# Table of Contents

**Moten, Annabelle**

Electronics and Information Technology – Heritage College

**MAZE-Runner ROBOT D.I.Y MANUAL**

FINAL PROJECT II

Table of Contents

Table of Contents 1

table of Figures 2

Table of Tables 3

Glossary of Key Terms 4

Project Overview 4

D.I.Y. Manual Overview 5

1 Parts & Components 6

* 1. Bill of Materials 6

1.2 Ordering the PCBs 7

1.2.1 Selecting Parameters for Sensor Array PCb 7

1.2.2 Selecting Parameters for Power Supply PCB 8

2 Prototype Assembly 8

2.1 Component Soldering 8

2.1.1 5-IR Sensor Array Component Soldering 8

2.1.2 Power Supply PCB Component Soldering 10

2.2 Chassis, Motors & Wheels Assembly 13

2.2.1 Attaching Wheels to Gear Motors 14

2.2.2 Chassis, Sensor Array & Power Supply PCB Assembly 18

2.3 Component Wiring 21

2.3.1 Component Wiring – Connecting The Sensor PCB 21

2.3.2 Component Wiring – Connecting The Power Supply PCB 21

3 Prototype Testing/troubleshooting 22

3.1 5-IR Sensor Array Testing/Troubleshooting. 23

3.1.1 testing – Output Monitoring 23

3.1.2 Testing – IR-Sensor Array calibration 25

3.2 Voltage regulator Circuit testing/Troubleshooting 25

3.2.1 testing – Output Monitoring 25

3.2.2 Troubleshooting – Unexpected Output Monitoring 26

4 User Guide 27

4.1 GUI Functionalities 28

4.1.1 Timer – Communicating between NUCLEO & PC 29

4.2 Other Important considerations 29

5 Future Considerations 30

5.1 Adding a Manual Mode 30

5.2 Revising the Voltage Regulator Circuit Design 30

5.3 Refine code & Implement Maze Loops 30

6 Annex A 31

6.1 Proteus Final Schematic & Layout Files 31

6.1.1 5-IR Sensor Array Proteus Files 31

6.1.2 Voltage Regulator Proteus Files 31

6.2 Prototype testing Code 31

6.2.1 5-IR Sensor Array Test Code 31

6.2.2 Gear Motor test Code 31

6.3 Final Microcontroller Code 31

6.4 Final GUI code 31

6.5 Physical Maze Examples 31

7 Annex B 33

7.1 Assembled Prototype – Sensor Array View 33

7.2 Assembled Prototype – Motors & Wheels 33

7.3 Assembled Prototype – Front Body View 34

7.4 Assembled Prototype – side Body View 34

# table of Figures

[Figure 1 – Component Soldering Locations A1](#Toc8396790) 9

[Figure 2 – Component Soldering Location B1](#Toc8396791) 10

[Figure 3 – Component Soldering Locations A2](#Toc8396792) 12

[Figure 4 -- Component Soldering Locations B2](#Toc8396793) 12

[Figure 5 -- Component Soldering Locations C2](#Toc8396794) 13

[Figure 6 -- Bottom Chassis Reference](#Toc8396795) 14

[Figure 7 -- Top Chassis Reference](#Toc8396796) 14

[Figure 8 -- Mounting the Motor to the Bracket](#Toc8396797) 15

[Figure 9 -- Rubber Wheel Cover](#Toc8396798) 15

[Figure 10 -- Mounting Wheel to Hub](#Toc8396799) 15

[Figure 11 -- Securing the Hub to the Shaft](#Toc8396800) 16

[Figure 12 -- Final Motor & Wheel Assembly](#Toc8396801) 16

[Figure 13 -- Bottom Chassis Mounting Diagram](#Toc8396802) 17

[Figure 14 -- 5-IR Sensor Array PCB Orientation Diagram](#Toc8396803) 18

[Figure 15 -- Power Supply PCB Orientation Diagram](#Toc8396804) 18

[Figure 16 -- Aligning Sensor PCB & Chassis](#Toc8396805) 19

[Figure 17 -- Aligning Top & Bottom Chassis](#Toc8396806) 19

[Figure 18 -- CoolTerm COM Port Setup](#Toc8396807) 23

[Figure 19 -- Monitored Sensor Data Output](#Toc8396808) 23

[Figure 20 -- Voltage Regulator Troubleshooting](#Toc8396809) 25

[Figure 21 -- MazeBot GUI Application](#Toc8396810) 26

[Figure 22 -- Maze Example A](#Toc8396811) 30

[Figure 23 -- Maze Example B](#Toc8396812) 30

[Figure 24 -- Maze Example C](#Toc8396813) 30

[Figure 25 -- Sensor Array View](#Toc8396814) 31

[Figure 26 -- Motor & Wheels View](#Toc8396815) 31

[Figure 27 -- Front Body View](#Toc8396816) 32

[Figure 28 -- Side Body View](#Toc8396817) 32

# Table of Tables

[Table 1 – Finalized Bill of Materials](#Toc8396818) 6

[Table 2 -- IR Sensor PCB Connections](#Toc8396819) 20

[Table 3 -- Voltage Regulator PCB Connections](#Toc8396820) 20

[Table 4 -- Motor Encoder Wire Functions](#Toc8396821) 21

# Glossary of Key Terms

* **BLDC** – also known as a ***Brushless DC motor*** & is a synchronous motor powered by DC power via an inverter or switching power supply which produces an AC current to drive each phase of the motor using loop control functionalities.
* **BOM** – also known as a ***Bill of Materials***, and is a list of the materials, sub-assemblies, intermediate assemblies, sub-components, parts, and the quantities of each needed to manufacture an end product
* **CPR** – refers to the amount of ***Counts per Revolution*** of a component (in this case its referring to the gear-motor encoder)
* **DMM** – also referred to as a ***Digital Multi-Meter*** and is used to test circuitry in terms of voltage, current, resistance, inductance, continuity, etc.
* **GUI** – also known as a ***Graphical User Interface*** which allows interaction between the hardware & software (i.e., provides communication functionalities between the microcontroller and the PC).
* **IC** – refers to an ***Integrated Circuit*** which is most commonly known as a chip or a microchip. This component is a semiconductor and contains miniature circuitry of its own (e.g., transistors, resistors, capacitors, etc.).
* **IFD** – refers to the ***Initial Facing Direction*** of the robot’s displayed orientation (on the GUI) relative to its physical orientation in the maze (i.e., North, South, East, West).  
  IR-Sensors – also known as Infrared Sensors
* **NUCLEO** – this is referring to the microcontroller used in the entirety of the project (e.g., the ***STM32-NUCLEOF411RE***).
* **PCB** – also known as a ***Printed Circuit Board*** & can be described as a board made for connecting electronic components together. Nowadays, PCBs are used in almost all electronic circuitry systems.
* **SPDT** – refers to a switch with one single pole & 2 throws (i.e., Single-Pole Double-Throw switch) and is used to switch between two separate circuits
* **UFD** – this refers to the displayed robot’s new or ***Updated Facing Direction*** which is, again, relative it’s the physical orientation.

# Project Overview

The Maze-Runner Robot is a project which allows a user to monitor a real-time mapping of a given maze (consists ultimately of black tape on a white surface), all while it is autonomously solving it. The idea is simple; every time it has successfully solved the maze, the path is stored; finally, the robot uses this acquired on its future runs, to solve for the shortest path possible.

**\*NOTE: This specific prototype does not account for maze loops as the code is much more difficult than a simple loop free maze. Thus, if creating your own maze, feel free to use the example mazes in the section below.**

# D.I.Y. Manual Overview

This manual consists **five major separate sections**; Parts & Component, Prototype Assembly, Prototype Testing/Troubleshooting, User Guide, and Future Considerations.

1. ***Parts & Components*** – this section will provide a full list of the required components, as well as where to purchase them, and an explicit final BOM to be used as reference.
2. ***Prototype Assembly*** – this section demonstrated how to fully assemble the prototype, in terms of PCBs, chassis, wheels, & motors.
3. ***Prototype Testing/Troubleshooting*** – this section describes how to test each major circuitry components individually (e.g., testing of voltage regulator circuit, testing IR-sensor array…).
4. ***User Guide*** – this section describes how the user can interact with the prototype (i.e., describes the GUI functionalities, how to calibrate sensors).
5. ***Future Considerations*** – this section touches on how the project can be modified in the future to improve its functionality.

# 1 Parts & Components

All the required components & hardware for this project are provided in the sub-sections below.

## Bill of Materials

Table 1 – Finalized Bill of Materials

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Maze-Runner Robot Final Bill of Materials** | | | | | |
| **Part Number** | **Manufacturer** | **Manufacturer Part Number** | **Quantity** | **Price (CAD)** | **Part Description** |
| 2822 | Polulu | RB-Pol-350 | 2 | $ 98.00 | 12V, 19:1 Gear Motor w/ 64 CPR Encoder |
| 1084 | Polulu | RB-Pol-03 | 1 | $ 10.36 | 37D mm Metal Gear motor Bracket (Pair) |
| 1083 | Polulu | RB-Pol-136 | 1 | $ 9.86 | Universal Aluminum 6mm Mounting Hubs (4-40) |
| 1435 | Polulu | RB-Pol-122 | 1 | $ 13.00 | Wheel 90 x 10mm Black (Pair) |
| X0017RSG9X | Gikfun | EK8443 | 1 | $ 11.99 | 5mm 940nm LEDs IR-receiver & photodiode pair (pack of 20pcs) |
| 1153 | Cytron | RB-Cyt-48 | 2 | $ 1.54 | 4-AA battery holder (in series) |
| 11829 | Sparkfun | RB-Spa-155 | 2 | $ 0.77 | SPDT mini power switch |
| **n/a** | **n/a** | **n/a** | 1 | $ 6.99 | Pack of 4-40 screws & lock nuts (12pcs each) |
| **n/a** | **n/a** | **n/a** | 1 | $ 12.99 | Pack of 10-AA batteries |
| **n/a** | **n/a** | **n/a** | 1 | $ 20.00 | STM32-NUCLEOF411RE |
| **n/a** | **n/a** | **n/a** | 1 | **n/a** | L298 Dual H-Bridge Motor Driver (from class part bins) |
| **n/a** | **n/a** | **n/a** | 3 | **n/a** | LM358 Dual Op-Amp (from class part bins) |
| **n/a** | **n/a** | **n/a** | 2 | $ 30.00 | Chassis modifications (drill holes & cuts) |
| **n/a** | **n/a** | **n/a** | 1 lot of 10pcs | $ 26.00 | Power Supply PCB (130.5 x 84.0 mm) |
| **n/a** | **n/a** | **n/a** | 1 lot of 10pcs | $ 26.00 | 5-IR Sensor Array PCB (100.0 x 46.0 mm) |
| **n/a** | **n/a** | **n/a** | 10 | **n/a** | Resistors (5 x 330-ohms & 5 x 10k-ohms) from class part bins |
| **n/a** | **n/a** | **n/a** | 5 | **n/a** | 10k-ohms potentiometers |
| **n/a** | **n/a** | **n/a** | 1 | **n/a** | LM1117T-5.0 voltage regulator (from class parts bin) |
| **n/a** | **n/a** | **n/a** | 2 | **n/a** | 10uF electrolytic capacitors (from class part bins) |
| **n/a** | **n/a** | **n/a** | 1 | **n/a** | 1000nF ceramic capacitor (from call parts bin) |
| **n/a** | **n/a** | **n/a** | 1 | **n/a** | 100nF ceramic capacitor (from call parts bin) |
| **n/a** | **n/a** | **n/a** | 8 | **n/a** | 1N4937 diodes (from call parts bin) |
| **n/a** | **n/a** | **n/a** | 3 | **n/a** | 8 POS DIP IC Socket (from call parts bin) |
| **n/a** | **n/a** | **n/a** | **n/a** | $ 40.00 | Total shipping fees |
| **n/a** | **n/a** | **n/a** | **n/a** | $ 8.99 | Wheel Caster |
| **Total Project Cost (CAD)** | | | | | **$ 324.63** |

## 1.2 Ordering the PCBs

If unsure of what PCB manufacturer to use, please consult the link provided below. This company is based in China however so make sure to order well in advance as most services will be unavailable during their holidays.

