Line Following Robot
using Arduino

Group Members: *Roll numbers:*

AZEEMA AZHAR **180916**

MUHAMMAD FAIQ NASIR **201075**

MUHAMMAD ABUBAKAR **201090**

MUHAMMAD WALEED TARIQ **201111**

BE MECHATRONICS (Session 2020-2024)

Project Supervisor

Engr. Iraj Kainat

Lab Instructor

DEPARTMENT OF MECHATRONICS ENGINEERING

FACULTY OF ENGINEERING

AIR UNIVERSITY, ISLAMABAD

Team Member Name	Role	Word count that each has written in assigned color code in report	Signature
Faiq	Team Lead	1000	
Waleed	Project manager	1000	
Azeema	Technical	1000	
Abubakar	Integrator and tester	3000	

ABSTRACT

Robots can be fixed robots or mobile robots. Mobile Robots are robots with a mobile base which makes the robot move freely in the environment. One of the advanced mobile robots is the Line Follower Robot. It is basically a robot which follows a particular path or trajectory with the help of sensors that are implemented to detect the path. The path can be a black line on the white floor (visible) or wise versa or even a magnetic field (invisible). This Robot can be integrated with more and more sensors to get full information of the movement, Voltage, Current, Speed, Obstacles and store them in SD Card. Its applications start from basic domestic uses to industrial uses, etc. The use of line following robotic vehicle is transport the materials from one place to another place in the industries. This robot movement completely depends on the pre-determined track.

Table of Contents

Chapter 1- Preliminaries

Proposal

The purpose of this project is to design a line following robot that can cover any given path in minimum possible time while displaying full details of the voltage, current, speed, Obstacle, transmit all this data to a smart phone and then display them all on the smart phone via a serial monitor app. Design an Embedded System for a line following robot that can go on white line on black surface (or it may be reverse of that) from Point A to Point B, upon reaching Point B which is a T junction with all black line and stop, which it uses IR sensors which detects the line and H bridge which controls the working of the wheels, the Voltage sensor, current sensor, ultrasonic sensor and speed sensor show the data on the smart phone.

Initial feasibility

First of all we designed a schematic diagram of the project with all the components that we intended to use. We simulated our design with a pseudo code. Upon its working we designed a double copper PCB for our project.

Comparison table

Single IR sensor

- Detection Range: 1mm to 8 mm
- Focus Range: 2.5 mm
- Driving Ability: > 15 mA
- Working Voltage: 3.3 V to 5 V
- Output: digital switch outputs (0 and 1)
- PCB Size: 3.2 x 1.4 cm
- TCRT5000 IR reflex sensors
- Stable LM393 chipset in comparator

IR line tracking module

- 5-way reflective optical sensors mounted in line (TCRT5000 or equivalent)
- On-board hex inverter provide clean digital output
- Sensitive to dark color and infrared
- Operating Voltage : 5 V (recommended)
- Comes with M3 flexible mounting slot

ACS712

- Measures both AC and DC current
- Available as 5A, 20A and 30A module
- Provides isolation from the load
- Easy to integrate with MCU, since it outputs analog voltage
- Scale Factor

GY-471

- Model: GY-471
- Quantity: 1 piece
- Form Color: Purple
- Material: Copper clad laminate + components
- Application: Current module, current detection, current sensor, current measurement
- Working Voltage: 3~5V

Technical Standards (for example Speed of a standard automatic sliding door)

Speed of the motors was kept 255 using the PWM of the microcontroller. We are using the motors with speed under 255.using enable pins of the motor driver. These values maybe decreased if it give too high speed.

Team Roles & Work Breakdown Structure

Team Lead: Muhmmad Faiq, made the pseudo code for the primary schematics and designed the algorithm for the motion of the robot with PID and also finalized the program of the working of the project.

Project manager: Muhammad Waleed interfaced some of the sensors, searched and bought most of the components for the project, did the hardware implementation and designed the state diagram.

Technical: Azeema Azhar tested the working of the robot and helped in correcting the error and malfunctions faced in the testing.

Tester and integrator: Muhammad Abubakar designed the PCB layout for the project, integrated the hardware components and tested the working of the code.

Gantt Chart

Table 1: Gantt Chart of Project

Task Name	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7
Planning							
Research							
Design							
Implementation							
Troubleshooting							

Estimated budgets

As some of the expensive components were already present in our inventory so we estimated our project expenses to be around 5000 Rupees. We remained well under our budget with some mistakes we made with our PCB.

Chapter 2-Project Conception

Introduction

Robots are basically complete automatic machines. It starts working when sense something, decides to work according to the conditions and stop by its own due to other condition or senses. So, we can say robot are the replica of human being, as they work on phenomena as human being do. Robots are always made for the ease of human being. They can be made of many ways, but this is decided by the work, a robot will do. One of the most preferable methods is by using electronic control ways.

The line follower robot is one of the advance mobile robots, as we make its base moveable, and it can move from one place to another. Line follower robot follows a particular path or trajectory and decides its choices to interacts with the surface. Usually, the path is a black line on the white floor (or the opposite of it), but in other cases it may be a magnetic field which can't be seen by naked eyes. They can be mostly use in industry and in domestic chores as they can carry the parcels or materials from one place to another place.

List of features and operational specification of your project

We have used different components to optimize our robots. Their details are listed below.

Microcontroller

A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system. Microcontroller includes a processor, memory, and input/output (I/O) ports on a single chip. Some of the practical applications of microcontrollers are found in vehicles, robots, office machines, medical devices, mobile radio transceivers, vending machines, and home appliances, among other devices.

A microcontroller is embedded inside of a system to control a singular function in a device. It does this by interpreting data it receives from its I/O ports using its central processor. The temporary information that the microcontroller receives is stored in its data memory, where the processor accesses it and uses instructions and apply the incoming data. It then uses its I/O ports to communicate and enact the appropriate action.

Figure 1: Arduino Uno

DC motor

D.C. motors are built to operate in the steady state in a speed range close to their no-load speed this speed is generally too high for applications like the robot we are making so that is why we merge the DC motor with gearbox to reduce its speed. The basic working principle of the DC motor is that whenever a current carrying conductor places in the magnetic field, it experiences a mechanical force. The principle remains the same but using them both as a single component is very helpful in performing variety of functions.

Figure 2: DC Geared Motor

1. Rated Voltage: 3-6V DC.
2. Unloaded speed: 120 RPM.
3. Load current: 190 mA (250 mA MAX).
4. Maximum torque: 800 g. Cm min.
5. Chassis Specification:
6. Dimensions: 7.72 in x 4.13 in x 0.12 in (19.6 cm x 10.5 cm x 0.3 cm)

Motor driver

We have used L298N motor driver module in our robot. It is a high-power motor driver module for driving DC Motors. It consists of an L298 Motor Driver IC, 78M05 Voltage Regulator, resistors, capacitor, Power LED, 5V jumper in an integrated circuit. It can control up to 4 DC motors, or 2 DC motors with directional and speed control.

7805 Voltage regulators will be enabled only when the jumper is placed. When the power supply is less than or equal to 12V, then the internal circuitry will be powered by the voltage regulator and the 5V pin can be used as an output pin to power the microcontroller. The jumper should not be placed when the power supply is greater than 12V and separate 5V should be given through 5V terminal to power the internal circuitry.

ENA & ENB pins are speed control pins for Motor A and Motor B while IN1& IN2 and IN3 & IN4 are direction control pins for Motor A and Motor B.

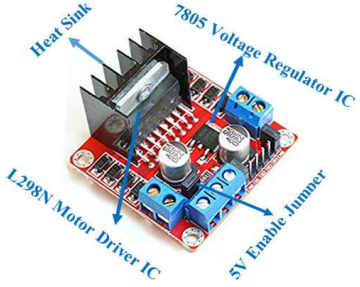


Figure 3: L298N Motor Driver

Ultrasonic sensor:

We have used the ultrasonic sensor to detect any obstacle in the circuit. For this purpose, the HC-SR04 Sensor was selected. Our robot will stop movement when it detects any obstacle. It is a 4-pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications were measuring distance or sensing objects are required.

