Project No&Topic: Design a car mobile robot via motion of your body parts

Name-Surname: Ayana Tleulenova No: 160412059
Burak Berke Çanaklı 160412023
Bedircan Kara 160412017
Group No : A (11:50 -12:05)

Introduction: The main idea of this project is to creat a car with an obstacle detection module which we can control via body part. We will use the MPU -6050 Gyro Accelerometer sensor to give the direction to the car, Arduino Nano for glove and Arduino Uno for car. When the car recognizes an obstacle our buzer will make a sound.

Equipment:

A **DC motor is** any of a class of rotary electrical machines that converts direct current electrical energy into mechanical energy.

L298N Driver: It is a dual H-Bridge motor driver which allows speed and direction control of two DC motors at the same time. The module can drive DC motors that have voltages between 5 and 35V, with a peak current up to 2A.

nRF24L01: is a single chip radio transceiver for the world wide 2.4 - 2.5 GHz ISM band. The transceiver consists of a fully integrated frequency synthesizer, a power amplifier, a crystal oscillator, a demodulator, modulator and Enhanced ShockBurst™ protocol engine.

Arduino Nano: comes with a crystal oscillator of frequency 16 MHz. It is **used** to produce a clock of precise frequency using constant voltage.

3.7 v 3000mah li-ion battery: is a type of rechargeable battery. Lithium-ion batteries are commonly used for portable electronics and electric vehicles and are growing in popularity for military and aerospace applications.

MPU-6050 Gyro Accelerometer sensor: are used in self-balancing robots, UAVs, smartphones, and more. IMU sensors help us get the position of an object attached to the sensor in three-dimensional space. These values are usually in angles to help us to determine its position.

Software:

The code of the glove:

```
#include <SPI.h>       //SPI library for communicate with the nRF24L01+

#include "RF24.h"      //The main library of the nRF24L01+

#include "Wire.h"      //For communicate

#include "I2Cdev.h"    //For communicate with MPU6050

#include "MPU6050.h"   //The main library of the MPU6050

//Define the object to access and cotrol the Gyro and Accelerometer (We don't use the Gyro data)

MPU6050 mpu;

int16_t ax, ay, az;

int16_t gx, gy, gz;

//Define packet for the direction (X axis and Y axis)

int data[2];

//Define object from RF24 library - 9 and 10 are a digital pin numbers to which signals CE and CSN are connected.

RF24 radio(9,10);

//Create a pipe addresses for the communicate

const uint64_t pipe = 0xE8E8F0F0E1LL;

void setup(void){

  Serial.begin(9600);

  Wire.begin();

  mpu.initialize();              //Initialize the MPU object

  radio.begin();                 //Start the nRF24 communicate

  radio.openWritingPipe(pipe);   //Sets the address of the receiver to which the program will send data.

}

void loop(void){

  //With this function, the acceleration and gyro values of the axes are taken.

  //If you want to control the car axis differently, you can change the axis name in the map command.
  mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);


  //In two-way control, the X axis (data [0]) of the MPU6050 allows the robot to move forward
and backward.
  //Y axis (data [0]) allows the robot to right and left turn.
  data[0] = map(ax, -17000, 17000, 300, 400 ); //Send X axis data
  data[1] = map(ay, -17000, 17000, 100, 200);  //Send Y axis data
  radio.write(data, sizeof(data));
}
```

The code of the car:

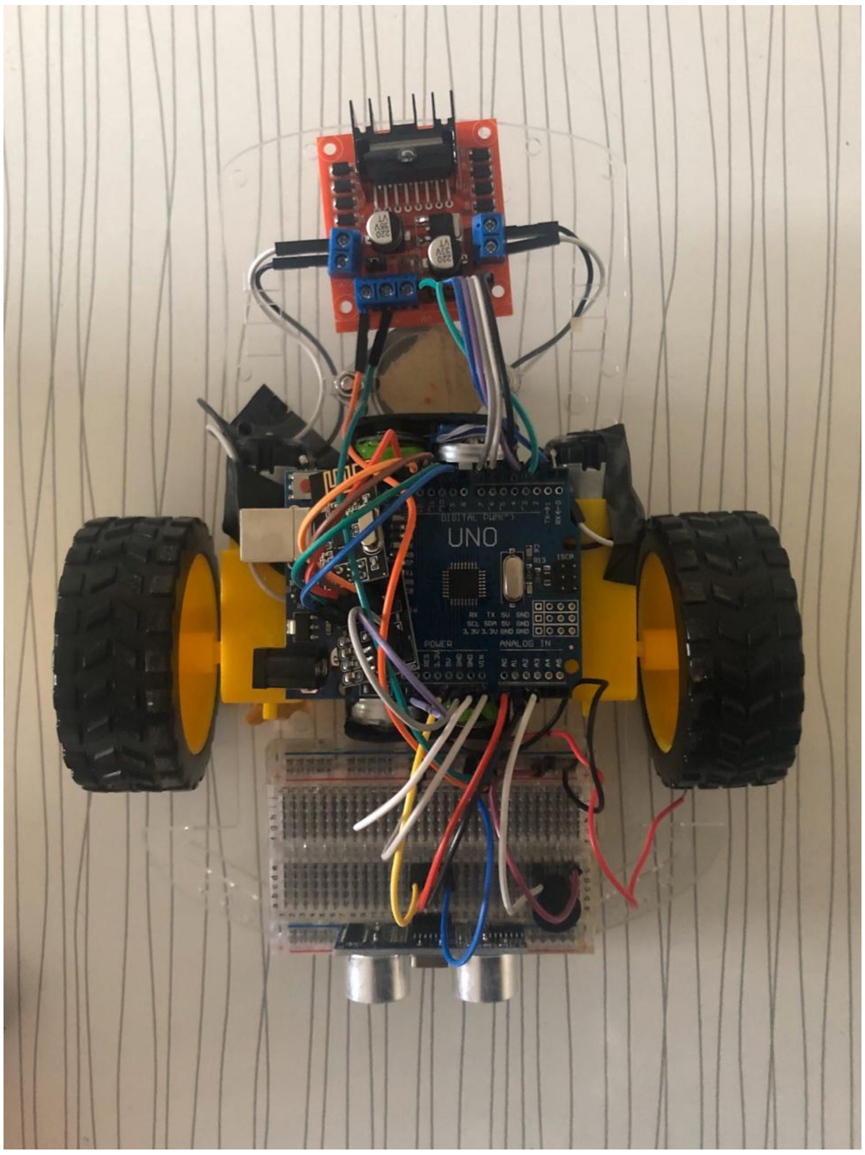
```
#include <SPI.h>      //SPI library for communicate with the nRF24L01+
#include "RF24.h"     //The main library of the nRF24L01+
//Define enable pins of the Motors
const int enbA = 3;
const int enbB = 5;
const int trigPin = A1 ;
const int echoPin = A0;
const int buzzer = A2;
//Define control pins of the Motors
//If the motors rotate in the opposite direction, you can change the positions of the following
pin numbers
const int IN1 = 2;    //Right Motor (-)
const int IN2 = 4;    //Right Motor (+)
const int IN3 = 7;    //Left Motor (+)
const int IN4 = 6;    //Right Motor (-)
//Define variable for the motors speeds
//I have defined a variable for each of the two motors
```

```
//This way you can synchronize the rotation speed difference between the two motors

int RightSpd = 120;

int LeftSpd = 145;

long duration;

int distance;

int safetyDistance;

//Define packet for the direction (X axis and Y axis)

int data[2];

//Define object from RF24 library - 9 and 10 are a digital pin numbers to which signals CE and
CSN are connected

RF24 radio(9,10);

//Create a pipe addresses for the communicate

const uint64_t pipe = 0xE8E8F0F0E1LL;

void setup(){

  //Define the motor pins as OUTPUT

  pinMode(enbA, OUTPUT);

  pinMode(enbB, OUTPUT);

  pinMode(IN1, OUTPUT);

  pinMode(IN2, OUTPUT);

  pinMode(IN3, OUTPUT);

  pinMode(IN4, OUTPUT);

 pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

pinMode(buzzer, OUTPUT);

  Serial.begin(9600);
  radio.begin();                  //Start the nRF24 communicate

  radio.openReadingPipe(1, pipe);   //Sets the address of the transmitter to which the program
will receive data.

  radio.startListening();

  }

void loop()

{

// Clears the trigPin

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance= duration*0.034/2;

safetyDistance = distance;

if (safetyDistance <= 20){

  digitalWrite(buzzer, HIGH);

}

else{

  digitalWrite(buzzer, LOW);

}
```