* <https://www.pcbgogo.com>

### 1.2.1 Selecting Parameters for Sensor Array PCb

Begin the instant quote by inputting the following parameters in their respective fields:

* Select the Dimensions | ***length = 100 mm & width = 46 mm***
* Select the Quantity | ***1 lot of 10pcs***
* Select the Amount of layers | ***2-layers*** |top & bottom layers
* Select the board Thickness | ***1.6 mm*** | by default

Now press the ‘Quote Now’ widget to move on to the next page. Moving on, select **ONLY** the following parameters for the PCB prototype and leave the rest with their default values:

* Size (single) | ***length = 100 mm & width = 46 mm***
* Quantity (single) | ***10***
* Solder Mask | ***Green*** by default | changing it will increase the price
* Surface Finish | ***HASL led free***

### 1.2.2 Selecting Parameters for Power Supply PCB

Begin the instant quote by inputting the following parameters in their respective fields:

* Select the Dimensions | ***length = 100 mm & width = 84 mm***
* Select the Quantity | ***1 lot of 10pcs***
* Select the Amount of layers | ***2-layers*** |top & bottom layers
* Select the board Thickness | ***1.6 mm*** | by default

Now press the ‘Quote Now’ widget to move on to the next page. Moving on, select **ONLY** the following parameters for the PCB prototype and leave the rest with their default values:

* Size (single) | ***length = 100 mm & width = 84 mm***
* Quantity (single) | ***10***
* Solder Mask | ***Green*** by default | changing it will increase the price
* Surface Finish | ***HASL led free***

Once the parameter selection is complete, add the board to the cart and select the shipping company of your preference. Before being able to pay, you will need to upload your design files and wait a few hours for the company to look over & process the request. Once this is complete and they’ve approved of the PCB design, you will be able to pay and complete the checkout process.

# 2 Prototype Assembly

At this point, all your hardware components should be received and ready to assemble. This section will be split-up into 3 sub-sections; component soldering, component wiring, and the final hardware connection.

## 2.1 Component Soldering

Since there are two boards to be assembled, this sections will be split in two respectively. The first will be for the 5-IR sensor array PCB and the second will be for the voltage regulator & motor driver board.

### 2.1.1 5-IR Sensor Array Component Soldering

This sub-section provides detailed description about the soldering of each component on this specific board. The following sub-section serves as a sort of legend, when referring to **Figure 1** & **Figure 2**.

#### 2.1.1.1 Component Soldering – Headers

The location of all headers to be soldered on this board are shown as yellow rectangles in **Figure 1**.

* Solder a **5-POS male header** in the yellow rectangle labelled as ‘**SENSOR\_DATA**’ in **Figure 1**.

#### 2.1.1.2 Component Soldering – Resistors & Potentiometers

The location of all resistors to be soldered on this board are shown as green rectangles in **Figure 1**.

* Solder the **330-ohm** resistors in the green rectangles labelled **R2, R4, R6, R8, R10** in **Figure 1.**
* Solder the **10k-ohm** resistors in the green rectangles labelled **R1, R3, R5, R7, R9** in **Figure 1.**
* Solder the **10k-ohm** potentiometers in the green rectangles labelled **RV1, RV2, RV3, RV4, RV5** in **Figure 1.**

#### 2.1.1.3 Component Soldering – Diodes

The location of all the IR-emitter LED to be soldered will be in orange circles & the location for all photodiodes are shown as blue circles in **Figure 2**.

* Solder the **IR-emitter LEDs** in the orange circles labelled **D6-D10** in **Figure 2**.
* Solder the **photodiodes** in the blue circles labelled **D1-D5** in **Figure 2**.

**\*NOTE: When soldering the diodes, one MUST make sure that all LEDS are in fact at the same height. You can do this with the use of a perfectly flat surface when soldering to ensure the best possible accuracy.**

**\*Note: Also make sure to solder the diodes on the bottom side of the board as well AS IN the proper orientation as they are polarized. Recall that the ‘a’ denotes the anode (+) & the ‘k’ denotes the cathode (-)**

#### 2.1.1.4 Component Soldering – Sockets & Connectors

The location of all the DIP-sockets to be soldered will be in pink rounded rectangles & the location for all connectors are shown as purple rounded rectangles in **Figure 2**.

* Solder the **8-POS DIP IC sockets** in the pink rounded rectangles labelled as **U1-U3** in **Figure 2**.
* Solder the **2-POS connector** in the purple rounded rectangle labelled as ‘**POWER\_TERMINAL**’ in **Figure 2**.

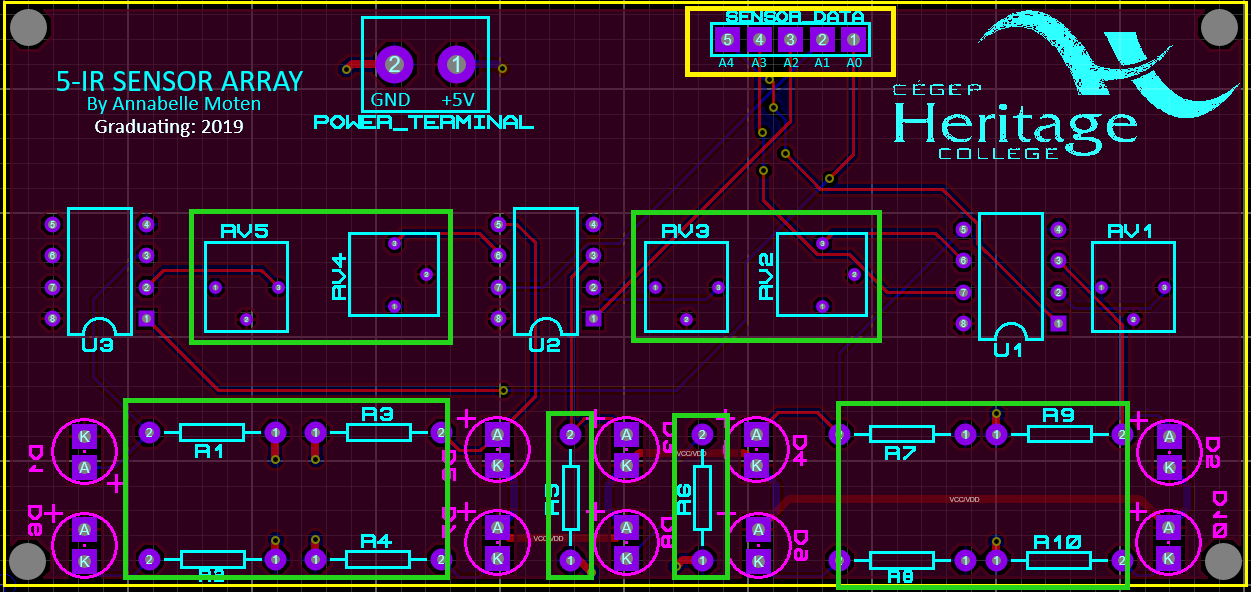


Figure 1 – Component Soldering Locations A1

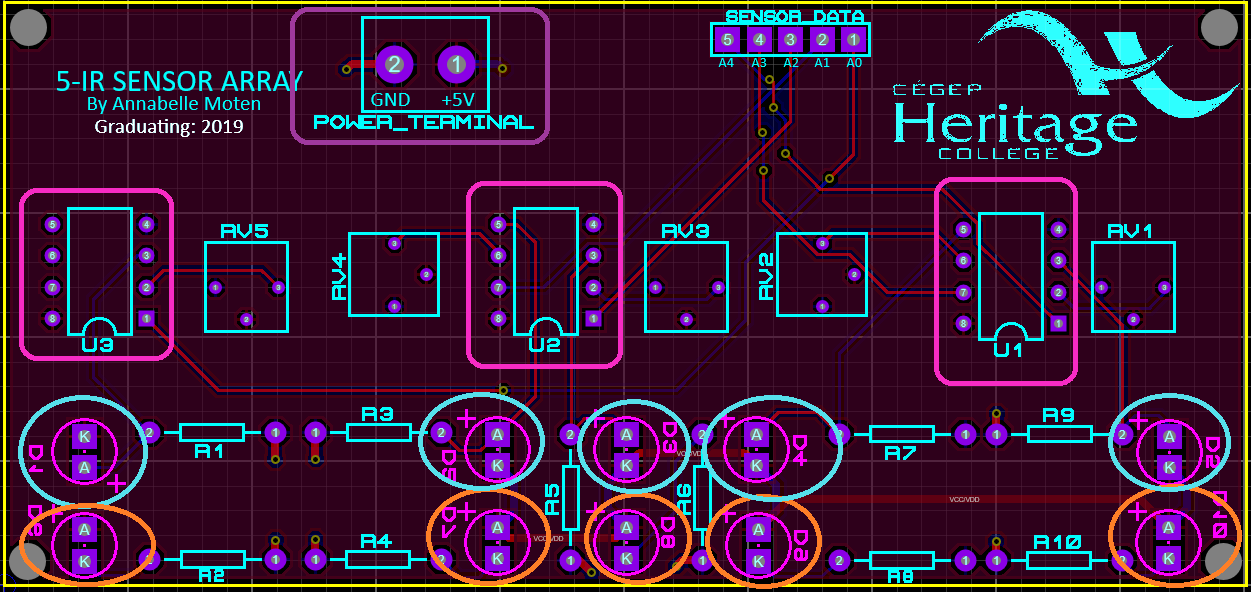


Figure 2 – Component Soldering Location B1

### 2.1.2 Power Supply PCB Component Soldering

This sub-section provides detailed description about the soldering of each component on this specific board. The following sub-section serves as a sort of legend, when referring to **Figure 3, Figure 4** & **Figure 5**.

**\*note: The pcb designed used here has an error in terms of the power LED & current limiting resistor placement. As such, both components are to be replaced by jumper wire to prevent a short from damaging the remaining circuitry.**

#### 2.1.2.1 Component Soldering – Headers

The location of all headers to be soldered on this board are shown as yellow rectangles in **Figure 3**.

