NUS ECE CG1111A Engineering Principles and Practice I
The Amazing Race Project

Table of Contents

1. Introduction

1.1. The A-maze-ing Race Project

The A-maze-ing Race Project is a maze challenge where all groups are to design and build a robot that finds its way through the maze in the shortest time. The robot faces a number of challenges at intermediate waypoints where the robot has to make its decision to make turns based on the colour detected underneath the robot. Within the course of three weeks, our group has built and programmed a robot that successfully navigated the maze. In this report, we will discuss the working principles of the design and algorithm of our robot.

1.2. Our Robot

Our robot consists of:

1. 1 x Metallic structure for body of mBot
2. 1 x Ultrasonic sensor
3. 1 x Infrared (IR) emitter and detector
4. 3 x LEDs (Red, Green, Blue)
5. 1 x Light Dependent Resistor (LDR)
6. 1 x HD74LS139P 2-to-4 Decoder IC Chip
7. 1 x SN754410 H-Bridge Chip
8. 5 x Resistors of varying resistance
9. 1 x Line sensor

10. 2 x DC Motors and wheels

11. 1 x mCore

Figures 1.2.1 to 1.2.4 show the completed robot and the position of the various components. Figure 1.2.5 shows the schematic for circuit logic of our robot, excluding line sensor, ultrasonic sensor, and DC motors.

Figure 1.2.1: Top view of robot.	Figure 1.2.2: Bottom view of robot.

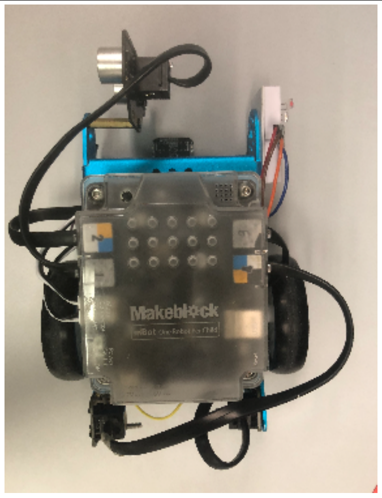
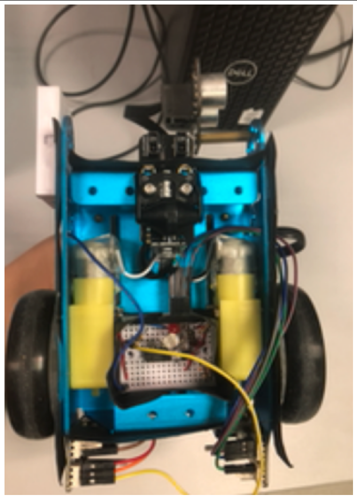

	