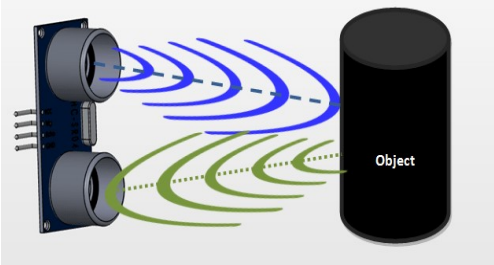
Figure 4: Ultrasonic Sensor

Working:

The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The sensor works with the simple high school formula that

Distance = Speed × Time

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module as shown in the picture below,



Now, to calculate the distance using the above formulae, we should know the Speed and time. Since we are using the Ultrasonic wave we know the universal speed of US wave at room conditions which is 330m/s. The circuitry inbuilt on the module will calculate the time taken for the US wave to come back and turns on the echo pin high for that same particular amount of time, this way we can also know the time taken. Now simply calculate the distance using a microcontroller or microprocessor.

IR sensor:

An infrared (IR) sensor is an electronic device that measures and detects infrared radiation in its surrounding environment. Infrared radiation was accidentally discovered by an astronomer named William Herchel in 1800. While measuring the temperature of each color of light (separated by a prism), he noticed that the temperature just beyond the red light was highest. IR is invisible to the human eye, as its wavelength is longer than that of visible light (though it is still on the same electromagnetic spectrum). Anything that emits heat (everything that has a temperature above around five degrees Kelvin) gives off infrared radiation.

There are two types of infrared sensors: active and passive. Active infrared sensors both emit and detect infrared radiation. Active IR sensors have two parts: a light emitting diode (LED) and a receiver. When an object comes close to the sensor, the infrared light from the LED reflects off of the object and is detected by the receiver. Active IR sensors act as proximity sensors, and they are commonly used in obstacle detection systems (such as in robots).

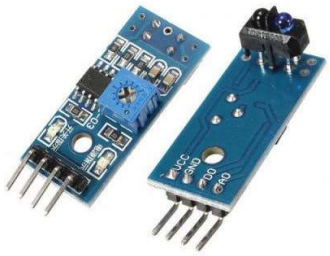


Figure 5: TRCT5000 IR Sensor

Bluetooth module:

Wireless communication is swiftly replacing the wired connection when it comes to electronics and communication. Designed to replace cable connections HC-05 uses serial communication to communicate with the electronics. Usually, it is used to connect small devices like mobile phones using a short-range wireless connection to exchange files. It uses the 2.45GHz frequency band. The transfer rate of the data can vary up to 1Mbps and is in range of 10 meters.

The HC-05 module can be operated within 4-6V of power supply. It supports baud rate of 9600, 19200, 38400, 57600, etc. Most importantly it can be operated in Master-Slave mode which means it will neither send or receive data from external sources.

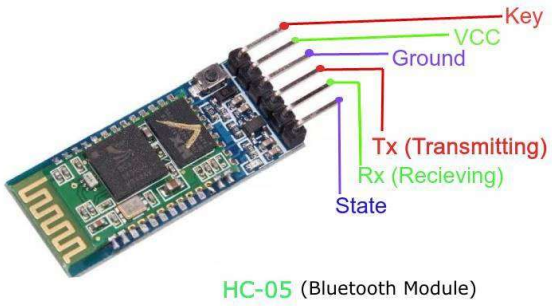


Figure 6: HC-05 Blurooth Module

The HC-05 Bluetooth Module can be used in two modes of operation: Command Mode and Data Mode.

- Command Mode

In Command Mode, you can communicate with the Bluetooth module through AT Commands for configuring various settings and parameters of the Module like get the firmware information, changing Baud Rate, changing module name, it can be used to set it as master or slave.
A point about HC-05 Module is that it can be configured as Master or Slave in a communication pair. In order to select either of the modes, you need to activate the Command Mode and sent appropriate AT Commands.

- Data Mode

Coming to the Data Mode, in this mode, the module is used for communicating with other Bluetooth device i.e. data transfer happens in this mode.

Relay

A relay is an electrically operated switch. It consists of a set of input terminals for a single or multiple control signals, and a set of operating contact terminals. The switch may have any number of contacts in multiple contact forms, such as make contacts, break contacts, or combinations thereof.

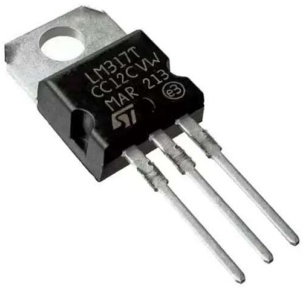
Relays are used where it is necessary to control a circuit by an independent low-power signal, or where several circuits must be controlled by one signal. Relays were first used in long-distance telegraph circuits as signal repeaters: they refresh the signal coming in from one circuit by transmitting it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

Figure 7: Relay SPDT 6V

Voltage regulator:

A voltage regulator is a system designed to automatically maintain a constant voltage. A voltage regulator may use a simple feed-forward design or may include negative feedback. It may use an electromechanical mechanism, or electronic components. Depending on the design, it may be used to regulate one or more AC or DC voltages.

Electronic voltage regulators are found in devices such as computer power supplies where they stabilize the DC voltages used by the processor and other elements. In automobile alternators and central power station generator plants, voltage regulators control the output of the plant. In an electric power distribution system, voltage regulators may be installed at a substation or along distribution lines so that all customers receive steady voltage independent of how much power is drawn from the line.

