Final Report

Micro-mouse: Initialisation, Calibration, Localisation, Surrounding Awareness, Movement Decisions and Path Optimisation.

Prepared by:

Kashan Pillay PLLKAS005

- Executive Summary: Problem Description, Scope, Limitations Requirements and Specifications
- Subsystem Design (Navigation): Design, Mitigation of Limitations
- Subsystem Design (Optimisation): Design, Mitigation of Limitations
- Conclusion: Recommendations

Priya Moodley MDLPRI040

- Executive Summary: Achievement Matrix
- Subsystem Design (Navigation): Acceptance Testing Procedure, Critical Analysis
- Subsystem Design (Optimisation): Acceptance Testing Procedure, Critical Analysis
- Conclusion: Conclusion

Prepared for:

EEE3097S

Department of Electrical Engineering

University of Cape Town

October 22, 2024

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

October 22, 2024

Kashan Pillay — Date

October 22, 2024

Priya Moodley — Date

Contents

1 Executive Summary **1**
- 1.1 Problem Description . . . 1
- 1.2 Scope and Limitations . . . 1
- 1.3 Requirements . . . 2
- 1.4 Specifications . . . 2
- 1.5 Achievements Matrix . . . 2

2 Subsystem Design: Navigation **3**
- 2.1 Design . . . 3
  - 2.1.1 Line Following . . . 3
  - 2.1.2 Surrounding Awareness . . . 5
  - 2.1.3 Decision Based Movements . . . 7
- 2.2 Mitigation of Limitations . . . 8
- 2.3 Acceptance Testing Procedure . . . 9
  - 2.3.1 Tests . . . 9
  - 2.3.2 Critical Analysis of Testing . . . 10

3 Subsystem Design: Optimisation **13**
- 3.1 Design . . . 13
  - 3.1.1 Optimisation Algorithm . . . 13
- 3.2 Mitigation of Limitations . . . 15
- 3.3 Acceptance Testing Procedure . . . 15
  - 3.3.1 Tests . . . 15
  - 3.3.2 Critical Analysis of Testing . . . 15
  - 3.3.3 Outcomes of Demonstration . . . 16

4 Conclusion **17**
- 4.1 Recommendations . . . 17

5 Appendix **18**

Bibliography **23**

Chapter 1

Executive Summary

1.1 Problem Description

This project is a task in which a robust software system is to be designed and developed for utilisation on a fully functioning micro-mouse.

The overarching project of this course focuses on developing a micro-mouse, which is essentially a fully autonomous robot designed to navigate and solve mazes. Traditionally, these robots have been tasked with exploring a 16x16 grid of cells, autonomously finding a path from the starting point to the center of the maze without any external assistance.[1] This course is centered around designing, testing, and implementing software that integrates the sensing, power, and processing subsystems of the micro-mouse, with the motherboard serving as the hub connecting all essential components. The processor board, utilising an STM32L476, will serve as the control module for the robot, where the software will be executed. The power module supplies energy to all components of the robot. Lastly, the sensing subsystem is responsible for detecting the environment and enabling the robot's line-tracking capabilities. The focus of this faction of the project - being Milestone 03 and Milestone 04 - is on the software to be utilised in integrating all aspects of the micro-mouse to ensure accurate and efficient initialisation, calibration, surrounding awareness and decision-based movement of the micro-mouse to enable it to solve a 7x7 maze. In keeping with the goal of this project, the software developed should be able to detect the black line underneath the robot, detect the presence of an obstacle (maze wall) in front of and to either side of the robot and finally, integrate these aspects and - when placed in a maze - make decisions to rotate in a specific direction to reach the center. In addition to this, the software solution must abide by all the set-out requirements and specifications within the scope of work.

1.2 Scope and Limitations

The scope and extent of this design project, specifically the milestones three and four are to design and implement a software solution that builds on previous achievements. These previous achievements are the ability to utilise the real time inputs supplied to a system by a dedicated sensing subsystem to allow for an accurate calibration and initialisation method, tracking and following of a black line and obstacle detection to the front and either side of an assembled micro-mouse. In this, the final milestone of the project, the initially designed software solution is to now incorporate movement, decision making and turning corners in a maze while optimising the path taken which encompasses minimising errors and time taken. There are several limitations that need to be considered:

- Design Limitation: The recommended software tool was MatLab which, while having ease of use, exhibits limitations in code flexibility and manipulation.
- Time Limitation: The complete and working robot is to be demonstrated on 18/10/2024.
- Environmental Limitation: External conditions, for example - Ambient Light which changes, may impact the project execution.
- Technical Limitation: The simplicity of the infrared sensors used in the design allows for extreme susceptibility to noise.
- Technical Limitation: Due to the processor board only being able to process 12-bit ADC values (4095 ($2^{12} = 4096$)), the accuracy of readings by already stunted components is low.

1.3 Requirements

The requirements of the micro-mouse for Milestone 03 and 04 are described in Table 1.1 below.

Table 1.1: Requirements[R] of the micro-mouse.

Requirement ID	Description
R01	The micro-mouse must be able to identify and follow (hold) a black line.
R02	The micro-mouse must have surrounding awareness (obstacle detection)
R03	The micro-mouse must have calibration and initialization functions.
R04	The micro-mouse must be able to move through a maze.
R05	The micro-mouse must be able to rotate/turn around a corner
R06	The micro-mouse must be able to optimise the path through the maze.

1.4 Specifications

The specifications, refined from the requirements in Table 1.1, are described below in Table 1.2.

Table 1.2: Specifications[SP] of the micro-mouse

SP ID	Description
SP01	The design must detect a black line using the right and left downward facing IR sensors - 22mm apart from each other and sampling at a period of 0.2s and follow the line using the motors running at a minimum of 80% duty cycle.
SP02-1	The design must detect an obstacle in front using the forward facing IR sensors - 62mm apart from each other - and sampling at 0.2s.
SP02-2	The design must detect an obstacle to either side using the side, both left and right, facing IR sensors sampling at a period of 0.2s.
SP03	The design must assign the threshold values to their respective variables at the press of SW2.
SP04	The design must be able to move through the maze, avoiding collisions with maze walls, with motors running at a minimum of 80% duty cycle.
SP05	The design must be able to make a 90-180 degree turn/rotation using the motors running at a minimum of 85% duty cycle each in the opposite direction(polarity).
SP06	The design must utilise a shortest path or guaranteed-to-solve algorithm to allow for an optimum path through the maze with minimal collisions and time.

1.5 Achievements Matrix

The achievements matrix (Table 1.3) below outlines the results of each milestone, giving a brief overview of the performance of the design. KEY: green = fully achieved, orange = partly achieved, red = not achieved.

Table 1.3: Achievements Matrix containing achieved and outstanding aspects for each milestone

Requirements	IR Emitters	IR Sensors	Motors	Onboard LEDs
M2: Initialization				
M2: Calibration				
M2: Line Identification				
M2: Line Following				
M2: Obstacle Detection				
M3: Move, decide and make turns				
M4: Optimization				

Chapter 2

Subsystem Design: Navigation

In this section, the first design aspect is to be outlined, with various considerations, selection criterion and solutions explored. This aspect encompasses the design and implementation of the navigation faction of the fully operational micro-mouse (milestone three) in conjunction with the previously set out design decisions for line tracking, obstacle detection, calibration and initialisation functions and routines. All of the above, together, are used to accurately drive the micro-mouse through a walled maze.