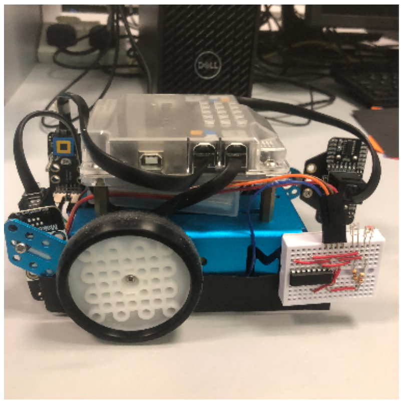
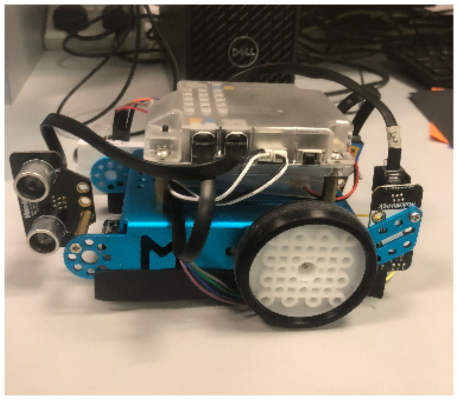
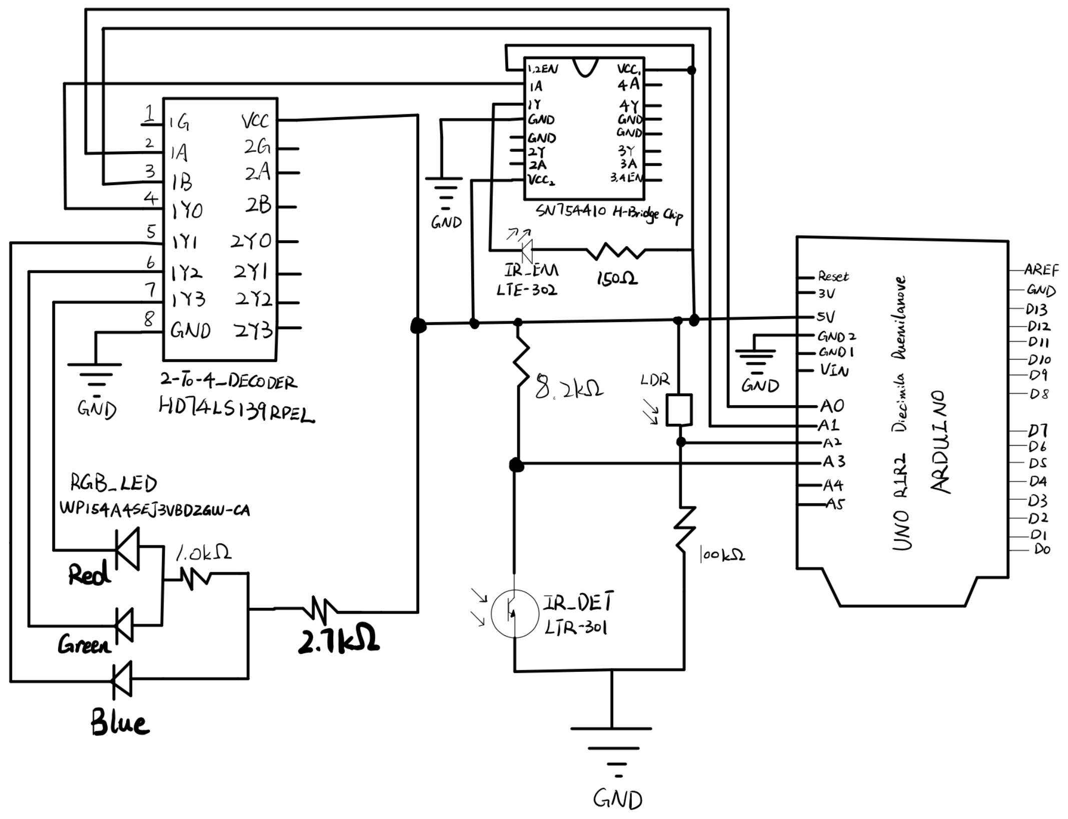
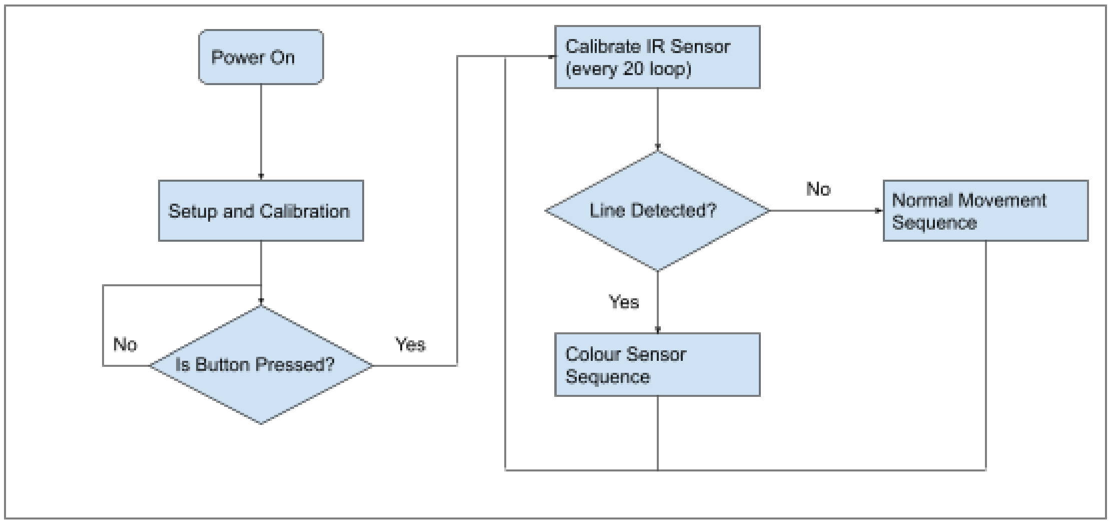
Figure 1.2.3: Right side view of robot.	Figure 1.2.4: Left side view of robot.
	

Figure 1.2.5: Schematic of components used for colour sensor and IR proximity sensors.



2. Overall Algorithm of robot

Figure 2.1: Overall workflow of algorithm used



2.1. Initial Power On

When the robot is powered on, the respective pins for button push, internal LED, those that lead to the 2-to-4 decoder, and measure values from the colour and IR sensors are set up accordingly. Serial communication will also be set up for debugging. An initial ambient IR reading is measured and stored in a global variable for use. Finally, calibration for the colour sensor would also be done in the sequence of white then black values (line 287 of arduino code). The built-in LED of the robot will then blink to denote the end of the set up sequence.