* Solder a **5-POS female header** in the yellow rectangle labelled as ‘**SENSOR\_DATA**’ in **Figure 3**.
* Solder a **5-POS male header** in the yellow rectangle labelled as ‘**SENSOR\_DIG\_DATA\_MICRO\_CONN**’ in **Figure 3**.
* Solder a **10-POS DIP male header** in the yellow rectangle labelled as ‘**SENSOR\_ANL\_DATA\_MICRO\_CONN**’ in **Figure 3**.
* Solder a **6-POS female header** in the yellow rectangle labelled as ‘**LEFT\_MOTOR**’ in **Figure 3**.
* Solder a **6-POS female header** in the yellow rectangle labelled as ‘**LEFT\_MOTOR**’ in **Figure 3**.

**\*Note: The footprint used for the l298 motor driver ic, for this specific design, does not have the proper drill hole size, thus it has to be mounted on the 8-POS & 7-POS female headers described below.**

* Solder an **8-POS female header** in the yellow rectangle labelled as ‘**U2**’ in **Figure 3**.
* Solder a **7-POS female header** in the yellow rectangle labelled as ‘**U2**’ in **Figure 3**.

#### 2.1.2.2 Component Soldering – Resistors & Capacitors

The location of all resistors to be soldered on this board are shown as green rectangles in **Figure 3**.

* Solder the **jumper wire** in the green rectangles labelled **R4** in **Figure 3.**
* Solder the **10uF** electrolytic capacitors in the green rectangles labelled **C1** & **C3** in **Figure 3.**
* Solder the **100nF** ceramic capacitor in the green rectangles labelled **C2** in **Figure 3.**
* Solder the **1000nF** ceramic capacitor in the green rectangles labelled **C4** in **Figure 3.**

#### 2.1.2.3 Component Soldering – Diodes

The location of all 1N4937 diodes to be soldered will be in orange circles & the location for the power LED is shown as a blue circle in **Figure 4**.

* Solder the **jumper wire** in the blue circle labelled **‘POWER\_LED’** in **Figure 4.**
* Solder the **1N4937** diodes in the orange circles labelled ‘**D1-D5**’ & ‘**D7-D9**’ in **Figure 4**.

**\*Note: Also make sure to solder the diodes on the bottom side of the board as well AS IN the proper orientation as they are polarized. Recall that the ‘a’ denotes the anode (+) & the ‘k’ denotes the cathode (-)**

#### 2.1.2.4 Component Soldering – Connectors & ICs

The location of all connector terminals to be soldered will be in pink rectangles & the location of all ICs to be soldered will be in white rectangles

* Solder the **2-POS connector block** in the pink rectangle labelled ‘**12V\_BATTERY**’ in **Figure 5**.
* Solder the **SPDT-switch** in the pink rectangle labelled ‘**POWER\_SWITCH**’ in **Figure 5**.
* Solder the **3-POS connector block** in the pink rectangle labelled ‘**5V VMOT GND**’ in **Figure 5**.
* Solder the **LM1117T IC** in the white rectangle labelled ‘**U3**’ in **Figure 5**.
* Place the **L298 IC** in the headers seen in the white rectangle labelled ‘**U2**’ in **Figure 5**.

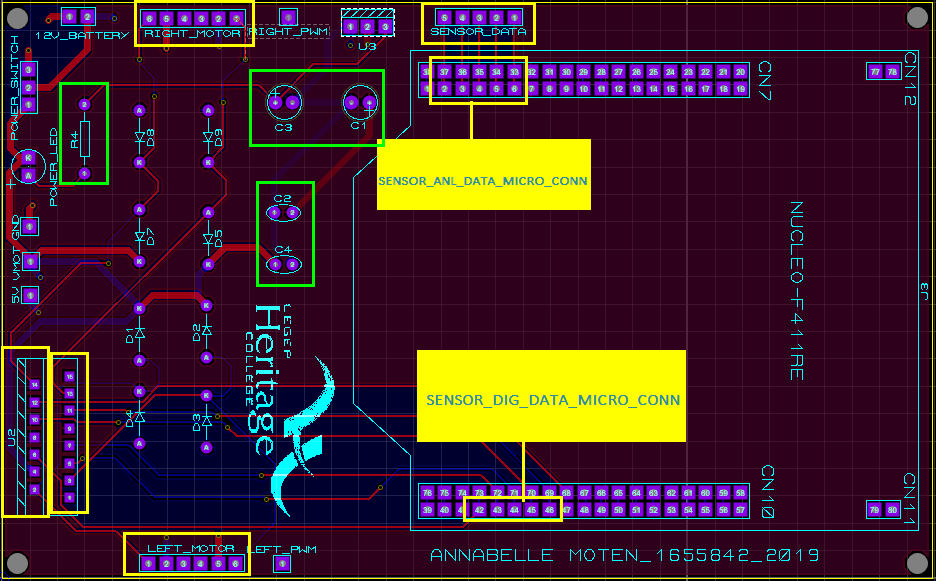


Figure 3 – Component Soldering Locations A2

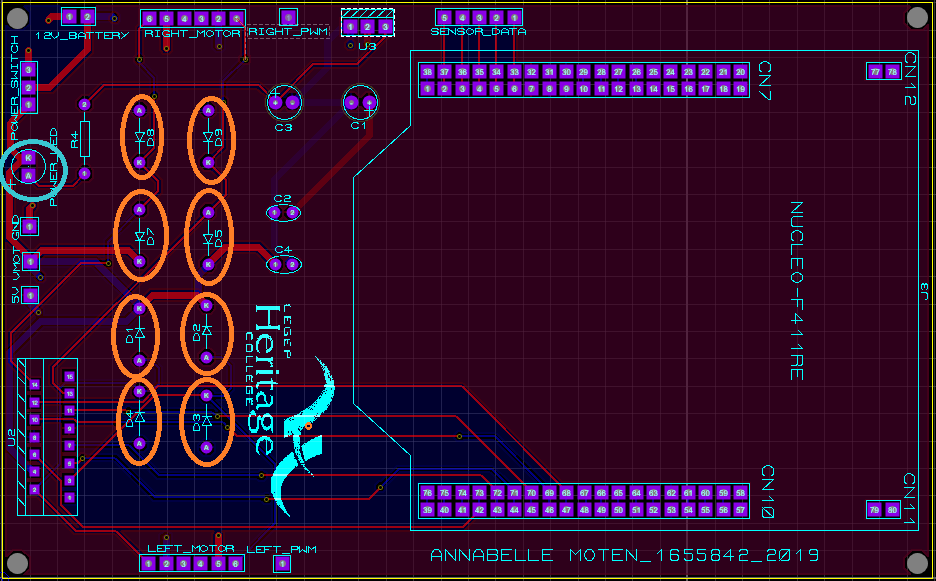


Figure 4 -- Component Soldering Locations B2

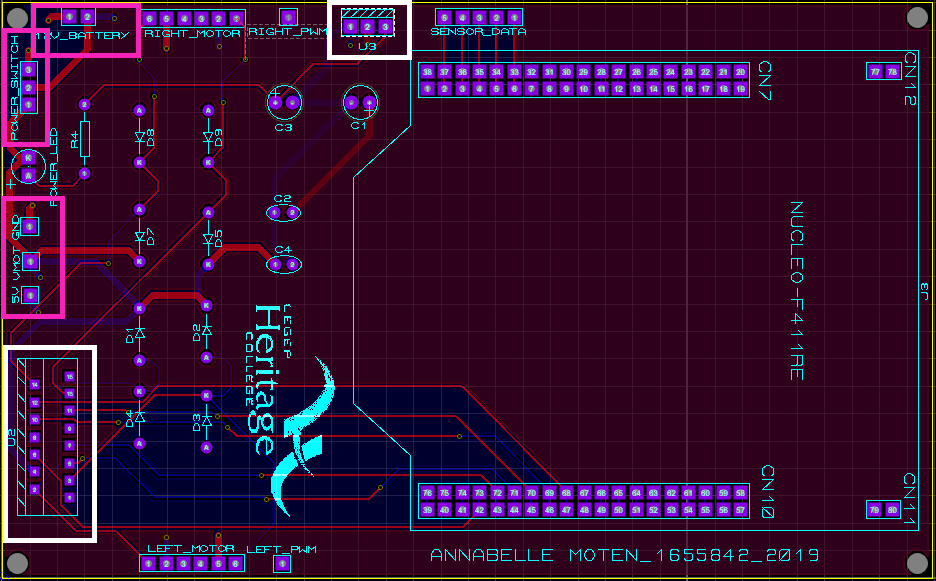


Figure 5 -- Component Soldering Locations C2

## 2.2 Chassis, Motors & Wheels Assembly

Now that the soldering portion of the project is completed, it is time to begin assembling the body of the MazeBot (i.e., assembling the motors & wheels onto the chassis). Before beginning the assembly, it is important to understand the proper orientation of the chassis. Two reference images are provided below.

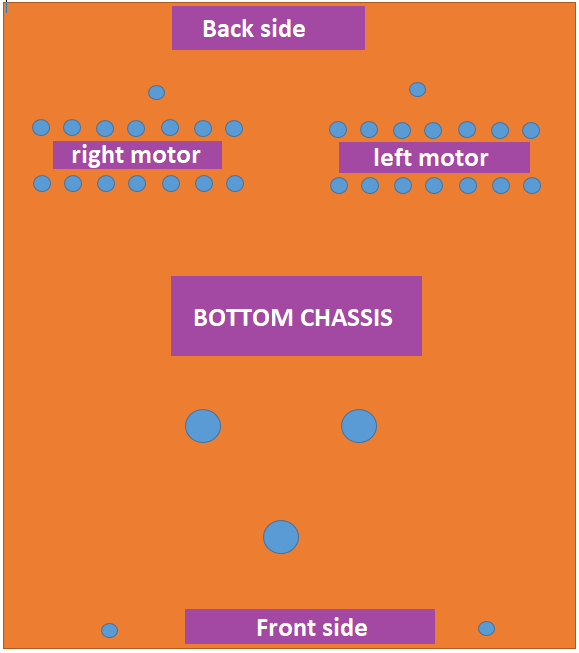


Figure 6 -- Bottom Chassis Reference

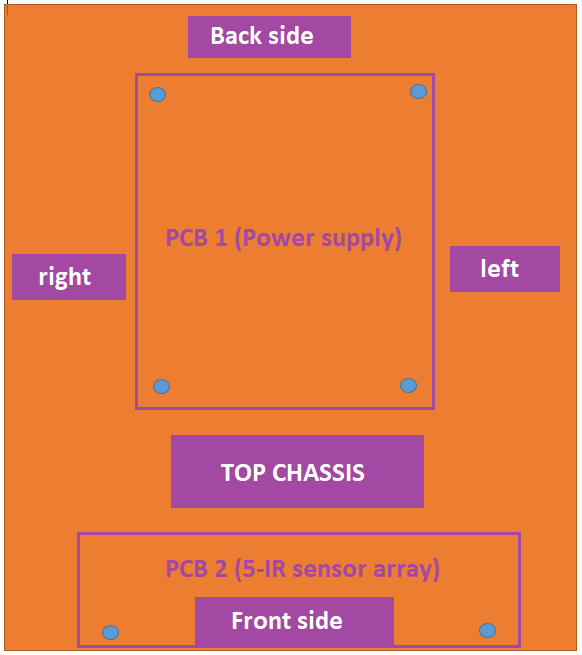


Figure 7 -- Top Chassis Reference

### 2.2.1 Attaching Wheels to Gear Motors

Let’s begin by attaching the wheels to the gear motor shafts. For this sub-section, you will need the following components:

* + DC Gear Motor (2)
  + 19 mm Wheel (pair)
  + Caster Wheel
  + 37 mm L Gear Motor Bracket (pair)
  + Universal Aluminum 6mm Mounting Hubs (pair)
  + #4-40 machine screws (16 in total – 8 for each one)
  + #4-40 machine screw nuts (8 in total – 4 for each one)
  + M3 screws (12 in total – 6 for each one)
  + M6 screws (3 in total)

Now that all the parts have been gathered, divide the components making sure each motor is to be paired with one wheel, one L-bracket, one mounting hub, 8 #4-40 machine screws, 4 #4-40 screw nuts, and 6 M3 screws.

