EEE 307 Microprocessors Experimental Project

Lane Assist to Sensitive Traffic Lights

Buğrahan İSMAİLOĞLU 170441010

Ömer ARSLAN 170441025

CONTENTS

1. Abstract

Nowadays, our lives are affected by the developing technology. In line with these effects, the technology in cars has improved considerably. In particular, interest in autonomous cars, which will increase safety, is increasing. Brands are developing these systems that will increase security considerably. The main ones among these systems are the systems that will make the job of the driver easier. Lane tracking system, sudden stop systems, systems that detect and decide on signs and traffic lights, etc. This project aims to build a car with some of these systems.

The aim of the project is it is planned that the vehicle will continue on its way by staying in the lane during long-distance driving in a fully automatic manner. In addition, it stops or continues depending on the traffic light when it approaches the traffic light in urban use. It will also stop by braking when it detects pedestrians to prevent accidents caused by pedestrians jumping on the road suddenly. After the pedestrian leaves the road, he will continue on his way.

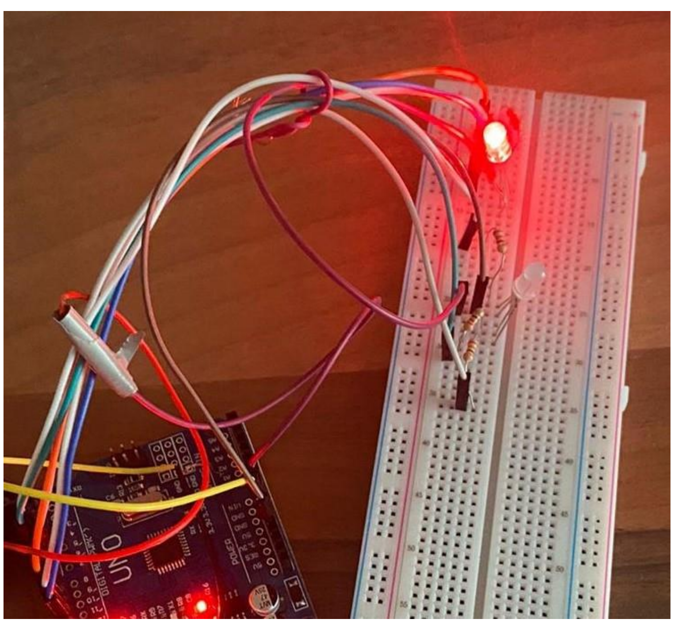
2. Introduction

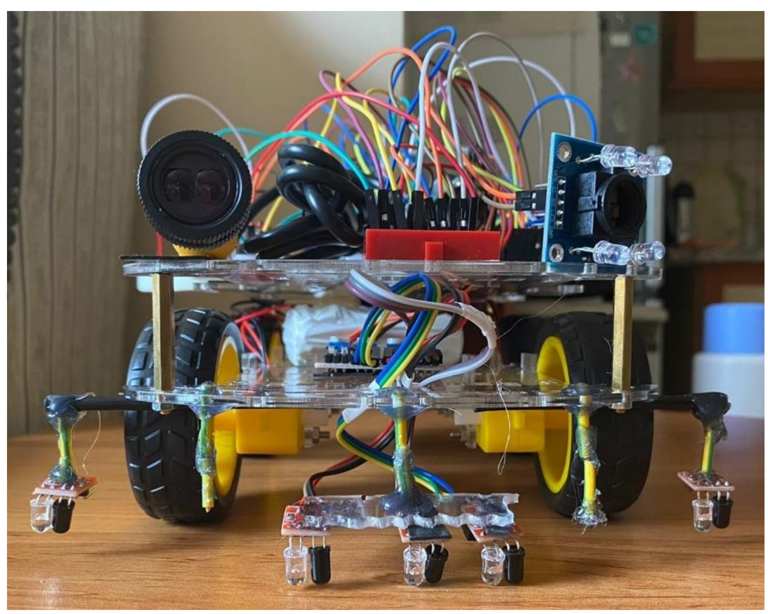
As stated in the summary section, the aim of the project is to design a vehicle that can make decisions on its own and act according to the decision taken according to the road condition. With the sensors on this vehicle and the development board Arduino Mega 2560, it can take these decisions and take act. The development board in the project is Arduino Mega 2560 and the lines of code in it make the right decision and apply it. The most important part of this project is the properties of the sensors used and the compatibility of these sensors with each other. The reason is that the project is a smart car. The main sensors used in the project are as follows. Infrared Sensor, Tracking Sensor, Color Sensor. Each of these sensors used has different purposes and different functions. The general capabilities of the car in the project are as follows. The ability to follow the lane and turn, react according to the state of traffic lights and react to a sudden object on the road. The vehicle moves in line with the command given and provides an output that explains how the vehicle moves accordingly with the LCD on it.

Other circuit elements used in the project and the features of the vehicle made will be explained in more detail in the following sections.

2.1 Parts Required

- Arduino Mega 2560 for car,
- Arduino Uno for traffic lights,
- Motor Driver Board (L298N),
- DC Motor (DC-3-6V),
- Infrared Sensor (MZ80),
- Tracking Module (YL-70),
- Tracking Sensor (YL-73),
- Color Sensor (TCS3200/GY-31),
- Buzzer,
- 9V Battery,
- 18650 Battery 2000 mAh,
- 3S BMS,
- 16x2 LCD screen,
- 10 kilohm resistor
- Robot Car Platform Kit,
- Wheel,
- Switch,
- RGB Led (Green and Red Light),
- 100 ohm Resistance, eight 220 ohm Resistance.





3. Materials and Methods

Arduino Mega 2560: The reason we used Arduino Unp in this project is that there are not enough pins. That's why we used Arduino Mega 2560.

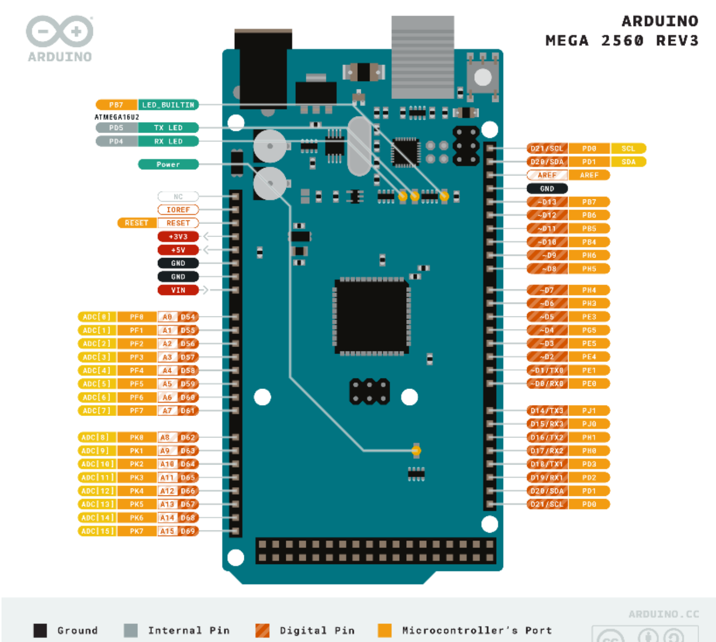


Figure 3.1

- Based on ATmega2560
- 54 Digital input/output pins
- 15 PWM output
- 16 Analog inputs
- 4 Hardware serial port (UARTs)
- 16 MHz crystal oscillator
- USB connection
- Power jack
- ICSP header
- Reset button.
- Work with +5V.

Arduino Uno: We used Arduino Uno for traffic light construction.

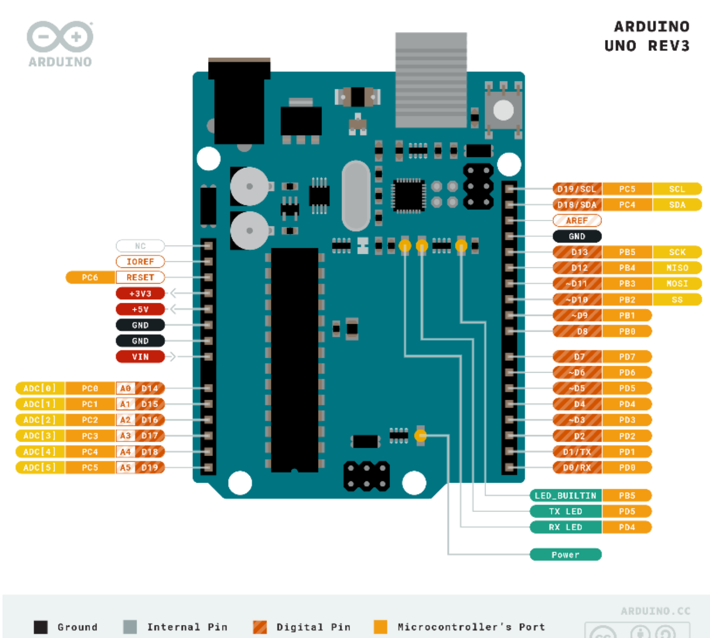


Figure 3.2

- Based on ATmega328P
- 14 Digital input/output pins
- 6 PWM output
- 6 Analog inputs
- 16 MHz ceramic resonator
- USB connection
- Power jack
- ICSP header
- Reset button.
- Work with +5V.

