Carleton University
Department of Systems and Computer Engineering

The Autocart: An Autonomous Shopping Cart with Indoor Localization

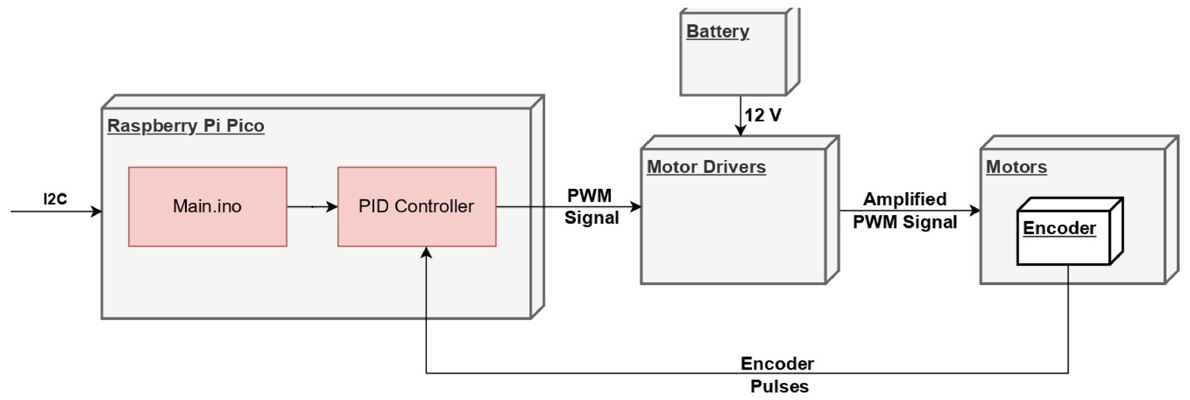
Ahmed Abdelaziz, Nadia Ahmed, Bilal Chaudhry, Cam Frohar, Mohamed Kaddour, Sebastian Marleau, Akuei Minyang, Kousha Motazedian, Bren Gelyn Padlan, Prianna Rahman
Supervised by: Dr. Ramy Gohary and Dr. Ian Marsland

SYSC 4907
Groups 53 and 54

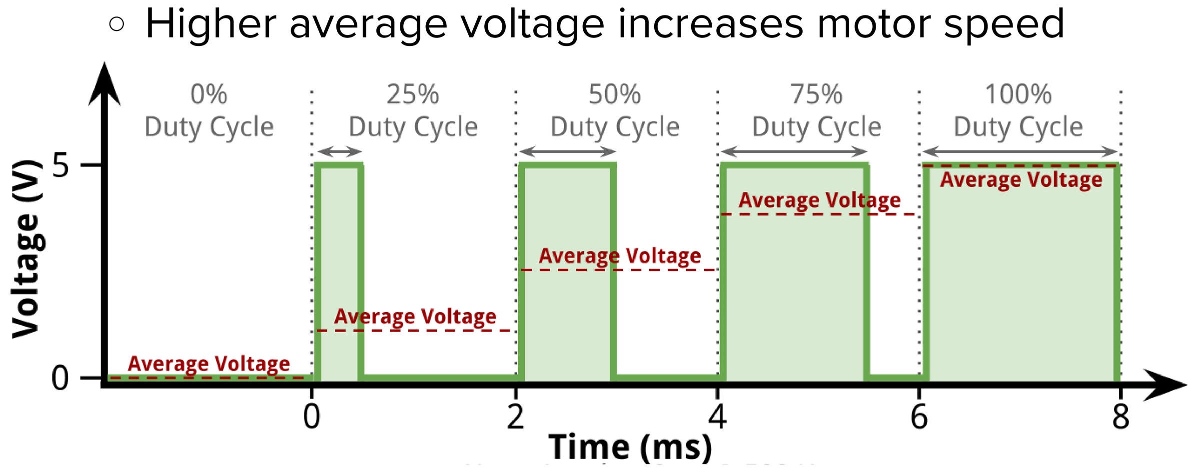
BACKGROUND

- The rise of smart systems in everyday life has extended to grocery shopping, with technology aiming to provide convenience through autonomous systems.
- The key challenge for implementing an autonomous cart is achieving accurate indoor localization relative to grocery aisles.
- This project aims to develop an autonomous cart capable of navigating within a store and recognizing its location.
- The project is divided into two main components: autonomous driving, which includes object detection, avoidance, and navigation, and indoor localization, focusing on mapping and establishing the cart and user's position within the store.

