

# INF2004-Embedded Systems Programming

GABRIEL LAU HIAN (2200749), JOYCE TENG MIN LI (2203242), ONG SHI YA (2201282),  
WONG KAR LONG KEAGAN (2200517)

## Navigation and Mapping

Keagan

Technologies we plan to use:

- HC-SR04 Ultrasonic Sensor
- IR Line Tracking Module

### **HC-SR04 Ultrasonic Sensor**

Used to detect and avoid obstacles on the maze tracks. Should an obstacle be detected, the car should automatically take another path.

### **IR Line Tracking Module**

Uses light as reflection in order to detect the black coloured lines in order to define the path to take. Line following algorithm should decide whether the robot will move left, right or go straight based on whether they are able to detect the “walls”.

### **Obstacle collision**

The ultrasonic sensor should detect whether there is an obstacle in front of the car. The data is received by pulsing and is displayed to us in cm. Upon receiving the data, it should be used accordingly in the navigation algorithm to decide next best course of action

### **Wall detection**

The 2 IR line sensors on each side of the car will receive input on whether there is a black line detected. If it is detected on either side or both sides, data should be passed to the algorithm to decide on next path to take.

### **Navigation Algorithms**

Motion Planning: Using the HC-SR04 to detect whether the obstacle is within a certain threshold range. If it is too close, the car will react by going for the next best path. Black tape will be used as the walls, in which case the IR Line Tracking Module should be able to recognise it and decide which way to go. The HC-SR04 must be able to detect obstacles that appear in the maze and choose the next best path for completion.

### **Mapping Algorithms**

A 2D occupancy grid map based on simulated sensor measurements can be set up and implemented. By initialising the grid we would be able to break the maze down into specific sections which will then be able to decide which path to go if it is blocked. This data will be mapped out and show the best path to take.

### **Example Steps:**

1. Place the car in the starting position.
2. Trigger HC-SR04 sensor to measure distance periodically.
3. IR Line tracking module checks whether an obstacle is in front.
4. If distance to obstacle is:
  - a. Lower than threshold, stop the car.
  - b. Higher than threshold, continue moving forward
5. If obstacle is:
  - a. Right side, turn left
  - b. Left side, turn right
  - c. Right, left and front, reverse the car and turn till no obstacle in front
6. Maintain direction by checking whether left and right side have obstacle

### **Example Algorithms**

- Dijkstra's and A\*
- Wall follower
  - Follows the left or right side of a maze all the way to the end
  - Depends on maze complexity, used for simpler mazes
  - Depth first in order traversal
- Trémaux's
  - Marks the entrances to decide whether to carry on
  - 100% success rate
  - Can be very slow
- Floodfill algorithm

## **Barcode Recognition**

### **Joyce**

The primary objective of integrating barcode recognition is to empower the robot car with the ability to detect and interpret barcodes on the ground. To accomplish barcode recognition, we plan to leverage infrared sensors that can effectively capture and interpret barcode patterns.

### **Code 39**

Every Barcode is 9 bars

Start with black end with black

5 black bar (+0-9) , 4 white bar (+0,+10,+20,+30)

### **Code Structure**

- Initialisation
  - ADC for the IR sensor and GPIO for the object sensor are initialised to begin barcode detection
- Detection
  - IR sensor reads analog signals representing black and white bars of a barcode

- When an object is detected, an interrupt is triggered, starting the barcode reading process
- Reading process
  - The system reads the duration of black and white bars based on the IR sensor's analog signals
  - Duration are measured by using a timer
- Output
  - The decoded character are the printed out representing the data encoded in the barcode

## PID Controller

### Gabriel

A Proportional-Integral-Derivative (PID) controller in a robot car, is a feedback control system used to adjust the car's speed and direction based on the difference between its desired setpoint and the resultant setpoint. This can be achieved by using the IR infrared rotary speed sensing module to measure the actual rotation of the motor. The PID controller algorithm then calculates the wheel's expected rotation speed and compares it to the actual IR reading and adjusts the duty cycle to match its desired speed. This will be done independently for both wheels to allow for direction changes. Our intended PID controller algorithm will not be using any libraries and will be coded by us.

## Communications & User Interface

### Shiya

Technologies we plan to use:

- Onboard Wifi Module: Infineon CYW43439
- Lightweight IP Stack: LwIP Libraries
- Node.js

#### **WIFI**

- pico/cyw43\_arch.h

The Infineon CYW43439 module is utilised for establishing connectivity with the WiFi network.

#### **Lightweight IP Stack**

LwIP libraries are employed to access essential networking services, including HTTP and TCP.

#### **TCP Server**

- lwip/pbuf.h
- lwip/tcp.h

For effective communication, TCP is utilised for the seamless transfer of packets.

### **HTTP Server**

- lwip/apps/httpd.h
- pico\_lwip\_makefsdata

An HTTP server is created on Pico W for data display and handling. JavaScript is used to fetch data from the api server asynchronously at fixed intervals. CGI facilitates calling functions upon interacting with UI buttons. In the absence of a connected TCP client, SSI is employed for manual refresh to update data.

### **Express**

- Nodejs (cors, express, http, net)

#### **Setup:**

1. Make sure that the TCP server is running.
2. Navigate to the `/driver/wifi/express` directory.
3. In the terminal, run the command `node index.js` to start the Express server.

Externally, Express serves as middleware to manage raw packets transmitted from Pico W TCP. The Net Module is used to establish a TCP client. Express also functions as an API server, facilitating data retrieval through Pico W HTTP.