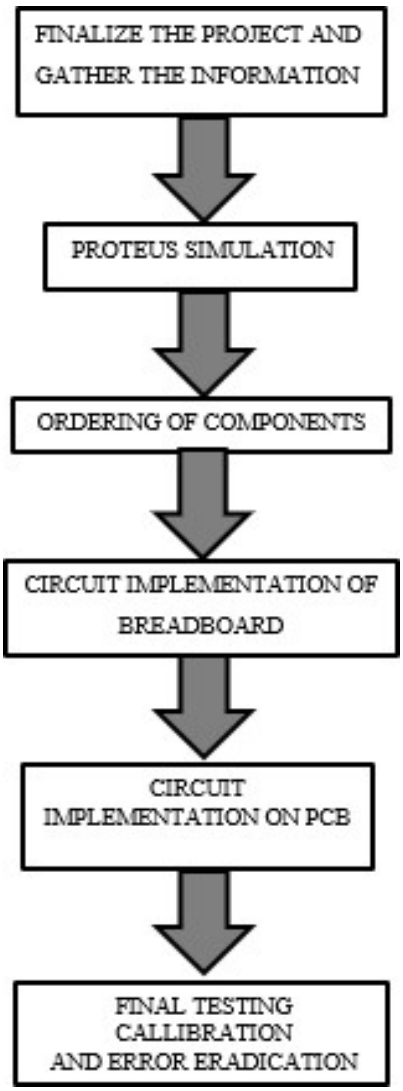


Project Development Process

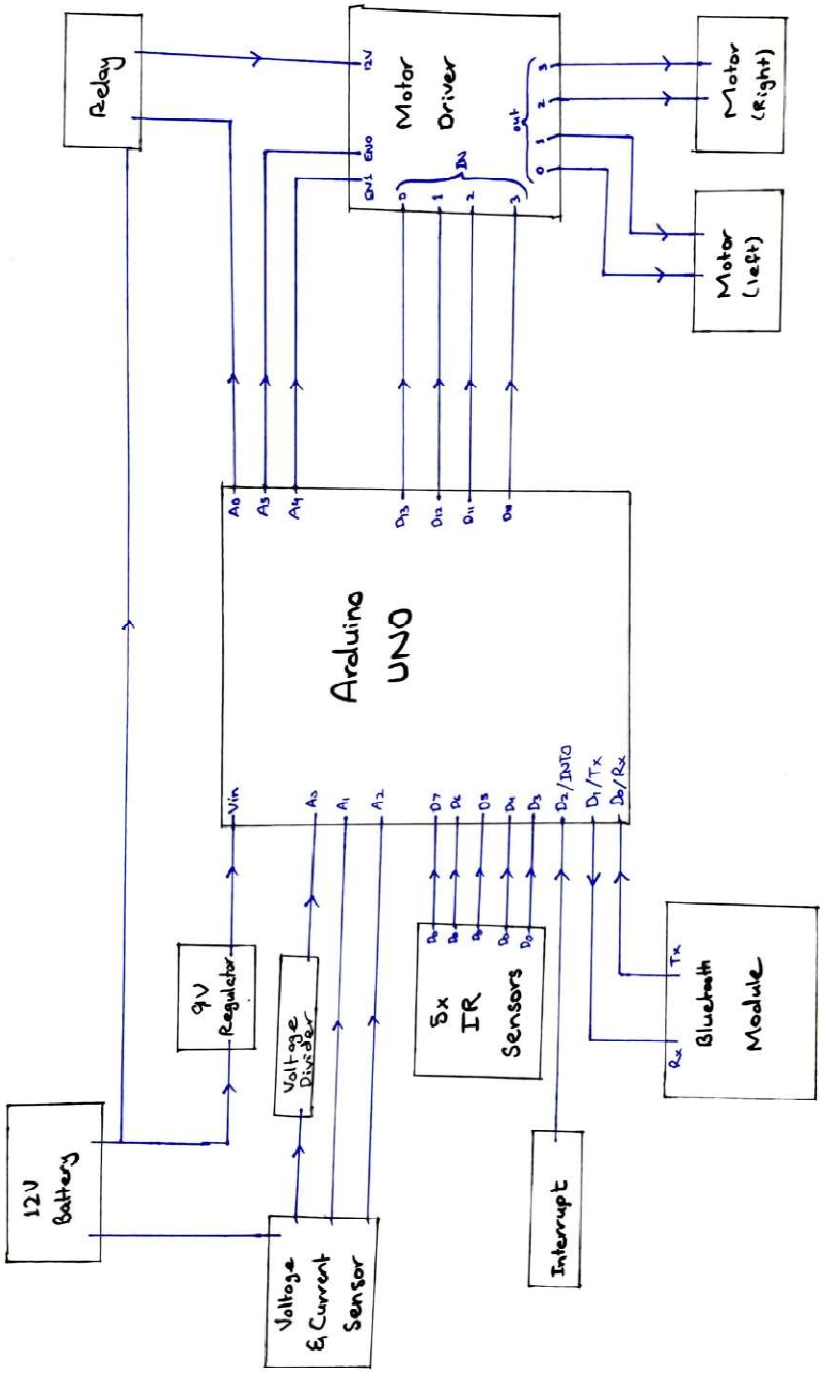
In this portion we are going to discuss that on what method we are going to design and assemble our project. Secondly, we will also highlight that what software were used which helped us in the designing phase of our project. We will also discuss what parts or components are used in the hardware designing of our project and the most important thing is the approach which will play a beneficial role in designing phase of our project.

Now before going to the designing method of the robot, we will discuss the importance of ultra-sonic sensors and other types of sensors used in the project.

This section discussed the compositions of the hardware components and software implementations used for designing and constructing the project.



Basic Block Diagram



Chapter 3-Product Design

System Consideration for the Design

As our project is based on the line following logic so the sensors used in our project were based on building up a logic for line following so we used truth table to describe all possible situations. These cases will be the reason of our robot to follow the desire path.

A truth table is a handy little logical table. The notation may vary depending on what discipline you're working in, but the basic concepts are the same. Then the results obtained from truth table were simplified were used to program the Microcontroller.

Now the design may contain 2 Arduino UNO or 1 ATMEGA2560. Choosing anyone can affect the design in a lot of ways in PCB specifically.

Other sensor and modules like motor driver, voltage, current, speed, ultrasonic sensor or the voltage booster will change the design accordingly.

Specifications of motor were considered so to check whether the motor can bear the load of our project and it is working requirements like Voltage and current needed

Criteria for Component Selection

Microcontroller:

A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system. Microcontroller includes a processor, memory, and input/output (I/O) ports on a single chip. Some of the practical applications of microcontrollers are found in vehicles, robots, office machines, medical devices, mobile radio transceivers, vending machines, and home appliances, among other devices.

A microcontroller is embedded inside of a system to control a singular function in a device. It does this by interpreting data it receives from its I/O ports using its central processor. The temporary information that the microcontroller receives is stored in its data memory, where the processor accesses it and uses instructions and apply the incoming data. It then uses its I/O ports to communicate and enact the appropriate action.

IR sensor:

As we are asked to make the line follower robot with using any microcontroller, so the best selection of the sensor in this regard is IR sensors for line detection. In our case of arena, the sensor will detect black line as path while white area as obstacle, so we must configure our circuit accordingly. The simplest case will be if we use 2 IR sensor and can work, even 1 IR sensor works if we have only black or white surface area. But as we are advised to used minimum 5 or 6 sensors or more sensors, that will increase the complexity of the circuit and robot, but it will improve the accuracy of our robot. That is why we will 5 sensors in our robot.

Ultrasonic Sensor:

We have used the ultrasonic sensor to detect any obstacle in the circuit. For this purpose, the HC-SR04 Sensor was selected. Our robot will stop movement when it detects any obstacle. It is a 4-pin

module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications were measuring distance or sensing objects are required. It can Operating voltage on +5V. Its theoretical Measuring Distance is from 2cm to 450cm. Its accuracy is 3mm. its measuring angle covered is <15°. Its operating Current is <15mA.

Voltage and current sensor:

We want to check the voltage of the circuit at any point then we will use this sensor. Voltage sensor basically measure the voltage at any certain. Voltage sensors can determine the AC voltage or DC voltage level. The current sensor is used in our robot to check if proper amount of current is provided to the circuit for robot working or not.

For this purpose we could you two separate sensors for each current and voltage measurement but to optimize the minimum use of space on PCB we opted for **GY-471** which can tells us value of the current and also voltage with just some slight adjustments.

Motor driver:

We have used L298N motor driver module in our robot. It is a high-power motor driver module for driving DC Motors. It consists of an L298 Motor Driver IC, 78M05 Voltage Regulator, resistors, capacitor, Power LED, 5V jumper in an integrated circuit. It can control up to 4 DC motors, or 2 DC motors with directional and speed control.

7805 Voltage regulator will be enabled only when the jumper is placed. When the power supply is less than or equal to 12V, then the internal circuitry will be powered by the voltage regulator and the 5V pin can be used as an output pin to power the microcontroller. The jumper should not be placed when the power supply is greater than 12V and separate 5V should be given through 5V terminal to power the internal circuitry.

DC motor:

D motors are most easy to control as compared to AC, and they can do the required work. One dc motor requires only 2 signals for its operation. If we want to change its direction just reverse the polarity of the power supply across it, we can vary speed by varying the voltage across motor.

Chapter 4- Mechanical Design

Mechanical design play very important rule in any project of engineering. Our line following robot is a sensitive engineering project robot that can be affected by a lame or bad model of mechanical design. The mechanical design is responsible for the sustainability, weightlifting, long lasting life of the robot. So, designing the robot in a most sufficient way so that external condition or internal condition on that material used and that design are minimum.

Mechanism selection

Mechanism, we adopted for our line follower robot is connecting the wheels right on the PCB and not use any kind of base. The PCB itself is main body of the robot. This allowed our robot to remain as compact and small as possible because the size affects the speed and overall battery performance of the robot. The lower side of the PCB is connected to 2 Motors on each back side (Left and Right) by a rectangular shape connecter These Motors Then connected to the Tires which are responsible for moment (Front, Left or right).

Now the front side of the line follower robot PCB is directly connected to a freely moving wheel (Figure 8) in any direction according to the moment of backward tires. Over the PCB, there will be the Circuit that is microcontroller, voltage, current sensor, Bluetooth module, relay and regulator etc. Bults and Nuts (Figure 9) are used in different places for tightness. The 5 IR Sensors Figure 10) are mounted on the front end of the PCB according to the truth table condition.

Platform Design

So, there are so many designs of which an individual can think of for a line follower robot. Some of them may be available in the market in easy cast or we can design some by ourselves. The design maybe in rectangular form, maybe a triangular shape, maybe pentagonal shape or maybe a circular shape. So, it's just depending upon our feasibility. That is because the design wouldn't affect our project, this is because our project contains very simple configuration where everything is mounted right onto the PCB. The design is enough as far as it has the space to accommodate these components. The design was made so it could easily adjust the motors and free wheels on underside of the board. Now the motors and Tires are below the, so they got their own space to work in the project. So, the Platform Design was thus Optimized.

Material Selection and choices

Material selection is very important thing to do while doing an engineering project, especially when the project is of high level or there are a lot of loads etc. over it. The commonly used materials used in different projects are aluminum, Glass, Plastic, Iron, Steel, Brass etc. different material have different properties according to which they are selected.

Now in our case of project, we have chosen the PCB of plastic, this is because it has very less weight as compared to other, and our project can be easily mounted over it without breakage etc.