MOTORS

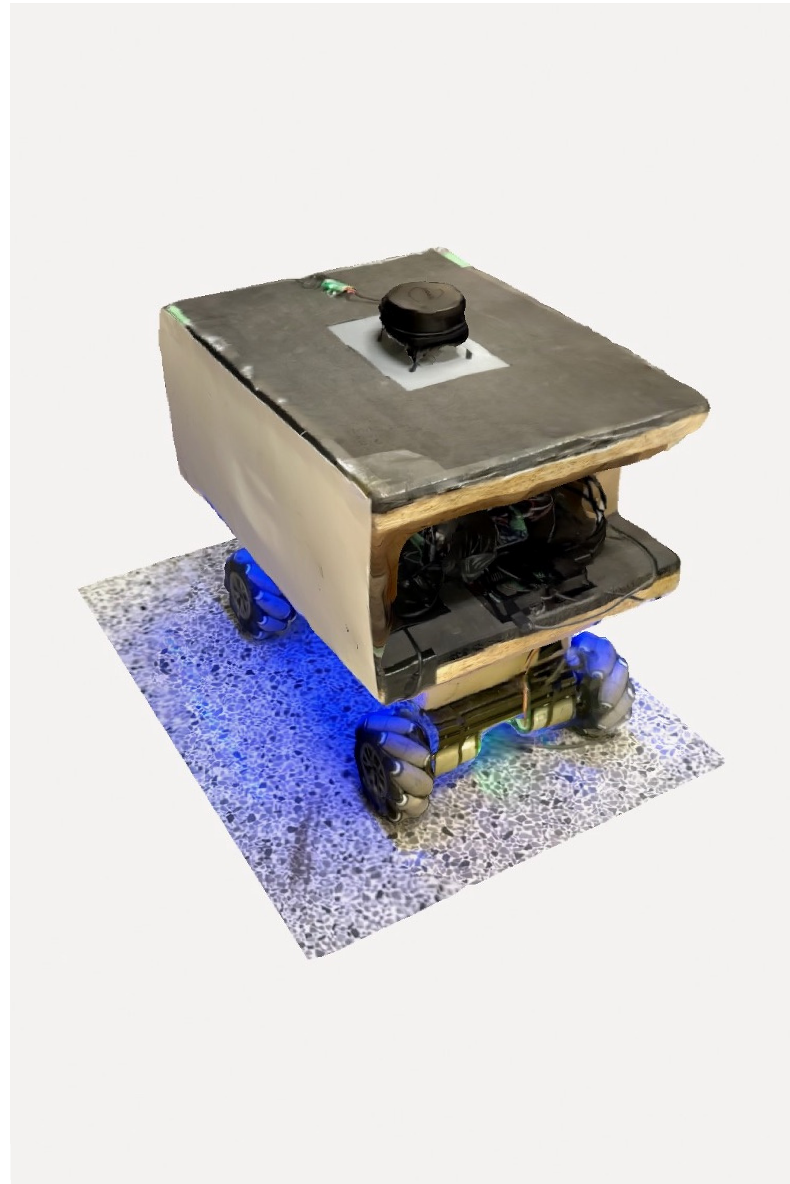


- **Encoder:** Device connected to the motor that sends multiple signals for each rotation of the motor shaft.
  - Track motor speed, perform precise turns
- **PID Controller:** Algorithm used to dynamically adjust PWM signal to drive motors at the **set-point** speed.
  - **P**roportional, **I**ntegral and **D**erivative Components
- **PWM (Pulse Width Modulation):** Provides control over average voltage supplied using the Duty Cycle
  - Higher average voltage increases motor speed

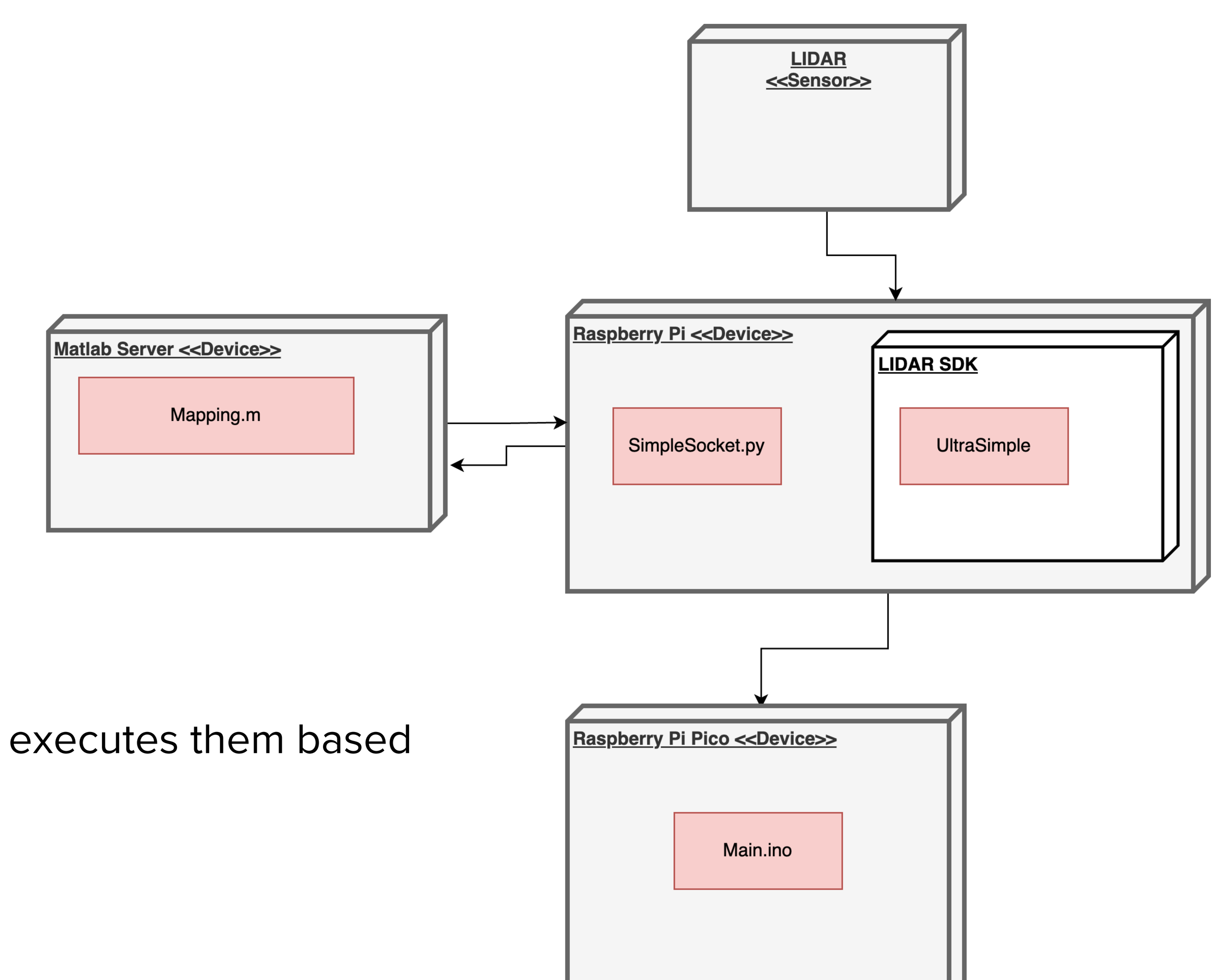


- **Drivers:** Combine PWM signal from the board with supply voltage from battery to drive the motors.
  - Direction control using **H-Bridge**
  - Overcurrent Protection
- **Interface:** Receives an angle and the desired state over I2C protocol. The angle is used to direct cart towards the Point of Interest. The state is utilized to tell the motors when to stop in the case of a detected object or to turn at a point of interest.