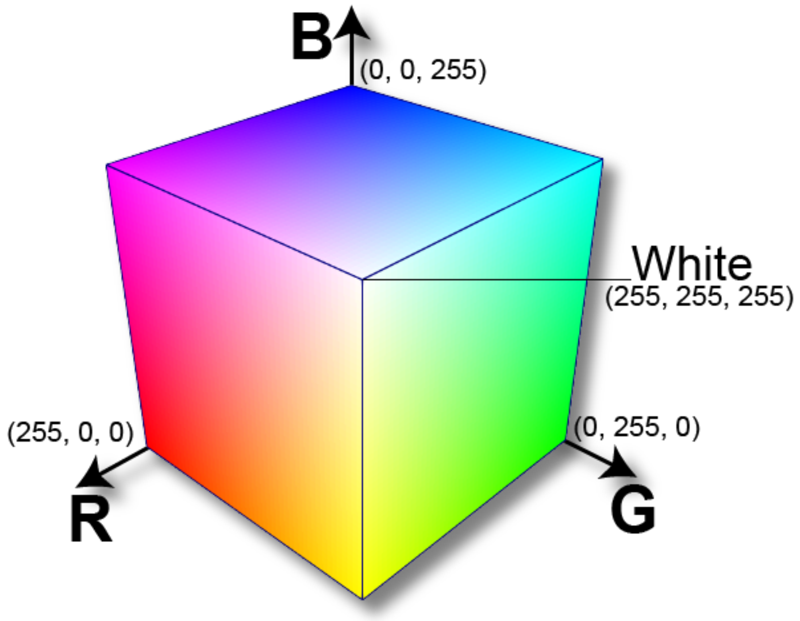
2.2. Moving Forward

When the robot is running, it enters a loop that starts with a condition check to see if the line tracker detects a black line. If the conditional check fails, the algorithm will check if the robot is too close to the walls. The ultrasonic sensor checks the distance between the robot and the left wall. If the distance is less than 7 centimetres, the robot steers right to the centre of the track. Similarly, the robot steers left if the IR proximity sensor detects that the robot is too close to the right wall. The turning action is carried out by stopping one of the wheels and maintaining the speed of the other wheel. If the robot is not too close to both walls, the robot moves with both wheels at the same speed. The process continues until the black line is detected.

2.3. Colour Detection

Upon detecting a black line by the line sensor, the vehicle stops and initiates the colour detection sequence. The LEDs are lit in the sequence of Red, Green then Blue, and the voltage measurements as a result of the resistance values of the Light Determinant Resistor in the circuit is measured. The measurements are repeated 5 times for the duration that each LED is lit and the average values are recorded. The values are then converted into RGB values based on the calibrations using white and black papers. The RGB values are then compared with the experimental RGB values of predefined colours of the colour tasks (Red, Green, Orange, Purple, Light Blue). We determine which of the colours is closest to the recorded colour using the concept of RGB colour space as shown in Figure 2.3.1, where the distance of the recorded colour from each of the predefined colours is calculated and the shortest distance from which will be labelled as the identified colour. However, a few special comparisons are included after we determine the identified colour due to inconsistency in differentiating between some of the colours during our testings. In particular, the differentiation between Red and Orange is done by comparing the G and B values (line 379-385 in arduino code), the differentiation between Orange and Green is done by comparing the R and G values (line 386-392 in arduino code). After identifying the colour, the corresponding task for each colour is initiated.