The connecters are also of plastic (Glass), as in our case, it can work easily without any problem. These materials are chosen because they were easily accessible in the market, and they were of less cost as compared to other and are feasible with loads in our project.

Actuators with speciation and datasheet

An actuator is a part of machine/device which is responsible to convert any type of energy into mechanical Force.

In our Project, the only actuator we have used is Motor. The quantity is 2, one at the left side and other at the right side of the base. These motors work with the help of Motor Driver which in our case used is L298.

The Positive of motor should be connected to the Positive output of motor driver and vice versa, so it will work in proper direction (Let suppose clockwise). If you may want to change its direction (let suppose Anti Clockwise) then just invert the connections, i.e., Positive of motor to the negative of motor driver output.

https://www.google.com/aclk?sa=l&ai=DChcSEwiAx5CwjvD0AhWCjFEKHSMbB-sYABABGgJ3cw&ae=2&sig=AOD64_0snlp29GpSSwZEY9S1GC5BFjNS4w&q&adurl&ved=2ahUKEwiutoiwjvD0AhUGlRQKHWCuB2IQ0Qx6BAgGEAE

Chapter 5- Electronics Design and Sensor Selections

Component Selection

Component selection was done after we read the report that listed all the requirements of the user, The user wanted a line following obstacle avoiding robot that could display its information on a screen and show the currents and voltages.

To tackle this problem keeping in mind the low budget and wanted to build a compact sized robot we chose to build the robot using a Arduino uno with the following components:

1. Bluetooth sensor: To display all the information on the phone screen as an LCD screen takes up more pins on the controller which we need to fulfil all the other requirements.

2. MAX 471: a combined current and voltage sensor to reduce the number of sensors on the board to make it easier to build a small compact sized robot.

3. Ultrasonic sensor: Obstacle avoidance and detection can be best done using an ultrasonic sensor.

4. IR sensors: For line detection

5. Motor Driver L298N: to control motors and using the microcontroller

6. 2 DC motors

7. PCB board

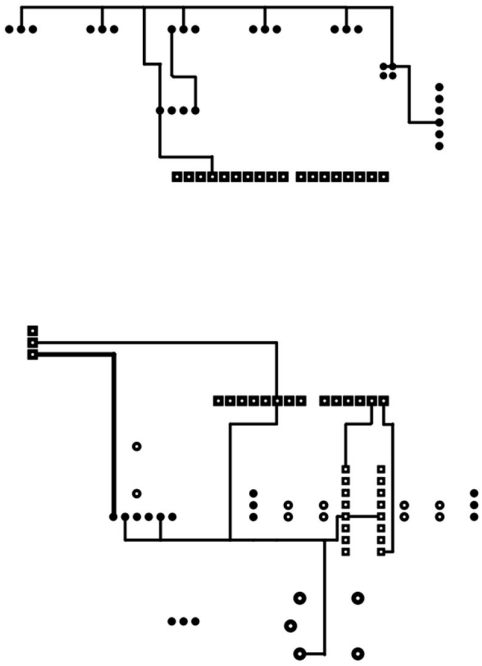
8. Arduino Uno

9. 9v Regulator

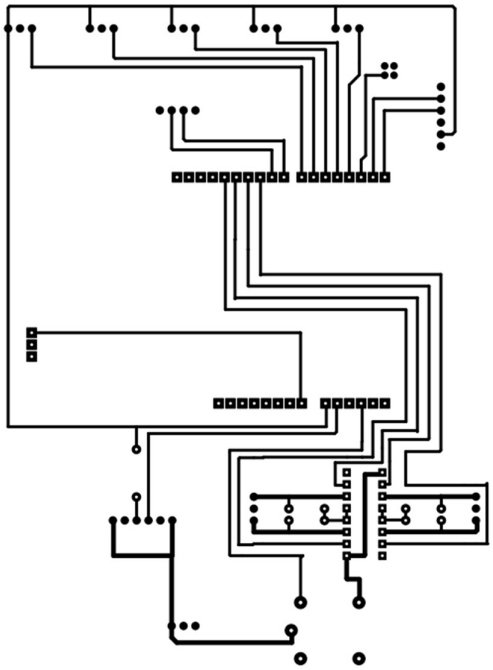
10. 12V battery

11. Relay

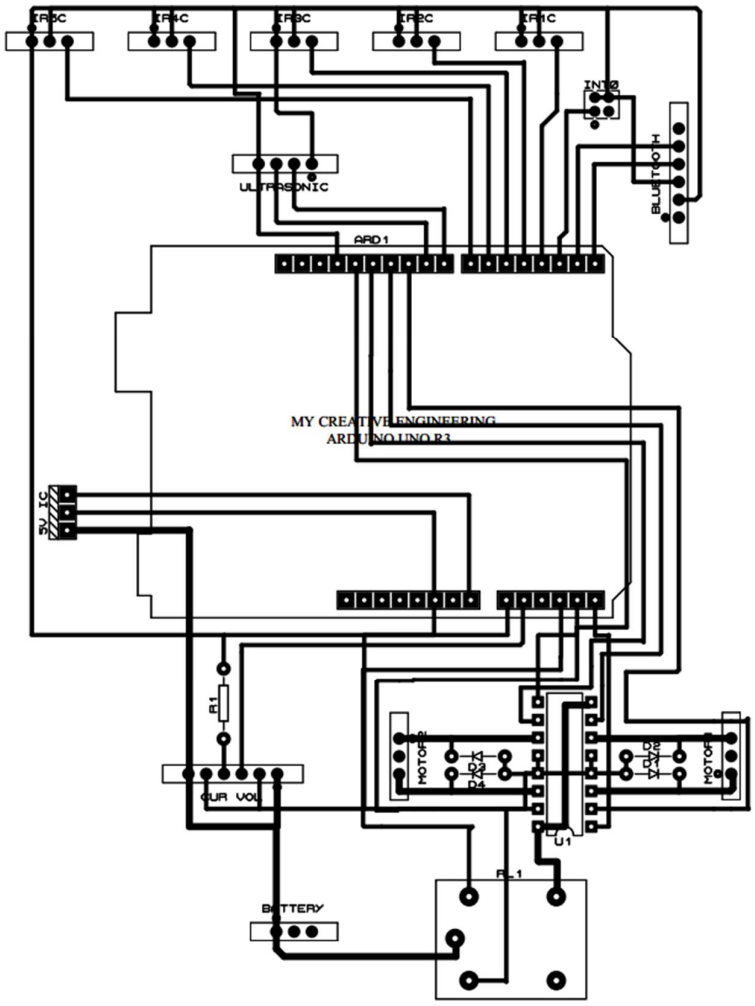
Deliverable of Complete Electronics Design



Top copper



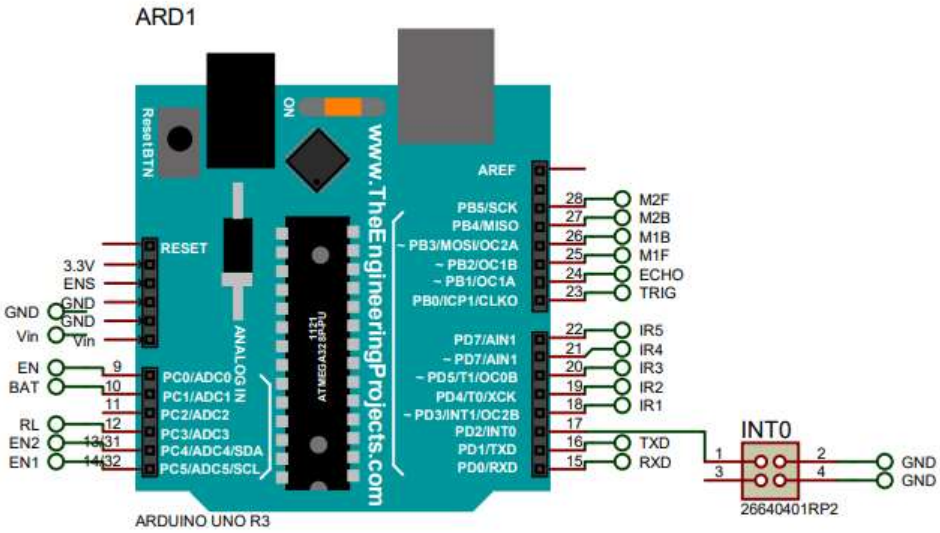
Bottom copper



Complete PCB Wiring

Chapter 6- Software/Firmware Design

Input Output pin outs



Controller Selections with features

We chose to use an Arduino uno for our controller because we already had one at hand and it was sufficient for the requirements of the project that we were given according to the plan we made to execute this project. The components we chose and all the requirements we filled would be perfectly accommodated using and Arduino uno without the need for us to purchase a new different microcontroller and hence we chose to use this microcontroller.