#### 2.2.1.1 Attaching the Motor to the L-Bracket

* **STEP 1** – Align the L-Bracket’s holes to the six hole found on the gear motor. Once this is complete, simply fasten the motor onto the bracket with the use of 6 M3 screws.

**\*\*\*** Please ignore the mounting hub in the figure below as its assembly has yet to be descried.

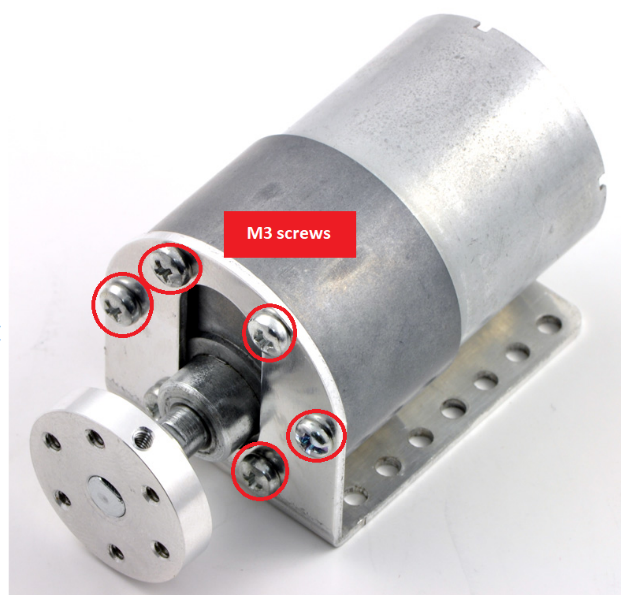


Figure 8 -- Mounting the Motor to the Bracket

#### 2.2.1.2 Attaching the wheel to the Mounting Hub

* **STEP 1** – Place the rubber cover onto the wheel

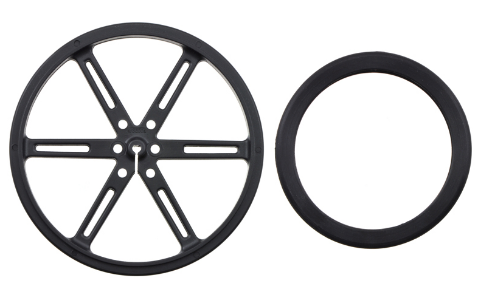


Figure 9 -- Rubber Wheel Cover

* **STEP 2** – Attach the wheel to the universal mounting hub, using four #4-40 screws as seen in the figure below.



Figure 10 -- Mounting Wheel to Hub

* **STEP 3** – With the use of the two set screws (provided with the mounting hubs) secure the wheel onto the motor’s shaft, as illustrated in both the figures below.

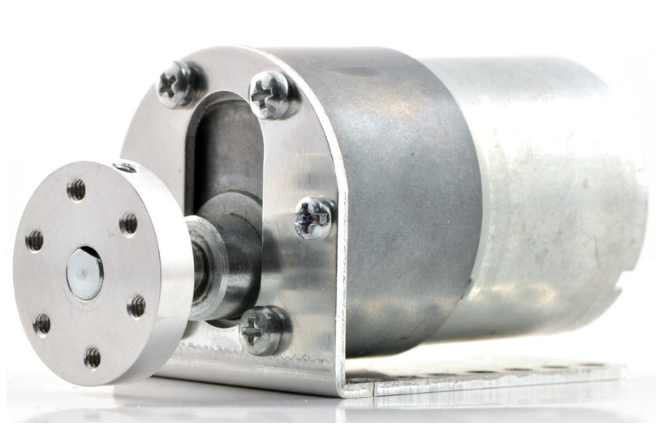
 

Figure 11 -- Securing the Hub to the Shaft

In The end, you should get something that looks like the figure below (with a metal L-Bracket of course).



Figure 12 -- Final Motor & Wheel Assembly

#### 2.2.1.3 Securing the Motors & Caster Wheel to the Chassis

* **STEP 1** – Secure the bracket to the chassis with the use of 4 #4-40 screws & nuts for each bracket (shown in red in the figure below).
* **STEP 2** – Secure the wheel caster to the chassis by fixing the 3 M6 screws & nuts to the drill holes (shown in green in the figure below)

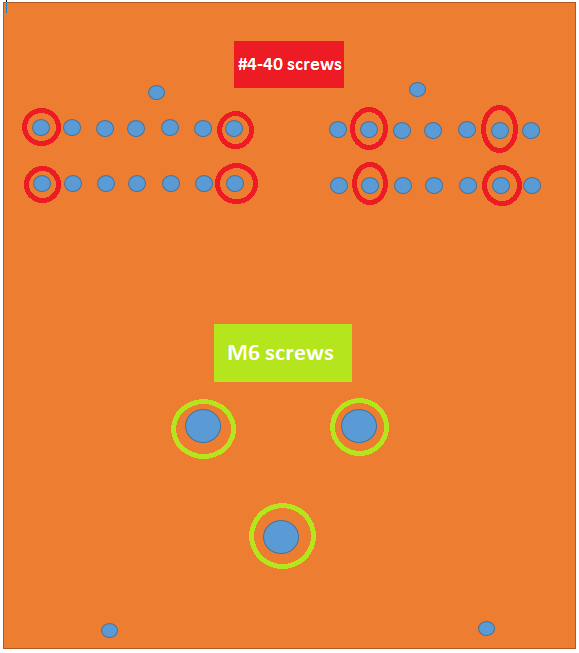


Figure 13 -- Bottom Chassis Mounting Diagram

**\*NOTE: when fixing the castor wheel to the chasis, one must make sure it is of the same height as the two other wheels to ensure all sensors will be at the same height.**

### 2.2.2 Chassis, Sensor Array & Power Supply PCB Assembly

This sub-section provides a detailed step-by-step description for mounting the PCBs onto the chassis. For this sub-section, you will need the following components:

* 8 #4-40 machine screws (between ¾ - 1 inch)
* 8 #4-40 machine screw threaded sleeves
* 2 #4-40 machine screws (¼ – ½ inch)
* 10 #4-40 screw nuts

Also, please take a look at the figures below which can be used as references in terms of each PCB’s drill hole orientations.

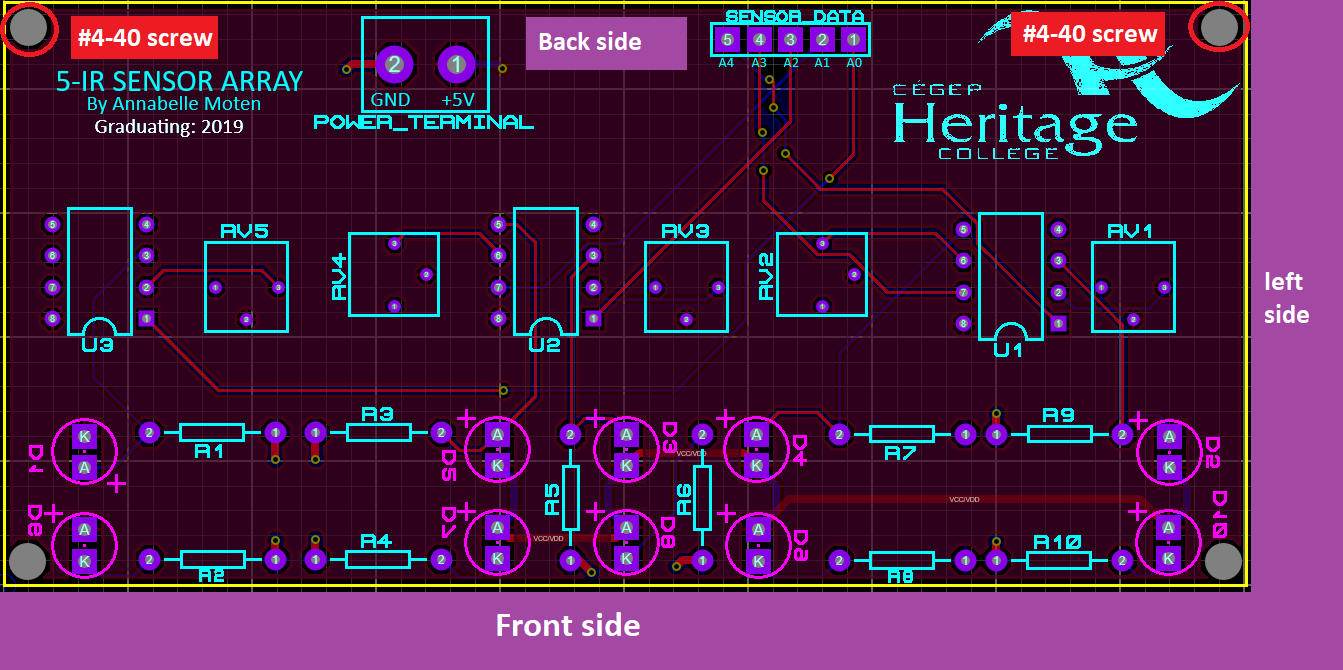


Figure 14 -- 5-IR Sensor Array PCB Orientation Diagram

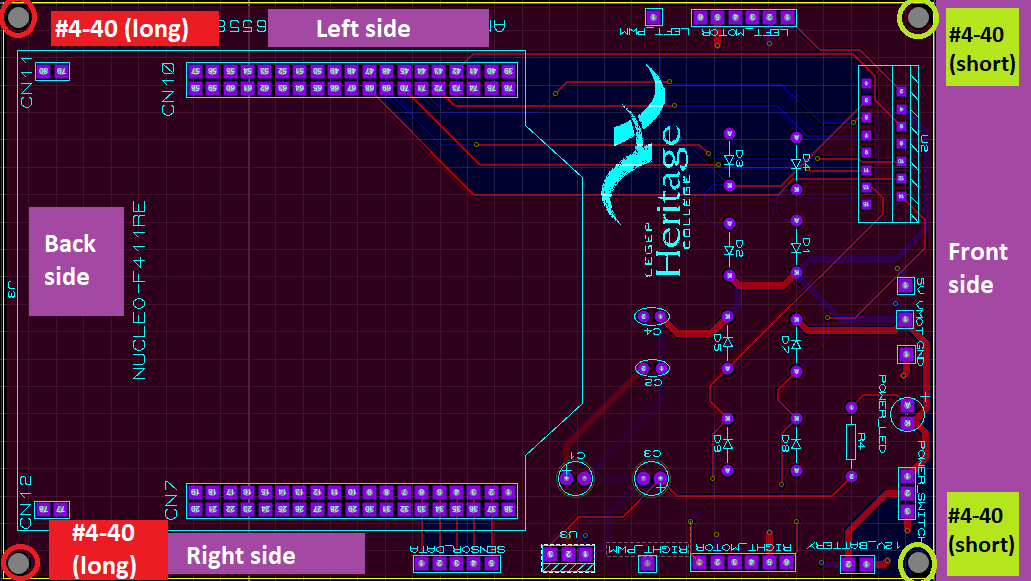


Figure 15 -- Power Supply PCB Orientation Diagram

#### 2.2.2.1 Attaching the PCBs to the Chassis

Since the whole bottom part of the chassis is almost all built, it’s a good idea to start by mounting the sensor array PCB and to finish with mounting the power supply PCB.