2.1 Design

A 'corrective' approach was taken in the design of this software solution and it can be broken down in to 3 major sections: Line Following, Surrounding Awareness and Decision Based Movements.

2.1.1 Line Following

Calibration and Initialisation

In an initialisation and calibration function - seen in the appendix Figure 5.6 (the blocks titled initialisation and calibration)- all the necessary variables that are to be utilised in the main function are created. An arbitrary value - generally 0 - is stored in them or the necessary data to be used in the main solution is allocated to the respective variables.
In Table 2.1 below, a list of the variables created and initialised to be used in the broader line following function can found.

Table 2.1: Initialised Variables

Variable	Description and Value
leftWheel	This variable's purpose is to store and set the duty cycle at which the left motor will run and has an initial value of 0%.
rightWheel	This variable's purpose is to store and set the duty cycle at which the right motor will run and has an initial value of 0%.
RThreshold	This variable's purpose is to store a threshold value to be used as a comparison value for the right downward facing sensor. This value is set in the calibration state and serves its purpose in determining the respective adjustment movement of the micro-mouse when following a black line. It has an initial value of 0.
LThreshold	This variable's purpose is to store a threshold value to be used as a comparison value for the left downward facing sensor. This value is set in the calibration state and serves its purpose in determining the respective adjustment movement of the micro-mouse when following a black line. It has an initial value of 0.

In this state of the algorithm (calibration), the necessary data in order for the primary function to be executed is captured and stored. The functionality of this state extends to recording the value of each of the right and left downward facing IR sensor as a 'threshold'. Since the calibration state is to be conducted with the micro-mouse atop a black line, this reference value is crucial in determining the correction movement of the line tracking system. Of the two considered calibration methods discussed in the previous milestone, the individual value method was selected as it provides more accurate black line detection. Since each of the two emitter/sensor pairs react differently to environmental conditions and are subject to component tolerances, it was found that the micro-mouse responded better to having a separate threshold value (RThreshold and LThreshold) per sensor/emitter pair.

Line Following Functionality

As previously outlined, first an Initialisation and Calibration function is run with the latter being controlled by the action of pressing SW2. With the micro-mouse placed atop a black line, the system may be successfully calibrated and from there automatically moved into a 'rotation' and 'waiting' state which officially begins the movement state of the solution.

Table 2.2: Pros and Cons of the considered line following functions.

	Black Line Tracking	Gyroscope Alignment
Pro	-Able to ensure movement coincides exactly with intended path set out by a maze.	-Allows for extreme accuracy in turns/rotations being able to detect 90 or 180 degrees.
Con	-Easily influenced by ambient light, low battery or fast movements	-More complex to implement, requiring an integrating function as well as a PID controller.

From the above considerations - Table 2.2 - it was decided that the Black Line Tracking line following function would be more ideal to implement in order to reduce complexity and allow for easy integration between different line following problems. Below, in Table 2.3 the basic functionality of the line tracking solution can be seen.

Important Points to Note:

- An important aspect to note in this solution is that DOWN_LS and DOWN_RS represent the left and right of the micro-mouse when looking at it from the bottom and are inputs to this state chart coming directly from the ADC's of the micro-controller.
- A threshold value is recorded and since black 'absorbs' more of the IR light, anything less than the threshold value implies that a black line is detected and anything greater than said value implies that a black line has not been detected.
- Since the emitters now have a sample time of 0.05s, there was an increased likelihood of the mouse veering off the black line, and so in addition to the standard corrective measures in place, over corrective measures were put in place.
- Since a PWM signal is utilised to turn the IR Emitters on and off (as seen below in Surrounding Awareness and Figure 5.5 in the Appendix), a condition is utilised in order to ensure that the sensor values are utilised at the same time that its corresponding emitter is turned on, this is purpose of IR_LED_PATTERN(2) and IR_LED_PATTERN(4) in the final solution below.

Table 2.3: Basic Functionality of Line Tracking Solution seen in Figure 5.6 - Movement1 Chart.

CONDITION	ACTION	STATE
The current left and right readings provided by DOWN_RS and DOWN_LS are less than their respective threshold values.	Movement to be straight ahead.	'Straight'.
The current left reading provided by DOWN_LS is greater than its respective threshold value and the right reading provided by DOWN_RS is less than its respective threshold value.	Adjust the mouse to the left.	'TurnLeft'

CONDITION	ACTION	STATE
The current left reading provided by DOWN_LS is less than its respective threshold value and the right reading provided by DOWN_RS is greater than its respective threshold value.	Adjust the mouse to the right.	'TurnRight'
Both the left and right real time detected values are greater than their respective threshold value and the previous state was 'TurnRight'.	The movement will over-correct the robot to the right to ensure it stays on track.	'TurnRight1'
Both the left and right real time detected values are greater than their respective threshold value and the previous state was 'TurnLeft'	The movement will over-correct robot to the left to ensure it stays on track.	'TurnLeft1'

2.1.2 Surrounding Awareness

Initialisation and Calibration

In an initialisation and calibration function (as seen in Figure 5.8 of the Appendix) , all the necessary variables that are to be utilised in the main function are created and an arbitrary value - generally 0 - is stored in them or the necessary data to be used in the main solution is allocated to the respective variables. In Table 2.4 below, a list of the variables created and initialised to be used in the broader surrounding awareness function can found. In this state of the algorithm (calibration), the necessary

Table 2.4: Initialised Variables

Variable	**Description and Value**
LSThreshold	This variable's purpose is to store a threshold value for the left side IR sensor to be used as a reference in order to ascertain obstacle detection (sets a 'distance' at which an obstacle should trigger the system) - and turn on LED2 - and is set in the calibration state.
RSThreshold	This variable's purpose is to store a threshold value for the right side IR sensor to be used as a reference in order to ascertain obstacle detection (sets a 'distance' at which an obstacle should trigger the system) - and turn on LED0 - and is set in the calibration state.
FLThreshold	This variable's purpose is to store a threshold value for the front left IR sensor to be used as a reference in order to ascertain obstacle detection (sets a 'distance' at which an obstacle should trigger the system) - and turn on LED1 - and is set in the calibration state.
FRThreshold	This variable's purpose is to store a threshold value for the front right IR sensor to be used as a reference in order to ascertain obstacle detection (sets a 'distance' at which an obstacle should trigger the system) - and turn on LED1 - and is set in the calibration state.
LED0	This variable is used to alter the state (on/off) of its respective on-board LED in order indicate obstacle detection in a particular direction.
LED1	This variable is used to alter the state (on/off) of its respective on-board LED in order indicate obstacle detection in a particular direction.
LED2	This variable is used to alter the state (on/off) of its respective on-board LED in order indicate obstacle detection in a particular direction.
FL_Off	This variable is used as a measurement in the aspect of this solution that accounts for ambient light and is set to zero.
FL_On	This variable is used as a measurement in the aspect of this solution that accounts for ambient light and is set to zero.
FR_Off	This variable is used as a measurement in the aspect of this solution that accounts for ambient light and is set to zero.
FR_On	This variable is used as a measurement in the aspect of this solution that accounts for ambient light and is set to zero.

data in order for the primary function to be executed is captured and stored. The functionality of this state extends to recording the value of each of the right, left and front facing IR sensor as a 'threshold'. Since the calibration state is to be conducted with the micro-mouse facing a walled off corner, this reference value is crucial in determining the success of the surrounding awareness function. Of the two considered calibration methods discussed in the previous milestone, the individual value method was

selected since it provides more accurate surrounding awareness. Since each of the two emitter/sensor pairs react differently to environmental conditions and are subject to component tolerances, it was found that the micro-mouse responded better to having a separate threshold value (RSThreshold, LSThreshold, FLThreshold and RLThreshold) per sensor/emitter pair.