Figure 2.3.1 Colour space representation



3. Implementation Details of Subsystem

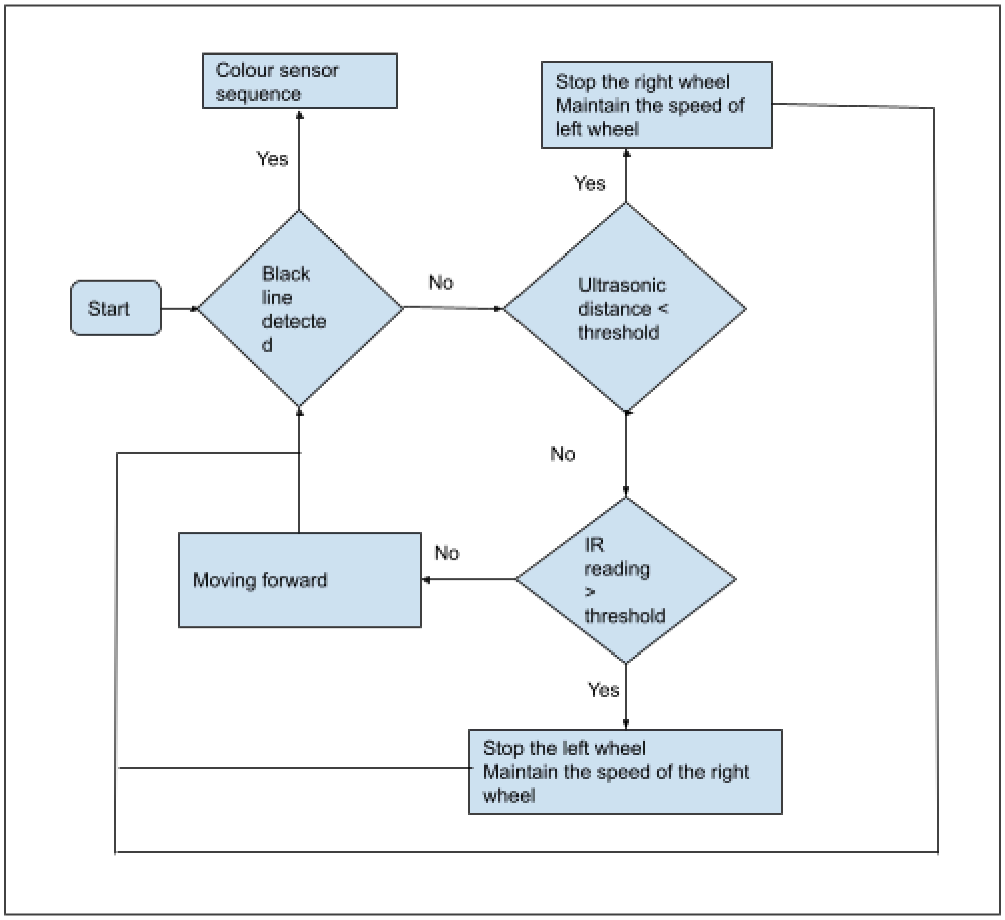
In this project, we utilize subsystems to meet the project requirements.

1. Ultrasonic and IR sensors ensure that the robot moves in a straight line.
2. Colour sensor detects the colour of the paper at each waypoint.
3. Line tracker checks the presence of the black line.

3.1. Keeping the robot centred

To keep the robot at the centre of the track, we use both the ultrasonic sensors and the IR proximity sensor. Figure 3.1.1 below shows the looping algorithm. The code corresponding to this flowchart can be found in FinalCode.ino , line 316-319 and line 463-500.

Figure 3.1.1: Algorithm to keep mBot centred


Colour sensor sequence
Stop the right wheel Maintain the speed of left wheel
Yes
Yes
Start
Black line detecte d
No
Ultrasonic distance < threshold
No
Moving forward
No
IR reading > threshold
Yes
Stop the left wheel Maintain the speed of the right wheel

3.1.1. The Left Side

We use an ultrasonic sensor to check the distance between the robot and the left wall. Provided that the minimum working distance is 3cm for ultrasonic sensors, placing the sensor on the left edge of the robot results in inaccurate readings when the robot is near the left wall, especially during left turns. Therefore, we mount the ultrasonic sensor between the ledges at the front of the robot, which ensures the distance between the sensor and the left wall is always more than 3 cm. The built-in library function of the robot was used to read in the distance as detected by the ultrasonic sensor.

The Figure 3.1.2 and Figure 3.1.3 below shows the pseudocode for the ultrasonic sensor. The threshold value is a defined constant of 7 cm away from the wall. The code also checks whether distance is more than is more than 0 cm. This conditional check fails when the left wall is absent as the ultrasonic sensor is not able to receive the sound wave and the calculated distance is 0. If the ultrasonic reading returns a valid value and the distance is less than the threshold distance of 7 cm, we steer the robot to the right by stopping the motion of the right wheel and maintaining the speed of the left wheel.

Figure 3.1.2: Code snippet for using ultrasonic sensor (Final Code,ino Line 493-502).

```
float get_ultrasonic_distance() {
return ultrasonic reading;
}
```

Figure 3.1.3: Code snippet for left steer (Final Code,ino Line 512-515).

```
(if the robot is too close to the left wall) {
The speed of the right motor becomes 0;
The speed of the left motor remains unchanged;
}
```

3.1.2. The Right Side

An IR proximity sensor checks if the robot is too close to the right wall. Since the limitation of the effective range was not applicable to the IR proximity sensor, we position the IR sensor on the right edge of the robot to face the right wall. Due to the fluctuation of ambient IR at different locations on the maze, we have to eliminate the influence of ambient IR to ensure that the decisions are made based purely on the reflected IR. The elimination of the influence of ambient IR would be elaborated in Section 4. When the IR sensor reading exceeds our threshold value, the robot is too close to the right wall. The speed of the left wheel becomes 0 and the speed of the right wheel remains unchanged. Similar to the ultrasonic sensor, the threshold value is a predetermined constant. We conducted multiple trials to test the output voltage at different distances between the IR sensor and the right wall. The change in voltage is insignificant when the distance is large. After multiple trials under different ambient settings, we decide on a threshold value of 350 (analog value). The pseudo code for the IR proximity sensor is shown in Figure 3.1.4. The overall code for how the robot remains at the centre is shown in Figure 3.1.5.