Motor Driver Board (L298N): We chose to use this motor driver so that we can adjust the required power to our motors.

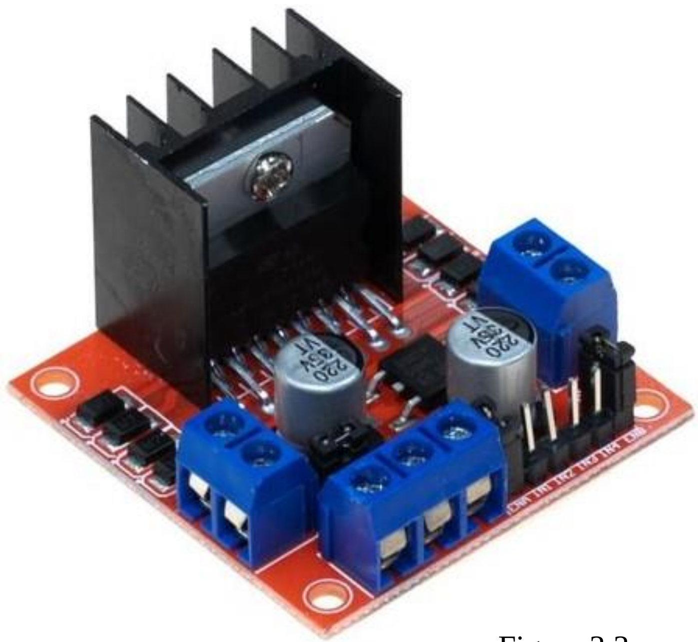


Figure 3.3

- ENA: Left motor channel activation pin.
- ENB: Right motor channel activation pin.
- IN1: Left engine 1st input
- IN2: Left engine 2nd input
- IN3: Right engine 1st input
- IN4: Right engine 2nd input
- VCC: Supply voltage (4.8V-24V)
- GND: Ground
- 5V: 5V output

Infrared Sensor (MZ80): The biggest factor in our preference for this sensor was that the distance range was up to 80cm.

Figure 3.4

- The MZ80 consists of a receiver and a transmitter. The infrared transmitter emits light and the receiver detects the infrared light returning from objects. In this way, it understands whether there is an object or not.
- Work with 5V.
- 25mA current output.
- 3-80 work range.

Tracking Module (YL-70): We used this module to keep track of the lanes.

- Operating voltage: DC 3.3V-5V

- Operating Current: >1A

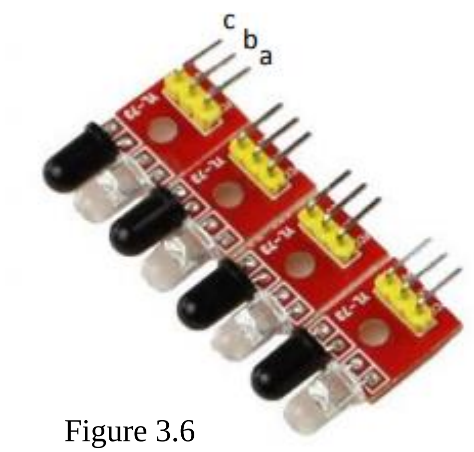
- 1. VCC: 3.3V-5V DC

- 2. GND: ground

- 3, 4, 5, 6 high/low outputs.

Figure 3.5

Tracking Sensor (YL-73)



- The YL73 transmitter, which consists of a pair of transceivers, decides whether the light it sends bounces back and comes to the receiver.

- Detection distance: 1mm to 60cm adjustable

Figure 3.6

Color Sensor (TCS3200)

- TCS3200 is a circuit element that uses photodiodes. It converts the values from the photodiodes into a square wave in direct proportion to the light intensity using a current to frequency converter. These values go to Arduino and find out which color it is.

- Input Voltage: 2.7-5V

- Interface: Digital TTL

Figure 3.7

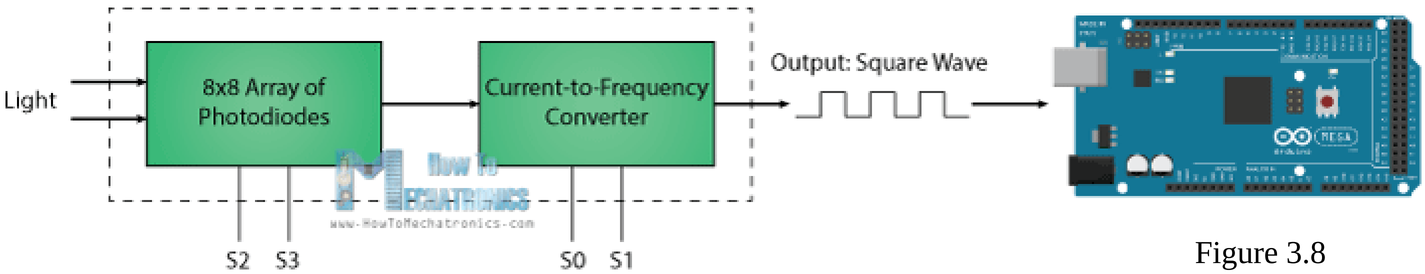


Figure 3.8

3S BMS

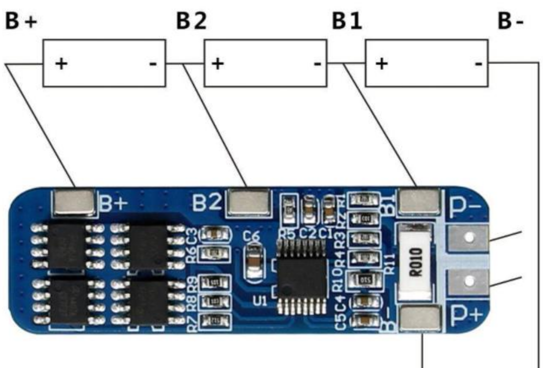


Figure 3.9

- This circuit element provides over charge, discharge and current protection.

- Nominal Voltage: 11.1V

- Maximum Voltage: 12.6V

16x2 LCD

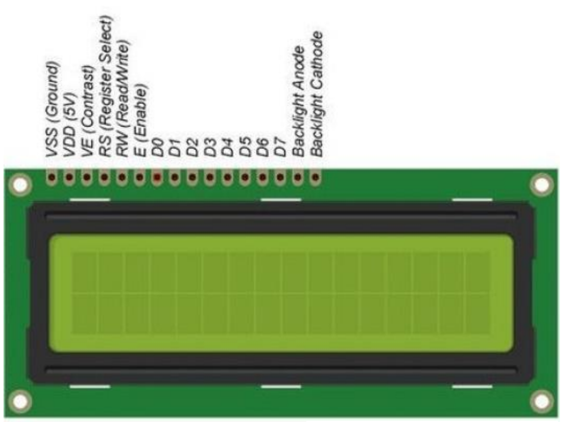


Figure 3.10

- I2C Address: 0x20-0x27

- Input Voltage: 5V

- Interface: I2C/TWI

18650 Battery: We tried a few different batteries, but we preferred to use this battery so that our motors can work efficiently. Thanks to these batteries, no additional battery was required o power our Arduino.

Figure 3.11

- Nominal Voltage: 3.6V
- Maximum Voltage: 11.1V
- Nominal Capacity: 2000mAh

DC Motor

- Input Voltage: 3-6V
- Input Current: 0.35-0.4A
- 250 RPM

Figure 3.12

200 ohm – 100 ohm – 10 Kohm Resistor:

We used these resistors in the use of LCD, LEDs and buzzers.

Figure 3.13

Figure 3.14

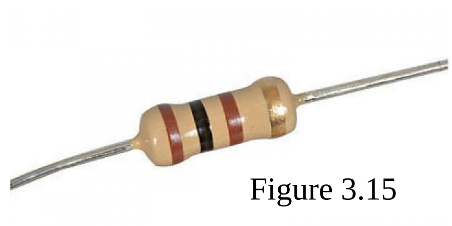


Figure 3.15

Buzzer

Figure 3.16

- Input Voltage: 5V

Switch

Figure 3.17

- When pressed, the circuit switches to transmission, if not pressed, it does not transmit.