Ambient Light Mitigation:
In this small section of the surrounding awareness function of the micro-mouse, the goal is to mitigate the effect of ambient light from the surrounding environment to ensure optimal obstacle/wall detection. It is accomplished by only comparing IR light emitted by the emitters and discarding that of any other IR light in the surrounding environment. This encompasses taking a reading with the emitters on, and off and finding the difference in this value This is done in the following manner:

Ambient Light Mitigation: (as per Figure 5.8 of the Appendix)

- First all the necessary variables are initialised (performed upon entry to the state), these variables are to be used as a measured value for two states, one with the respective emitters on, and the other with them off. The variables are FL_On, FL_Off, FR_On, FR_Off, R_On, R_Off, L_On and L_Off.

- Next, using the array , IR_LED_PATTERN(), index as a method to ensure that the values are sampled at the correct time, the two readings are taken (with the emitters being on and then off) and are recorded in their respective variables.

- Lastly, the reference variable (FL_Ambient, FR_Ambient, R_Ambient or L_Ambient) is calculated by subtracting the value recorded when the emitters were turned off from when the emitters were turned on. All of this is done under the 'during' heading so it is constantly updated and utilised when necessary in order to account for real time changes in ambient lighting/environment.

Obstacle Detection Functionality
This faction of the overall navigation element of the micro-mouse involves the detection of obstacles - most likely a maze wall - and reporting and indicating such a detection. The initialisation, calibration and ambient light mitigation functions are all used in conjunction with this main state in order to provide a successful and robust wall/object detecting ability for the area surrounding the micro-mouse. Various solutions to obstacle detection were proposed as seen in Table 2.5 below.

Table 2.5: Pros and Cons of the considered detection functions.

	ALL EMITTERS ON	PWM SIGNAL EMITTERS	SELECTED EMITTERS ON
Pro	-Seen as efficient since no processing is done.	-Allows for significantly less 'cross-exposure' to IR light from an over designed sensor board.	-A combination of reduced exposure and process utilisaton.
Con	-High levels of cross-exposure on all sensors since all emitters are on.	-Higher processing power since a PWM signal is to be generated and delayed respectively.	-Heavily affected by ambient light.

SELECTED METHOD: PWM SIGNAL EMITTERS

Optimising on the previously utilised 'selected emitter' method, the above titled (pwm signal emitters) was selected since it allows for the least amount of exposure to IR light from the surrounding emitters and in conjunction with the utilised ambient light mitigation algorithm, allows for an extremely successful and robust surrounding awareness mechanism. What this solution entails is a PWM based signal with adjusted delays in order to allow for certain emitters to have an input be high at different times in a given period, avoiding any overlap between the front, left, right and downward facing emitters. This can be seen in Figure 5.5:'PWM Signal used to turn on Emitters at a specific time delay' of the Appendix. In order to increase the accuracy of the line following solution, the downward facing emitters were turned on twice in a period (during its dedicated portion and simultaneously with the left and right motor emitters). In order to ensure that the sensor values were being used exactly when their respective emitters were turned on, the on-off states of the emitters were stored in the array IR_LED_PATTERN and when it was required that a sensor value be used, the respective emitter would have to be on (true) at that point in time. However, due to various factors, primarily processing delays, the sensor to detect a signal from emitter 1 was only receiving that reading when emitter 2 was on. This resulted in the following condition needing to be met IR_LED_PATTERN(n+1) in order to record a value from the sensor when emitter n is on. The process that this algorithm follows is firstly the initialisation and calibration states are run, both the wall detection and ambient lighting calibration states, triggered by SW2. The micro-mouse should be placed with the maze walls surrounding the front, left and right (backwards at the starting point - as the movement function will rotate it to the correct starting position). Once successfully calibrated, this system may run in conjunction with the line tracking capability of the micro-mouse.

Important Points to Note:

- A crucial characteristic to clarify is that 'L' and 'R' represent the left and right of the micro-mouse when looking at it from the underneath (so it would be inverted when looking at it from the top).
- LS, RS, FWD_LS and FWD_RS are inputs to this state chart coming directly from the ADC's of the micro-controller.
- When calibrating the micro-mouse, there should be obstacles, or maze walls surrounding the front, left and right side of it. This implies that for either L, R, FL, FR- _Ambient being less than R, F, FL, FR-Threshold is indicative of no obstacle/wall being detected.
- When calibrating the micro-mouse, there should be obstacles, or maze walls surrounding the front, left and right side of it. This implies that for either L, R, FL, FR-_Ambient being greater than R, F, FL, FR-Threshold is indicative of an obstacle/wall being detected.
- In order to allow for concurrent surrounding awareness on all the necessary sides of the micro-mouse as well as while the rest of the navigational tasks are being processed each LED algorithm is in its own state flow chart on the main template page. The basic functionality of surrounding awareness can be seen below in Table 2.6.

2.1.3 Decision Based Movements

Unlike the previous two sections of the navigational component of this project, the last section is not as complex. By simply utilising the outcomes and actions of the previous two extensive solutions, this system is able to allow for the micro-mouse to move through a maze, allow for basic localization

Table 2.6: Basic Functionality of Surrounding Awareness Solution - seen in Figure 5.8

CONDITION	ACTION	STATE
The current right readings provided by R_Ambient is greater than the RSThreshod value	LED2 turns on, indicating an obstacle to the right.	'WALLinRightSide'.
The current left readings provided by L_Ambient is greater than the LSThreshod value	LED0 turns on, indicating an obstacle to the left.	'WALLinLeftSide'.
The current front readings provided by FR or FL_Ambient is greater than the FR or FLThreshod value	LED1 turns on, indicating an obstacle to the front.	'WALLinFRONT'.
The current right readings provided by R_Ambient is less than the RSThreshod value	LED2 turns off, indicating no obstacle to the right.	'NoWallinRightSide'.
The current left readings provided by L_Ambient is less than the LSThreshod value	LED0 turns off, indicating no obstacle to the left.	'NoWallinLeftSide'.
The current front readings provided by FR or FL_Ambient is less than the FR or FLThreshod value	LED1 turns off, indicating no obstacle to the front.	'NoWallinFRONT'.

capabilities and then make a decision as to what action to complete. The basic functionality of this subsystem can be seen below in Table 2.7.

Table 2.7: Basic Functionality of Decision Based Movements Solution - seen in Figure 5.6: Turning Chart.

CONDITION	ACTION	STATE
LED1 has been turned on (indicating an obstacle/wall detected to the front of the micro-mouse).	Stop	'Wait'.
Only LED1 has been turned on.	Make an arbitrary right turn.	'RotateRight1'
Only LED0 and LED1 have been turned on (indicating an obstacle/wall detected to the front and left)	Rotate right to avoid wall.	'RotateRight1'
LED0, LED1 and LED2 have been turned on (indicating that the mouse is now enclosed from the front.)	Rotate a full 180 degrees.	'Rotate'
Only LED1 and LED2 have been turned on.	Rotate left to avoid wall.	'RotateLeft'
After movement, LED1 has been turned off and provided a black line has been detected while sampling at the correct time.	Stop and return to moving straight	'Hold' and return to previous movement chart.

