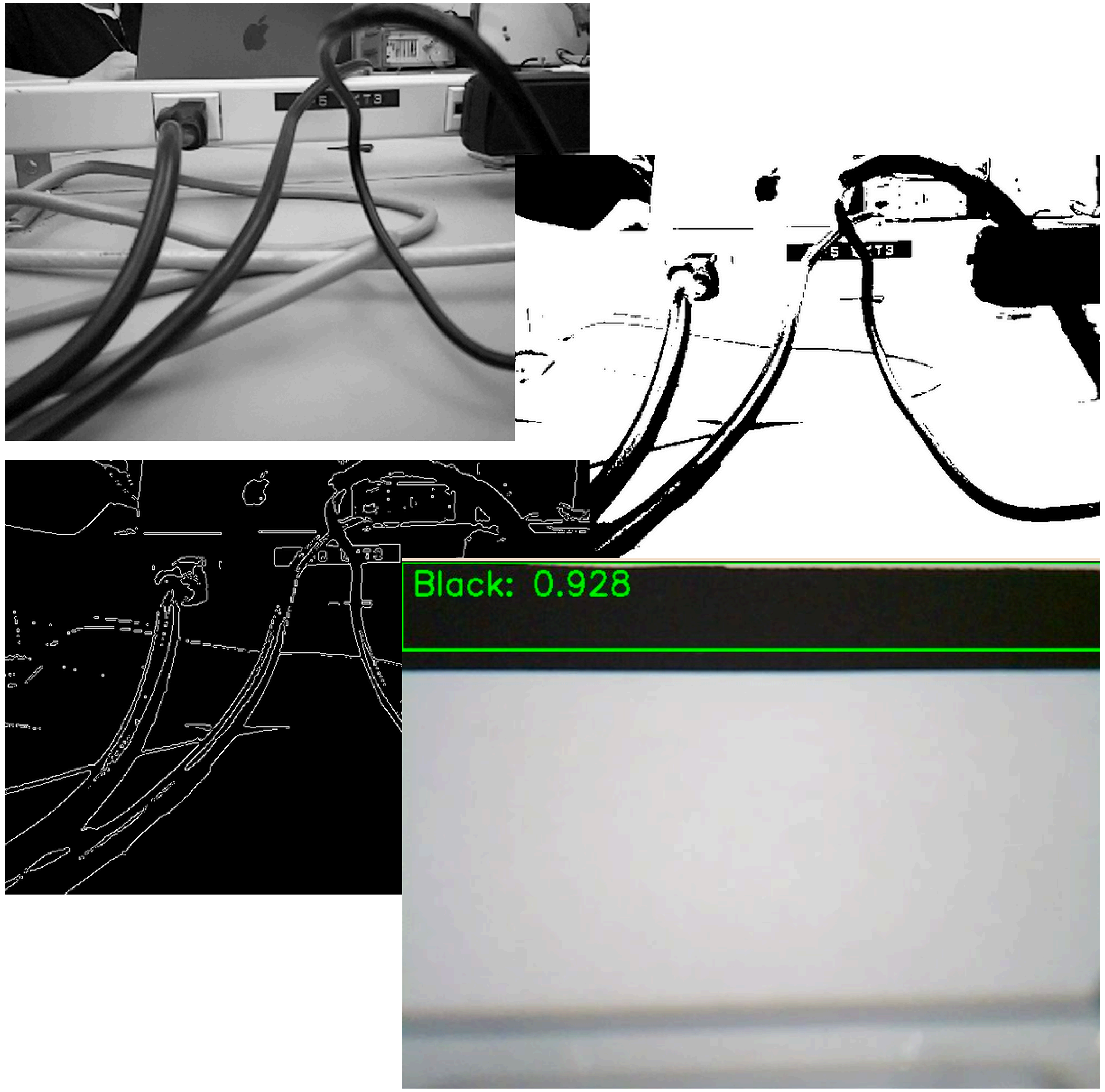
TT Motors
Main DC motors

Raspberry Pi 4
Connects to each component using GPIO

IMU
Gyroscope used for rotation

MAIN FUNCTIONS

- Connects to Maze Program using TCP.
- Recieves movement commands from the Maze Program.
- Uses Camera Vision for forward movement.
- Uses IMU gyroscope data for rotation.