Potentiometer

Figure 3.18

- It is one of the adjustable resistors whose value changes as you turn the shaft on it. It is often used where intermediate resistance value is required or in current and voltage control circuits. The resistance value varies between the middle leg and the lateral legs. While the resistance between the middle leg and any leg is 0 Ohm, the resistance between the other leg is at the maximum level.

LED

Figure 3.19

- Color: Red

- Size: 10 mm

- Working Voltage: 1.5-3V

RGB LED

Figure 3.20

- Can give 16 colors

- Long pin can be anode or cathode

4. Results

If we examine the working principle of the project in 3 separate paragraphs:

4.1 Line Tracking Sensors (YL_70/YL-73)

Line tracking sensors consist of a receiver and a transmitter. The transmitter continuously emits beams and the receiver collects this beam. The beam falling on a black background is absorbed and does not reflect back. The beam falling on a white background is reflected back. This sensor generates an output according to these two conditions and works that way.

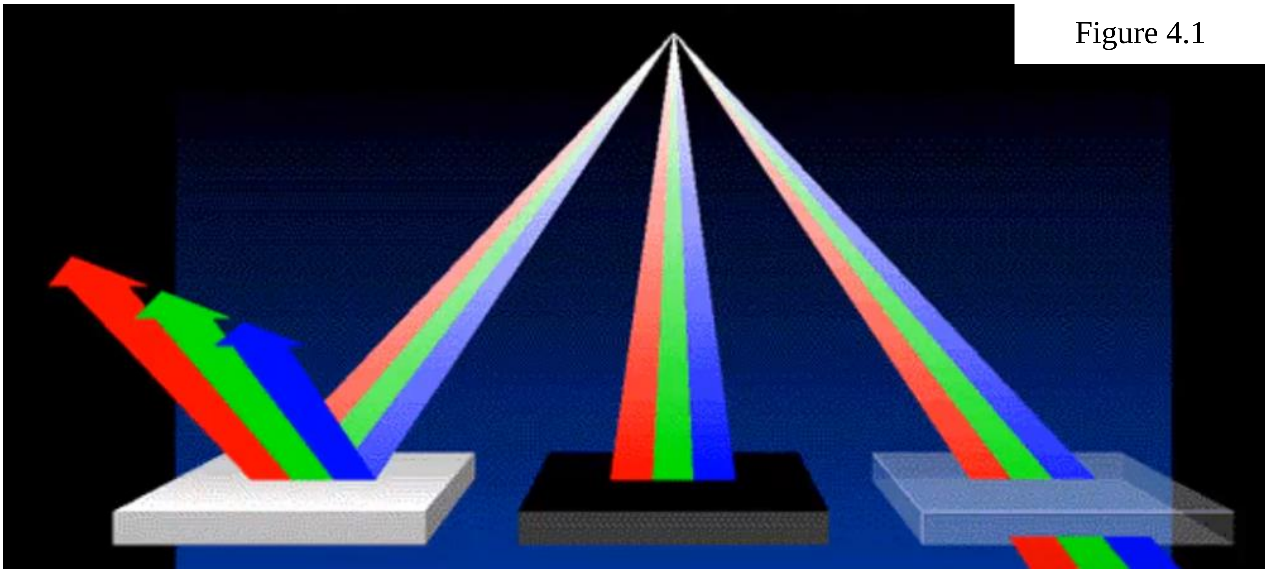


Figure 4.1

Line tracking sensors in the project detect the lines on the path and act accordingly. Line tracking sensors in the project detect the lines on the path and act accordingly. Line Tracking sensors read the line on which it is located and creates 1-0 values. In this project, values of 1 read black ground and 0 values read white ground. When the sensor in the middle of the 3 sensors used in the project reads the white line (0) and the other two sensors read the black ground (1), the engines are given full power, and the car moves. In case a turn is encountered, the vehicle performs the rotation process in line with the values from the other two sensors. For example, while turning left, the sensor on the left will read the white ground (0) and therefore cut the power on the motors for a short time and power the motors on the right. When the rotation is completed, the sensor on the left will read the black ground (1) again and the sensor in the middle will start to read the white background (0). In this case, the engines will be powered again and the vehicle will continue. While turning to the right, the sensor on the right will read the white ground (0) and therefore cut the power on the motors for a short time and power the motors on the left side. When the rotation process is completed, the sensor on the right will read the black ground (1) again and the sensor in the middle will start to read the white ground (0). In this case, the engines will be powered again and the vehicle will continue.

4.2 Infrared Sensor (MZ80)

IR sensors consist of two parts. The first is a light emitting diode (LED) that emits continuous infrared rays. The second is the photodiode that collects these rays. If the beam sent by the LED hits an object, it is reflected back and the photodiode collects the rotating rays and thus detects whether an object is present.

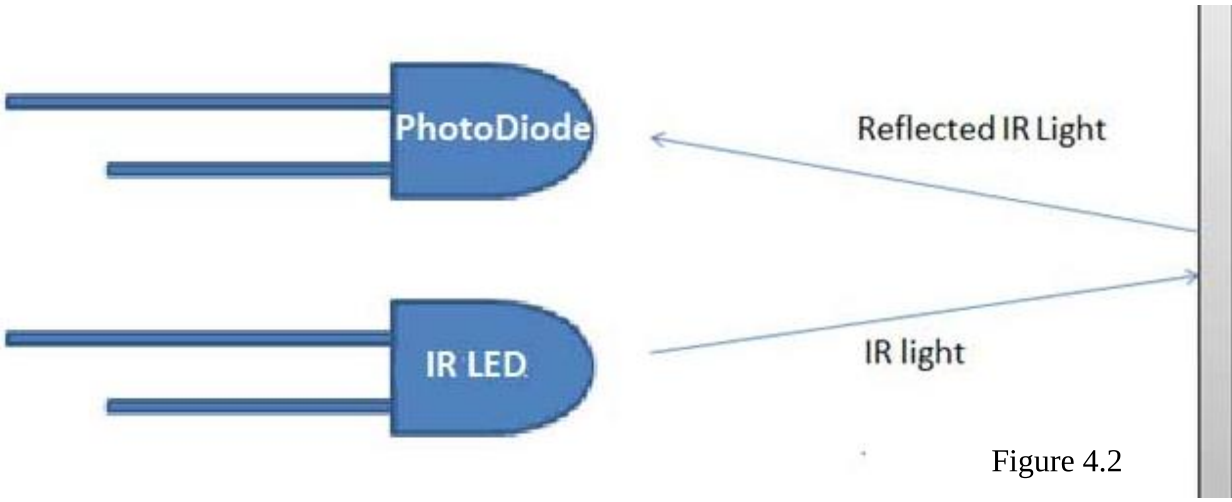


Figure 4.2

The infrared sensor (MZ80) in the project while the car is running. When the line tracking system is active, if an object comes across the car at a distance of 0-20 cm, it stops the car. During this period when the car stopped, two brake lights in the car turn on. In addition, as soon as the car sees the object, it makes a sound with the buzzer on it and warns the object that is suddenly on the road.

4.3 Color Sensor (TCS3200)

The TCS3200 has a photodiode section made up of 64 small cells in total. 16 of them are red, 16 of them are blue, 16 of them are green and 16 of them are transparent. The beam coming on these photodiodes is transformed into a square wave form with a current to frequency converter. These 16 photodiodes are connected in parallel to each other. The color to be read is specified with S2 and S3 pins. This is determined by the signal given to pins S2 and S3. The other two pins in the tcs3200 are S0 and S1. These pins determine the frequency value of the output.