OUR PROTOTYPE



SYSTEM ARCHITECTURE



Raspberry Pi "Main Brain"

Runs both the LiDAR SDK and the Python Program

- **LiDAR SDK:** The software provided by the LiDAR manufacturer to interface with the LiDAR
- **UltraSimple**: The program we use to collect the data from the LiDAR
- **SimpleSocket.py:** This program collects information from UltraSimple and communicates with the Matlab Server and the Raspberry Pi Pico

Raspberry Pi Pico

The microcontroller used to control the Motors

- **Main.ino:** Receives navigation commands over I2C and executes them based on the current state of the motors

LiDAR

Light **D**etection and **R**anging module

- Scans 360° with a sampling rate of 10 Hz, gathering distance, angle and quality for each object within 6 m

Matlab Server

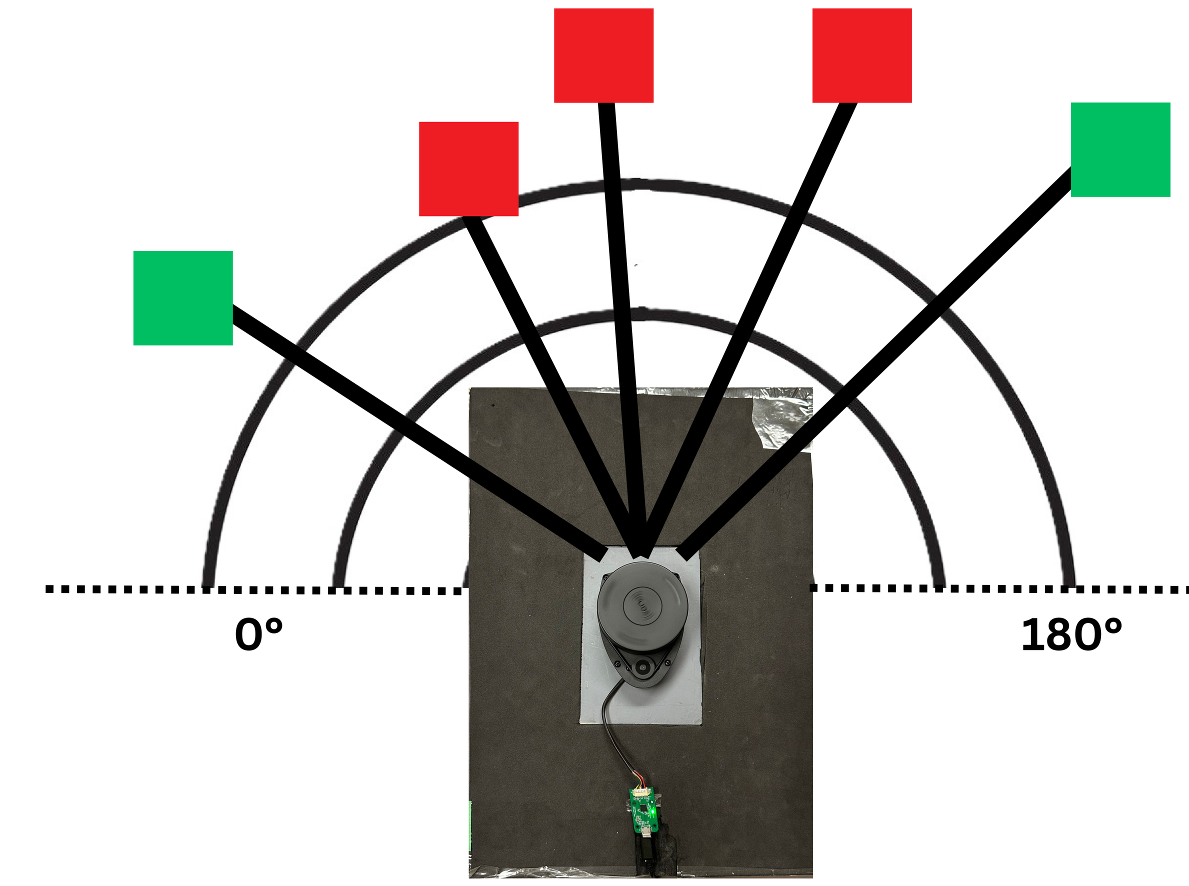
Receives LIDAR information from the Raspberry Pi

- **Mapping.m:** Interprets incoming LiDAR data over TCP/IP to determine and visualize the cart's approximate location. Sends back information on the cart's location

Communication

1. LiDAR SDK sends the distance and angles of objects around it to the Python Program (SimpleSocket.py).
2. The Python Program will send a list of that information to the MATLAB Server via socket and return the cart's coordinates and the angle it is facing.
3. The Python Program sends the Raspberry Pi Pico commands for the motor's movement.

COLLISION AVOIDANCE



Goal:

- Ensure safe navigation by accurately assessing obstacles in the robot's path and determining whether to proceed or halt accordingly.