2.2 Mitigation of Limitations

Table 2.8 below elaborates on the mitigation of limitations performed for navigation. KEY: EL = Environmental Limitation, TL = Technical Limitation.

Table 2.8: Mitigation of Limitations

LIMITATION	METHOD OF MITIGATION
EL: Accounting for ambient lighting	By ensuring that our calibration functions accurately measure and record the necessary readings, ambient light can be accounted for in its function.
TL 1: Exposure due to excess IR light (over designed solution)	By implementing the above mentioned PWM signal to turn emitters on and off, over exposure due to an over designed sensor board can be mitigated.
TL 2: 12-bit ADC Values	By adjusting the recorded values with an ADC constant, we were able to ensure that no readings were corrupted as a result of the 12-bit ADC value limitation.

2.3 Acceptance Testing Procedure

2.3.1 Tests

The table below (Table 2.9) outlines each testing procedure done, with a description and the pass/fail criteria.

Table 2.9: Navigation Acceptance Tests

AT ID	**Description**	**Pass/Fail Criteria**
AT01	Confirm that the IR sensors have a distinct change when on and off the black line (using debugger mode)	PASS: The values on the scope change significantly (passed a threshold value) when moved from on the black line to off the black line. FAIL: The values on the scope do not change when moved on and off the black line.
AT02-1	Verify that an obstacle can be detected by the front IR sensors by observing the middle onboard LED(or using the debugger mode)	PASS: LED 1 turns on. The value for the forward facing IR sensor changes when the wall is placed in front. FAIL: LED 1 is off. The scope shows no change in the forward facing sensor value.
AT02-2	Verify that an obstacle can be detected on the left by the left IR sensor by observing the left onboard LED.	PASS: LED 0 turns on. The value for the left facing IR sensor changes when the wall is placed to the left of the robot. FAIL: LED 0 is off. The scope shows no change in the left facing sensor value.
AT02-3	Verify that an obstacle can be detected on the right by the right IR sensor by observing the right onboard LED.	PASS: LED 0 turns on. The value for the right facing IR sensor changes when the wall is placed to the right of the robot. FAIL: LED 0 is off. The scope shows no change in the right facing sensor value.
AT03-1	Validate that the essential MATLAB variables are correctly initialised.	PASS: There should be variables defined with values in the Matlab workspace. FAIL: There are no values assigned to the variables in Matlab.
AT03-2	Confirm that the calibration function operates successfully by verifying the variables are within similar range as in debug mode.	PASS: The value of the threshold matches the expected range.FAIL: The value of the threshold is out of the bounds of the expected range.

AT ID	Description	Pass/Fail Criteria
AT04	Confirm that the robot can follow the black line correctly and react accordingly to its surroundings without colliding with the walls	PASS: The robot is able to follow(hold) the black line the entire time. It does not collide with any walls and it able to correct its motion when veering slightly off-course. FAIL: The robot is not able to follow(hold) the black line. It collides with walls and veers off the black line and cannot find its way back without intervention.
AT05-1	Check that the robot makes the correct turn/rotation with every possible wall configuration.	PASS:The robot is able to turn in the desired direction. FAIL: The robot does not turn in the desired direction and may collide with a wall.

2.3.2 Critical Analysis of Testing

AT01

Objective

This unit acceptance testing procedure was to confirm significant change in the downward facing IR sensors when placed on the black line and then moved away from the line.

Equipment

MATLAB Debugger file.

Micro-mouse.

Straight Black Line made from insulation tape attached to the floor.

Procedure

Open the MATLAB debugger file and ensure that the data-carrying micro-USB cable has been inserted into the micro-mouse's debugger port. Place the micro-mouse aligned on the black line and monitor the output on the IR sensors using the scopes for the left and right downward facing IR sensors. Adjust the position of the micro-mouse to the left so that the left down sensor is not aligned with the left side of the black line. Observe any difference in the left downward scope. Repeat for the right side.

Analysis

This ATP follows R01 and SP01: to pass this test, there must be a significant change in the scopes for both left and right downward facing sensors. Figure 5.1a indicates the values when perfectly aligned with the line -the threshold values- 1.8V for the left sensor and 2.45V for the right. Figure 5.1b displays the response when the micro-mouse has been moved to the right, so that the right sensor is no longer on the black line (seen by the value increasing to 3V which is higher than the threshold), and the left sensor is now on the black line(seen by the left sensor value of 1.5V which is lower than the threshold). This is expected as black absorbs the IR light and will reflect less light back to the sensor. In a similar manner, when the micro-mouse is placed to the left the opposite occurs. This expected operation allows for control of the movement of the micro-mouse. When one value drops below the threshold, it can be assumed that the related sensor is on the black line and the micro-mouse must adjust its position to realign with the black line. The selected design passes this test in its entirety, as seen by the variation in values in Figure 5.1. It is important to calibrate when aligned with the black line. A slight variation, shown in Figure 5.11 is acceptable as it is catered for in the code, however if the threshold values are unreliable this will impact the values and overall motion.

AT02-1, AT02-2, AT02-3

Objective

This unit testing procedure was to verify that obstacles to the front, left and right of the micro-mouse can be detected by the respective IR sensors.

Equipment

MATLAB Debugger file.

Micro-mouse.

'Wall' - a small sheet of cardboard.

Procedure

Open the debugger file in MATLAB and check that the data-carrying micro-USB cable has been inserted into the correct port (the debug port) on the robot. Monitor the movement on each scope when the 'wall' is not present and then placed in position. Calibrate the micro-mouse with the obstacles a certain distance away (around 4cm away is acceptable). Observe the state of each LED when removing the obstacles and placing them back into position.

Analysis

This ATP follows R02 and SP02-1, SP02-2, with each obstacle detection tested independently using the respective IR sensors. For a test to pass, a significant increase in sensor readings must be seen when an obstacle is detected, and the corresponding LED should turn on. In AT02-1, the debugger results (Figure 5.2a) show a clear increase in both front IR sensor values and LED 1 turns on when the obstacle is within the calibrated distance, confirming the test's success.

Similar results were observed for the right and left sensors in Figures 5.2b and 5.2c, where spikes in sensor values triggered LED 0 and 2. Under standardised testing conditions, the surrounding awareness solution outlined and implemented as described, completely passes all tests conducted prior to the demo on 18/10/2024.

AT03-1, AT03-2

Objective

This acceptance procedure was to validate that the essential MATLAB variables are correctly initialised as well as verify that the calibration function operates successfully.

Equipment

MATLAB runMeFirst file.

Micro-mouse.

Straight black line made from insulation tape attached to the floor.

'Walls' - three small sheets of cardboard

Procedure

Open the runMeFirst MATLAB file and check that status of the workspace variables. Open the main code chart and check that all variables are initialised to 0. Perform the calibration function by pressing SW2 after placing the micro-mouse in the maze or simulate the maze starting block: a black line in the center with three 'walls' in the front, left and right side.

Analysis

This ATP follows R03 and S03. Most of the necessary threshold values are environment dependent and will vary based on external conditions such as ambient light, 'wall' material and other factors. Thus, the calibration function must read real-time data and log those values for accurate operation. When

calibrating, the micro-mouse must be placed in a position where all threshold values can be tested. This was checked multiple times in different environmental conditions and led to the conclusion that the calibration function is completely successful. Additionally, files such as the runMeFirst, shown in Figure 5.3, must be run prior to flashing the code to the micro-mouse.This test is conducted and proven to be passed.