Maze Navigator POV

COMPUTER VISION

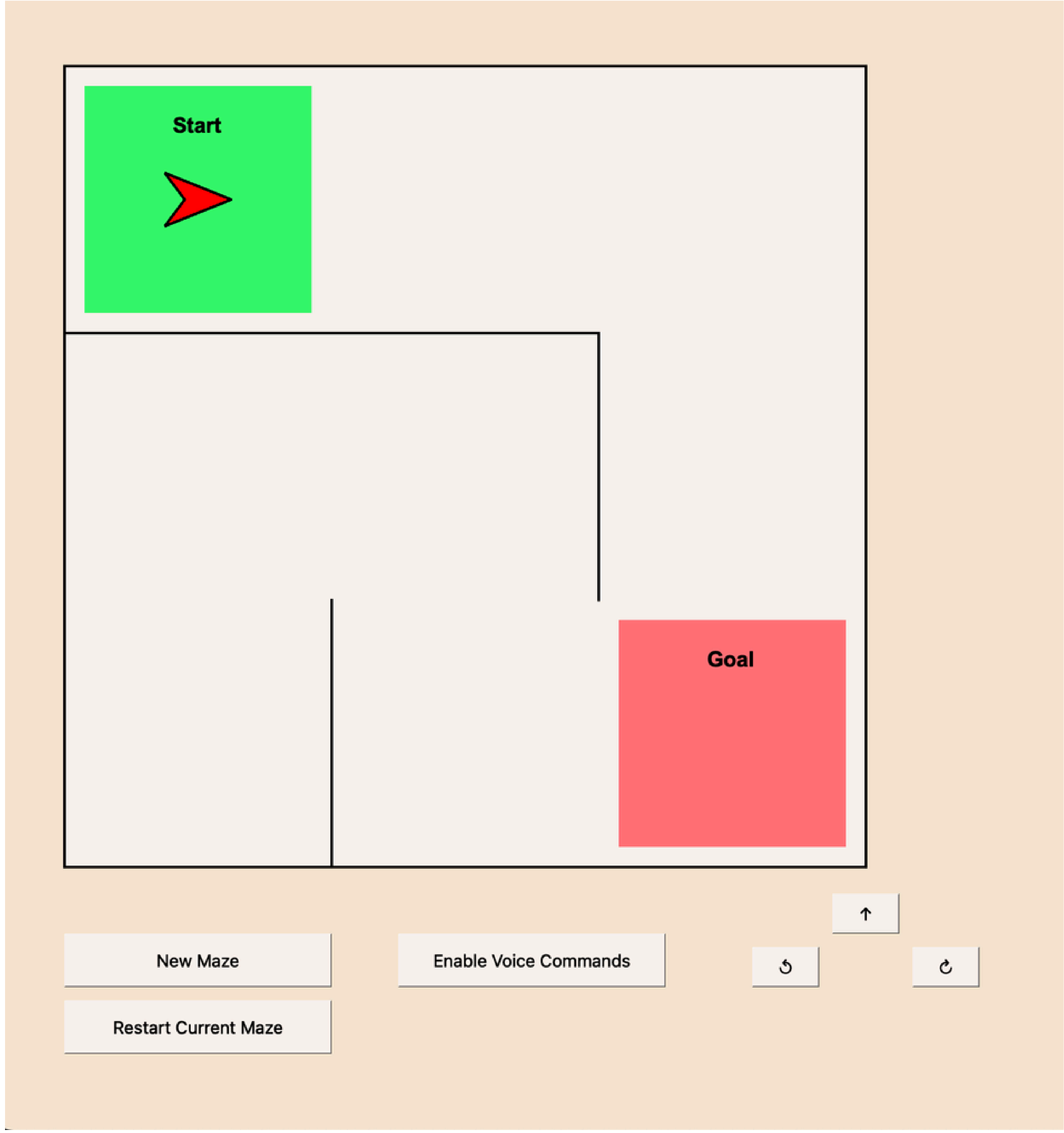
Responsible for controlling Navigator movement once in motion.