Algorithm:

- **Real-Time-Processing:**
  - Process each rotation of LiDAR data in real-time.
- **Detection Range:**
  - 0° to 180°
- **Trigonometric Calculations:**
  - Calculate the adjacent distance to detected objects based on the angle and hypotenuse provided by the LiDAR sensor.
  - $Adjacent = cos(\Theta) \cdot hypotenuse$
- **Decision Making:**
  - Identify obstacles located directly 50 cm ahead of the robot.
  - Analyze the adjacent distances obtained from LiDAR data to evaluate potential obstacles in the robot's path.

PROJECT GOALS

Objective	Description
Dynamic Movement and Control	Precise forward and backward movement, with accurate steering, variable speed settings, and responsive braking.
Autonomous Navigation	Autonomously navigate the store environment, detecting and avoiding obstacles in real-time, and planning efficient paths to designated destinations, including pick-up and drop-off areas.
Dynamic Mapping	Map its environment and accurately identify its location in real-time in reference to the map.
Indoor Localization	Determine the user's position with respect to the environment.
Object Detection	The vehicle must rapidly detect objects and obstructions, with object detection times under 1 second.

The Autocart: An Autonomous Shopping Cart with Indoor Localization

Carleton University
Department of Systems and Computer Engineering

Ahmed Abdelaziz, Nadia Ahmed, Bilal Chaudhry, Cam Frohar, Mohamed Kaddour, Sebastian Marleau, Akuei Minyang, Kousha Motazedian, Bren Gelyn Padlan, Prianna Rahman

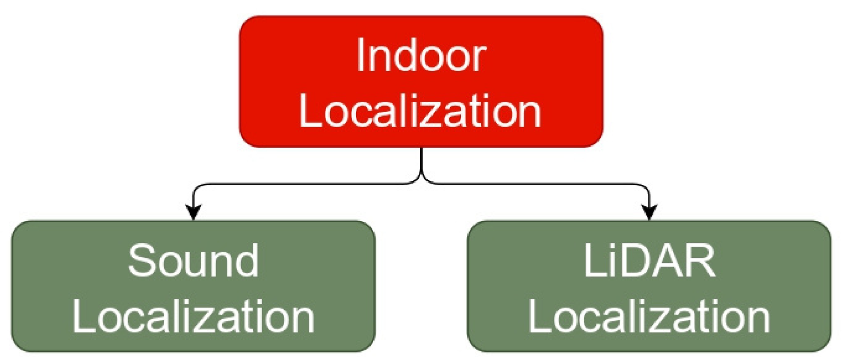
Supervised by: Dr. Ramy Gohary and Dr. Ian Marsland

SYSC 4907
Groups 53 and 54

INDOOR LOCALIZATION

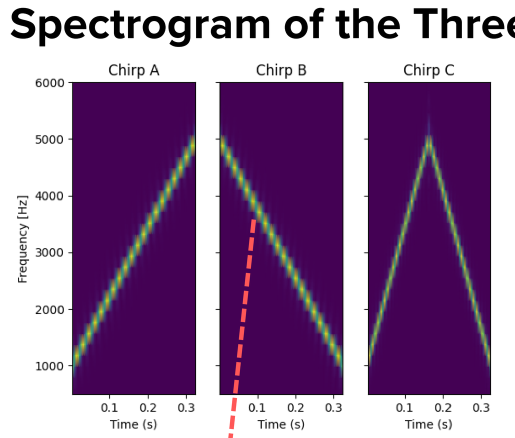
Motivation

- Global Positioning System (GPS) does not work well indoors, since buildings and walls block and weaken GPS signals significantly
- Some applications need more accuracy than what GPS provides



SOUND LOCALIZATION (CONT'D)