* **STEP 1**—Start by aligning the 2 drill holes from the sensor PCB (seen in **Figure 14**) to the ones on the bottom chassis as illustrated in the figure below

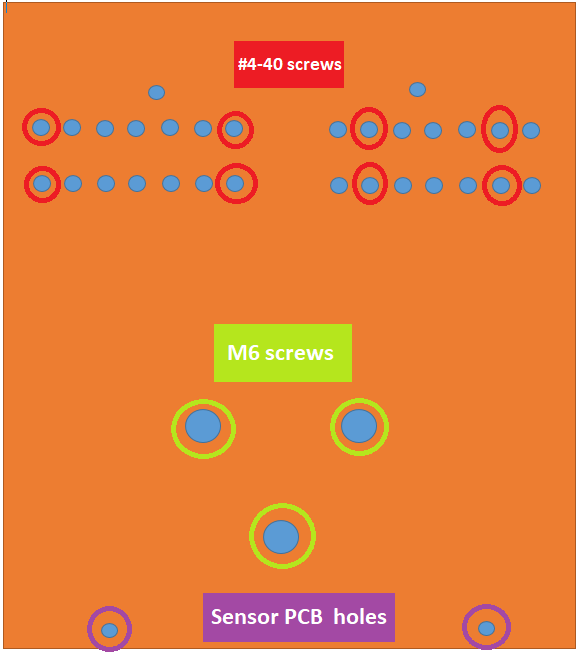


Figure 16 -- Aligning Sensor PCB & Chassis

* **STEP 2** – Now secure the boards with 2 screws (one for each hole) with its head facing down, and fasten them in place with the threaded screw sleeve.
* **STEP 3** – Move on by screwing the power supply PCB. This process is the same as in the previous step except the screw heads will be facing upright and the short screws only need to be fasten with a screw nut. The figure below provides an illustration of the top chassis, its drill holes, and their orientation.

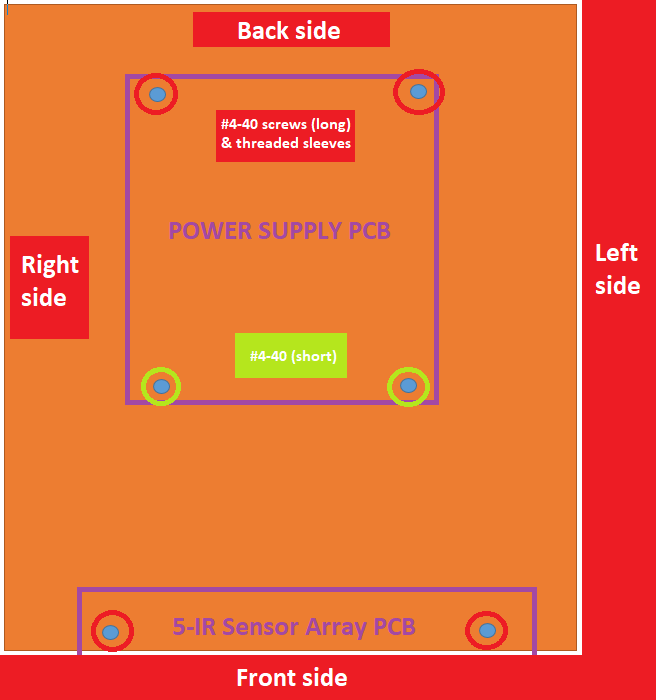


Figure 17 -- Aligning Top & Bottom Chassis

**\*Note: When fastening the long #4-40 screws, you must ensure the long screw is centered with the bottom board’s threaded screw sleeves as they will be fastened together.**

## 2.3 Component Wiring

Now that the body of the prototype is complete, it is time to wire the components in order to provide communication between each board and the microcontroller.

### 2.3.1 Component Wiring – Connecting The Sensor PCB

This section will provide a step by step description in terms of the 5-IR sensor array PCB and its connection to the main board & microcontroller. Please take a look at the table below which illustrates all the connections coming to/from this board.

Table 2 -- IR Sensor PCB Connections

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Header/Connector Label on Sensor PCB** | **Pin Number** | **Header/Connector Label on Power Supply PCB** | **Pin Number** | **Header Label on NUCLEO** | **Pin Number** |
| **POWER\_TERMINAL** | 1 | **n/a** | **n/a** | **5V** | **n/a** |
| **POWER\_TERMINAL** | 2 | **n/a** | **n/a** | **GND** | **n/a** |
| **A0** | 1 | **SENSOR\_DATA** | 33/6 | **D2** | PA\_10 |
| **A1** | 2 | **SENSOR\_DATA** | 34/5 | **D3** | PB\_3 |
| **A2** | 3 | **SENSOR\_DATA** | 35/4 | **D4** | PB\_5 |
| **A3** | 4 | **SENSOR\_DATA** | 36/3 | **D5** | PB\_4 |
| **A4** | 5 | **SENSOR\_DATA** | 37/2 | **D6** | PB\_10 |

**\*Note: When calibrating the sensors, use NUCLEO analog pins A0-A4 instead of the NUCLEO digital pins D2-D6**

### 2.3.2 Component Wiring – Connecting The Power Supply PCB

This section will provide a step by step description in terms of the Power Supply PCB and its connection to the sensor board & microcontroller. Please take a look at the table below which illustrates all the connections coming to/from this board.

Table 3 -- Voltage Regulator PCB Connections

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Header/Connector Label on Power Supply PCB** | **Pin Number** | **Motor Wire** | **Header Label on NUCLEO** | **Pin Number** |
| **12V\_BATTERY** | 1 | **n/a** | **GND** | **n/a** |
| **12V\_BATTERY** | 2 | **n/a** | **VIN** | **n/a** |
| **RIGHT\_MOTOR** | 1 | **RED** | **n/a** | **n/a** |
| **RIGHT\_MOTOR** | 2 | **BLUE** | **n/a** | **n/a** |
| **RIGHT\_MOTOR** | 3 | **YELLOW** | **n/a** | **n/a** |
| **RIGHT\_MOTOR** | 4 | **WHITE** | **n/a** | **n/a** |
| **RIGHT\_MOTOR** | 5 | **GREEN** | **n/a** | **n/a** |
| **RIGHT\_MOTOR** | 6 | **BLACK** | **n/a** | **n/a** |
| **LEFT\_MOTOR** | 1 | **RED** | **n/a** | **n/a** |
| **LEFT\_MOTOR** | 2 | **BLUE** | **n/a** | **n/a** |
| **LEFT\_MOTOR** | 3 | **YELLOW** | **n/a** | **n/a** |
| **LEFT\_MOTOR** | 4 | **WHITE** | **n/a** | **n/a** |
| **LEFT\_MOTOR** | 5 | **GREEN** | **n/a** | **n/a** |
| **LEFT\_MOTOR** | 6 | **BLACK** | **n/a** | **n/a** |

The column labelled ‘Motor Wire’ indicate which pins are connected to which motor/encoder wires based on each of their function.

Table 4 -- Motor Encoder Wire Functions

|  |  |
| --- | --- |
| **Red Wire** | Motor power (connects to one motor terminal) |
| **Black Wire** | Motor power (connects to other motor terminal) |
| **Green** | Encoder GND |
| **Blue** | Encoder Vcc (3.5 – 20V) |
| **Yellow** | Encoder A output |
| **White** | Encoder B output |

As for the external power supply, one must simply connect both of the battery packs’ terminals to the the connector block labelled ‘12V\_BATTERY’ on the voltage regulator PCB. The idea is when adding 8 alkaline batteries in series (rated at 1.5V each; 2 battery packs with 4 batteries each), their voltages add up to a total of 12V (i.e., 8 batteries \* 1.5V each = total 12V) necessary to power the motors.

Reference images of the final assembled & wired prototype are provided in **Annex B** at the end of this document.

# 3 Prototype Testing/troubleshooting

This section provides a descriptive breakdown of the testing & troubleshooting processes necessary to determining the prototype’s functionality.

The following section will be divided into 3 sub-sections, in terms of the three major components who require individual testing (i.e., the sensor array, the voltage regulator, and the motors).

## 3.1 5-IR Sensor Array Testing/Troubleshooting.

This sub-section will focus on the 5-IR sensor array PCB, notably how to test its functionality & troubleshoot the circuit if required.

### 3.1.1 testing – Output Monitoring

Before even writing the code to test this circuit, one must first understand what the expected outputs of the IR-sensors are. Here are a few important notes about these sensors’ desired behavior:

* The use of the LM358 dual OP-Amp enables the sensor data to be output as digital values (either LOGIC 0/LOW or LOGIC 1/HIGH);
* When the sensors detect a dark surface (black), the output will be HIGH (LOGIC 1);
* When the sensors detect a light surface (white), the output will be LOW (LOGIC 0);
* This enables the user to monitor the sensors output as a pattern (ex. 00100, 00110…);
* The output will be monitored through serial terminal (e.g., CoolTerm);
* The only time one will use the analog pins to read sensor data is while running calibration functions.

#### 3.1.1.1 Testing – MBed Coding and CoolTerm Setup

Now you may begin the sensor testing by running a simple mbed code which will read the sensors’ outputs and print them out as a pattern onto the CoolTerm serial application.

* **STEP 1** – First start by connecting the NUCLEO to the PC through a serial-to-USB cable and go to the official ***mbed compiler*** page and create a new project folder & file (make sure to include the mbed library).
* **STEP 2** – ***Copy & paste*** the test code provided in **Annex A** at the very end of this document. Now ***save & compile*** the code onto your NUCLEO.
* **STEP 3** – While the code is compiling, go to the ***CoolTerm*** application and select the ***options*** tab.
* **STEP 4** – Now select the proper COM port, if unsure, simply disconnect the cable from the microcontroller and verify which COM port is no longer an option. This is the desired port.
* **STEP 5** – Select the baud rate of 9600 and finally apply the previous changes by clicking the ***connect*** options.
* **STEP 6** – Once the mbed code is compiled, download it and ***send the build file*** to the NUCLEO. Once the progress bar reaches its end, you can go back to CoolTerm and monitor each sensor’s output by ***holding your hand in front of each one***.

The Figures provided below illustrate the steps described above as well as the code to test the sensor array.