S0	S1	Output Frequency Scaling
L	L	Power down
L	H	2%
H	L	20%
H	H	100%

S2	S3	Photodiode Type
L	L	Red
L	H	Blue
H	L	Clear (no filter)
H	H	Green

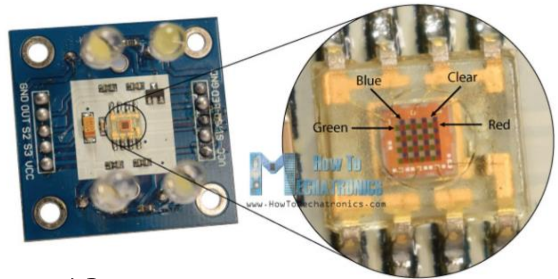
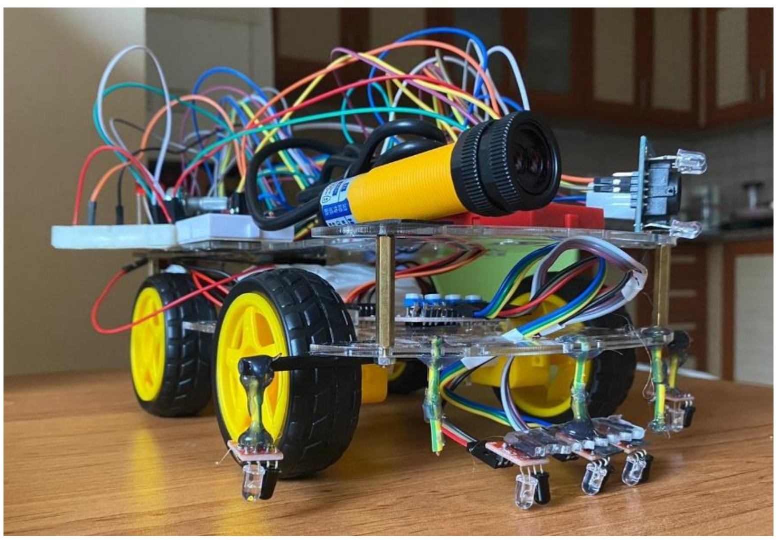
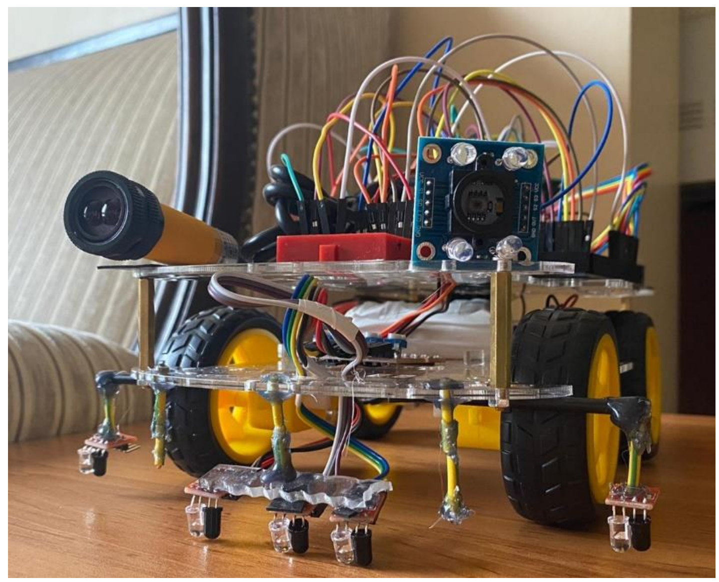


Figure 4.3

The color sensor (TCS3200) in the project reads the traffic lights and moves the car according to the color of the light. For example, if the car sees and detects red light while driving, the car cuts off the power to the engine and stops the car. Meanwhile, two red brake lights at the back are lit. Whenever the color of the traffic light turns green, the car moves again and continues on its way.





5. Discussion

This robot, realized and made in this project, is very open to development. For example, using rasberry pi and various camera systems, a robot can be designed similar to robots found at Amazon warehouse. Robots in Amazon can systematically transport the desired objects to the desired location with the help of the information they receive from the database. While doing these, all robots work systematically and do not hit each other.

When starting this project, the main purpose was to establish an artificial intelligence infrastructure together with image processing technology using a camera. However, this idea was abandoned due to the short project delivery time and the price of the equipment to be used. In the future, this idea will be tried to be implemented and the project will be developed.

6. Appendices

Two different Arduino's are used in this project. One of them is "Car Control" and the other is "Traffic Lights Control".

6.1 Car Control

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(9, 10, 11, 12, 13, 34);

#define SensorLeft 36
#define SensorRight 44
#define SensonFront 40

#define RightMotorForward 4
#define RightMotorBackward 8
#define RightMotorEnable 3

#define LeftMotorForward 6
#define LeftMotorBackward 7
#define LeftMotorEnable 5

#define ColorSensorS0 22
#define ColorSensorS1 24
#define ColorSensorS2 26
#define ColorSensorS3 28
#define ColorSensorOut 30

int RedFrequency, GreenFrequency, BlueFrequency  = 0;

#define InfraredData 53

int InfraredValue = 0;

#define LEDLeft 46
#define LEDRight 48

#define BuzzerHorn 2
```

Figure 6.1

First LCD's library has been added to the code. The library has started with the pin numbers connected to our Arduino. Identify the (left-middle-right) line sensors in the car and the pins (YL-70) connected to the Arduino. Identify the pins of the motors that will control the correct wheels connected to the Arduino (L298N).4 engines were used. Since it is not desired to control all of them separately, they connect the right and left motors to each other so that they move synchronously. The color sensor is defined by the pins we connect to the Arduino (TCS3200) located at the front of the car. Due to the working principle of the sensor, variables are defined for 3 main colors. We add an insignificant value when defining. We define the pin (MZ80) that we connect the infrared sensor to the Arduino at the top of the car. Variables are defined in order to be able to read the data received from the sensor. We add an insignificant value when defining. In addition, the LEDs we connect to the back of our car and the pins we connect to the Arduino are defined. Finally, we define the pin to which we connect the bell to be warned in case of an emergency. Other functions of these stages are that we want to use the names we have determined instead of the pin numbers we connect one by one. These operations make the changes more convenient. You can see the codes I mentioned in figure 6.1.

```
void setup() {

  lcd.begin(16, 2);

  Serial.begin(9600);

  pinMode(SensorLeft, INPUT);
  pinMode(SensorRight, INPUT);
  pinMode(SensonFront, INPUT);

  pinMode(RightMotorForward, OUTPUT);
  pinMode(RightMotorBackward, OUTPUT);
  pinMode(RightMotorEnable, OUTPUT);

  pinMode(LeftMotorForward, OUTPUT);
  pinMode(LeftMotorBackward, OUTPUT);
  pinMode(LeftMotorEnable, OUTPUT);

  pinMode(ColorSensorS0, OUTPUT);
  pinMode(ColorSensorS1, OUTPUT);
  pinMode(ColorSensorS2, OUTPUT);
  pinMode(ColorSensorS3, OUTPUT);
  pinMode(ColorSensorOut, INPUT);

  digitalWrite(ColorSensorS0, HIGH);
  digitalWrite(ColorSensorS1, LOW);

  pinMode(InfraredData, INPUT);

  pinMode(LEDLeft, OUTPUT);
  pinMode(LEDRight, OUTPUT);

  pinMode(BuzzerHorn, OUTPUT);
}
```

Figure 6.2

In the setup section, first of all, determined the number of columns and rows in the LCD which are used. Then initiate serial communication at 9600 bps to see the results while experimenting with colors. Project members determine the input-output information of the components use. Defined the lane tracking sensors, infrared sensor and the out pin of our color sensor as the input. Project members define motor driver, the pins of color sensors, buzzer, and LEDs for warning as outputs, which will enable the movement of the remaining motors. In addition, defined the S0 and S1 pins as HIGH and LOW respectively, so that can adjust the output frequency of color sensor. You can see the codes mentioned in figure 6.2.

```
void loop() {

  silencethehorn();

  describepedestrian();
}
```

Figure 6.3

Since having so many different commands in the project, created only two commands in the loop section by defining them in different functions. Project members thought this was better for further improvements and other developers. These are the functions silencethehorn () and describepedestrian (). You can see the codes mentioned in figure 6.3.

```
void stopspeed(){
  digitalWrite(RightMotorForward, HIGH);
  digitalWrite(RightMotorBackward, LOW);

  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(LeftMotorBackward, LOW);

  analogWrite(RightMotorEnable, 0);
  analogWrite(LeftMotorEnable, 0);
}

void dontchangespeed(){
  digitalWrite(RightMotorForward, HIGH);
  digitalWrite(RightMotorBackward, LOW);

  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(LeftMotorBackward, LOW);

  analogWrite(RightMotorEnable, 70);
  analogWrite(LeftMotorEnable, 70);

}

void turnright(){
  digitalWrite(RightMotorForward, HIGH);
  digitalWrite(RightMotorBackward, LOW);

  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(LeftMotorBackward, LOW);

  analogWrite(RightMotorEnable, 0);
  analogWrite(LeftMotorEnable, 70);
}

void turnleft(){
  digitalWrite(RightMotorForward, HIGH);
  digitalWrite(RightMotorBackward, LOW);

  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(LeftMotorBackward, LOW);

  analogWrite(RightMotorEnable, 70);
  analogWrite(LeftMotorEnable, 0);
}
```

Figure 6.4

First of all, defined the motion functions of the car. These functions used the activation pin, which determines the power going to the vehicle's engines. First, stopspeed () function. In this function, the minimum value is written on the activation pins.

Then write function, which is the opposite of stopspeed () function, which makes our tool moving forward. In the dontchangespeed () function, some power is supplied to the activation pins. By trying this value changed as you want. Since do not have a large area and to be able to conduct a controlled experiment, this value was given to 70.

Next are the return functions. In the turnright () function, power the left motors and pull the power from the right motor to 0. In turnright () function, the left motors are powered up and the power is pulled from the right motor to 0. You can see the codes mentioned in Figure 6.4.