According to our planning all the pins on the Arduino uno were going to be used to control some component on the board and we would not be left with any excess port on the board but if that fulfils out requirements then we don't need any extra ports anyways. The smaller footprint on the Arduino uno also made it easier to make a smaller sized bot with higher precision and speed to track the line.

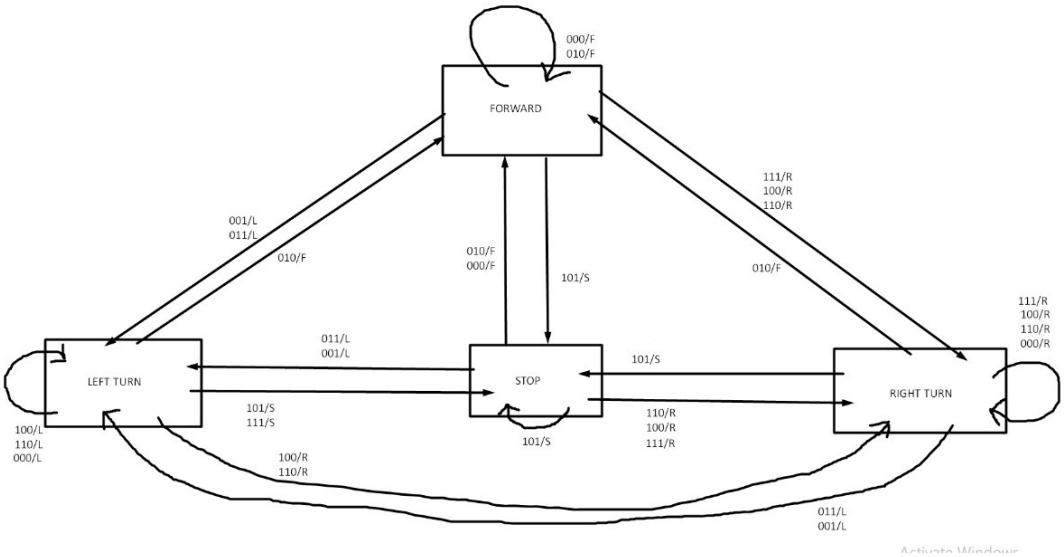
Software Design details & user Requirements

We had to develop out software or our code according to the requirements the user had given us that the bot had to have, some of which were that that bot needs to follow a white line on a table, it needs to be able to detect any obstacles in its path and avoid them by coming back to the starting position, it needs to display its data on a screen and it also needs to display the voltages and currents on the screen

We used a Bluetooth sensor to display everything on a phone as we did not have want to sped too much money on a bigger controller and a Bluetooth sensor requires less pins than a lcd screen, we used a MAX G471 current and voltage sensor to calculate the currents and voltages to reduce the number

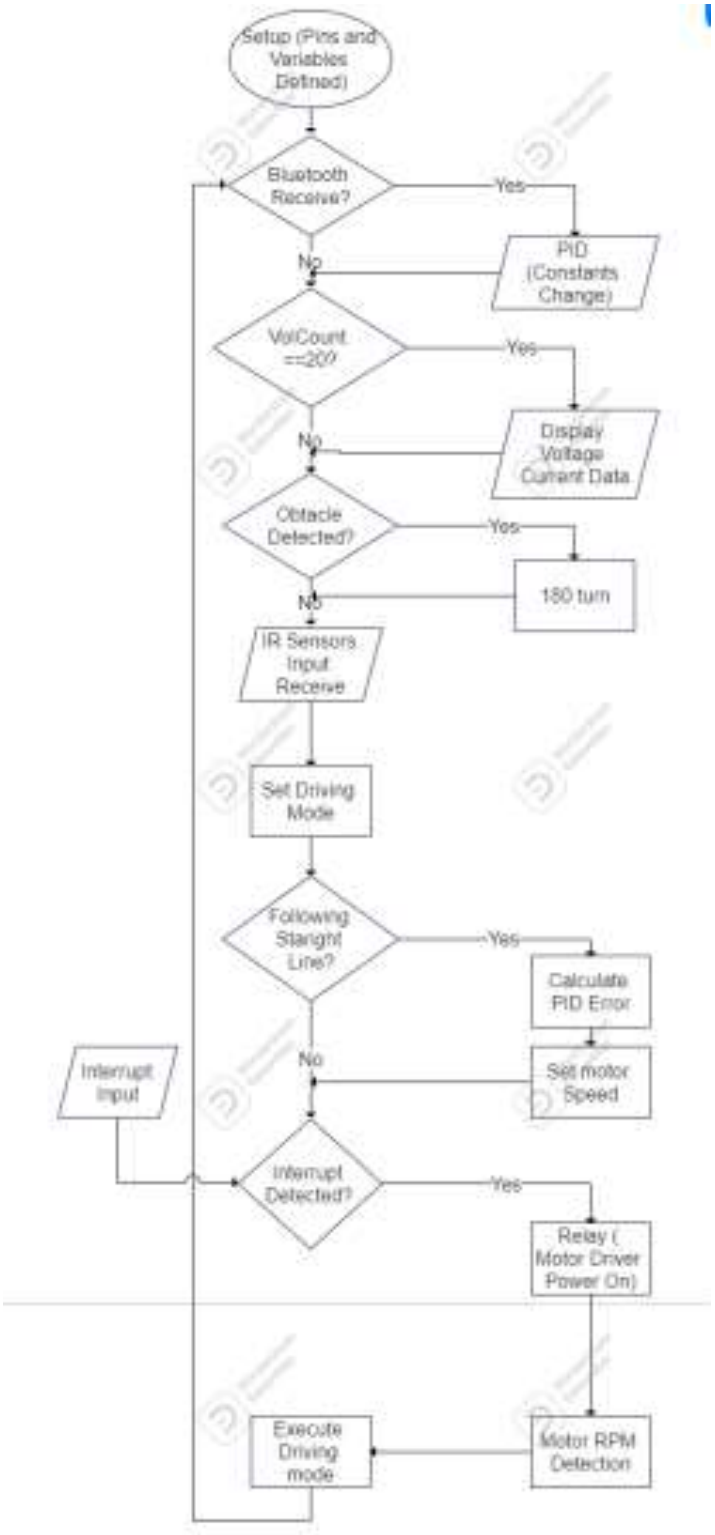
of sensors on the board and make it as compact as possible keeping in view the user requirements and used an Ultrasonic sensor to detect obstacles in the path, after selecting out components all there was left to do was to program what and how each sensor would work - under what conditions and what output they would give. Aside from the coding we had to use proteus to build the board that interconnected all the components with the Arduino in a manner that we could easily program them and control how everything works.

State Machine



State Diagram.