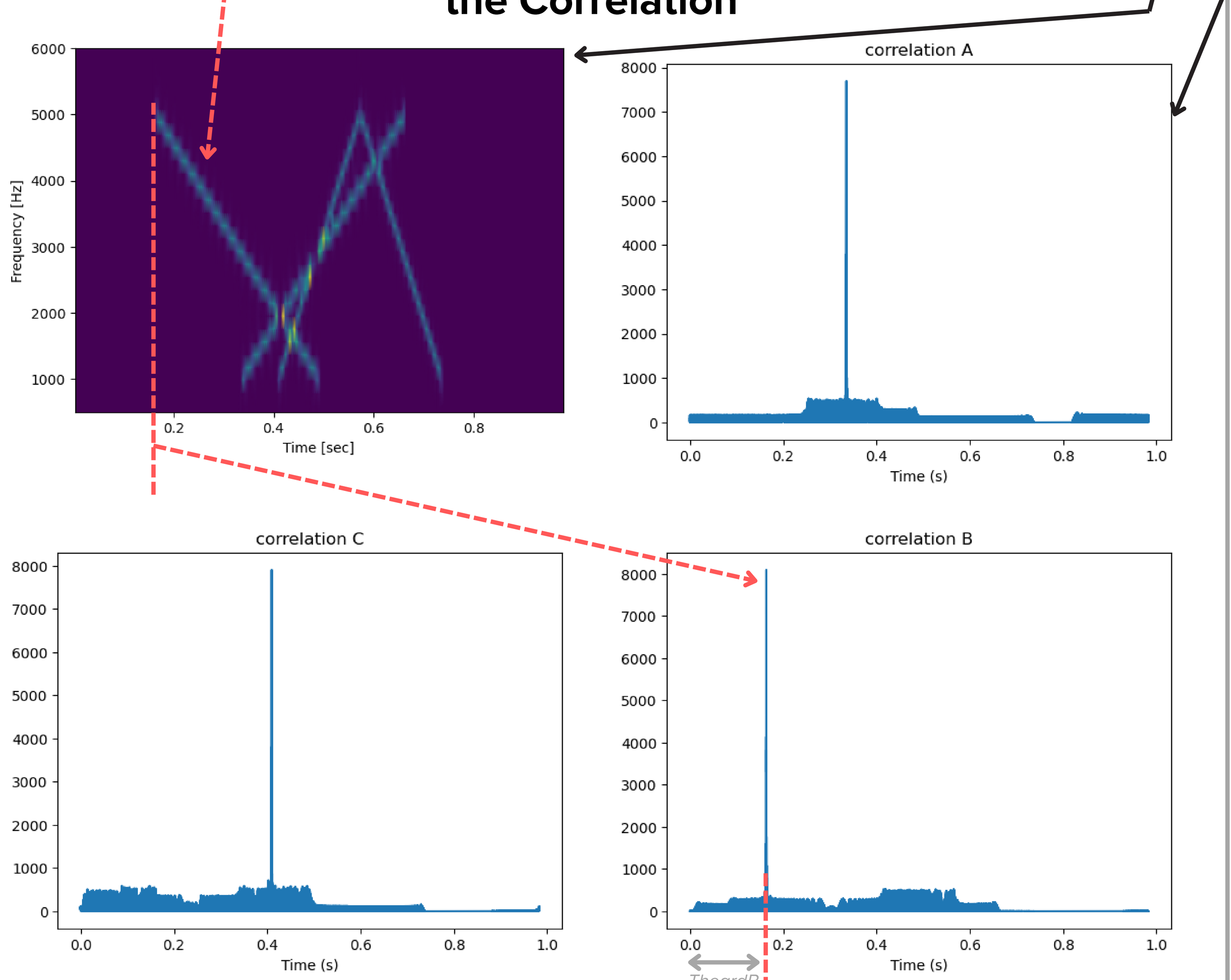
Spectrogram of the Three Chirps



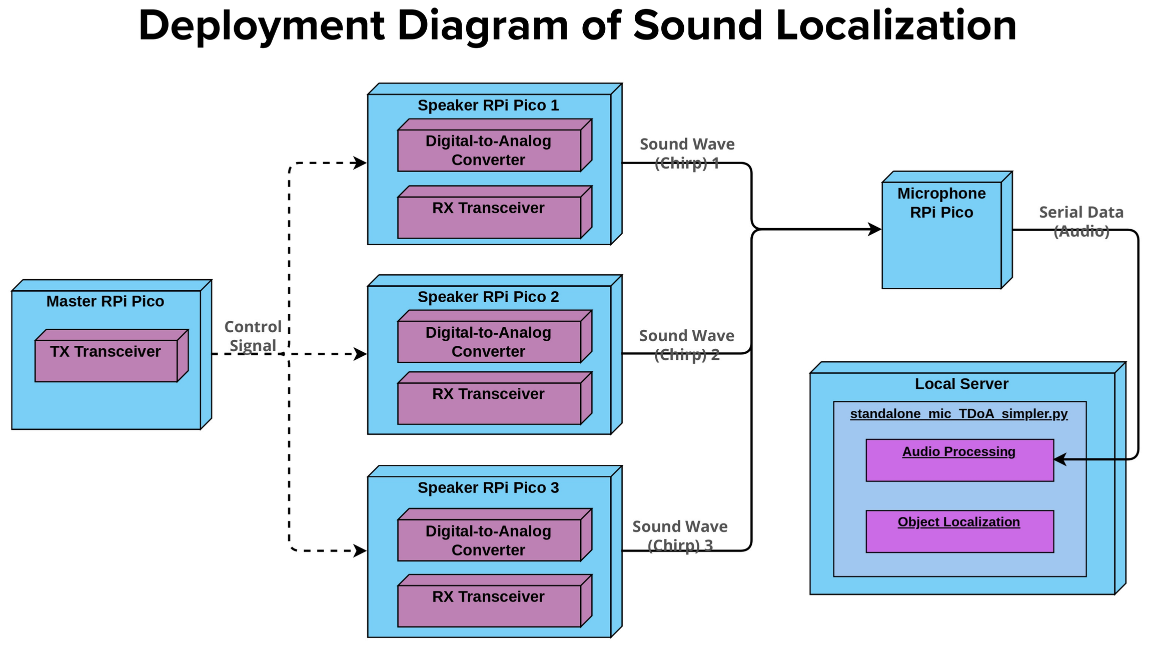
Correlation Equation

$$[f(t) \star g(t)](t) = [\overline{f(-t)} * g(t)](t)$$

Spectrogram of a Recorded Sound, and How it Relates to the Correlation

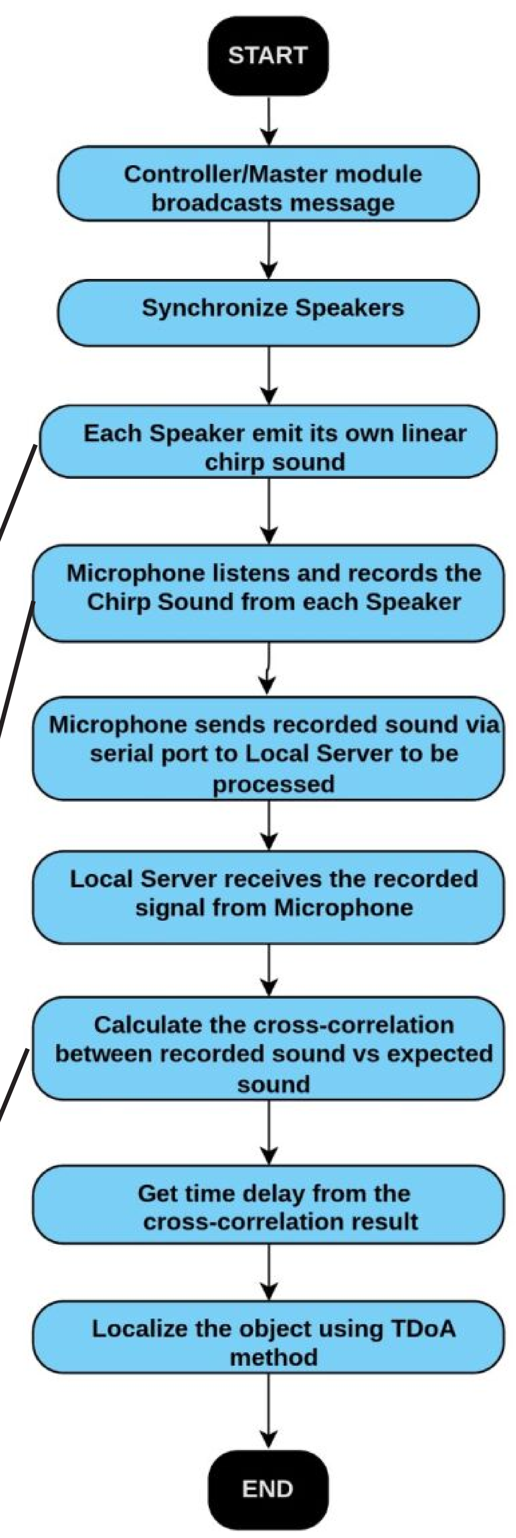


Deployment Diagram of Sound Localization



SOUND LOCALIZATION

Flow Diagram of Sound Localization



Time of Arrival (ToA)

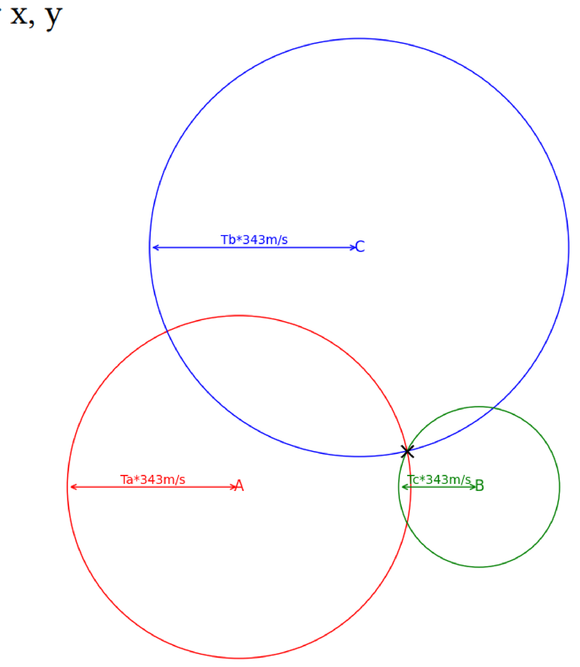
A, B, C = positions of speakers
T_s = time the sound started playing
T_{heard} = time the sound was heard at the microphone