Figure 3.1.4: Code snippet for IR proximity sensor (Final Code.ino, Line 516-520).

```
(if the robot is too close to the right wall) {
The speed of the left motor becomes 0;
The speed of the right motor remains unchanged;
}
```

Figure 3.1.5: Code snippet for IR and ultrasonic sensor (Final Code.ino, Line 493-524).

```
while(running && !reached_black_line()) {
check if the robot is too close to the sides
if (the robot is too close to left)  {
 steer right;
}   else if (the robot is too close to right) {
 steer left;
} else {
  move forward;
}
stop moving;
```

3.2. Checking for waypoint

A line sensor is used to determine whether the robot should continue moving forward. There is a black strip at every way point challenge. The line sensor checks the presence of the black strip while moving forward. In the case where the robot reaches the black strip at an angle, the robot stops when either or both sensors detect a black strip. The conditional check ensures that the robot does not crash into the wall regardless of its position on the black strip. The pseudo code for the line sensor is shown in Figure 3.2.1. Afterwards, the colour sensor is activated to solve the waypoint challenge.

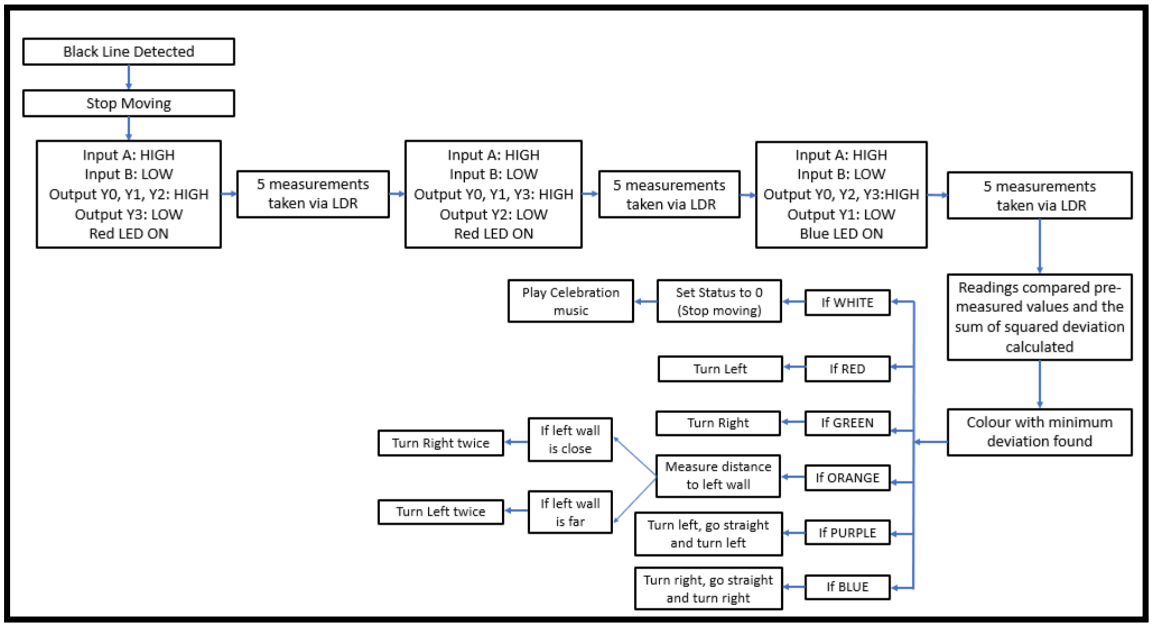
Figure 3.2.1: Code snippet for detecting black line (Final Code.ino, Line 337-339).

```
if (black line is detected)  {
The speed of both motors becomes 0;
}
```

3.3. Solving Waypoint Challenge

To solve the waypoint challenges, a colour sensor was used. To turn on the 3 different LED lights the 2-to-4 decoder IC was used. When the output of the decoder was low, the LED was switched on (the default being high). Each LED was turned on individually and the average of 5 different readings was taken for each colour to increase accuracy. The RGB values were then calculated based on earlier calibration. To determine which colour best matched the RGB values, we compared each of the measured RGB values with previous measurements for the paper and found the one with the minimum sum of error squared. This method was not always accurate for differentiating between red and orange as well as orange and green. Thus, special comparisons were carried out in those cases.