Code Flow Chart



Deliverable of Complete Commented Code

```
/*#include <SoftwareSerial.h>
SoftwareSerial Serial(0,1); // RX, TX*/

//driving mode of the car, it depends on tha band of the of the track
# define STOPPED 0
# define FOLLOWING_LINE 1
# define NO_LINE 2
# define BIG_ANGLE_LEFT 3
# define BIG_ANGLE_RIGHT 4
# define SMALL_ANGLE_LEFT 5
# define SMALL_ANGLE_RIGHT 6

//motor pin configuration
const int rightMotorEnable = 10;
const int leftMotorEnable = 11;
const int rightMotorBackward = A5;
const int rightMotorForward = A4;
const int leftMotorBackward = 12;
const int leftMotorForward = 13;

const int lineFollowSensor0 = 6; //righ most sensor
const int lineFollowSensor1 = 5;
const int lineFollowSensor2 = 4;
const int lineFollowSensor3 = 3;
const int lineFollowSensor4 = 7; //left most sensor
const int Interrupt= 2
const int Relay= A3;
const int busPin= A2;
const int outPin= A1;
const int signPin=A0;
const int Echo= 8;
const int Trigger= 9;

int leftMotorStartingSpeed = 70;
int rightMotorStartingSpeed = 80;
int manualSpeed = 65;

int mode = 0;
unsigned char IntCount=0;
unsigned char VolCount=0;
int distance;
bool obstacle;

int LFSensor[5]={0, 0, 0, 0, 0};

// PID controller
float Kp=25;
```

```
float Ki=0;
float Kd=15;

float pidValue = 0;

float error=0, P=0, I=0, D=0, PIDvalue=0;
float previousError=0, previousI=0;

int wheelCheck=0;
//String command;
String device;

String inputString = "";
String command = "";
String value = "";
boolean stringComplete = false;

void setup() {
  Serial.begin(9650);  //Set the baud rate to that of your Serial module.
  //Serial.begin(9650); //set the baud rate for your Serial module

  //all motor controller related pin should be output
  pinMode(rightMotorForward, OUTPUT);
  pinMode(rightMotorBackward, OUTPUT);
  pinMode(leftMotorForward, OUTPUT);
  pinMode(leftMotorBackward, OUTPUT);
  pinMode(rightMotorEnable, OUTPUT);
  pinMode(leftMotorEnable, OUTPUT);
  pinMode(Relay,OUTPUT);
  pinMode(Trigger, OUTPUT);

  //all sensor pin should be input
  pinMode(lineFollowSensor0, INPUT);
  pinMode(lineFollowSensor1, INPUT);
  pinMode(lineFollowSensor2, INPUT);
  pinMode(lineFollowSensor3, INPUT);
  pinMode(lineFollowSensor4, INPUT);
  pinMode(Interrupt,INPUT_PULLUP);
  pinMode(Echo, INPUT);
  pinMode(signPin, INPUT);
  pinMode(outPin, INPUT);
  pinMode(busPin, INPUT);
  attachInterrupt(digitalPinToInterrupt(2), Start, FALLING); // Interrupt

  //command for Serial receiver
  inputString.reserve(50);  // reserve 50 bytes in memory to save for
stringmanipulation
  command.reserve(50);
```

```
  value.reserve(50);
}

void loop() {

 //uncomment the following two lines for testing sensors
 //testLineFollowSensorsAndPIDvalue();
 //delay(1000);

 /***************This snippet is for PID tuning
usingSerial*************************/
 serialEvent(); //check if Serial receiver receives any data
 if (stringComplete) {
    delay(100);
    // identified the posiion of '=' in string and set its index to pos variable
    int pos = inputString.indexOf('=');
    // value of pos variable > or = 0 means '=' present in received string.
    if (pos > -1) {
      // substring(start, stop) function cut a specific portion of string from
startto stop
      // here command will be the portion of received string till '='
      // let received string is KP=123.25
      // then the receive value id for KP and actual value is 123.25
      command = inputString.substring(0, pos);
      // value will be from after = to newline command
      // for the above example Kp value is 123.25
      // we just ignoreing the '=' taking first parameter of substring as 'pos+1'
      // we are using '=' as a separator between command and vale
      // without '=' any other character can be used
      value = inputString.substring(pos+1, inputString.length()-1);  //
extractcommand up to \n exluded
      pidValue = value.toFloat(); //convert the string into float value
      //uncomment following line to check the value in serial monitor
      //Serial.println(pidValue);
      if(command == "KP"){
           Kp = pidValue;
           delay(50);
           }
      else if(command == "KI"){
           Ki = pidValue;
           delay(50);
           }
      else if(command == "KD"){
           Kd = pidValue;
           delay(50);
           }
       }
      inputString = "";
```

```
      stringComplete = false;//all the data is collected from serial buffer
    }

//Every 20th loop Voltage Current Value will be displayed
VolCount++;
 if (VolCount==20){
   VoltageCurrent();
   VolCount=0;
 }
 //if Obstacle detected in path
 obstacle= Ultrasonic();
 if (obstacle){
  Serial.println("OBSTACLE Detected");
  Serial.println("180 Turn");
  do{
    int x=0;
    wheelCheck=0;
    wheel_rotation(65,-65); //rotate right for small angle
    if (x==0)
    delay(300);  //delay determines the degree of angle
    x++;
  }while(digitalRead(lineFollowSensor2)==1);
 } /**********************This snippet is for Line
Following**************************************/
 readLFSsensorsAndCalculateError(); //read sensor, calculate error and set
drivingmode
 switch (mode)
   {
    case STOPPED:  //all sensors read black line
      wheelCheck=0;
      motorStop();
      previousError = error;
      Serial.println("STOPPED");
      break;

    case NO_LINE:  //all sensor are on white plane
      wheelCheck+=1;
      if(wheelCheck==1){
      forward();
      delay(2);
      }
      else if(wheelCheck==2){
      wheel_rotation(65,-65);
      delay(4);
      }
      previousError = 0;
      Serial.println("NO LINE");
      break;
```

```
    case BIG_ANGLE_RIGHT:
      wheelCheck=0;
      wheel_rotation(65,-65); //rotate right for small angle
      delay(5);  //delay determines the degree of angle
      Serial.println("BIG RIGHT");
      break;

    case BIG_ANGLE_LEFT:
      wheelCheck=0;
      wheel_rotation(-75,75); //rotate left for small angle
      delay(5);
      Serial.println("BIG LEFT");
      break;

    case SMALL_ANGLE_RIGHT:
      wheelCheck=0;
      wheel_rotation(65,-65);      delay(20); //rotate right for big angle
      previousError = 0;
      //Serial.println("SMALL RIGHT");
      break;

    case SMALL_ANGLE_LEFT:
      wheelCheck=0;
      wheel_rotation(-75,75);
      delay(20); //rotate left for big angle
      previousError = 0;
      //Serial.println("SMALL LEFT");
      break;

    case FOLLOWING_LINE: //everything is going well
      wheelCheck=0;
      calculatePID();
      motorPIDcontrol();
      Serial.println("STRAIGHT");
      break;
  }

}

void forward(){ //drive forward at a preset speed
  analogWrite(rightMotorEnable, manualSpeed);
  digitalWrite(rightMotorForward, HIGH);
  digitalWrite(rightMotorBackward, LOW);

  analogWrite(leftMotorEnable, manualSpeed);
  digitalWrite(leftMotorForward, HIGH);
  digitalWrite(leftMotorBackward, LOW);
```

```
}

void left(){ //left motor rotate CCW and right motor rotate CW
  analogWrite(rightMotorEnable, manualSpeed);
  digitalWrite(rightMotorForward, HIGH);
  digitalWrite(rightMotorBackward, LOW);

  analogWrite(leftMotorEnable, manualSpeed);
  digitalWrite(leftMotorForward, LOW);
  digitalWrite(leftMotorBackward, HIGH);
}

void right(){ //left motor rotate CW and left motor rotate CCW
  analogWrite(rightMotorEnable, manualSpeed);
  digitalWrite(rightMotorForward, LOW);
  digitalWrite(rightMotorBackward, HIGH);

  analogWrite(leftMotorEnable, manualSpeed);
  digitalWrite(leftMotorForward, HIGH);
  digitalWrite(leftMotorBackward, LOW);
}

void motorStop(){ //speed of both motor is zero
  analogWrite(rightMotorEnable, 0);
  digitalWrite(rightMotorForward, LOW);
  digitalWrite(rightMotorBackward, LOW);

  analogWrite(leftMotorEnable, 0);
  digitalWrite(leftMotorForward, LOW);
  digitalWrite(leftMotorBackward, LOW);
}

void readLFSsensorsAndCalculateError()
{
  LFSensor[0] = digitalRead(lineFollowSensor0);
  LFSensor[1] = digitalRead(lineFollowSensor1);
  LFSensor[2] = digitalRead(lineFollowSensor2);
  LFSensor[3] = digitalRead(lineFollowSensor3);
  LFSensor[4] = digitalRead(lineFollowSensor4);

  if((     LFSensor[0]== 1 )&&(LFSensor[1]== 1 )&&(LFSensor[2]== 1
)&&(LFSensor[3]==1 )&&(LFSensor[4]== 0 ))  {mode = FOLLOWING_LINE; error = -4;}
  else if((LFSensor[0]== 1 )&&(LFSensor[1]== 1 )&&(LFSensor[2]== 1
)&&(LFSensor[3]==0 )&&(LFSensor[4]== 0 ))  {mode = FOLLOWING_LINE; error = -3;}
  else if((LFSensor[0]== 1 )&&(LFSensor[1]== 1 )&&(LFSensor[2]== 1 )&&(LFSensor[3]==
0 )&&(LFSensor[4]== 1 ))  {mode = FOLLOWING_LINE; error = -2;}
  else if((LFSensor[0]== 1 )&&(LFSensor[1]== 1 )&&(LFSensor[2]== 0 )&&(LFSensor[3]==
0 )&&(LFSensor[4]== 1 ))  {mode = FOLLOWING_LINE; error = -1;}
```

```
  else if((LFSensor[0]== 1 )&&(LFSensor[1]== 1 )&&(LFSensor[2]== 0 )&&(LFSensor[3]==
1 )&&(LFSensor[4]== 1 ))  {mode = FOLLOWING_LINE; error = 0;}
  else if((LFSensor[0]== 1 )&&(LFSensor[1]== 0 )&&(LFSensor[2]== 0 )&&(LFSensor[3]==
1 )&&(LFSensor[4]== 1 ))  {mode = FOLLOWING_LINE; error = 1;}
  else if((LFSensor[0]== 1 )&&(LFSensor[1]== 0 )&&(LFSensor[2]== 1 )&&(LFSensor[3]==
1 )&&(LFSensor[4]== 1 ))  {mode = FOLLOWING_LINE; error = 2;}
  else if((LFSensor[0]== 0 )&&(LFSensor[1]== 0 )&&(LFSensor[2]== 1 )&&(LFSensor[3]==
1 )&&(LFSensor[4]== 1 ))  {mode = FOLLOWING_LINE; error = 3;}
  else if((LFSensor[0]== 0 )&&(LFSensor[1]== 1 )&&(LFSensor[2]== 1 )&&(LFSensor[3]==
1 )&&(LFSensor[4]== 1 ))  {mode = FOLLOWING_LINE; error = 4;}
  else if((LFSensor[0]== 0 )&&(LFSensor[1]== 0 )&&(LFSensor[2]== 0 )&&(LFSensor[3]==
0 )&&(LFSensor[4]== 0 ))  {mode = STOPPED; error = 0;}
  else if((LFSensor[0]== 1 )&&(LFSensor[1]== 1 )&&(LFSensor[2]== 1 )&&(LFSensor[3]==
1 )&&(LFSensor[4]== 1 ))  {mode = NO_LINE; error = 0;}
  //track goes right at an angle >90 degree
  else if((LFSensor[0]== 0 )&&(LFSensor[1]== 0 )&&(LFSensor[2]== 0 )&&(LFSensor[3]==
1 )&&(LFSensor[4]== 1 ))  {mode = BIG_ANGLE_RIGHT; error = 0;}
  else if((LFSensor[0]== 0 )&&(LFSensor[1]== 0 )&&(LFSensor[2]== 0 )&&(LFSensor[3]==
0 )&&(LFSensor[4]== 1 ))  {mode = BIG_ANGLE_RIGHT; error = 0;}
  //track goes right at an angle <90 degree
  else if((LFSensor[0]== 0 )&&(LFSensor[1]== 1 )&&(LFSensor[2]== 0 )&&(LFSensor[3]==
1 )&&(LFSensor[4]== 1 ))  {mode = SMALL_ANGLE_RIGHT; error = 0;}
  //track goes left at an angle >90 degree
  else if((LFSensor[0]== 1 )&&(LFSensor[1]== 1 )&&(LFSensor[2]== 0 )&&(LFSensor[3]==
0 )&&(LFSensor[4]== 0 ))  {mode = BIG_ANGLE_LEFT; error = 0;}
  else if((LFSensor[0]== 1 )&&(LFSensor[1]== 0 )&&(LFSensor[2]== 0 )&&(LFSensor[3]==
0 )&&(LFSensor[4]== 0 ))  {mode = BIG_ANGLE_LEFT; error = 0;}
  //track goes left at an angle <90 degree
  else if((LFSensor[0]== 1 )&&(LFSensor[1]== 1 )&&(LFSensor[2]== 0 )&&(LFSensor[3]==
1 )&&(LFSensor[4]== 0 ))  {mode = SMALL_ANGLE_LEFT; error = 0;}

}

void testLineFollowSensorsAndPIDvalue()
{
     int LFS0 = digitalRead(lineFollowSensor0);
     int LFS1 = digitalRead(lineFollowSensor1);
     int LFS2 = digitalRead(lineFollowSensor2);
     int LFS3 = digitalRead(lineFollowSensor3);
     int LFS4 = digitalRead(lineFollowSensor4);

     Serial.print ("LFS: L  0 1 2 3 4  R ==> ");
     Serial.print (LFS0);
     Serial.print (" ");
     Serial.print (LFS1);
     Serial.print (" ");
     Serial.print (LFS2);
     Serial.print (" ");
```