$$T_A = T_{\text{heardA}} - T_s$$
$$T_B = T_{\text{heardB}} - T_s$$
$$T_C = T_{\text{heardC}} - T_s$$

$$c = 343\frac{m}{s}$$
$$R_A = T_A \cdot c$$
$$R_B = T_B \cdot c$$
$$R_C = T_C \cdot c$$

$$R_A = \sqrt{(A_x - x)^2 + (A_y - y)^2}$$
$$R_B = \sqrt{(B_x - x)^2 + (B_y - y)^2}$$
$$R_C = \sqrt{(C_x - x)^2 + (C_y - y)^2}$$

Solve for x, y



Time Difference of Arrival (TDoA)

T_s is not known

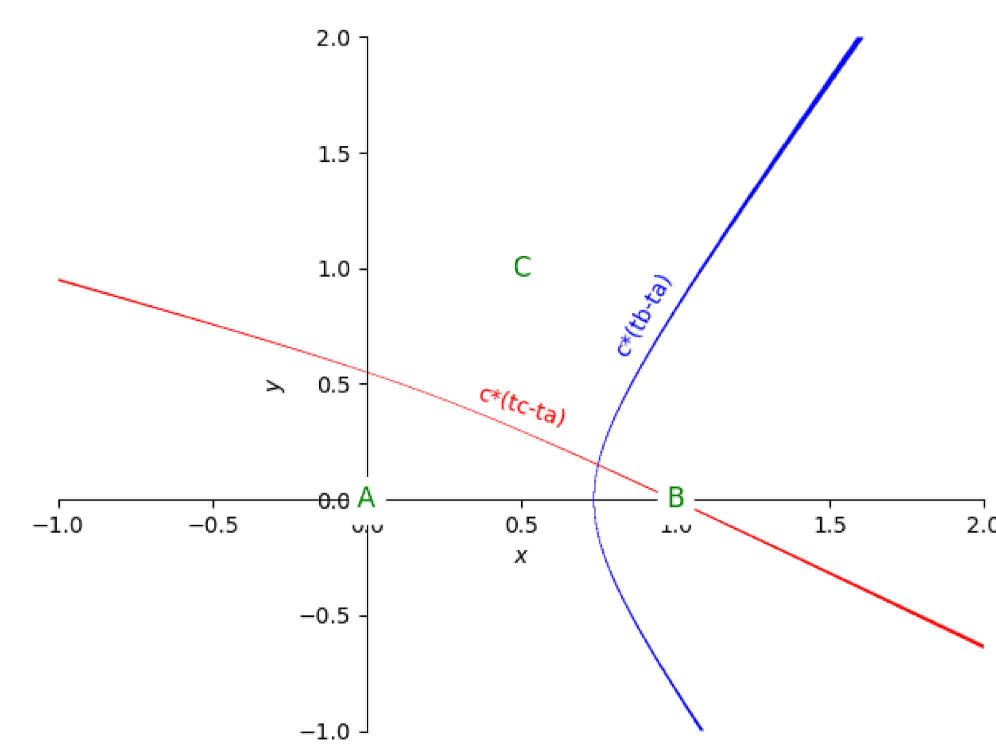
$$T_{\text{heardA}} = t_A = \frac{R_A}{c} + T_s$$
$$T_{\text{heardB}} = t_B = \frac{R_B}{c} + T_s$$
$$T_{\text{heardC}} = t_C = \frac{R_C}{c} + T_s$$