- The Maze Navigator uses Region of Interest (ROI) analysis to detect lines locally
- Pipeline: BGR > BW > Binary Threshold
- Once enough of the maze's boundary has been detected in the region, the car will recognize the boundary

GAME SOFTWARE

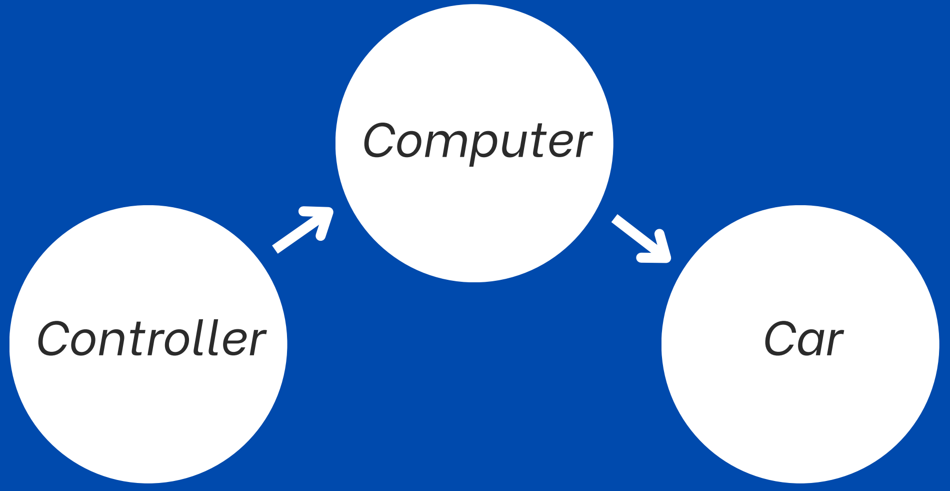
Provides an omniscient overview of the game.

- Shows all the maze walls and the player's real-time position for any spectators or if the player gives up
- Randomly generates mazes using recursive backtracking
  - Options to restart current maze or start over with new maze
- Behaves as middleman for controller commands
  - Program user can override the controller and toggle voice commands