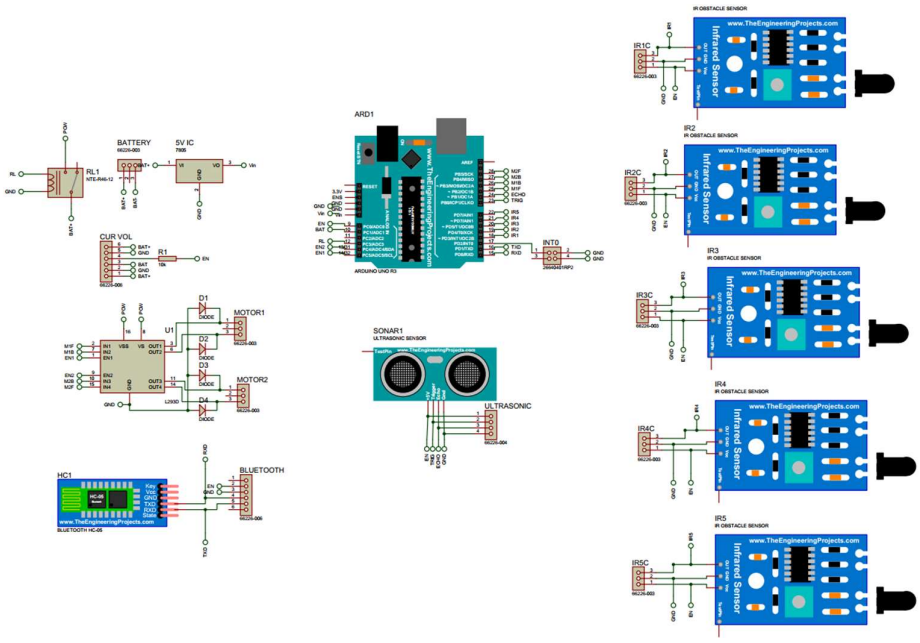
```
    Serial.print (LFS3);
    Serial.print (" ");
    Serial.print (LFS4);
    Serial.print ("  ==> ");

    Serial.print (" P: ");
    Serial.print (P);
    Serial.print (" I: ");
    Serial.print (I);
    Serial.print (" D: ");
    Serial.print (D);
    Serial.print (" PID: ");
    Serial.println (PIDvalue);
}

void calculatePID()
{
  P = error; //maximum error value is 4 and minimum is zero
  I = I + error;
  D = error-previousError;
  PIDvalue = (Kp*P) + (Ki*I) + (Kd*D);
  previousError = error;
}

void motorPIDcontrol()
{
  int leftMotorSpeed = leftMotorStartingSpeed + PIDvalue;
  int rightMotorSpeed = rightMotorStartingSpeed - PIDvalue;

  // The motor speed should not exceed the max PWM value
  constrain(leftMotorSpeed, 60, 100);
  constrain(rightMotorSpeed, 60, 100);

  //determing the rotation and direction of wheel
  wheel_rotation(leftMotorSpeed,rightMotorSpeed);
}

void wheel_rotation(int left, int right){
  if(left>0){
    analogWrite(leftMotorEnable, left);
    digitalWrite(leftMotorForward, HIGH);
    digitalWrite(leftMotorBackward, LOW);
    }
  else if(left<0){
    analogWrite(leftMotorEnable, abs(left));
    digitalWrite(leftMotorForward, LOW);
    digitalWrite(leftMotorBackward, HIGH);
    }
```

```
  if(right>0){
    analogWrite(rightMotorEnable, right);
    digitalWrite(rightMotorForward, HIGH);
    digitalWrite(rightMotorBackward, LOW);
    }
  else if(right<0){
    analogWrite(rightMotorEnable, abs(right));
    digitalWrite(rightMotorForward, LOW);
    digitalWrite(rightMotorBackward, HIGH);
    }
  }

void serialEvent() {
  while (Serial.available()) {
    // get the new byte:
    char inChar = (char)Serial.read();
    //Serial.write(inChar);
    // add it to the inputString:
    inputString += inChar;
    // if the incoming character is a newline or a carriage return, set a flag
    // so the main loop can do something about it:
    if (inChar == '\n' || inChar == '\r') {
      stringComplete = true;
    }
  }
}

void Start(){
  IntCount++;
  if(IntCount==1)
  {
    digitalWrite(Relay, HIGH);
  }
  else{
    digitalWrite(Relay, LOW);
    IntCount=0;
  }
}

bool Ultrasonic(){
  digitalWrite(Trigger, LOW);
  delayMicroseconds(2);
  // Sets the trigPin HIGH (ACTIVE) for 10 microseconds
  digitalWrite(Trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(Trigger, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  long duration = pulseIn(Echo, HIGH);
```

```
  // Calculating the distance
  distance = duration * 0.034 / 2; // Speed of sound wave divided by 2 (go and back)
  if ((distance>1) && (distance<=15))
  return 1;
  else
  return 0;
}

void VoltageCurrent(){
  float voltage = 0.0;
  unsigned long rawVal = 0;
  analogReference(INTERNAL);
  /* After switching you have to execute a dummy analogRead,
     followed by a short */
  rawVal = analogRead(outPin);
  rawVal = 0;
  delay(10);
  rawVal += analogRead(outPin);
  Serial.println("**********Battery Voltage Current***********");
  voltage = rawVal * 1100.0 / 1023.0;
  Serial.print("V Out         [mV]: ");
  Serial.println(voltage);
  float current_mA = voltage * 0.9452 + 1.0544; // Calibration parameters
  Serial.print("Busstrom      [mA]: ");  // Bus current
  Serial.println(current_mA);
  Serial.print("Stromrichtung:    : ");  // Current direction
  if(digitalRead(signPin)){
    Serial.println("RS+ -> RS-");
  }
  else{
    Serial.println("RS- -> RS+");
  }
  analogReference(DEFAULT);
  voltage = analogRead(busPin) * 5.0 / 1023.0;
  Serial.print("Busspannung    [V]: "); // Bus voltage
  Serial.println(voltage);
  float power_mW = (voltage * current_mA);
  Serial.print("Leistung      [mW]: ");  // Power
  Serial.println(power_mW);
  Serial.println("-------------------------------");
}
```