```
void soundthehorn() {
  tone(BuzzerHorn, 500);
}

void silencethehorn() {
  noTone(BuzzerHorn);
}
```

Figure 6.5

Two functions have been added to control the buzzer. One of them is the soundthehorn () function, which activates the buzzer. Using the tone () command, make a sound in the frequency range we want. Then, with the noTone () command in the slincethehorn () function, the buzzer is silenced. You can see the codes mentioned in Figure 6.5.

```
void calibrating() {

  digitalWrite(ColorSensorS2, LOW);
  digitalWrite(ColorSensorS3, LOW);

  RedFrequency = pulseIn(ColorSensorOut, LOW);

  Serial.print("Red= ");
  Serial.print(RedFrequency);
  Serial.print(" - ");
  delay(200);

  digitalWrite(ColorSensorS2, HIGH);
  digitalWrite(ColorSensorS3, HIGH);

  GreenFrequency = pulseIn(ColorSensorOut, LOW);

  Serial.print("Green= ");
  Serial.print(GreenFrequency);
  Serial.print(" - ");
  delay(200);

  digitalWrite(ColorSensorS2, LOW);
  digitalWrite(ColorSensorS3, HIGH);

  BlueFrequency = pulseIn(ColorSensorOut, LOW);

  Serial.print("Blue= ");
  Serial.println(BlueFrequency);
  delay(200);

}

void calibrated(){

  digitalWrite(ColorSensorS2, LOW);
  digitalWrite(ColorSensorS3, LOW);

  RedFrequency = pulseIn(ColorSensorOut, LOW);

  digitalWrite(ColorSensorS2, HIGH);
  digitalWrite(ColorSensorS3, HIGH);

  GreenFrequency = pulseIn(ColorSensorOut, LOW);

  digitalWrite(ColorSensorS2, LOW);
  digitalWrite(ColorSensorS3, HIGH);

  BlueFrequency = pulseIn(ColorSensorOut, LOW);
}
```

Figure 6.6

Next is the functions for the color sensor. The calibrating () function is written so that can only try and read the frequencies want. If give LOW to the s2 and s3 pins of sensor, it allows us to read the red values. Using the pulseIn () command, synchronize it to the RedFrequency variable to determined at the beginning.

If set the pins s2 and s3 HIGH, it allows us to read the green values. Using the PulseIn () command it synchronizes with the GreenFrequency variable to determine at startup.

If give LOW and HIGH to pins s2 and s3 respectively, it allows us to read the blue values. Using the pulseIn () command, we synchronize it to the BlueFrequency variable we determined at the beginning.

Use the Serial.print () command to see values, including our variables, on the serial port screen. The reason we do these operations is to be able to read the value range of the light we show.

Turn on the calibrated () function. The purpose of this function is a necessary function for our next function. From here assigns the values of the colors to the variables. You can see the codes mentioned in Figure 6.6.

```
void moveaccordingtocolors() {

 calibrated();

 if (RedFrequency > BlueFrequency and RedFrequency < GreenFrequency and GreenFrequency > BlueFrequency) {

  if (RedFrequency < 400 and 0 < RedFrequency){

   lcd.setCursor(0, 0);
   lcd.print("Red light");
   lcd.setCursor(0, 1);
   lcd.print("    STOP    ");

   stopspeed();

   digitalWrite(LEDLeft, HIGH);
   digitalWrite(LEDRight, HIGH);

   delay(100);
  }

  lcd.clear();
 }

 if (GreenFrequency < RedFrequency and GreenFrequency > BlueFrequency and BlueFrequency < RedFrequency) {

  if (GreenFrequency < 400 and 0 < GreenFrequency){

   dontchangespeed();

   lcd.setCursor(0, 0);
   lcd.print("Green light");
   lcd.setCursor(0, 1);
   lcd.print("    GO ON    ");

   digitalWrite(LEDLeft, LOW);
   digitalWrite(LEDRight, LOW);
  }

  lcd.clear();
 }
}
```

Figure 6.7

When the desired colors (red, green) are displayed on the sensor, we write our function, which we determine what we want it to do. By calling this function, we get the values read by the variables. The sensor is wanted to detect only the red color. We organize transactions to take place at regular intervals. We set the cursor to column 0, line 0, and column 0, line 1. and we print a message to the LCD on these lines. The code that stops our vehicle because it reads the red light is added. We turn on the LEDs as a warning to the vehicle behind. A little delay is added to make it work better. The screen is cleared before printing a new reply. You can see the codes mentioned in Figure 6.7.

```
void describepedestrian(){

  InfraredValue = digitalRead(InfraredData);

  if (InfraredValue == 0){

    lcd.setCursor(0, 0);
    lcd.print("There is a");
    lcd.setCursor(0, 1);
    lcd.print("pedestrian.");

    stopspeed();

    digitalWrite(LEDLeft, HIGH);
    digitalWrite(LEDRight, HIGH);

    soundthehorn();

    lcd.clear();
  }
```

Figure 6.8

The write function is added to the function we want the pedestrian to do when he sees it. This describepedestrian () function is the function that determines the entire movement of the vehicle. Then, when the value received from the sensor is converted into the assigned variable, the process continues. Returns 0 if a pedestrian is detected. so it enters the desired if command. Then it prints the message on the LCD that we added. Immediately after, it cuts the power in the car and stops it. LEDs added for warning purposes become active and give a warning sound with the soundthehorn () function. Finally, the message printed on the LCD is deleted. You can see the codes mentioned in Figure 6.8.

```
  else if (InfraredValue == 1){

    silencethehorn();

    digitalWrite(LEDLeft, LOW);
    digitalWrite(LEDRight, LOW);

    if(digitalRead(SensonFront) == 0){
     dontchangespeed();

     moveaccordingtocolors();
    }

    if (digitalRead(SensorLeft) == 0 or digitalRead(SensorRight) == 0) {

      if (digitalRead(SensorLeft) == 0) {

        lcd.setCursor(0, 0);
        lcd.print("The car");
        lcd.setCursor(0, 1);
        lcd.print("turns right.");

        turnright();

        moveaccordingtocolors();

        if (digitalRead(SensorLeft) == 1){
          stopspeed();
        }

      delay(10);
      }

      lcd.clear();

      if (digitalRead(SensorRight) == 0) {

        lcd.setCursor(0, 0);
        lcd.print("The car");
        lcd.setCursor(0, 1);
        lcd.print("turns left.");

        turnleft();

        moveaccordingtocolors();
      }
        if (digitalRead(SensorRight) == 1){
          stopspeed();
        }
      }