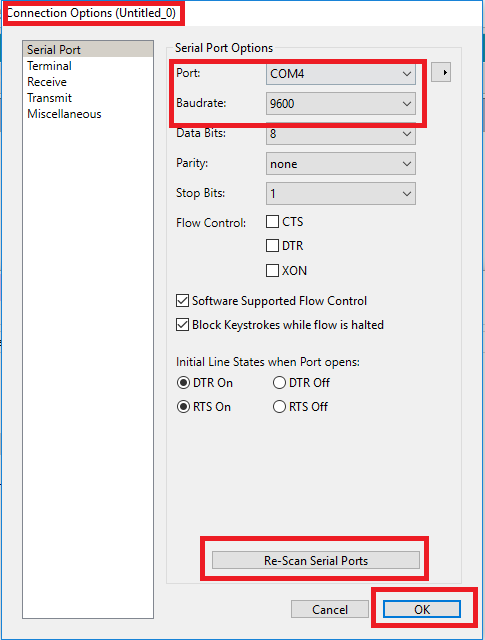
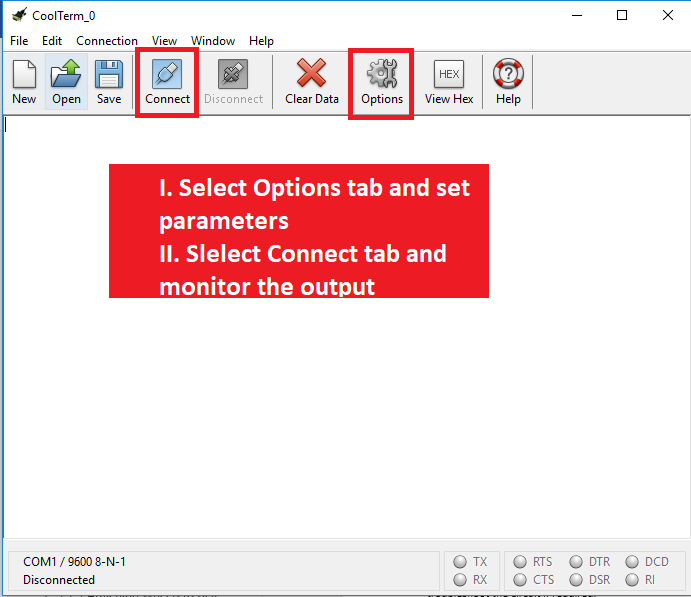


Figure 18 -- CoolTerm COM Port Setup

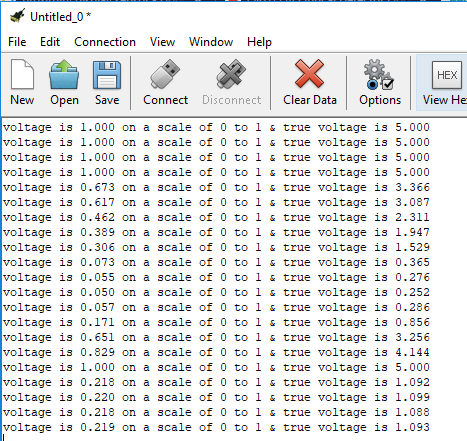


Figure 19 -- Monitored Sensor Data Output

### 3.1.2 Testing – IR-Sensor Array calibration

As for the board itself, I ran in to no problems whatsoever, thus there was no need to troubleshoot it (other than testing the current was flowing correctly & no short was present).

The only considerable initializing process for the array is calibration. Ultimately, this process consists of reading the varying analog voltages of each sensor as one turns the potentiometer. The calibration function can be described as:

* AnalogIn object for each sensor pin connected to NUCLEO pins A0-A4
* The function gathers the analog voltages (from 0-1023) of the pin from the lowest to the highest value and then uses the collected data to compute the average voltage of each sensor.
* The result of each sensors value is then displayed on the serial terminal application (CoolTerm) and these values are then used in future code, to ensure the robot reads the surface as accurately as possible.

## 3.2 Voltage regulator Circuit testing/Troubleshooting

This sub-section will focus on the power supply/voltage regulator PCB, notably how to test its functionality & troubleshoot the circuit if required.

### 3.2.1 testing – Output Monitoring

This circuit is quite simple and doesn’t require any coding, so all one needs to test the functionality of the voltage regulator is to follow the steps provided in the next sub-section. Here is a list of the components needed to test this board:

* Digital Multi-Meter
* Power Supply
* Voltage regulator PCB

Before even setting-up this circuit, one must first understand what the expected input/output of the voltage regulator are. Here are a few important notes about this circuit’s desired behavior:

* The voltage at the input should be of no more than 12V;
* The voltage at the output should be around 5V.
* If you see a current surge or spike, it’s important to shut off all power to the circuit as it’s not functioning as it should (i.e., may be a short circuit).

#### 3.2.1.1 Testing – Setting-up the Circuitry & Monitoring

You may begin the testing portion of this circuit by setting up the circuit as follows:

* **STEP 1** – Start by setting-up the power supply with a voltage of 12V and attach the power and ground cables to their respective terminals (labelled ***12V\_BATTERY*** on the PCB). **DO NOT APPLY POWER UNTIL INSTRUCTED\*\*\***
* **STEP 2** – Now set up the multi-meter: power it on, select the ***DCV option*** and connect the cables coming from the DMM to the test point labelled ***‘5V’ to monitor the output*** of the LM1117 or select the test point labelled ‘***12V’ to monitor the input*** on the PCB.
* **STEP 3** – Once the DMM is properly setup, one may now ***apply power to the circuit and monitor the input and/or output*** of the voltage regulator and verify that the input voltage is in fact equal to 12V and the output voltage is as expected (e.g., 5V).

### 3.2.2 Troubleshooting – Unexpected Output Monitoring

Seeing as the circuitry here is quite simple, there isn’t much that can go wrong, other than a short. Evidently, there were a couple errors in the design of the power supply circuit.

* If you take a look at the circuit schematic provided in the annex below, it’s clear the LED is place backwards and its current limiting resistor is placed incorrectly (as it should go on the opposite side of the diode).
* Also, the switch causes a short every time it’s in its OFF position as it is connected to ground rather than an open circuit

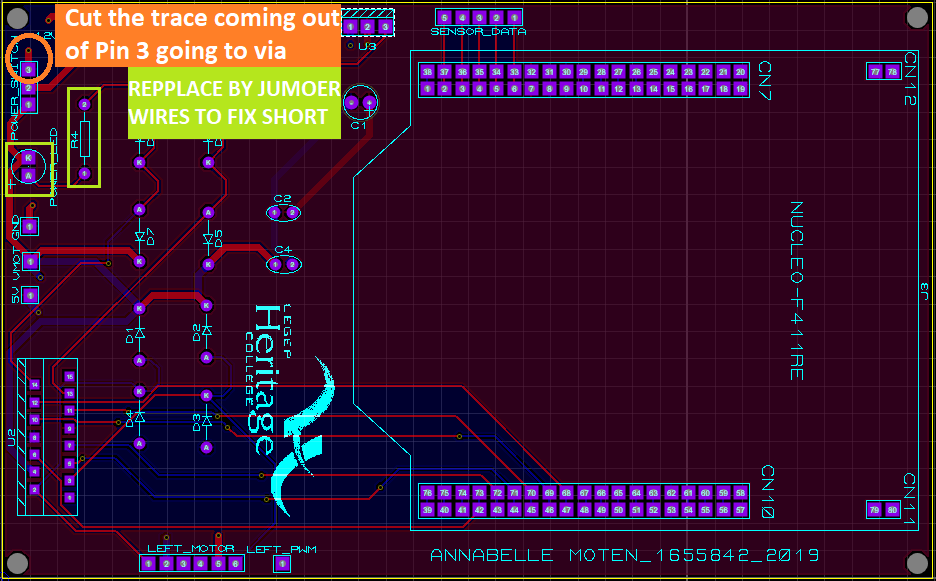


Figure 20 -- Voltage Regulator Troubleshooting

The following sub-section provides explanations for how I troubleshoot the voltage regulator circuitry.

#### 3.2.2.1 Troubleshooting – Fixing the Diode & Resistor

The incorrect placement of these components resulted in a very low voltage read at the input as well as the output of the voltage regulator. This however shouldn’t be a problem for anyone recreating the design with the help of this document, as I mitigated for this issue in the previous assembly steps. The steps to how exactly I was able to fix this issue are as follows:

* **STEP 1** – Power off and remove disconnect the external power supply & DMM
* **STEP 2** – De-solder the LED and the resistor (shown in the green rectangle in the figure above)
* **STEP 3** – In their place, solder 2 jumper wires

Ultimately, the way to fix the short was simple: replace the current limiting components with cables, thus allowing all power to flow through the circuit as required.

#### 3.2.2.2 Troubleshooting – Fixing the switch

Another short was caused by the improper design for my power switch, which you can also see in the annex at the end of this document. In brief, every time I connected my battery packs, I noticed they would often overheat in a matter of seconds, and would still be overheating when the switch was in the OFF position. This led me to conclude it was due to the fact it was connected to ground rather than an open circuit. The fix was easy and the steps are as follows:

* **STEP 1** – Disconnect all external power from the circuit.
* **STEP 2** – With the use of an X-Acto (i.e., a precision knife), simply cut the trace coming out of the switch’s pin 3 and going into the via.
* **STEP 3** – Apply power but keep the switch in the OFF position and verify the power supply to ensure it isn’t in overload mode, thus the short is no more.

# 4 User Guide

Here is where one might start putting all the parts (i.e., motors, sensors, power supply) together and monitoring how they interact with one another. To do so, one must begin by compiling the final project code onto the NUCLEO and launching the GUI applications. The code used to test the entirety of my prototype is provided in the annex (section 6.3).

At this point you may begin building your physical maze with the use of any white surface of your choice, and some black tape. Some simple maze examples have been provided in Annex A

* **STEP 1** – Start by connecting the microcontroller to the PC & compiling the code provided in Annex A (section 6.3)
* **STEP 2** – Now it’s time to launch the GUI application; for this project, the GUI application will be built using Qt code. Here is what my final application looks like:

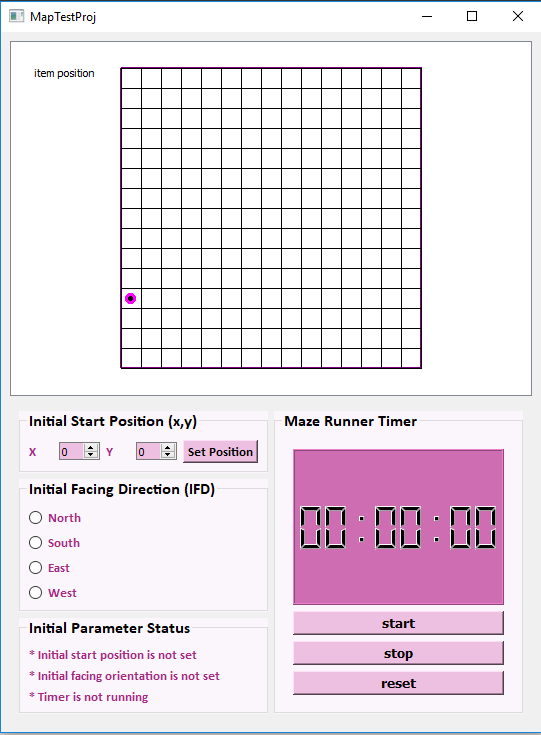


Figure 21 -- MazeBot GUI Application

## 4.1 GUI Functionalities

Since this is an automated robot project, there was no manual mode implementation. The idea here is that after the user sets up the required fields and the robot starts solving the maze, then my GUI is supposed to display the robot’s relative position. Here are the main functionalities of this application described in more detail:

* The user starts by placing the robot where he/she wishes it to begin running the maze.
* At this point, you can power up the robot or leave it turned off, as it only starts once the timer starts.
* The user now needs to enter what the initial start position of the displayed robot relative to the one of the physical prototype, the default value for these coordinates are (0,0).
* Now the User is prompted to choose the IFD of the displayed robot relative to the one of the physical prototype, there is no default value for this parameter thus the user must choose one.
* At this point the robot is ready to start and the user may start the timer, ultimately allowing the robot to begin running around the maze to try and solve it. This is when the user may monitor the path taken by the prototype as it’s being mapped on the screen.