Figure 3.3.1: Algorithm for colour detection.



Black Line Detected
Stop Moving
Input A: HIGH Input B: LOW Output Y0, Y1, Y2: HIGH Output Y3: LOW Red LED ON
5 measurements taken via LDR
Input A: HIGH Input B: LOW Output Y0, Y1, Y3: HIGH Output Y2: LOW Red LED ON
5 measurements taken via LDR
Input A: HIGH Input B: LOW Output Y0, Y2, Y3:HIGH Output Y1: LOW Blue LED ON
5 measurements taken via LDR
Readings compared pre-measured values and the sum of squared deviation calculated
Colour with minimum deviation found
If WHITE
Set Status to 0 (Stop moving)
Play Celebration music
If RED
Turn Left
If GREEN
Turn Right
If ORANGE
Measure distance to left wall
If left wall is close
Turn Right twice
If left wall is far
Turn Left twice
If PURPLE
Turn left, go straight and turn left
If BLUE
Turn right, go straight and turn right

Based on the colour functions, the motors will turn according to the required direction by changing the speed and direction of each motor independently. To turn right or left, the motors turned either both clockwise (for left) or both anti-clockwise (for right). To turn 180 degrees 2 consecutive turns were taken. To determine which direction to turn 180, the ultrasonic sensor is used to find the distance to the left wall. The direction is determined by whether the distance is more or less than a certain threshold. For the purple and blue waypoints, after the first turn the motors were turned for a fixed duration (calibrated separately).

Figure 3.3.1: Code snippet for solving waypoint challenge (Final Code.ino, lines 316-463).

```
{
After black line is detected
The speed of both wheels become 0;
read rgb values
identify color using minimum deviation as compared to previous measurements
if closest colour is red or orange analyse the blue and green value of the measurement
if closest colour is green check if the green value is larger than the blue value
turn the robot based on identified color
}
```

3.4. Reaching end of the maze

The end of the maze is identified when a black line is reached and the colour detected is white

Figure 3.4.1: Code snippet for end of maze (Final Code.ino, lines 450-460).

```
if identified color is white {
```

```
The robot stops;
Celebratory tune plays;
}
```

4. <u>Calibration of Custom-Built Sensors</u>

To obtain a reliable gauge of the distance of the right wall from the vehicle as it moves through the maze, a threshold IR value will have to be updated in real-time. The threshold IR value is determined on the voltage value of the IR sensor resulting from the presence of ambient IR. The IR emitter is turned off for 30 milliseconds to allow time for the IR detector to adjust to the ambient IR, after which the measurement is recorded as our threshold value and the IR emitter is turned back on. The calibration is done every 20 loops of the robot main algorithm sequence.

<u>Figure 4.1: Pseudo code for calibrating IR sensor (lines 316-329 in arduino code)</u>.

```
{
Turn IR emitter off
Wait for 30 milliseconds
Record readings
Turn IR emitter on
}
```

The calibration for the colour sensor is done by determining the RGB values when detecting white and black coloured papers and getting a range of viable values for each

RGB spectrum. The range is then translated into the standard RGB range of 0 to 255. The white and black coloured papers are placed under the colour sensor during the calibration for the respective colours.