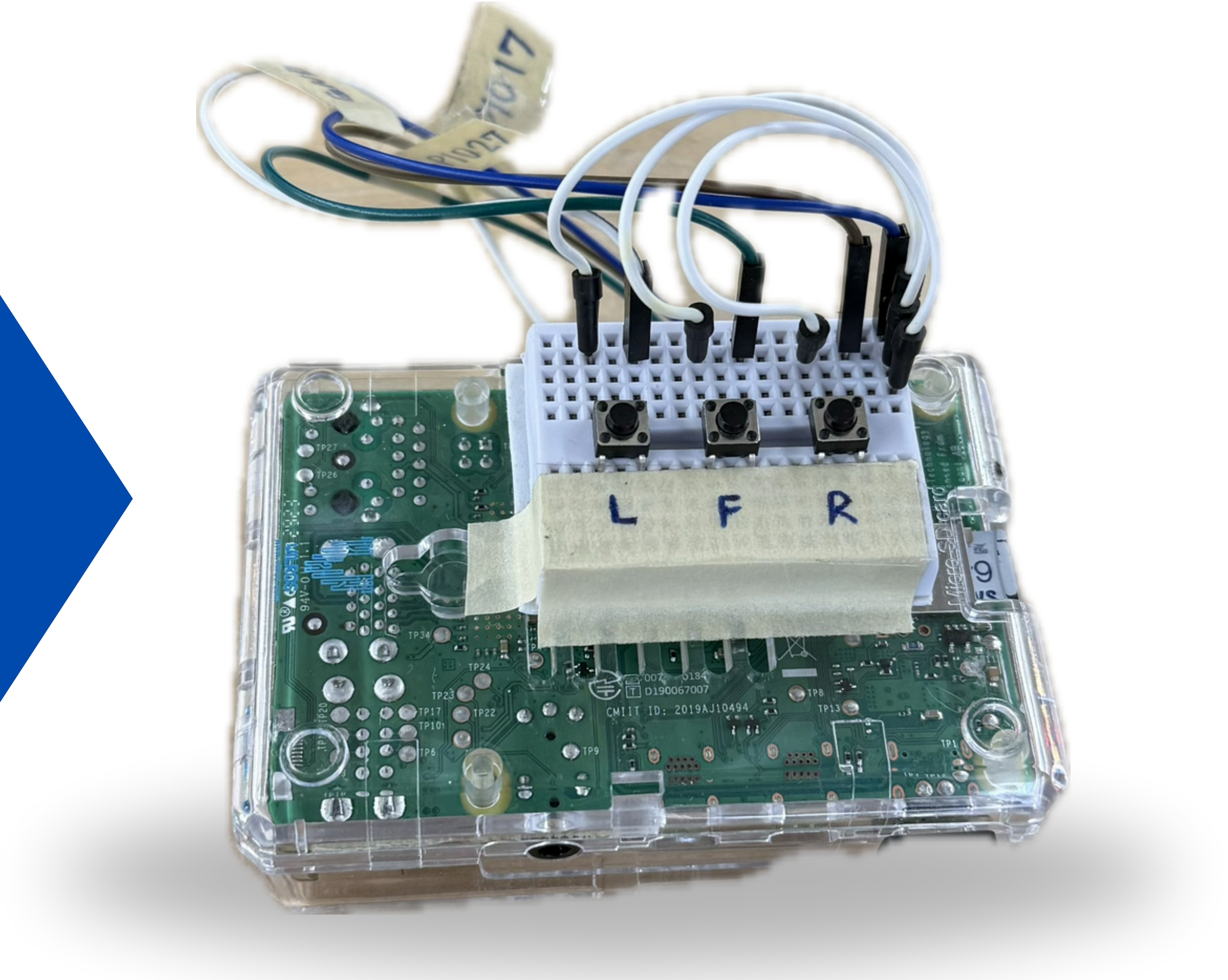


CONTROLLER

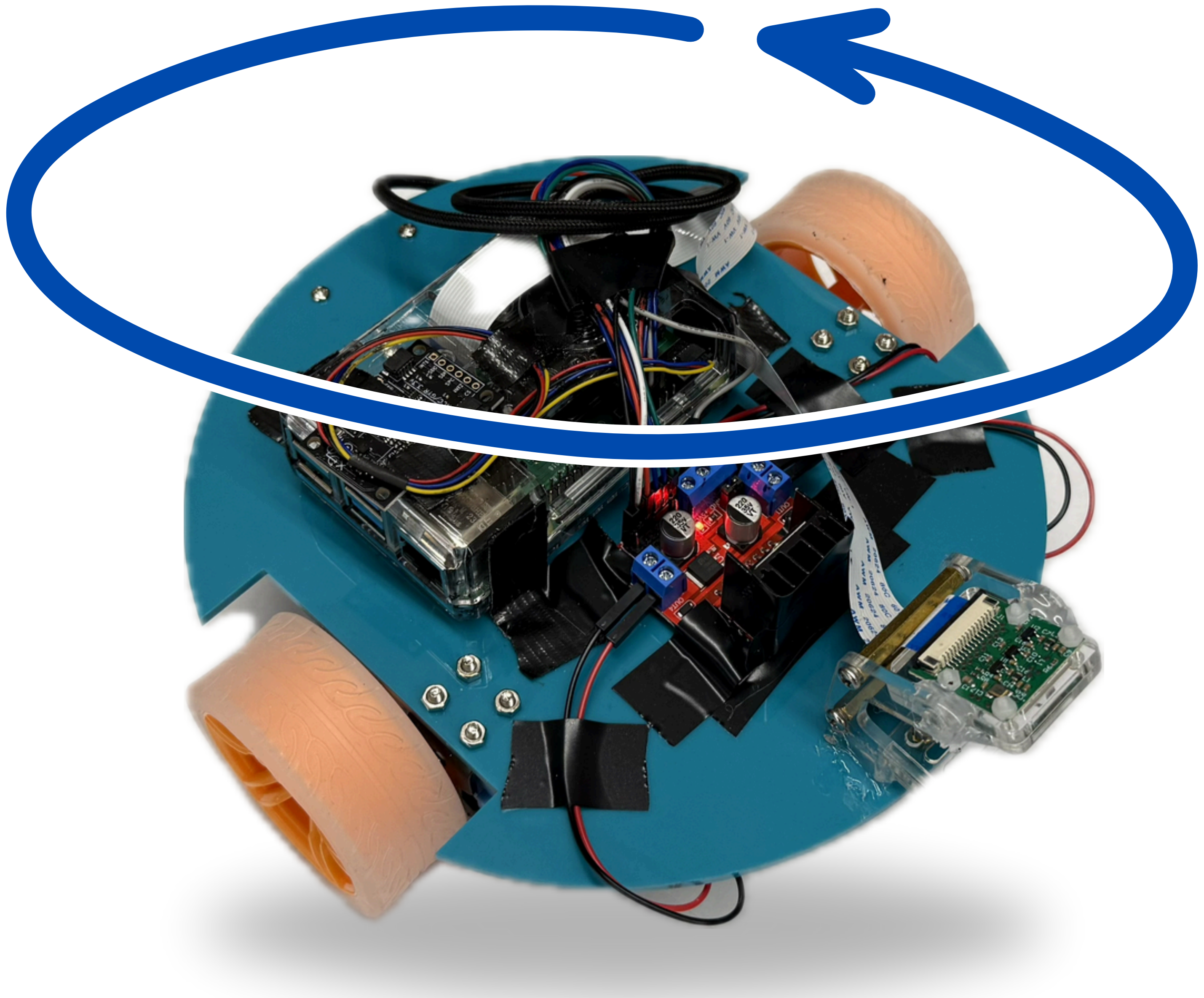
- Commands
  - L: Turn left
  - F: Go forward
  - R: Turn right
- Connected to the computer via TCP
- Controller replaces need for player to use keyboard while playing