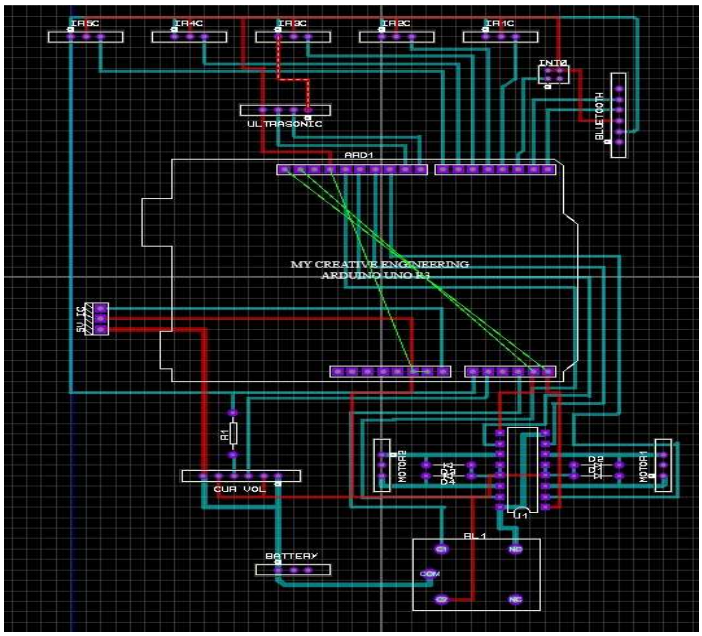
Chapter 7- Simulations and final Integrations

Simulations



Simulation and Schematic and Diagram.

Simulation PCB



PCB layout

Chapter 8- System Test phase

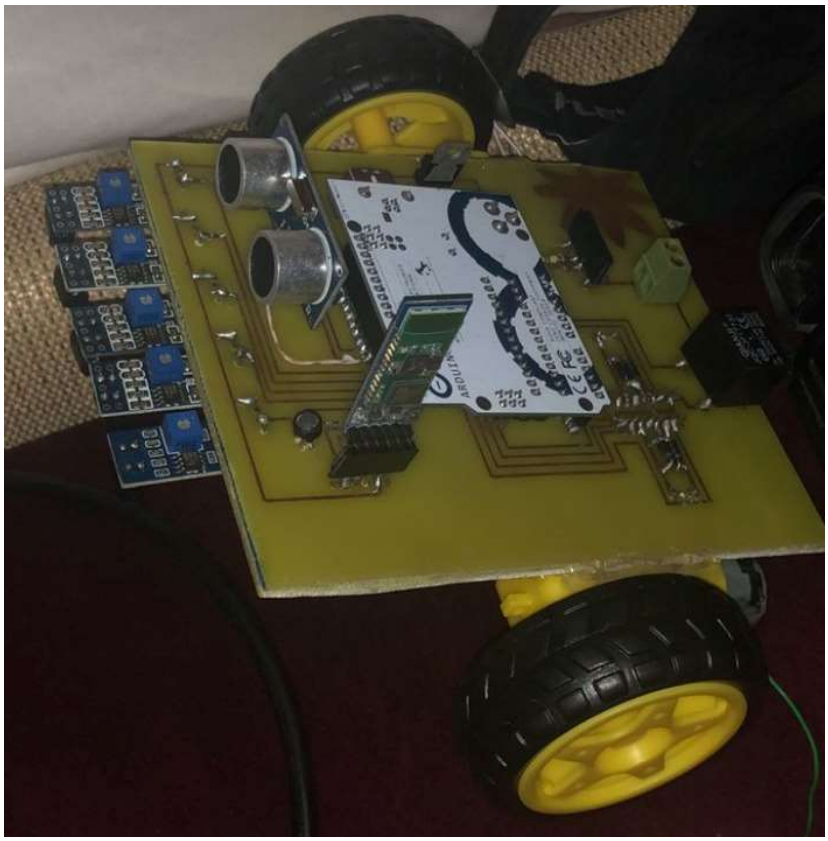
Final testing

After finally completing the robot we tested the Bot in a controlled setting with a power supply connected to the robot. We put the bot on the line following arena and connected all the sensors. After connecting the robot with our phone for Bluetooth data transmission we calibrated the robot on the line and then placed it on the line for it to start following the line as wanted, there were some issues in the code which was causing the bot to turn in the wrong directions but we diagnosed and corrected that mistake quickly. All the data was being shown on the phone using the Bluetooth sensor and the bot was following the line we intended it to.

Things that are questionable and get burned again and again

There was one component which was unreliable for us which was the 9v regulator, the 9v regulator gets very hot while the robot is running even on voltages lower than its recommended load rate, and if we use it for too long it would burnout. We had to replace 2 regulators due to this reason. Other than the regulator all our other components seem to be reliable and worked great without causing any issues.

Project actual Pictures



Conclusion:

The overall end objective was to make the robot follow a white track just by using microcontrollers and the sensors attached to it. The aim of the thesis is to provide simple guidelines to the new students and beginners who are interested in this type of project. Although the outcome was simple, as mentioned earlier, the project makes the new students familiar with proteus and how to make PCB, the working mechanism of it and future aspects of it in a simple and understandable way. Although the thesis project is very little about the robot's use in real world, with the help of guidelines and the abundance of resources the outcome, it could be very beneficial for many people and different sectors of the world depending on the sensors and features required as per necessity. The goal of this project is to get students interested in and excited about the fields of engineering, mechatronics, and software development as they design, construct, and program an autonomous robot. The guidelines provided are very simple to use and understand thus, making it very easy for the new students to build a foundation in their Robotics learning. Even being a powerful little device, microcontroller would be nothing without its hardware. It is just liked a brain that operates a body whereas in this case, the body is made up of many components, including motors, lights, and sensors. The motors provide locomotion, the sensors act as triggers for the motors, and the lights are used primarily as warning device