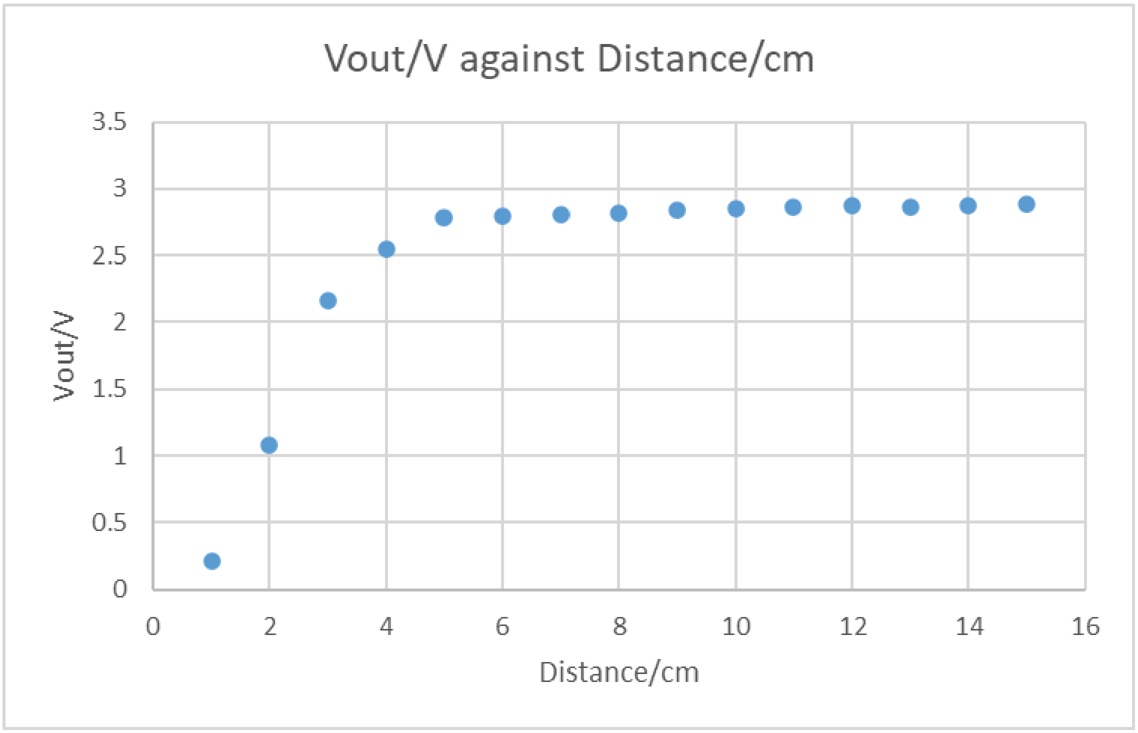
Figure 4.2: Pseudo code for calibrating colour sensor (lines 540-583 in arduino code).

```
{
Turn each LED colour on and record the corresponding readings for white coloured paper
Turn on built-in LED to indicate end of white calibration
Turn each LED colour on and record the corresponding readings for black coloured paper
Turn on built-in LED to indicate end of black calibration
}
```

4.1. IR Sensor

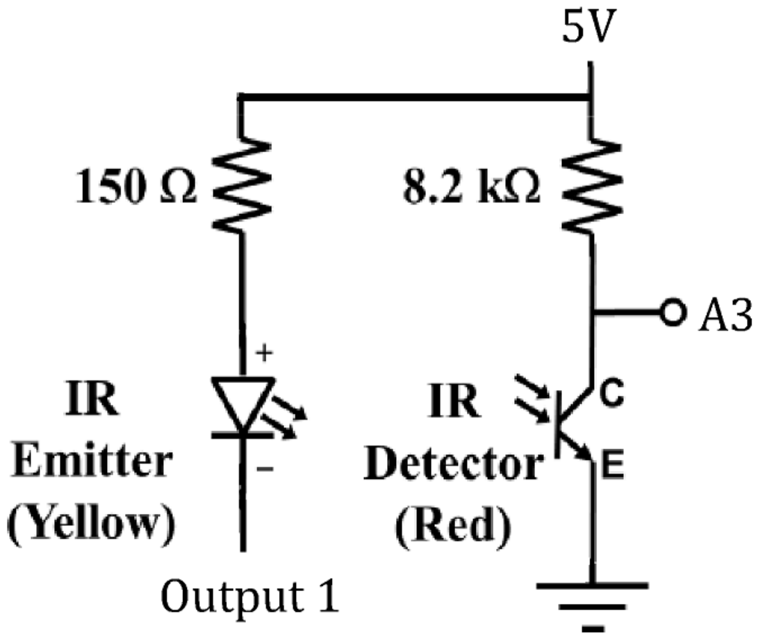
Measurements from the IR sensor are recorded through analog pin A3 which provides a value in the range from 0 to 1023 corresponding to the voltage readings of 0V to 5V in the IR sensor circuit. Figure 4.1.1 below illustrates the general relationship of the voltage readings of the custom-built IR sensors to the distance of the obstacle from the IR sensor.

Figure 4.1.1: Relationship between voltage value and distance of obstacle



The IR emitter starts to have a significant effect on the IR detector once the obstacle comes within around 5cm of the IR sensor. A distance of 2.5cm between the IR sensor and the wall corresponds to roughly a 1.7V drop from the threshold value (maximum value). The 1.7V corresponds to around a 350 drop in terms of analog reading value, hence 350 below the threshold value is selected as our condition for adjusting the vehicle towards the left side. Figure 4.1.2 below showcases the circuitry of our IR sensor.