Project Network



IMU-BASED TURNING



End of Q1

- *Turned based off an arbitrary timer*

Middle of Q2

- *Able to turn roughly 90 degrees based on the Gyroscope using BerryIMU library*

End of Q2

- *Fixed minor unreliability, due to a sample speed bug*

VOICE-COMMANDS

Allows the player to control the Maze Navigator using natural sentences.

- OpenAI’s Whisper:
  - Offline, tiny model.
  - Very fast, but less accurate.
- Fuzzy matching is used to make up for inaccuracies.
- Parse sentences looking for keyword matches.
- Words that are “close enough” are compared and the closest match is returned.

TECH STACK

Language: Python

MAZE PROGRAM

1. PyQt5: GUI framework.
2. Maze Algorithm: Recursive backtracking.
3. Socket library: TCP connection (Client).
4. Speech Recognition library: Whisper.
5. difflib library: Fuzzy matching.
6. PyInstaller: Executable.

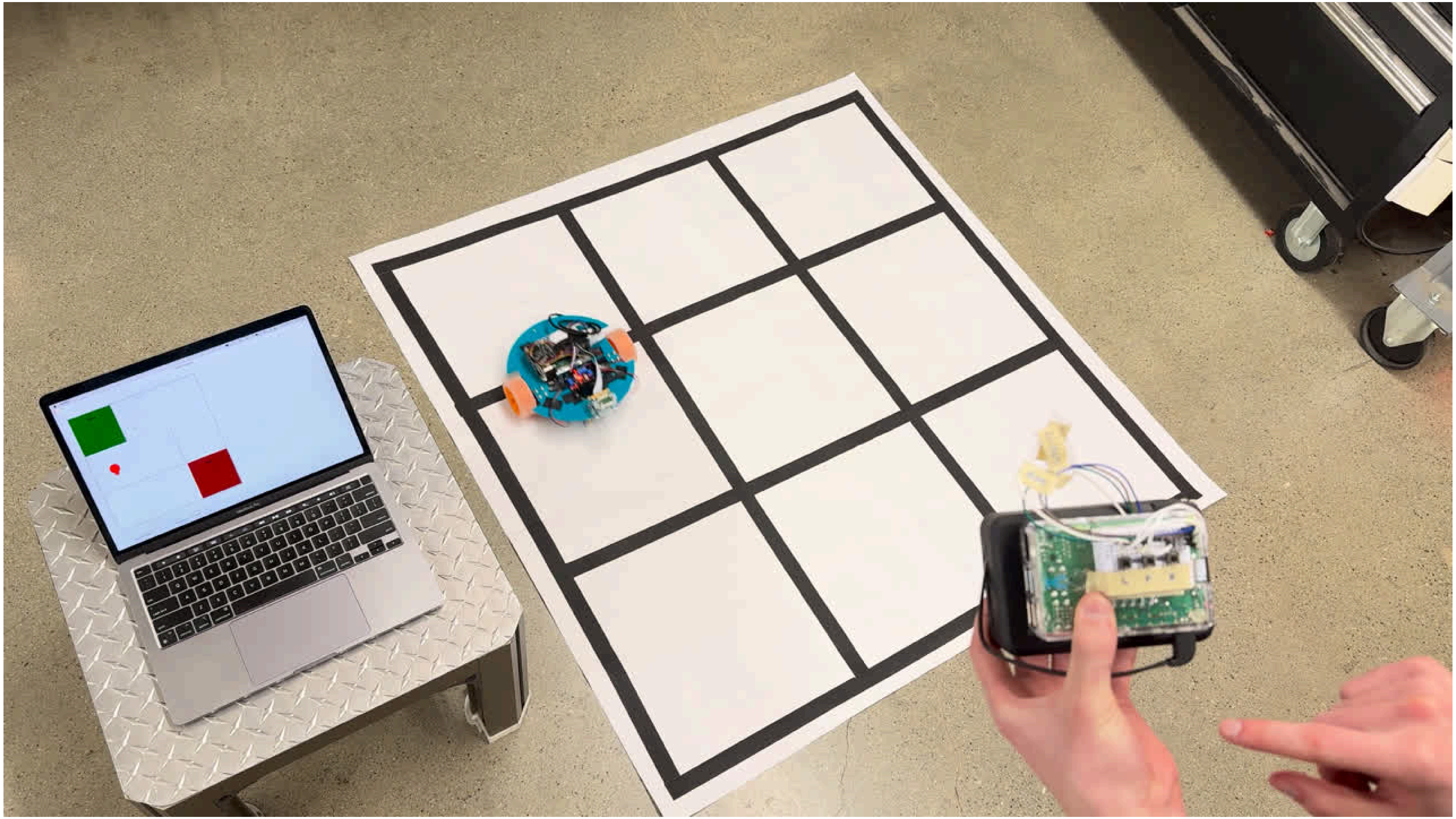
MAZE NAVIGATOR

1. RPi.GPIO: Motor control.
2. Socket library: TCP connection (Server).
3. PiCamera2 and OpenCV: Capture and process frames for line detection.
4. Custom BerryIMU library: Gyroscope data.
5. Systemd: Run program on boot.

CONTROLLER

1. RPi.GPIO: Interface with physical buttons.
2. Socket library: TCP connection (Server).
3. Threading: Periodically sends/receives data to monitor connection with maze program.
4. Systemd: Run program on boot.

LIVE DEMO



TIMELINE

Q1 Week 1 - 5: Learning & Project Proposal

Q1 Week 6 - 7: Create Maze Program & Initial Prototype

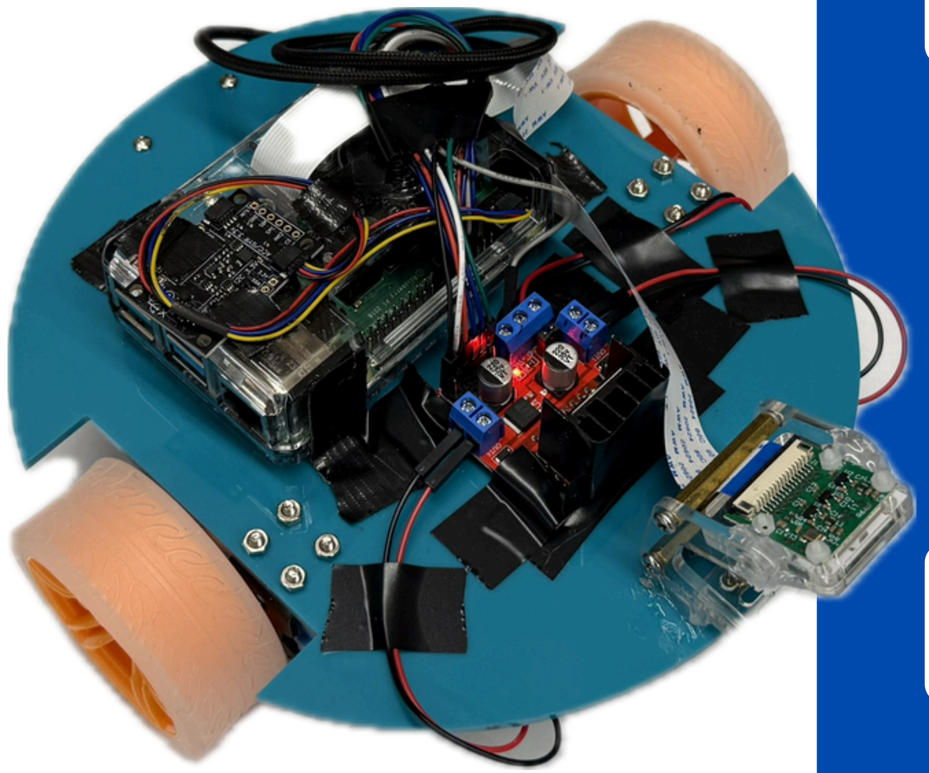
Q1 Week 8 - 10: Create Project Network, attach Camera & IMU