      lcd.clear();
   }
 }
```

Figure 6.9

Returns 1 if no pedestrian is detected. First of all, an active horn is muted. Then its LEDs are turned off. It is checked whether your vehicle can read the line. If the vehicle reads the lane, the engines are powered up. In meanwhile the color sensor reads the values and gives the necessary inputs with this function mentioned earlier. The red or green light is controlled.

It is checked whether the vehicle is off the road or not. If the left sensor reads the value, what needs to be done is written in the if block. Its cursor is set as column 0, row 0 and column 0, row 1, and a message is printed on the LCD over these lines. The vehicle is turned to the right. Meanwhile, the color sensor is reading the values. It takes other actions depending on whether the red or green light is controlled. If the sensor starts reading black again, it will stop the car. If the middle sensor cannot enter the road with this command, the vehicle can be controlled. Some lag has been added to make it run better. The screen is cleared before printing a new response.

It is checked whether the vehicle is off the road or not. If the correct sensor reads the value, what needs to be done is written in the if block. The cursor is set as column 0, row 0 and column 0, row 1, and a message is printed on the LCD through these lines. The car is turned to the left. Meanwhile, the color sensor is reading the values. It takes other actions depending on whether the red or green light is controlled. If the sensor starts to read black again, stop the car. If the middle sensor cannot enter the road with this command, we can control the vehicle. Some lag has been added to make it run better. The screen is cleared before printing a new response. You can see the codes mentioned in Figure 6.9.

In this image, you can see the connection scheme of our circuit.

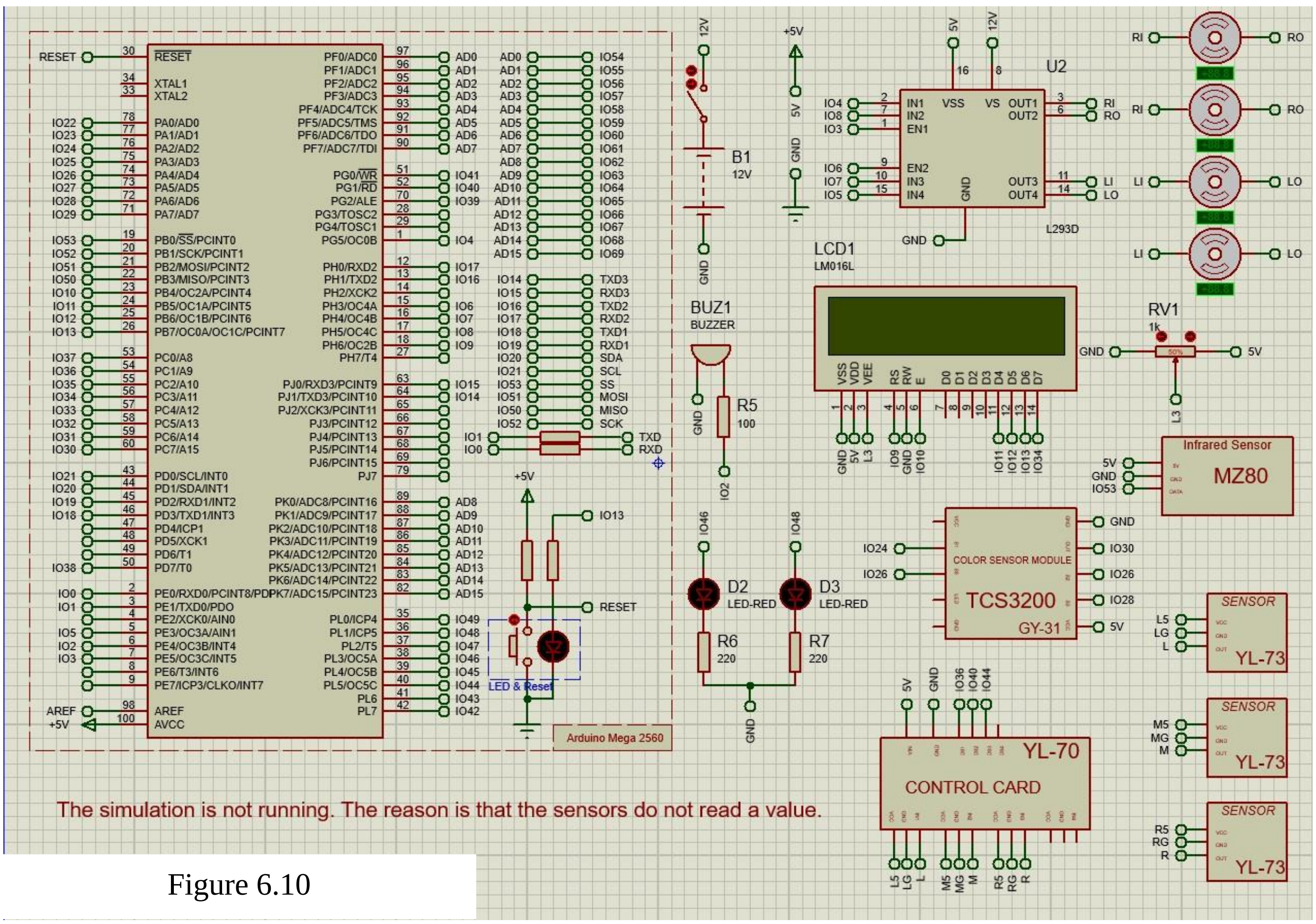


Figure 6.10

6.2 Traffic Lights

```
#define RGBred1 3
#define RGBgreen1 5
#define RGBblue1 6

#define RGBred2 9
#define RGBgreen2 10
#define RGBblue2 11
```

Figure 6.11

First, pins connected to RGB LEDs are defined. Their links were defined for the green and red colors, respectively. You can see the codes mentioned in Figure 6.11.

```
void setup() {

 pinMode(RGBred1,OUTPUT);
 pinMode(RGBgreen1,OUTPUT);
 pinMode(RGBblue1,OUTPUT);

 pinMode(RGBred2,OUTPUT);
 pinMode(RGBgreen2,OUTPUT);
 pinMode(RGBblue2,OUTPUT);
}
```

Figure 6.12

In the setup section, the output information of RGB LEDs is determined. You can see the codes mentioned in Figure 6.12.

```
void loop() {

 analogWrite(RGBred1,255);
 analogWrite(RGBgreen1,0);
 analogWrite(RGBblue1,255);
 analogWrite(RGBred2,255);
 analogWrite(RGBgreen2,255);
 analogWrite(RGBblue2,255);

 delay(20000);

 analogWrite(RGBred1,255);
 analogWrite(RGBgreen1,255);
 analogWrite(RGBblue1,255);
 analogWrite(RGBred2,0);
 analogWrite(RGBgreen2,255);
 analogWrite(RGBblue2,255);