<u>Figure 4.1.2: Circuitry of IR sensor</u>

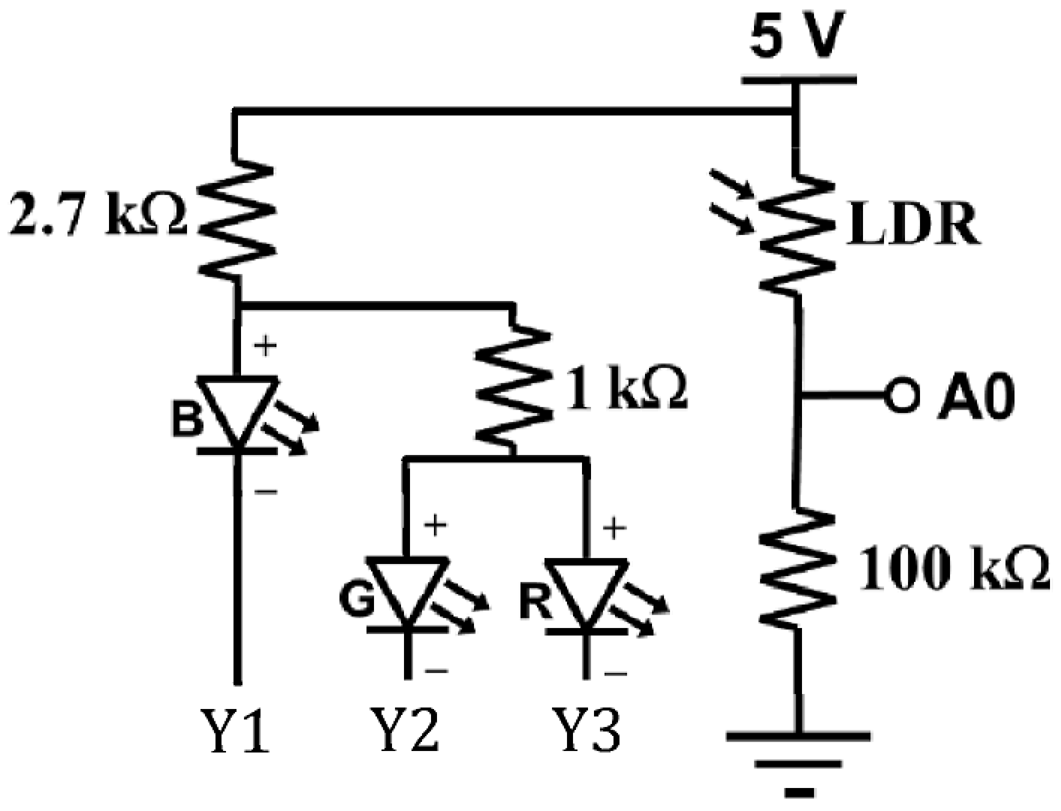


Output 1 is connected to 1Y of L293D which is controlled by 1A of L293D connected to Y0 of HD74LS139P. When Y0 is at high signal, the IR emitter will be off, and conversely if Y0 is at low signal, the IR emitter will be on.

4.2. Colour Sensor

Measurements from the colour sensor are obtained through analog pin A2 and converted into values in the range 0-255 for each of the RGB colours. The cathodes of the Red, Green and Blue LEDs are connected to Y3 to Y1 of the HD74LS139P respectively for controlling the turning on of individual coloured LEDs. With different combinations of input received by the 2-to-4 decoder, each of the LEDs is able to turn on one at a time for the recording of the RGB values. Experimentally, the Red and Green LEDs emit higher intensity of light as compared to that of the Blue LED, thus an additional 1k ohms resistor is added to the anodes of the Red and Green LEDs to bring down the intensity for a larger range between white and black paper calibration. The circuitry of the colour sensor is as shown below in Figure 4.2.1.

Figure 4.2.1: Circuitry of colour sensor



5. Work Division

Shanay:

- IR sensor circuitry and programming
- Arduino code integration and debugging
- Report

Yangda:

- Line sensor circuitry and programming

- Ultrasonic sensor circuitry and programming
- Celebration tune
- Report

Jingxi:

- Colour sensor circuitry and programming
- Colour tasks
- Report

6. Challenges Faced

Initially, we encountered a difficulty where the colour sensor was not able to distinguish orange and red colour at the waypoints. To solve the problem, we decided to add an additional conditional check. When the closest colour is red or orange, the blue and green values detected are compared for an additional round before a colour, either orange or red, is returned. Moreover, when the robot underwent a 180-degree turn at the waypoint where orange-coloured paper is placed, the robot bumped into side walls easily. We decided to programme the robot to turn towards the side that is more spacious. To do so, we use our ultrasonic sensor to find the distance to the left wall. The turning direction is determined by whether the distance is more or less than a certain threshold.