### 4.1.1 Timer – Communicating between NUCLEO & PC

In the previous section, there was a brief mention of the timer, which in short, is the widget allowing communication between the microcontroller and the PC. The major idea is simple and as follows:

* When the user presses the start button of the timer, a signal is sent to the microcontroller (with the use of RPC commands) allowing the motors to begin rotating.
* At this point, the user has two choices;
* Either wait until the robot reaches the end of the maze, recognizes that it has reached its desired destination before coming to a complete stop. As soon as this occurs, the microcontroller emits a signal which is connected to a timer SLOT function which will stop the timer, without the user having to do it himself.
* The other choice would be if the user, for whatever reason, wanted to stop the robot from running around the maze without having solved it yet. This is done by clicking the stop button on the GUI application timer. The timer then emits a signal connected to a SLOT which simply sends a message to the microcontroller resulting in the robot coming to a full stop. This is useful if anything goes wrong, for instance if the robot stops behaving as expected.

## 4.2 Other Important considerations

It’s important to note that the robot is using the Left Hand Rule, meaning if ever the MazeBot encounters an intersection with multiple turn possibilities, then it will prioritize turning left, then straight, then back, then right.

Also, when placing the robot onto the maze surface, it is important that the 3 mid-sensors (line following) are on top of a black surface, and the outer-sensors (wall following) are on top of a white surface.

# 5 Future Considerations

This section describes a few considerations to take into account when recreating the MazeBot prototype in the future to come.

## 5.1 Adding a Manual Mode

Though the goal of this project is to have a fully functional automated maze solving robot, I do believe it might have been a good idea to implement a manual mode, in order to simplify the testing of the motors.

## 5.2 Revising the Voltage Regulator Circuit Design

Another parameter worth revising when recreating this project in the future to come would be the voltage regulator PCB in terms of its design. The problems with this board were fairly simple: short circuits, not enough test points, and inaccurate footprint for the microcontroller. Thus I would recommend revising the entirety of the schematic & layout for this board, notably the power switch, the power LED circuitry, and of course the traces between microcontroller pins, as they risk damaging the microcontroller’s internal circuitry.

## 5.3 Refine code & Implement Maze Loops

Another interesting and increasingly challenging approach to this project would be to refine the maze following code, in order for this MazeBot to be able to solve any given maze, even the one with loops. The code provided at the moment functions perfectly except when dealing with loops which is why it may be a fun challenge to try and figure out exactly how this can be implemented.

# 6 Annex A

This annex contains all the necessary files to build this project (e.g., schematics, layouts, code files)

## 6.1 Proteus Final Schematic & Layout Files

Here are the proteus project files required to recreate the MazeBot prototype

### 6.1.1 5-IR Sensor Array Proteus Files

* SENSOR\_ARRAY.pdsprj

### 6.1.2 Voltage Regulator Proteus Files

* POWER\_SUPPLY.pdsprj

## 6.2 Prototype testing Code

These are simple test code program that test the functionality of the sensors & motors individually.

### 6.2.1 5-IR Sensor Array Test Code

* sensor\_testing.cpp

### 6.2.2 Gear Motor test Code

* motorTest.cpp

## 6.3 Final Microcontroller Code

* FinalProjMazeCode.cpp

## 6.4 Final GUI code

* MazeBotProjDir.zip

## 6.5 Physical Maze Examples

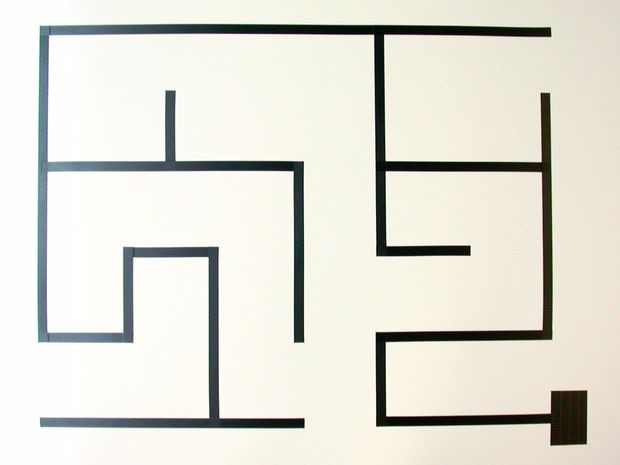


Figure 22 -- Maze Example A

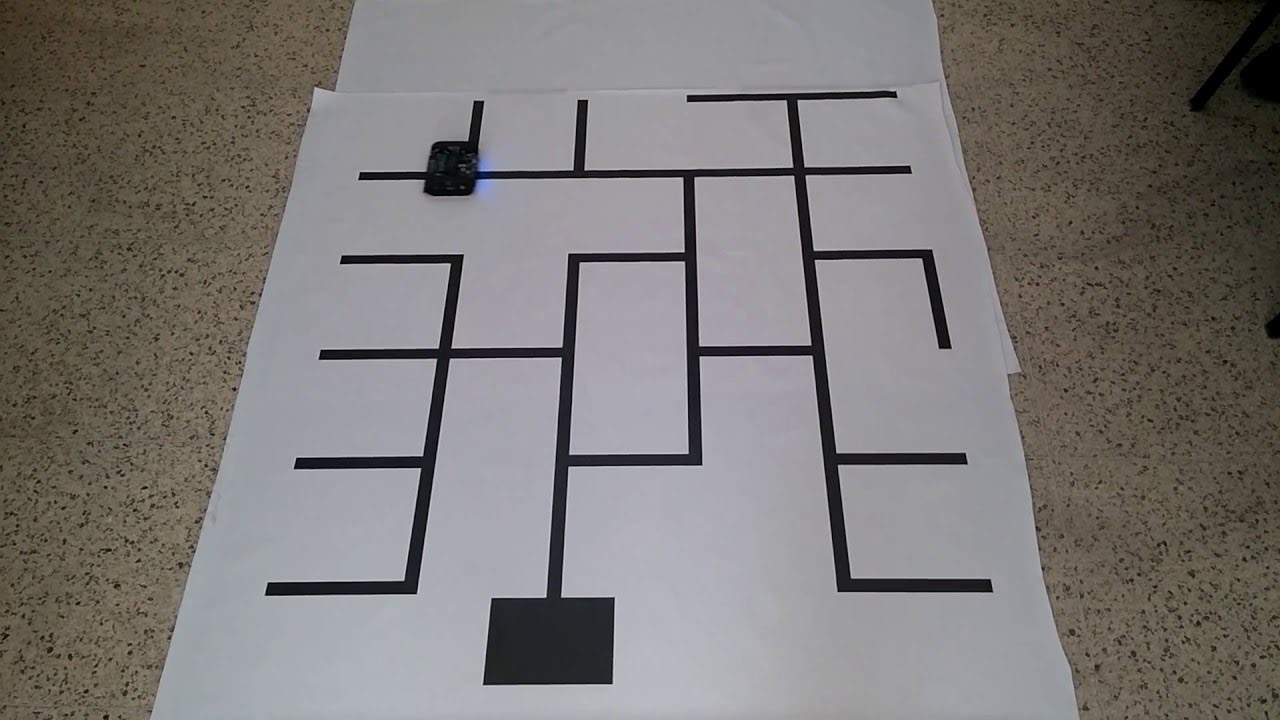


Figure 23 -- Maze Example B

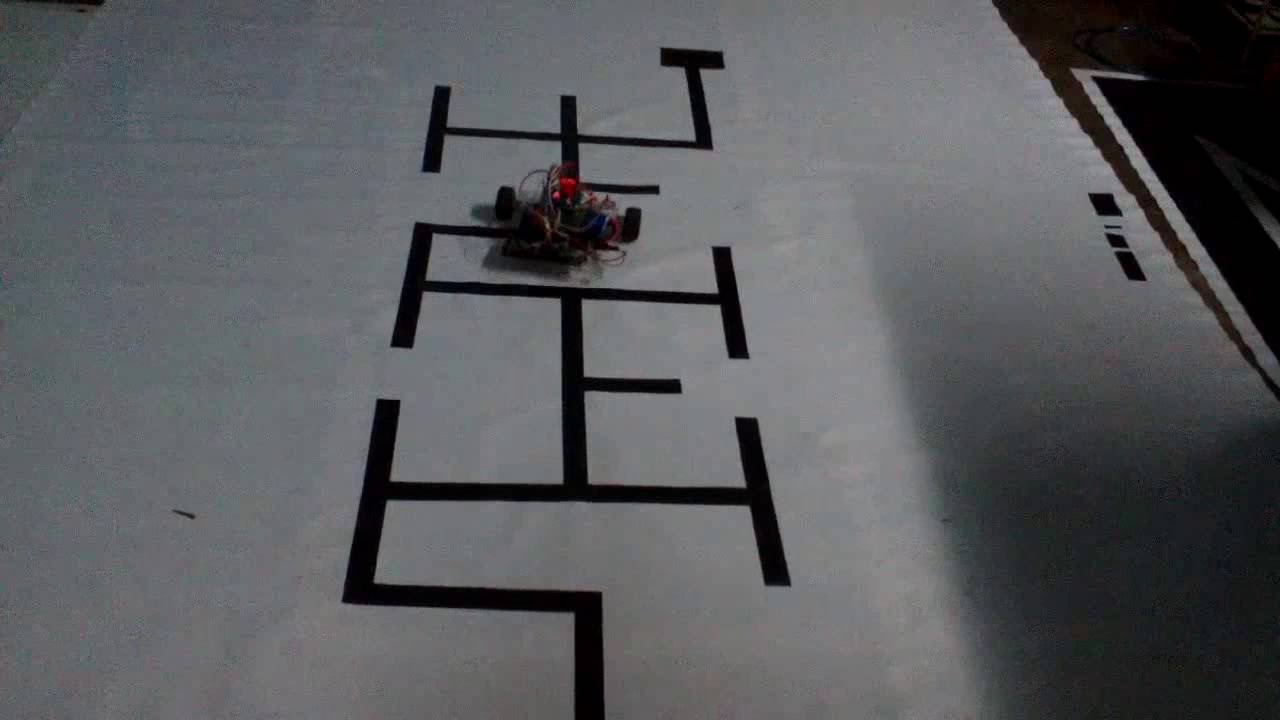


Figure 24 -- Maze Example C

# 7 Annex B

This annex contains images of the fully assembled prototype that one may use as references when recreating this project.