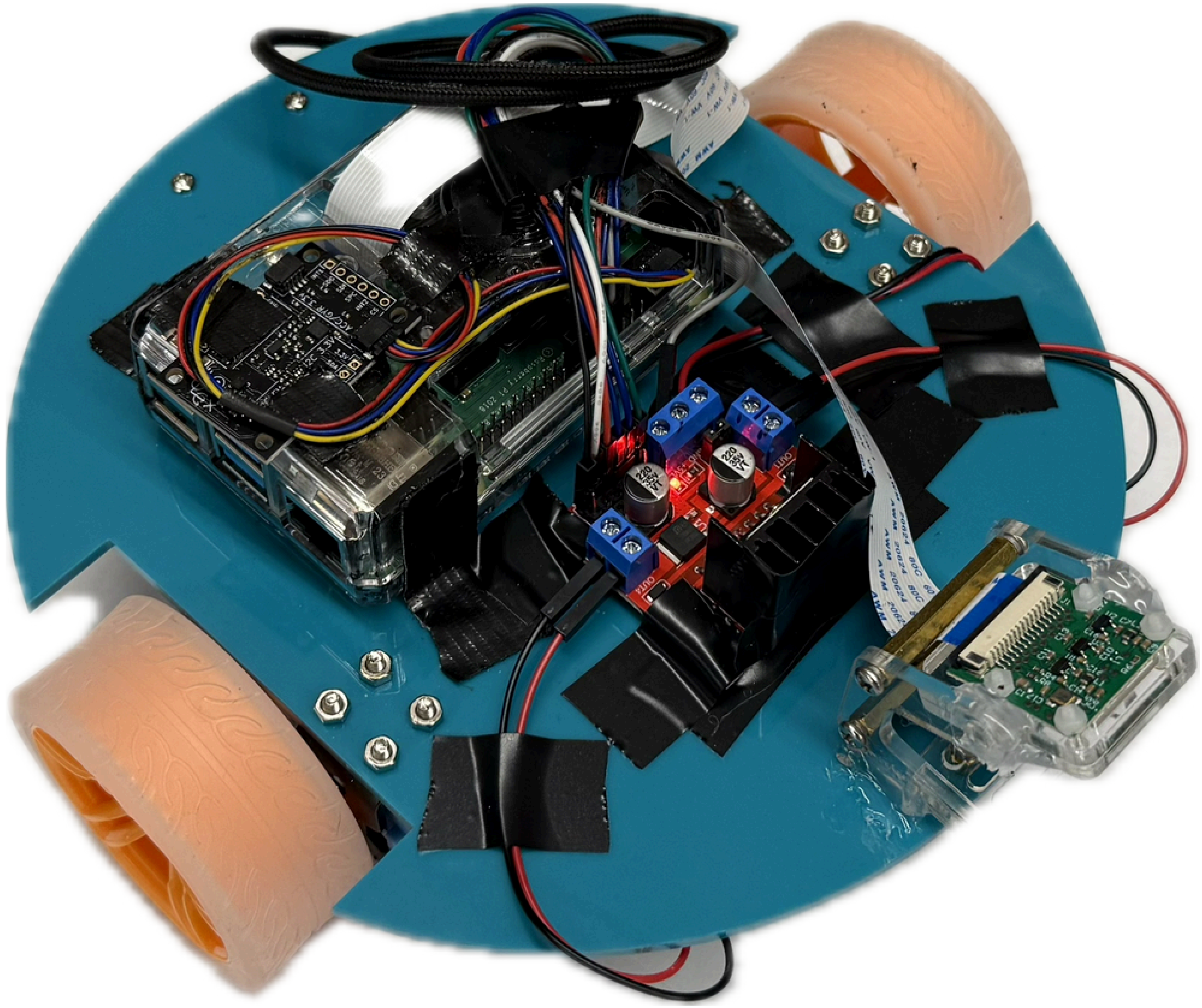
Q2 Week 1 - 2: Programmed Computer Vision, Voice Recognition & IMU