AT04

Objective
This acceptance testing procedure was to confirm that the micro-mouse can follow the black line correctly and react accordingly to its surroundings without colliding with the walls.
Equipment
Micro-mouse.
Maze: black insulation tape on the floor in a grid-like pattern and sheets of cardboard as 'walls'.
Procedure
Place and calibrate the micro-mouse in the maze. Observe the motion of the micro-mouse, focusing especially when it veers off the course. An example of the maze path used can be found in Figure 5.9.
Analysis
This ATP follows R04 and SP04, focusing on the micro-mouse's ability to navigate the maze without collisions. The procedure involved placing the micro-mouse in the maze and ensuring it corrects its path if it veers slightly off the black line, whether to the left or right. The goal is to test its ability to traverse straight segments of the maze error-free. The micro-mouse passed this test by maintaining alignment with the black line, adjusting its position as needed and turning when detecting a wall ahead.

AT05-1

Objective
This acceptance testing procedure checks that the micro-mouse is able to turn/rotate and continue motion in an available direction.
Equipment
Micro-mouse.
Maze with a black line grid and adjustable walls to create different obstacle configurations.
Procedure
Place and calibrate (Press SW2) the micro-mouse in the maze. Observe the motion of the micro-mouse.
Analysis
This ATP follows R05 and SP05 to verify the micro-mouse's ability to handle directional changes. Previously, it moved straight ahead; now it must observe its surroundings and decide which way to turn. The optimal rotation speed is specified as 85% as lower speeds can cause misalignment with the black line, while high speeds may lead to errors. Opposite motor polarity ensures smoother rotation and better alignment with the black line. Initially, the micro-mouse moved straight until it encountered a wall, then scanned its surroundings and chose a direction arbitrarily. It only changed direction when a front obstacle was detected. Since AT02-1 was successful, this test also passes, as the front-facing IR sensors would correct any error in navigation by triggering a rotation if an obstacle is detected.

Chapter 3

Subsystem Design: Optimisation

In this, the final section of the overarching micro-mouse project, the maze-solving algorithm is going to be outlined with various considerations, selection criterion and solutions explored. This aspect encompasses less design and more implementation of an optimal methodology which the micro-mouse must utilise in order to optimise the path taken through the maze, minimize errors and minimize time taken to complete the maze. Building on the arbitrary direction movements discussed in the decision-based movement section (subsection 2.1.3) of Navigation (Chapter 2), the following considerations were leveled and implemented to optimise the previous solution.

3.1 Design

3.1.1 Optimisation Algorithm

Various optimisation techniques and solutions were considered, these can be see below.

Left Wall Follower

This technique involves simply following the left wall. Since it is known that the left wall is the boundary wall of the maze. This is a guaranteed solution for all closed mazes (no loops) and is simply an application of 'depth first in order' tree traversal. Should there be a maze wall detected to the left of the micro-mouse, the mouse should continue to move straight until it encounters a complete surround, in which case it would have rotate by 180 degrees. Should there be no wall detected to the left of micro-mouse, it should rotate left in order to find the wall again, and continue following it.

Flood Fill

This technique involves mapping the entire maze - which implies that all information about the start and end points are required. As the mouse moves through pixels of the maze, the mapping of the maze is updated to include newly discovered obstacles/walls. What the mapping includes is the shortest per pixel distance to the destination and determining which neighbouring cells offer the smallest value (representative of n-1 pixels to the centre) with the mouse being directed along this path.

A-Star Search

This technique is by far the most complex but offers guarantees that the other two do not. This is a directed search algorithm in which the shortest distance is calculated, and the best direction to move is determined. This algorithms ability to back track allows for not only the discovery of an optimised path to the destination but a path guaranteed to be the shortest and least time consuming.

Important Points to Note:

- Since the task at hand was simply to optimise the path to be taken through the maze while reducing time and error, this function works in conjunction with the previously discussed Navigation function, and builds on it.
- From the Surrounding Awareness function, and LED being on is an indication that obstacle has been detected in its respective direction with LED0 - LEFT, LED1 - FORWARD and LED2 - RIGHT.
- The first three conditions are to transition from the movement state to the decision making state and so the last as well as the 5th to the 8th conditions also result in a transition back to the

movement state. As seen in Figure 5.7 of the Appendix.

Table 3.1: Pros and Cons of the considered optimisation functions.

	LEFT WALL FOLLOWER	FLOOD FILL	A-STAR SEARCH
Pro	- Always guaranteed to solve a closed maze (no loops). - Uses minimal memory	- Guaranteed to minimise time taken and reduce errors. - Makes use of multi-dimensional arrays, an easy to manipulate data structure.	- Follows the path with the lowest heuristic cost. - Very fast algorithm
Con	- The path taken is not always the most efficient path available. - For a disjoint maze, the algorithm could allow for a loop that never reaches the centre.	- Requires all the information regarding the maze. - Requires continuous update resulting in high processing utilisation. - Requires proper planning.	-Extremely complex implementation methodologies required. - The use of Djikstra's and BFS result in high memory usage.

Based on the above listed pros and cons - Table 3.1 - it was decided that the optimisation solution most suited for this project is the LEFT WALL FOLLOWER since there is minimal memory usage - perfect for the processing limitations of the micro-controller and it is always guaranteed to solve a closed maze. Table 3.2 below highlights the basic functionality of the optimisation algorithm as seen in Figure 5.7 in conjunction with Figure 5.8 which highlights the final working optimised solution.

Table 3.2: Basic Functionality of the Optimisation Solution - seen in Figure 5.7: Turning Chart

CONDITION	ACTION	STATE
Rflag and Lflag are true (indicating that the downward facing sensors have detected a reading less than a prescribed amount which is indicative of a 'junction' in the maze).	Stop	'Wait'.
Only LED1 has been turned on.	Stop	'Wait'
LED2 has been turned off (a left wall is no longer detected)	Stop	'Wait'
LED0 and LED1 are turned on, however LED2 has been turned off	Rotate 90 degrees to the right.	'RotateRight'
LED1 has been turned off and LED0 and LED2 are turned on.	Stop and then continue straight with black line detection.	'Hold' then 'Straight'
LED0 has been turned on and LED1 and LED2 are turned off.	Stop and then continue straight with black line detection.	'Hold' then 'Straight'
All three LEDs (0,1,2) have been turned off.	Stop and then continue straight with black line detection.	'Hold' then 'Straight'
LED1 has been turned off and LED0 and LED2 are turned on.	Stop and then continue straight with black line detection.	'Hold' then 'Straight'
LED0 has been turned off and LED1 and LED2 are turned on.	Rotate 90 degrees to the left	'RotateLeft'
All three LEDs (0,1,2) have been turned on	Rotate 180 degrees	'Rotate180'
LED1 has been turned on and LED0 and LED2 are turned off.	Rotate 90 degrees to the left	'ForceLeft'

CONDITION	ACTION	STATE
LED2 has been turned on and LED0 and LED1 are turned off.	Rotate 90 degrees to the left	'ForceLeft'
LED0 has been turned on and LED1 and LED2 are turned off.	Rotate 90 degrees to the left	'ForceLeft'
If the previous condition was RotateRight, RotateLeft, Rotate180 or ForceLeft	Stop and then continue to straight when black line is found.	'Hold' then 'Straight'

3.2 Mitigation of Limitations

In Table 3.3 below, various ways in which limitations are mitigated are discussed.