$$t_B - t_A = \frac{R_B}{c} + T_s - \left(\frac{R_A}{c} + T_s\right) \qquad \textit{Ts cancels out}$$
$$t_C - t_A = \frac{R_A}{c} + T_s - \left(\frac{R_B}{c} + T_s\right)$$

$$c \cdot (t_B - t_A) = R_B - R_A$$
$$c \cdot (t_C - t_A) = R_C - R_A$$

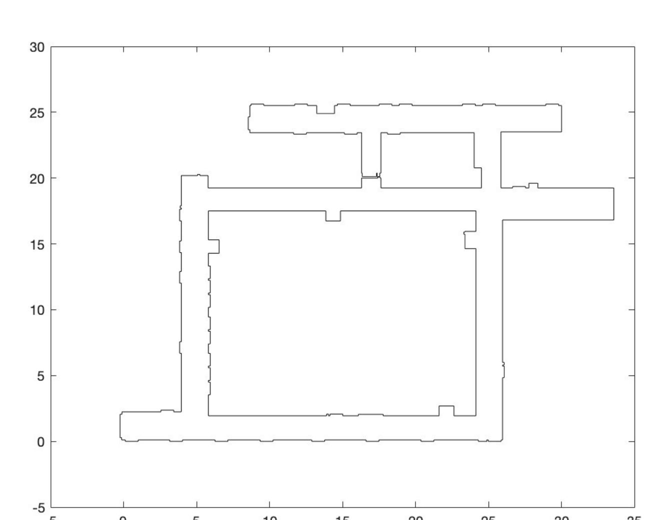
$$c \cdot (t_B - t_A) = \sqrt{(B_x - x)^2 + (B_y - y)^2} - \sqrt{(A_x - x)^2 + (A_y - y)^2}$$
$$c \cdot (t_C - t_A) = \sqrt{(C_x - x)^2 + (C_y - y)^2} - \sqrt{(A_x - x)^2 + (A_y - y)^2}$$