Q2 Week 3 - 5: Continued Programming, Constructed Controller & Reassembled Car

Q2 Week 6 - 7: Bug Fixing and Fine Tuning Car Movement Parameters

Q2 Week 8 - 10: Revitalized GUI & Finalized Car Design





WORK DISTRIBUTION

Hardware: Torin & Emanuel

Computer Vision: Steve, Aamina, and Torin

IMU: Torin & Emanuel

Voice Recognition: Torin

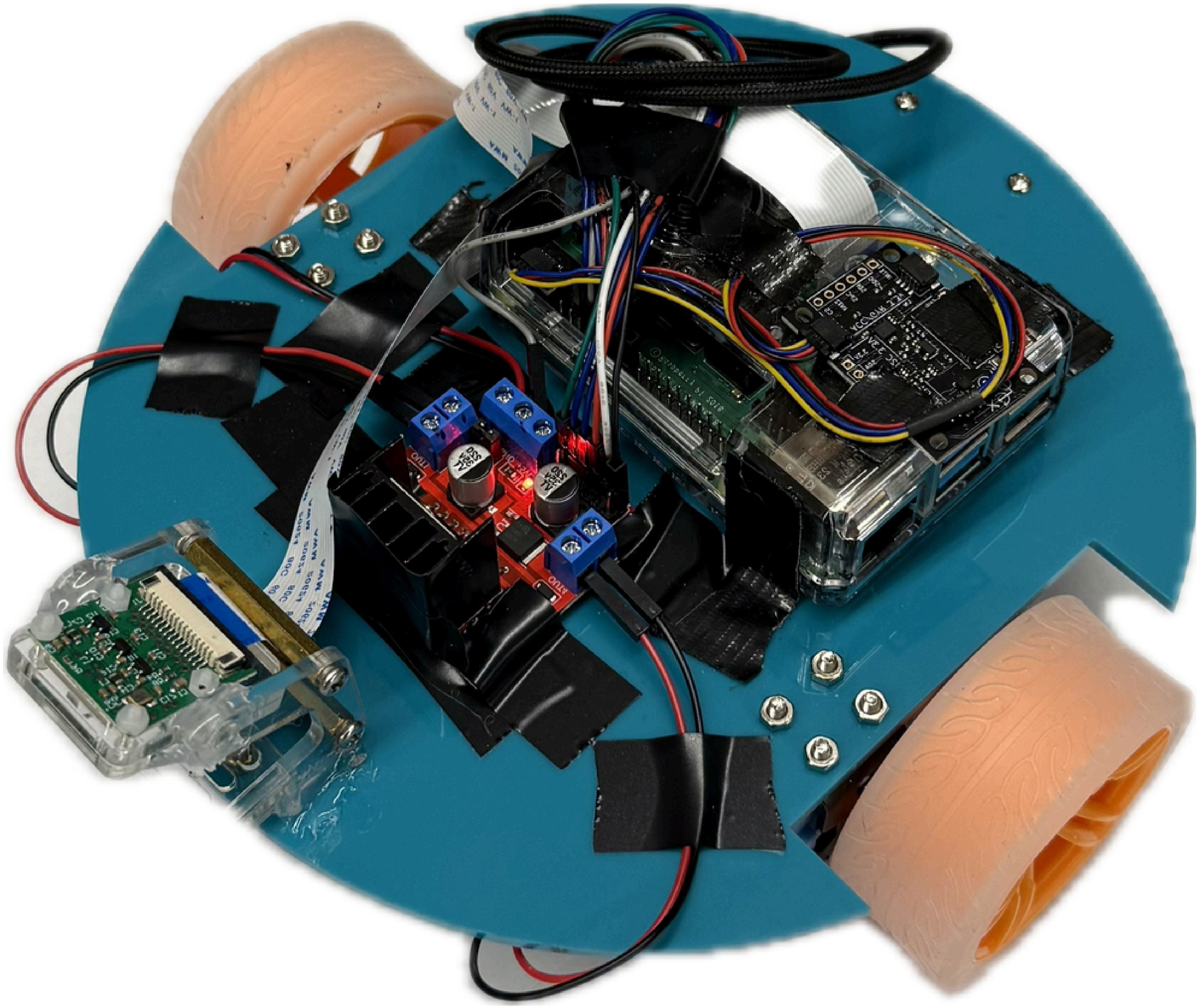
Networking: Torin, Emanuel, and Aamina

User Manual & Final Report: Aamina, Emanuel, Steve, and Torin

Maze Program: Emanuel, Steve, and Torin

FUTURE WORK

- *Improve aesthetics of the hardware*
  - *Cover for the navigator*
  - *custom hardware for controller*
- *Automatic resetting*
  - *At a press of a button, the car can automatically navigate back to the start cell*
- *Improved motors*
  - *Wider range of speed control*
  - *Encoders for better feedback*



QUESTIONS?