Table 3.3: Mitigation of Limitations

LIMITATION	METHOD OF MITIGATION
Design Limitation: Utilisation of MatLab	By completing the necessary MatLab on-ramp courses we were equipped with the necessary skills to tackle this limitation.
Time Limitation: Completion by 20/09/2024	By ensuring consistent and thorough work, we were able to complete this milestone task by the dedicated date.

3.3 Acceptance Testing Procedure

3.3.1 Tests

Table 3.4: Optimisation Acceptance Tests

AT ID	**Description**	**Pass Criteria**
AT05-2	Check that the robot makes the correct turn/rotation with every possible wall configuration.	PASS:The robot is able to turn in the desired direction. FAIL: The robot does not turn in the desired direction and may collide with a wall.
AT06	Ensure that the robot finds a shorter path when solving the maze by tracking its movement and verifying that it follows the selected algorithm.	PASS:The robot follows the algorithm movements and solves the maze. FAIL: The robot does not follow the algorithm movements and may not reach the center of the maze(does not solve the maze).

3.3.2 Critical Analysis of Testing

AT05-2

Objective

This acceptance testing procedure checks that the micro-mouse makes the correct turn/rotation with every possible wall configuration.

Equipment

Micro-mouse.

Maze with a black line grid and adjustable walls to create different obstacle configurations.

Procedure

Place the micro-mouse in the maze. Calibrate the micro-mouse at an appropriate distance away from the obstacles and perfectly aligned with the black line. Press SW2 and observe the motion of the micro-mouse.

Analysis

This ATP follows R05 and SP05. Like AT05-1, it tests the micro-mouse's ability to handle directional changes. Previously, it only changed direction when encountering a front obstacle, which contradicts the Left-Wall-following algorithm. The algorithm requires the micro-mouse to always keep a wall on its left by turning left whenever no obstacle is detected. Each possible configuration (shown in Figure 5.4 was tested using adjustable maze walls. These were assembled in the maze and tested, starting from the position shown in 5.10. Under standardised testing conditions conducted prior to the final demo on 18/10/2024, all surrounding awareness functions correctly and the micro-mouse is able to keep track of the left wall and follow it until eventually reaching the center of the maze, thus solving it.

AT06

Objective

This user acceptance testing procedure is to ensure that the micro-mouse finds a shorter path when solving the maze by tracking its movement and verifying that it follows the selected algorithm.

Equipment

Micro-mouse.

Maze with black grid lines and 'walls'.

Procedure

Place the micro-mouse in the maze. Calibrate the micro-mouse by pressing SW2 and observe its movement through the maze.

Analysis

This ATP follows R06 and SP06. It tests the overall motion and the micro-mouse's ability to solve the maze efficiently. When tested under standard conditions, the micro-mouse is able to move through the maze, making decisions on whether or not to turn: prioritising a left turn so as to always keep a wall on the left side. This ATP is thus passed for the series of tests performed on the micro-mouse prior to the final demo on 18/10/2024.

3.3.3 Outcomes of Demonstration

At the final demonstration, the ambient light appeared to be much more prominent than in the series of tests conducted. This limited the abilities of the surrounding awareness function of the micro-mouse. As a result, those functions dependent on the complete execution of the surrounding awareness were impaired.

Chapter 4

Conclusion

This project is a task in which a robust software system is to be designed and developed for utilisation on a fully functioning micro-mouse. This specific software solution attempts to utilise black line detection and obstacle detection to solve the maze. This report indicates the process followed to design, test and implement the software for each of the given requirements. Within the sensing subsystem, 6 of out the 8 IR LEDs are used in the final design. They are used in a periodic pattern, chosen for maximum accuracy and to reduce the noise interference brought about by having all on simultaneously. While the line detection and surrounding awareness was executed to above a satisfactory standard when testing, the obstacle detection proved to be partially troublesome in demonstration. This is due to the effect of ambient light disrupting the values read by the IR sensors. This caused an avalanche effect as the accuracy of the surrounding awareness influences the functionality of the decision-based turns. This led to small errors in the final maze solving attempt resulting in the path taken being different to what was intended. Overall, the project's acceptance tests (which were conducted in a controlled environment) AT01, AT02-1, AT03-1, AT03-2, AT04 and AT05-1, AT02-2, AT02-3, AT05-2 and AT05 are passed. This is shown in the respective sections, but summarised in the Table 5.1 found in the Appendix. The micro-mouse is able to identify and follow(hold) a black line to a near perfect degree. As discussed previously, the surrounding awareness has room for improvement for the left and right sides. The calibration and initialisation functions are fully functioning. The micro-mouse is able to successfully maneuver through a maze without colliding into the walls, and rotate around corners correctly. The path optimisation outlined is a decent option but its execution relied too heavily on the constant success of the other aspects which, ultimately fell ever so short of expectations. There were certain limitations that were overcome, such as the design, time and technical limitations, while others proved to be more of a challenge: the environmental limitation (ambient light). The design produced a micro-mouse which is able to navigate through a maze successfully; it needs time to find the center, thus requiring more accurate surrounding awareness for better optimisation. All of the specifications were adhered to and their respective requirements were met and the thorough testing procedure allowed for a detailed understanding of exactly which aspects of the micro-mouse fell short of the set out requirements in demonstration.

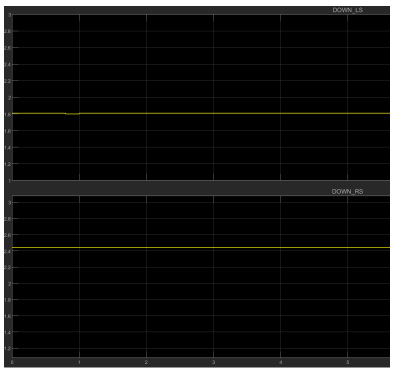
4.1 Recommendations

Throughout the design, testing and implementation processes required for this project, the following has been noted to implement as an improvement on the existing design:

- Developing a more robust method of dealing with ambient light from the environment.
- Approaching the design of the system as a whole in a more modular fashion could allow for less interdependence between systems so should one fail, the remainder are still able to operate successfully.
- Increasing the frequency (decreasing the period) at which the solution sampled data could also for more accuracy and less interference and noise.
- Simultaneously writing up report sections as the design and testing procedures commence can ensure that all the required information can be retrieved for future use.