Solve for x,y

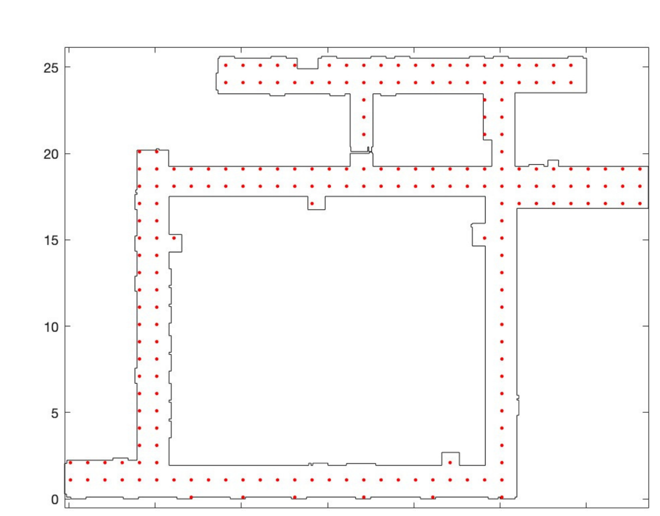


LiDAR LOCALIZATION

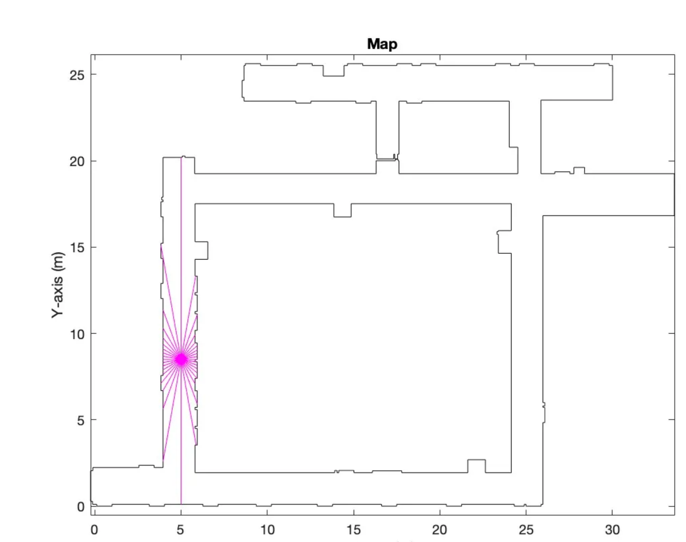
(a) Plain map of Mackenzie hallways



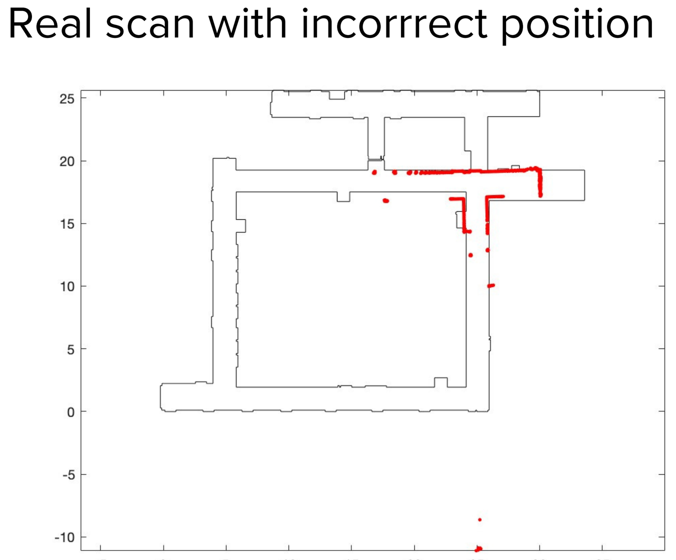
(b) Plain map with reference positions



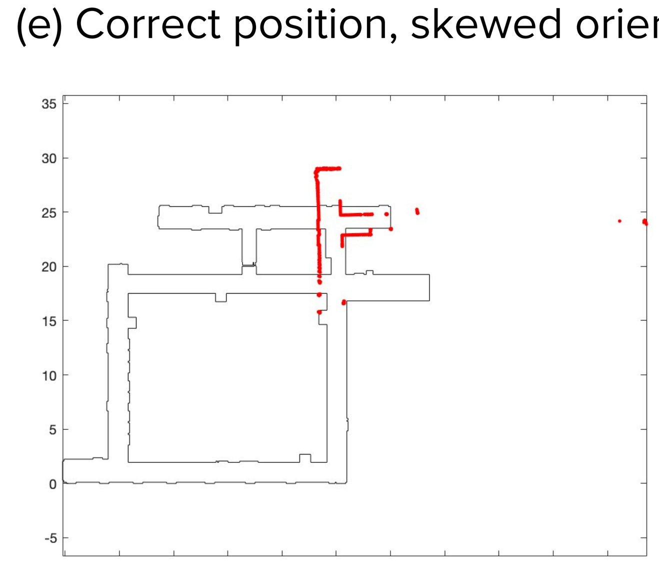
(c) LiDAR scan of simulated test point



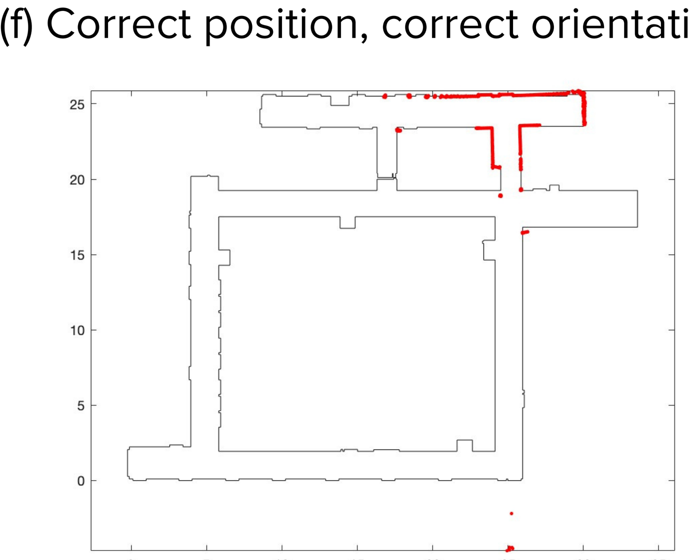
(d) Real scan with incorrrect position



(e) Correct position, skewed orientation



(f) Correct position, correct orientation



LiDAR LOCALIZATION (CONT'D)

Introduction

Light Detection and Ranging (LiDAR) is a sensor technology used to measure distances by analyzing the time it takes for emitted light pulses to return after reflecting off surrounding objects. The RPLiDAR A1M8 was used in the project and was very helpful due to its range and robustness. The sensor has a variety of uses but has been used to localize/position the cart within the known space. If the cart is localized, it can navigate through 2D space. Achieving a high standard of precision in localization allows the cart not only to maneuver through aisles for collecting groceries tailored to user preferences, but also to be adapted for a wide array of applications.

Positioning Algorithm

(a) Fourth floor Mackenzie was measured manually to map out the floor, realizable through MATLAB
(b) Populate the map with reference positions throughout, simulating every possible position the LiDAR can be in within 10 cm
(c) Reference positions have a LiDAR ray emitting every 10 degrees, creating reference LiDARs all throughout the map
(d) Actual LiDAR measurement is compared with all reference positions, a metric is calculated for the error at every reference position
(e) Reference position with lowest error is selected, and LiDAR ray is offset to get the best orientation value for the real LiDAR, removing any skew
(f) LiDAR is localized by figuring out its correct position and orientation

Error for Initial Position:

$$\text{metric} = \sum_{\theta=0}^{360} \left| \sqrt{x_{\text{actual}}^2 + y_{\text{actual}}^2} - \sqrt{x_\theta^2 + y_\theta^2} \right| \times \text{angle\_frequency}_\theta$$

Correction for Non-Initial Position

$$\text{metric} = \sum_{\theta=270}^{90} \left| \sqrt{x_{\text{actual}}^2 + y_{\text{actual}}^2} - \sqrt{x_\theta^2 + y_\theta^2} \right| \times \text{angle\_frequency}_\theta$$

Reference Distance Equation

$$\text{test\_pos}_x = \text{structure}_x - \text{dist}_j \cdot \cos(j^\circ) \qquad j = 0:10:360$$
$$\text{test\_pos}_y = \text{structure}_y - \text{dist}_j \cdot \sin(j^\circ)$$

CONTRIBUTIONS

Motors and Mechanics: Akuei, Bilal, Mohamed
LiDAR Localization: Ahmed, Akuei, Nadia, Sebastien
Sound Localization: Ahmed, Bren, Prianna, Sebastien
Collision Avoidance: Cam, Kousha, Nadia
Integration: Ahmed, Cam, Kousha, Mohamed, Sebastien