 delay(20000);
}
```

Figure 6.13

In the loop part, we turn on the green light first, not the red light. Then we turn off the green light and turn on the red light. A waiting period of 20 seconds was added to these processes. Thus, traffic light is made. You can see the codes mentioned in Figure 6.13.

In this visual, you can see the connection diagram of our circuit.

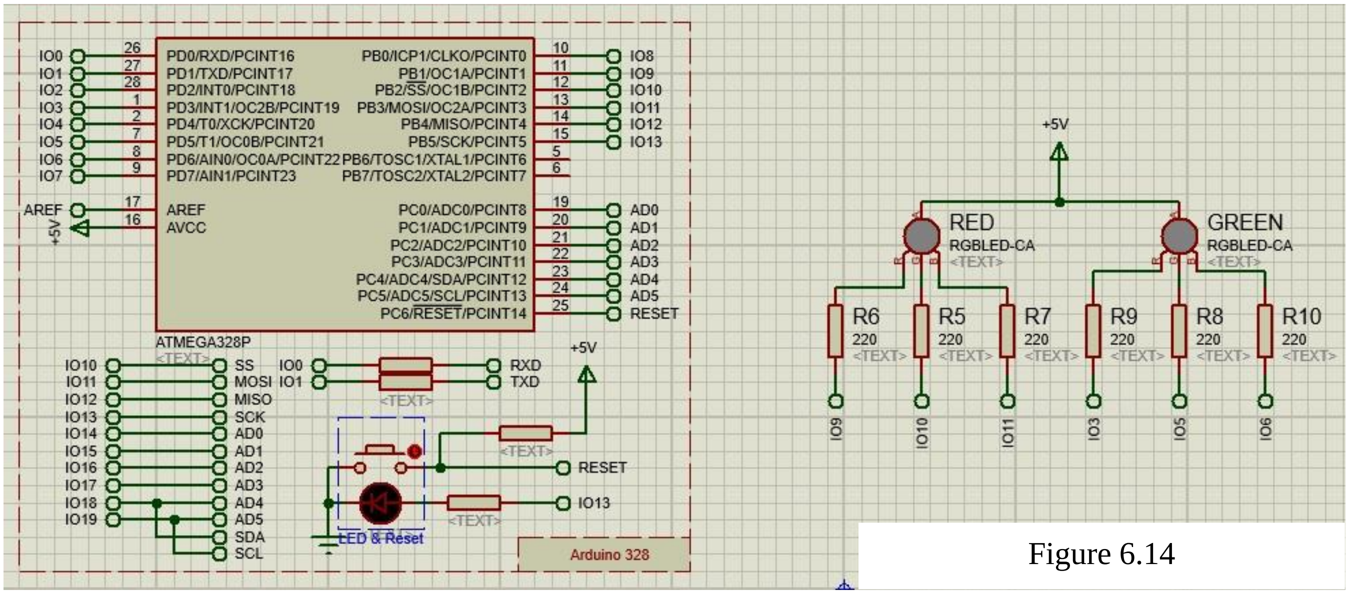
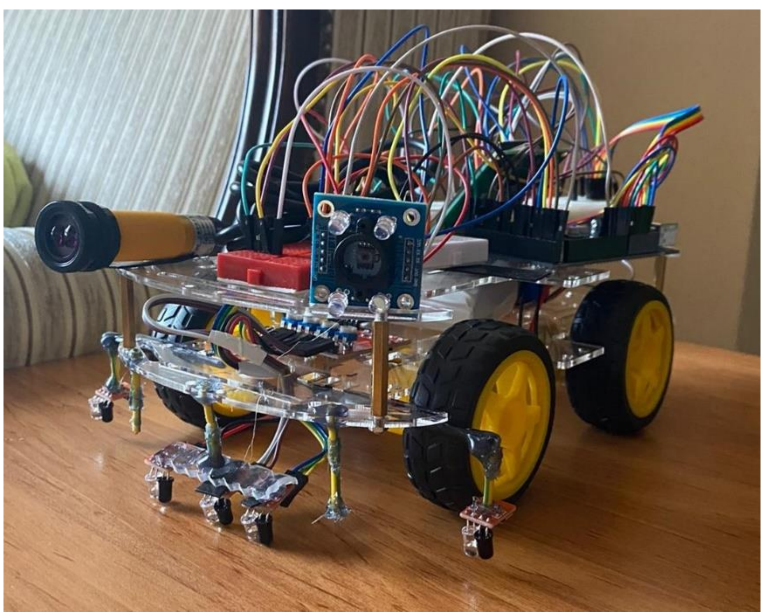
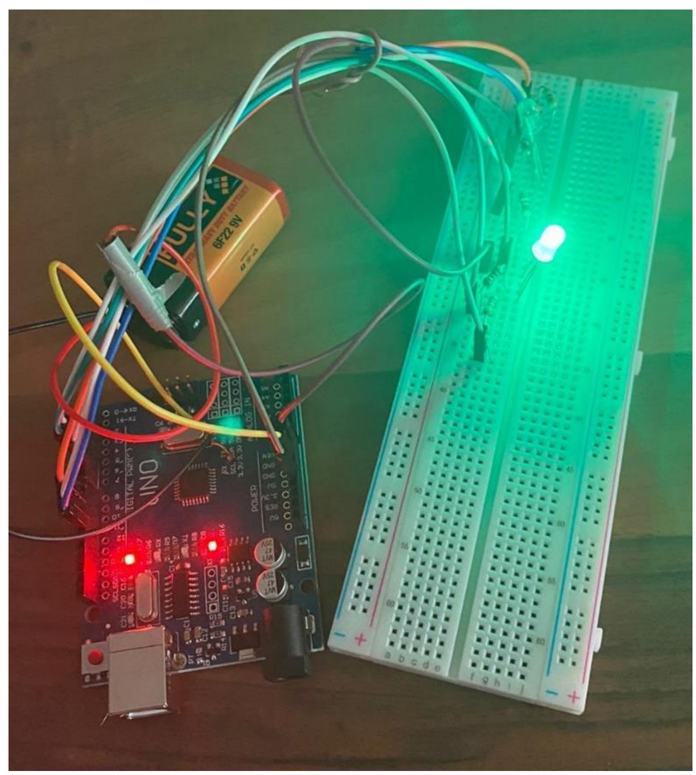


Figure 6.14





7. Literature Cited

Figure 3.1: https://store.arduino.cc/usa/mega-2560-r3
https://www.google.com/search?q=ARDUINO+MEGA+2560+REV3&tbm=isch&ved=2ahUKEwiAveqSv9ztAhWByKQKHXukCs0Q2-cCegQIABAA&oq=ARDUINO+MEGA+2560+REV3&gs_lcp=CgNpbWcQDFDy5QVY8uUFYPfvBWgAcAB4AIABhAGIAYQBkgEDMC4xmAEAoAEBqgELZ3dzLXdpei1pbWfAAQE&sclient=img&ei=fzrfX8CBGYGRkwX7yKroDA&bih=937&biw=958&rlz=1C1CHBD_trTR908TR908#imgrc=41BTQpVR1aPx4M

Figure 3.2: https://store.arduino.cc/usa/arduino-uno-rev3

https://www.google.com/search?q=ARDUINO+UNO+REV3&rlz=1C1CHBD_trTR908TR908&sxsrf=ALeKk00qzY2MyqGL1HbRdV18HowkCC9V8g:1608465021810&source=lnms&tbm=isch&sa=X&ved=2ahUKEwj4q4iSv9ztAhUQ4aQKHRtHBV8Q_AUoAXoECAYQAw&biw=958&bih=937#imgrc=bf64WmAfNx-DpM

Figure 3.3: https://www.robotistan.com/l298n-voltaj-regulatorlu-cift-motor-surucu-karti

Figure 3.4:
https://www.google.com/search?q=K%C4%B1z%C4%B1l%C3%B6tesi+Sens%C3%B6r+80cm+(MZ80+Infrared+Sensor+E18-D80NK)+Y%C3%BCksek+Kaliteli+Model+(KS132)&tbm=isch&ved=2ahUKEwi6jL7Bv9ztAhWHyKQKHZpaDYcQ2-cCegQIABAA&oq=K%C4%B1z%C4%B1l%C3%B6tesi+Sens%C3%B6r+80cm+(MZ80+Infrared+Sensor+E18-D80NK)+Y%C3%BCksek+Kaliteli+Model+(KS132)&gs_lcp=CgNpbWcQDFCqlwVYqpcFYIeeBWgAcAB4AIABeIgBeJIBAzAuMZgBAKABAaoBC2d3cy13aXotaW1nwAEB&sclient=img&ei=4TrfX_qXD4eRkwWatbW4CA&bih=937&biw=958&rlz=1C1CHBD_trTR908TR908#imgrc=VAEyLx5ViWrwEM

Figure 3.5/3.6: http://www.energiazero.org/arduino_sensori/4_channel_line_tracker_sensor.pdf

Figure 3.8/4.3: https://howtomechatronics.com/tutorials/arduino/arduino-color-sensing-tutorial-tcs230-tcs3200-color-sensor/

Figure 3.7:
https://www.google.com/search?q=color+sensor+tcs3200&tbm=isch&ved=2ahUKEwiDo5KdwdztAhUNOuwKHQHeCygQ2-

cCegQIABAA&oq=color+sensor+T&gs_lcp=CgNpbWcQARgBMgQIABATMgQIABATMggIABAFEB4QEzIICAAQCBAeEBMyCAgAEAgQHhATMggIABAIEB4QEzIICAAQCBAeEBMyCAgAEAgQHhATMggIABAIEB4QEzIICAAQCBAeEBM6BAgjECc6CAgAELEDEIMBOgUIABCxAzoCCAA6BAgAEB5QlLcHWPDFB2Cc1wdoAHAAeACAAc8BiAHWEJIBBjAuMTMuMZgBAKABAaoBC2d3cy13aXotaW1nwAEB&sclient=img&ei=rTzfX8OYN430sAeBvK_AAg&bih=937&biw=958&rlz=1C1CHBD_trTR908TR908#imgrc=-TTK1rc9tIAxXM

Figure 3.9:
https://www.google.com/search?q=3s+bms&tbm=isch&ved=2ahUKEwjP9o7awdztAhUB76QKHcBQAfsQ2-cCegQIABAA&oq=3s+&gs_lcp=CgNpbWcQARgAMgQIIxAnMgQIIxAnMgIIADICCAAyAggAMgIIADICCAAyAggAMgIIADICCAA6BQgAELEDUKTFBFjU0ARgyNwEaABwAHgAgAGtAYgB3gOSAQMwLjOYAQCgAQGqAQtnd3Mtd2l6LWltZ8ABAQ&sclient=img&ei=LT3fX4-sL4HekwXAoYXYDw&bih=937&biw=958&rlz=1C1CHBD_trTR908TR908#imgrc=-dD-TLtBFV6ogM

Figure 3.10:
https://www.google.com/search?q=16x2+LCD&tbm=isch&ved=2ahUKEwiVsoqAwtztAhUH26QKHd_pCSsQ2-cCegQIABAA&oq=16x2+LCD&gs_lcp=CgNpbWcQAzICCAAyAggAMgIIADICCAAyAggAMgIIADICCAAyAggAMgQIABAeMgQIABAeOgQIIxAnOggIABCxAxCDAToFCAAQsQNQlP0BWKyQAmCdkgJoAHAAeACAAZUBiAHLCJIBAzAuOJgBAKABAaoBC2d3cy13aXotaW1nwAEB&sclient=img&ei=fT3fX9X_F4e2kwXf06fYAg&bih=937&biw=958&rlz=1C1CHBD_trTR908TR908#imgrc=DLFG81e6nMiAMM

Figure 3.11:
https://www.google.com/search?q=18650+Battery&tbm=isch&ved=2ahUKEwjTq7CRwtztAhWFgKQKHZ8QCXcQ2-cCegQIABAA&oq=18650+Battery&gs_lcp=CgNpbWcQDDICCAAyAggAMgIIADICCAAyAggAMgQIABAeMgQIABAeMgQIABAeMgQIABAeMgQIABAeUKeyAViXvQFg6cMBaABwAHgAgAGRAYgBkAKSAQMwLjKYAQCgAQGqAQtnd3Mtd2l6LWltZ8ABAQ&sclient=img&ei=oT3fX5PXKIWBkgWfoaS4Bw&bih=937&biw=958&rlz=1C1CHBD_trTR908TR908#imgrc=JXMeDuLz1XgN9M

Figure 3.12:
https://www.google.com/search?q=dc+motor&tbm=isch&ved=2ahUKEwi_tvqdwtztAhXIm6QKHe3yD84Q2-cCegQIABAA&oq=dc+motor&gs_lcp=CgNpbWcQAzIECCMQJzIHCAAQsQMQQzIECAAQQzICCAAyAggAMgIIADICCAAyAggAMgIIADICCAA6BQgAELEDOggIABCxAxCDAVDWpQ9Yia0PYMSuD2gAcAB4AIABmAGIAY0IkgEDMC44mAEAoAEBqgELZ3dzLXdpei1pbWfAAQE&sclient=img&ei=vD3fX7_oAsi3kgXt5b_wDA&bih=937&biw=958&rlz=1C1CHBD_trTR908TR908#imgrc=EFG57I_T_y7YQM

Figure 3.13:
https://www.google.com/search?q=220+ohm+resistance&tbm=isch&ictx=1&tbs=rimg:CXhfjuVbwU0jIgh4X47lW8FNIyoSCXhfjuVbwU0jEZVz_195K-cZz&hl=tr&sa=X&ved=2ahUKEwjqm6yJxNztAhUUOuwKHT6fCowQiRx6BAgAEAQ&biw=941&bih=937#imgrc=TchhCVZ-rX7WwM

Figure 3.14:
https://www.google.com/search?q=10k%20ohm%20resistance&tbm=isch&ictx=1&tbs=rimg:CR1owPEAT046IggdaMDxAE9OOioSCR1owPEAT046Eb3wzhVuejTx&hl=tr&sa=X&ved=0CAEQiRxqFwoTCKCx5frD3O0CFQAAAAAdAAAAABAE&biw=941&bih=937#imgrc=lUusORQE6wrHfM

Figure 3.15:
https://www.google.com/imgres?imgurl=https%3A%2F%2Fwww.twinschip.com%2Fimage%2Fcache%2Fcatalog%2FProducts%2520Twins%2520Chip%2520Store%25202020%2F100%2520Ohm%2520Resistor%2520X%25205%2520Pieces%2F100%2520Ohm%2520Resistor%2520X%25205%2520Pieces%2520-%2520Twins%2520Chip%25201-600x315w.jpg&imgrefurl=https%3A%2F%2Fwww.twinschip.com%2F100_Ohm_Resistor&tbnid=zwK40AumNIZJ8M&vet=12ahUKEwj92sqUnNztAhWOtKQKHeXuD2sQMygAegUIARDEAQ..i&docid=2K30VCCYhgWLZM&w=600&h=315&q=100%20ohm%20&ved=2ahUKEwj92sqUnNztAhWOtKQKHeXuD2sQMygAegUIARDEAQ

Figure 3.16:
https://www.google.com/search?q=buzzer+arduino&rlz=1C1CHBD_trTR908TR908&sxsrf=ALeKk033kJTEHdoKDl-

tg50cYYJm7v5yOA:1608467351092&source=lnms&tbm=isch&sa=X&ved=2ahUKEwi5teDox9ztAhVimMMKHSWgDAMQ_AUoAnoECBIQBA&biw=958&bih=937#imgrc=NbjkOaIgiDr6mM

Figure 3.17:
https://www.google.com/search?q=potansiyometre&rlz=1C1CHBD_trTR908TR908&hl=tr&sxsrf=ALeKk03obIEQtHNGZiLTcF6tjvNaIK919Q:1608467898326&source=lnms&tbm=isch&sa=X&ved=2ahUKEwiXh9ntydztAhVC2KQKHZdEC_IQ_AUoAXoECBIQAw&biw=1920&bih=947#imgrc=imVo0ectB1UDzM

Figure 3.18:
https://www.google.com/search?q=k%C4%B1rm%C4%B1z%C4%B1+led&tbm=isch&ved=2ahUKEwjOnKPwytztAhWR-qQKHT8ZAi0Q2-cCegQIABAA&oq=k%C4%B1rm%C4%B1z%C4%B1+led&gs_lcp=CgNpbWcQAzICCAAyAggAMgIIADICCAAyAggAMgIIADICCAAyAggAMgIIADICCAA6BQgAELEDOgQIABBDOgYIABAHEB5Q7SZY0S5g2TZoAHAAeACAAYEBiAHKB5IBAzAuOJgBAKABAaoBC2d3cy13aXotaW1nwAEB&sclient=img&ei=zEbfX86GCpH1kwW_sojoAg&bih=947&biw=1920&rlz=1C1CHBD_trTR908TR908&hl=tr#imgrc=BSciJv1D9WkuKM

Figure 3.19:
https://www.google.com/search?q=rgb+led%2C&tbm=isch&ved=2ahUKEwiXu5X0ytztAhVTP-wKHS7qDOYQ2-cCegQIABAA&oq=rgb+led%2C&gs_lcp=CgNpbWcQAzIECCMQJzIECCMQJzIHCAAQsQMQQzICCAAyAggAMgIIADICCAAyAggAMgIIADICCABQ57oGWJHABmDFwAZoAHAAeACAAY4BiAGOAZIBAzAuMZgBAKABAaoBC2d3cy13aXotaW1nwAEB&sclient=img&ei=1EbfX5eBFNP-sAeu1LOwDg&bih=947&biw=1920&rlz=1C1CHBD_trTR908TR908&hl=tr#imgrc=egd8Ff5lvDRFQM

Figure 3.20:
https://www.google.com/search?q=switch+anahtar&tbm=isch&ved=2ahUKEwig7c3dy9ztAhUD7qQKHWtnCtEQ2-cCegQIABAA&oq=switch+&gs_lcp=CgNpbWcQARgBMgQIABBDMgIIADICCAAyBQgAELEDMgIIADIECAAQQzIECAAQQzICCAAyBAgAEEMyBAgAEENQniFYniFggjRoAHAAeACAAYEBiAGBAZIBAzAuMZgBAKABAaoBC2d3cy13aXotaW1nwAEB&sclient=img&ei=sUffX-

DQG4PckwXrzqmIDQ&bih=947&biw=1920&rlz=1C1CHBD_trTR908TR908&hl=tr#imgrc=ALh0aWTns2GiVM

Figure 4.1: https://www.youtube.com/watch?v=RDD1OktBrLM

Figure 4.2: https://www.google.com/search?q=how+to+work++Infrared+Sensor+&tbm=isch&ved=2ahUKEwiktd3yxNztAhVPuKQKHTMGDzkQ2-cCegQIABAA&oq=how+to+work++Infrared+Sensor+&gs_lcp=CgNpbWcQA1DC9ApY5osLYNuMC2gAcAB4AIABhQGIAfsBkgEDMC4ymAEAoAEBqgELZ3dzLXdpei1pbWfAAQE&sclient=img&ei=hkDfX6TaKs_wkgWzjLzIAw&bih=937&biw=941&hl=tr#imgrc=xhUbMon7iM7dEM

https://pages.mtu.edu/~suits/notefreqs.html

https://techzeero.com/arduino-tutorials/buzzer-with-arduino/

https://www.instructables.com/How-to-use-a-Buzzer-Arduino-Tutorial/

https://www.instructables.com/How-to-Make-a-3S-18650-Battery-Pack/

https://www.robolinkmarket.com/16x2-lcd-ve-i2c-aparati-yesil?gclid=Cj0KCQiA5vb-BRCRARIsAJBKc6J2EiQG3XLa1cE-j_rDmS-ITccuEfP7odqItXCfKp8VUXJqKlIxxs8aAnwNEALw_wcB

https://www.18650batterystore.com/collections/18650-batteries/products/samsung-20s

https://www.fierceelectronics.com/sensors/what-ir-sensor#:~:text=Active%20infrared%20sensors%20both%20emit%20and%20detect%20infrared%20radiation.&text=When%20an%20object%20comes%20close,(such%20as%20in%20robots)

https://robu.in/ir-sensor-working/#:~:text=IR%20sensor%20is%20an%20electronic,some%20form%20of%20thermal%20radiation.

https://www.youtube.com/watch?v=er_i-MaEoKU

https://maker.robotistan.com/

https://gomulumuhendis.blogspot.com/

https://create.arduino.cc/projecthub/muhammad-aqib/arduino-rgb-led-tutorial-fc003e