Chapter 5

Appendix

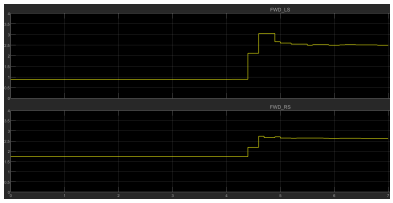


(a) Left and Right Downward Scope - Perfectly aligned with black line

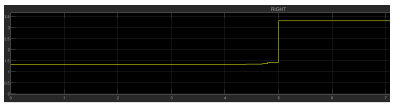
(b) Left and Right Downward Scope - Moved to the right, onto the black line

(c) Left and Right Downward Scope - Moved to the left, onto the black line

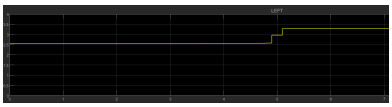
Figure 5.1: Downward IR sensor scopes



(a) Forward scopes showing the IR sensors picking up the obstacle in front



(b) Right side scope showing the IR sensor picking up the obstacle to the right



(c) Left side scope showing the IR sensor picking up the obstacle to the left

Figure 5.2: Corresponding IR sensor scopes spiking when obstacles are detected

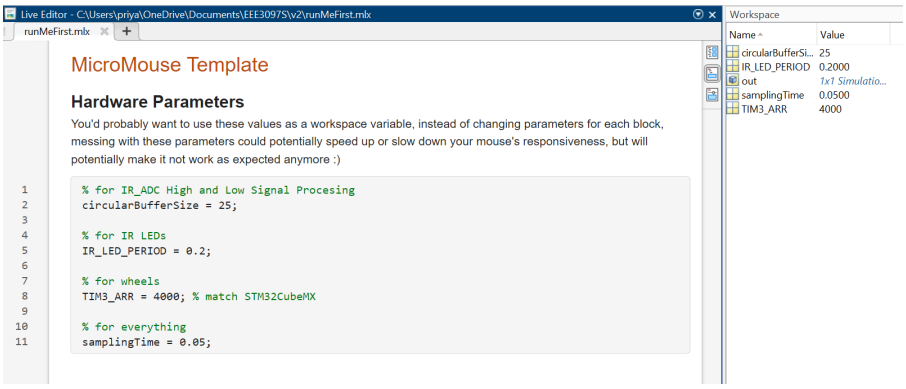


Figure 5.3: Matlab Home screen with certain variables initialised

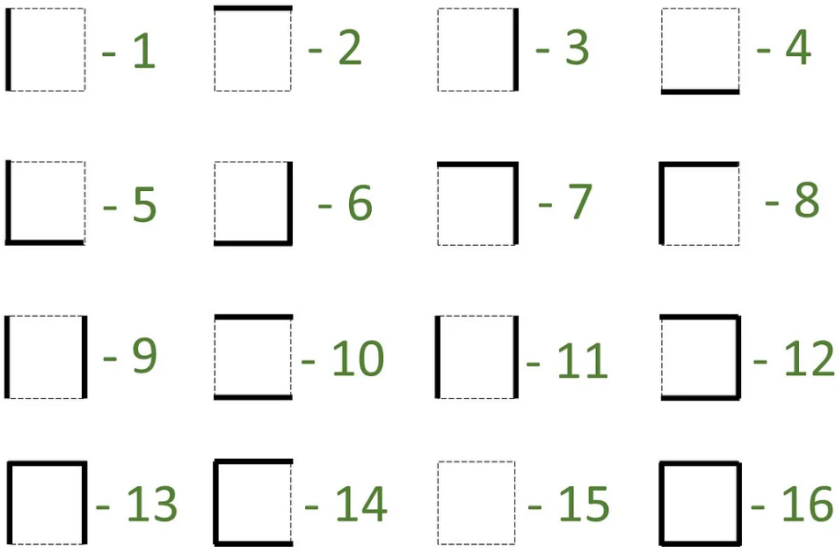


Figure 5.4: All possible obstacle configurations[2]

Table 5.1: Subsystem Acceptance Test Results

AT ID	**Description**	**Result**
AT01	Confirm that the IR sensors have a distinct change when on and off the black line (using debugger mode)	PASS
AT02-1	Verify that a front obstacle can be detected by the front IR sensors by observing the middle onboard LED.	PASS
AT02-2	Verify that an obstacle on the left can be detected by the left IR sensor by observing the left onboard LED.	PASS
AT02-3	Verify that an obstacle on the right can be detected by the right IR sensor by observing the right onboard LED.	PASS
AT03-1	Validate that the essential MATLAB variables are correctly initialised.	PASS
AT03-2	Verify that the calibration function operates successfully by confirming the threshold values are within a similar range as seen in debugger mode	PASS
AT04	Confirm that the micro-mouse can follow the black line correctly and react accordingly to its surroundings without colliding with walls.	PASS
AT05-1	Check that the micro-mouse is able to turn/rotate and continue motion in an available direction.	PASS
AT05-2	Check that the micro-mouse makes the correct turn/rotation with every possible wall configuration.	PASS
AT06	Ensure that the micro-mouse finds a shorter path when solving the maze by tracking its movement and verifying that it follows the selected algorithm.	PASS

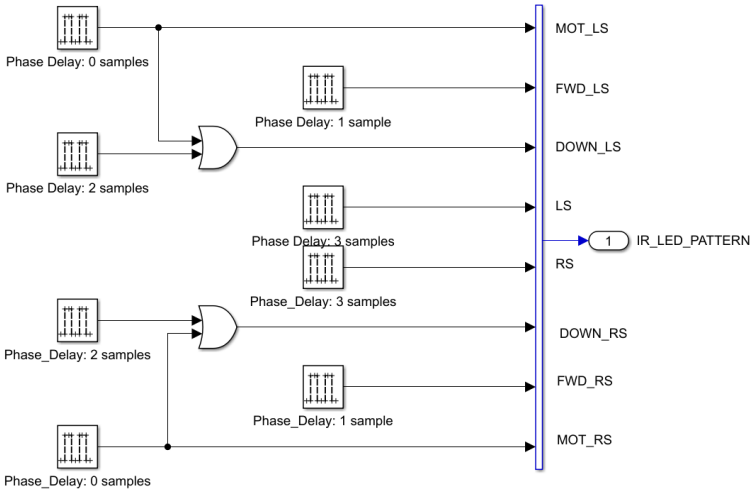


Figure 5.5: PWM Signal used to turn on Emitters at a specific time delay.