## 7.1 Assembled Prototype – Sensor Array View

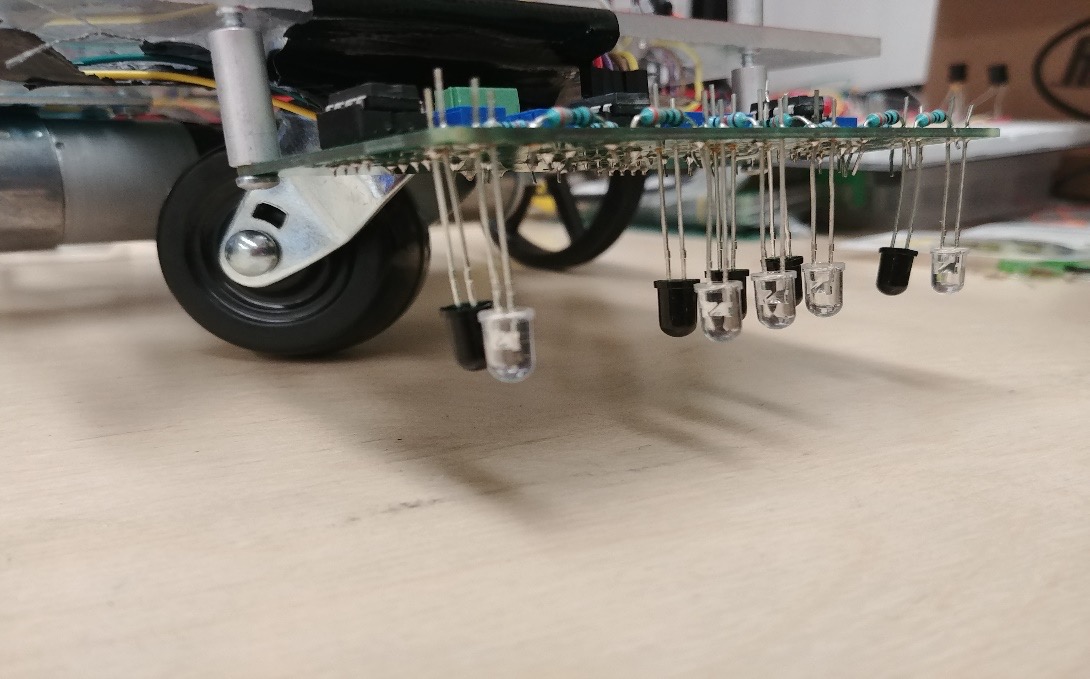


Figure 25 -- Sensor Array View

## 7.2 Assembled Prototype – Motors & Wheels

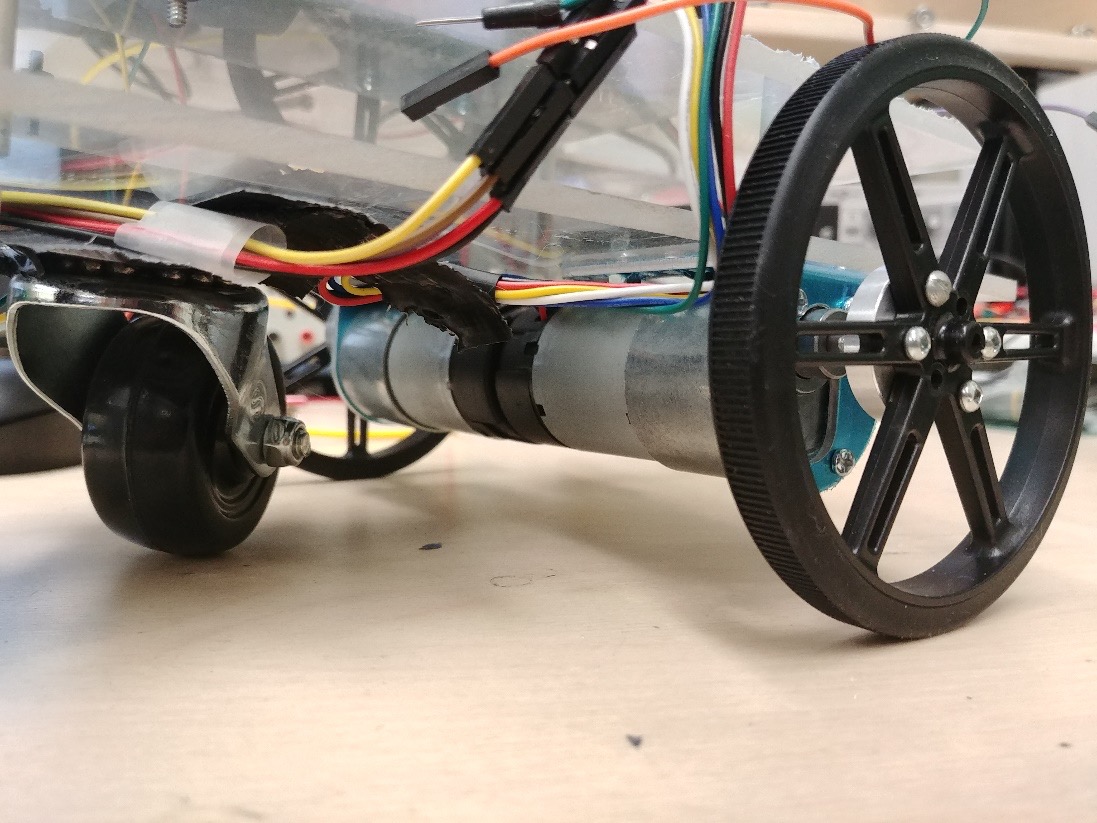


Figure 26 -- Motor & Wheels View

## 7.3 Assembled Prototype – Front Body View

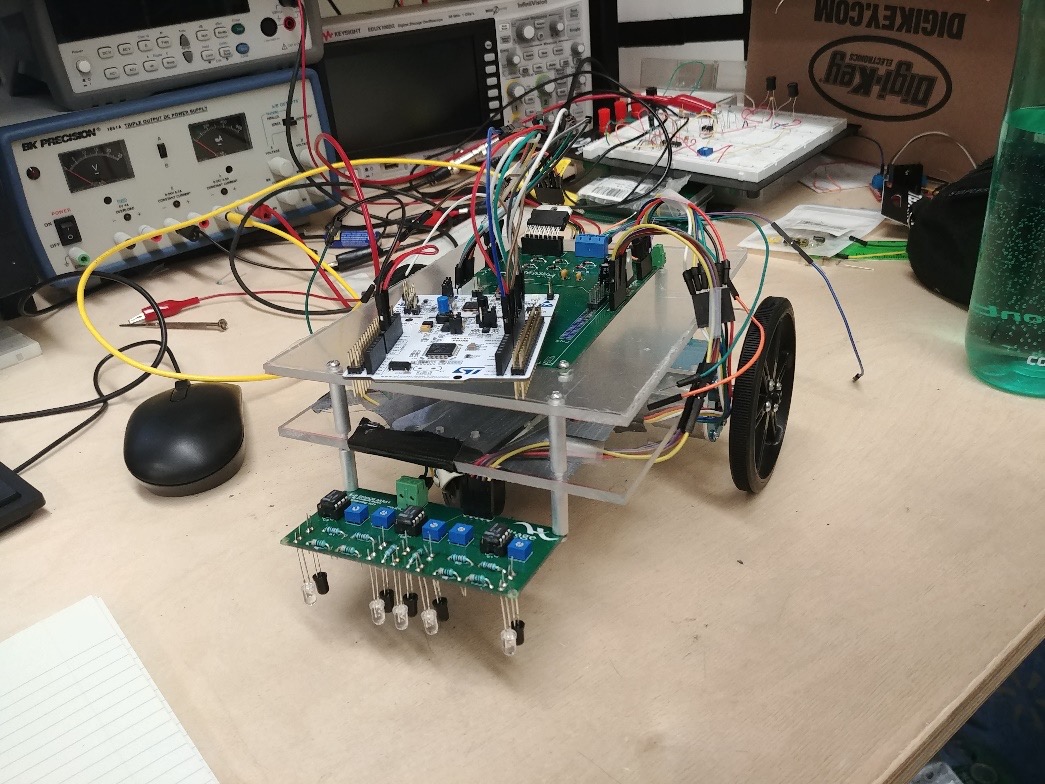


Figure 27 -- Front Body View

## 7.4 Assembled Prototype – side Body View

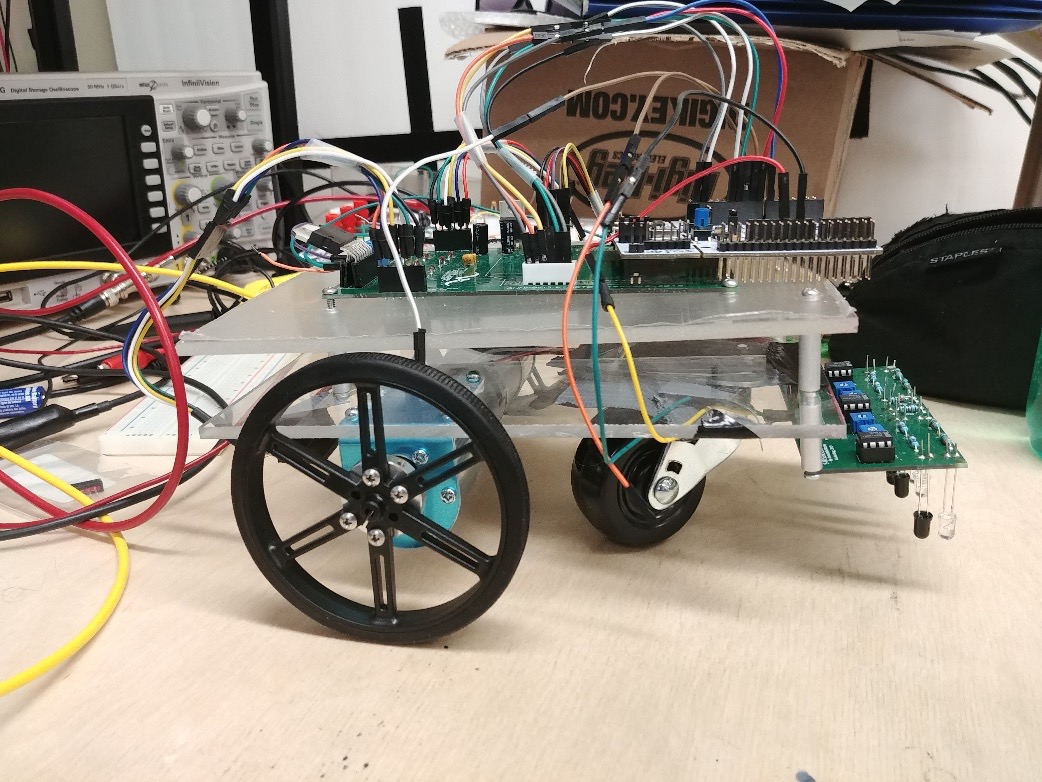


Figure 28 -- Side Body View