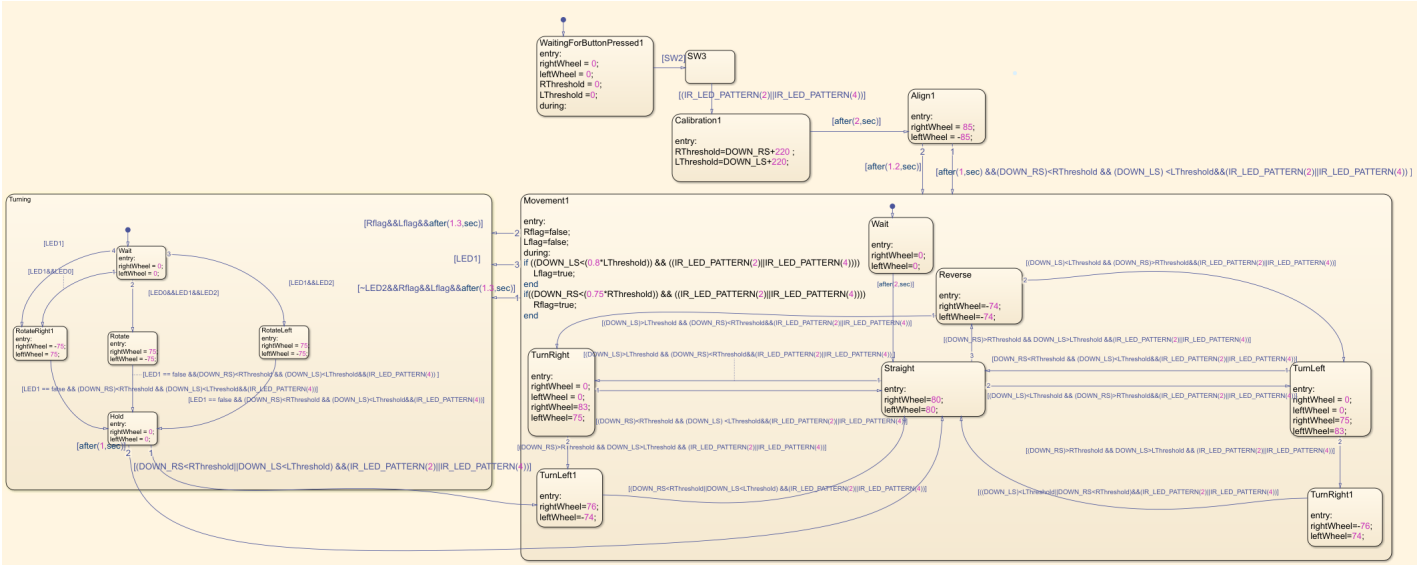


Figure 5.6: Final Line Tracking and Decision Based Movement State Flow Diagram

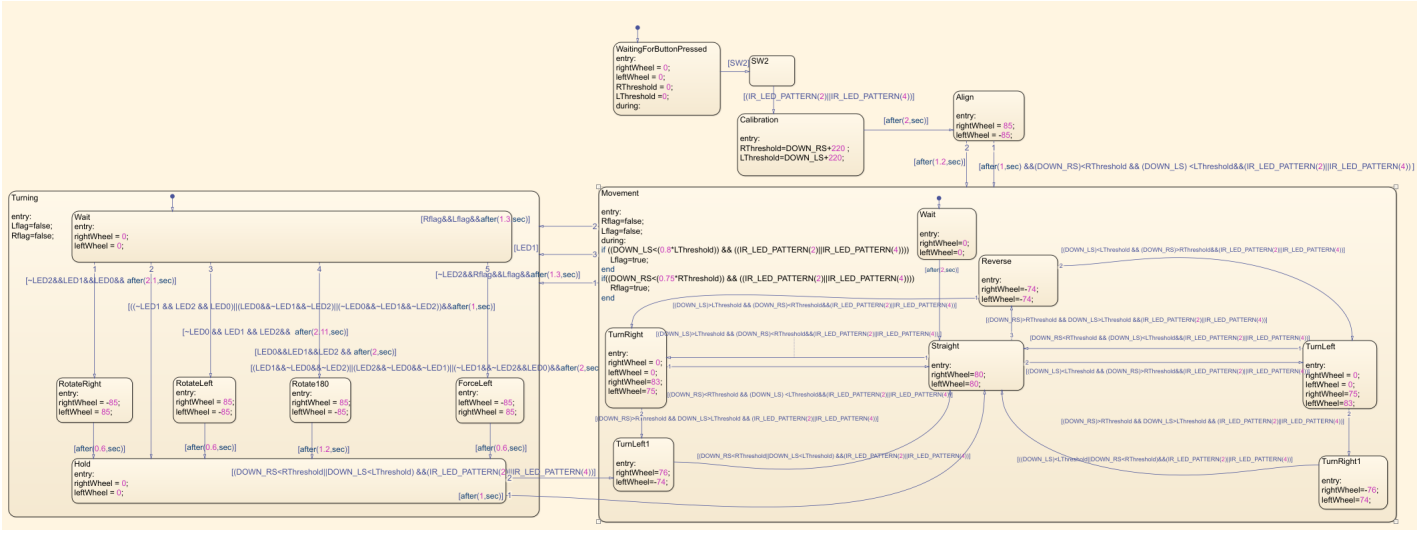
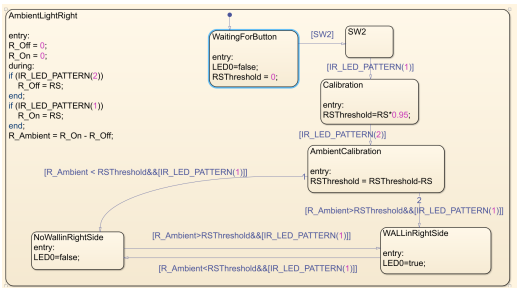
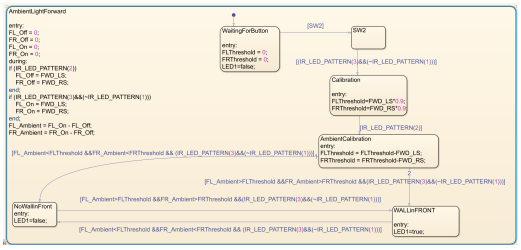


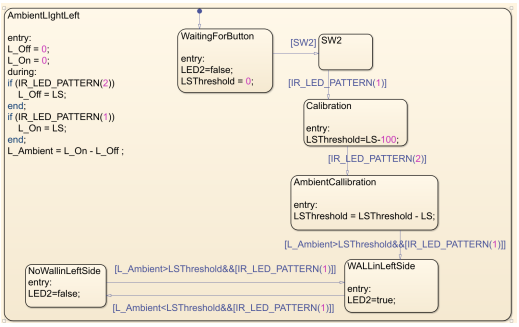
Figure 5.7: Final Line Tracking and Optimisation Algorithm State Flow Diagram



(a) LED0



(b) LED1



(c) LED2

Figure 5.8: Final Surrounding Awareness Solution

Figure 5.9: Micro-mouse in Straight Section of Maze

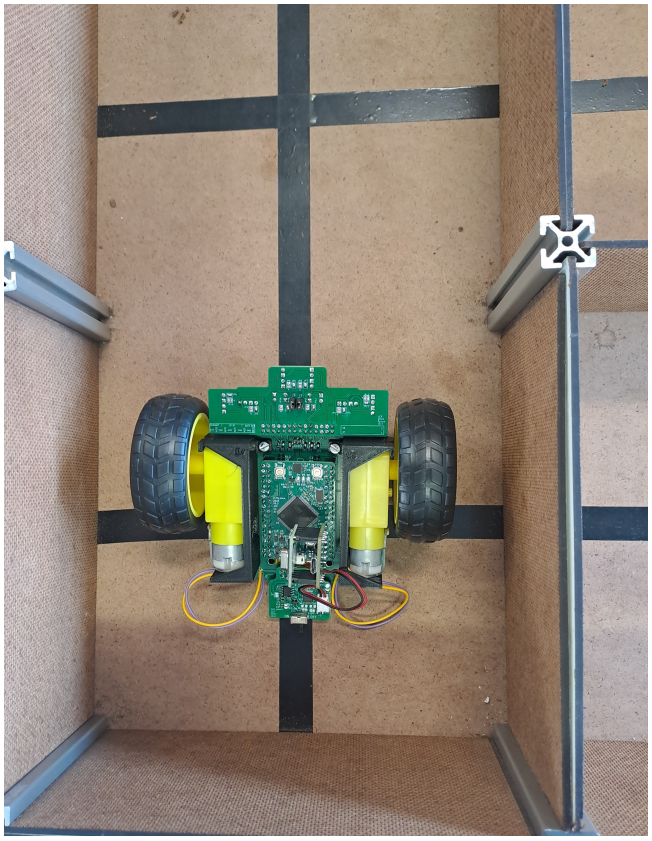


Figure 5.10: Micro-mouse in starting position, after calibration and rotation

Figure 5.11: Micro-mouse slightly misaligned with black line

Bibliography

[1] N. Ocean. Classic micromouse. [Online]. Available: https://ukmars.org/contests/micromouse/

[2] M. A. Dharmasiri. Micromouse from scratch. [Online]. Available: https://medium.com/@minikiraniamayadharmasiri/micromouse-from-scratch-algorithm-maze-traversal-shortest-path-floodfill-741242e8510