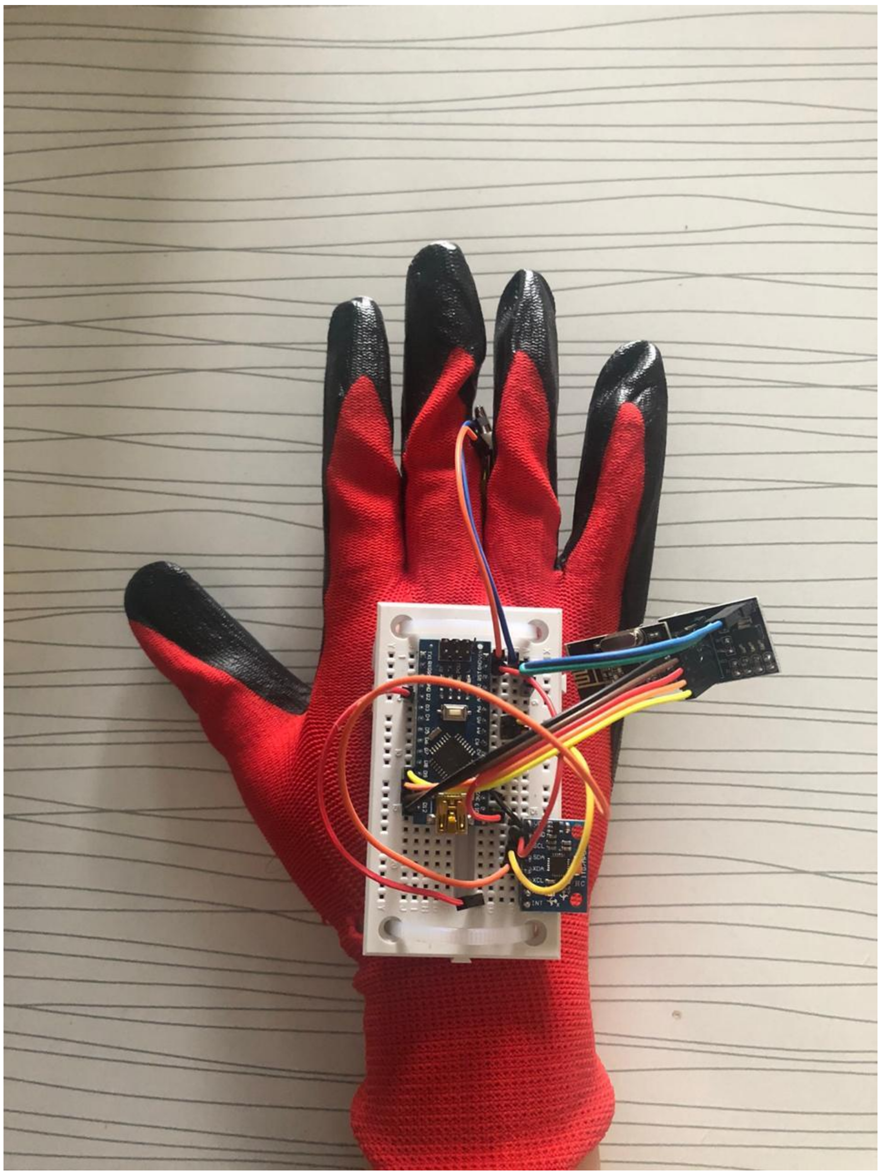
```
// Prints the distance on the Serial Monitor

Serial.print("Distance: ");

Serial.println(distance);

 if (radio.available()){

    radio.read(data, sizeof(data));

if(data[0] > 380){

      //forward

      analogWrite(enbA, RightSpd);

      analogWrite(enbB, LeftSpd);

      digitalWrite(IN1, HIGH);

      digitalWrite(IN2, LOW);

      digitalWrite(IN3, HIGH);

      digitalWrite(IN4, LOW);

    }

    if(data[0] < 310){

      //backward

      analogWrite(enbA, RightSpd);

      analogWrite(enbB, LeftSpd);

      digitalWrite(IN1, LOW);

      digitalWrite(IN2, HIGH);

      digitalWrite(IN3, LOW);

      digitalWrite(IN4, HIGH);

    }

    if(data[1] > 180){
      //left
      analogWrite(enbA, RightSpd);
      analogWrite(enbB, LeftSpd);
      digitalWrite(IN1, LOW);
      digitalWrite(IN2, HIGH);
      digitalWrite(IN3, HIGH);
      digitalWrite(IN4, LOW);
    }
 if(data[1] < 110){
      //right
      analogWrite(enbA, RightSpd);
      analogWrite(enbB, LeftSpd);
      digitalWrite(IN1, HIGH);
      digitalWrite(IN2, LOW);
      digitalWrite(IN3, LOW);
      digitalWrite(IN4, HIGH);
    }
if(data[0] > 330 && data[0] < 360 && data[1] > 130 && data[1] < 160){
      //stop car
      analogWrite(enbA, 0);
      analogWrite(enbB, 0);
    }
  }
}
```

Design and Assembly:

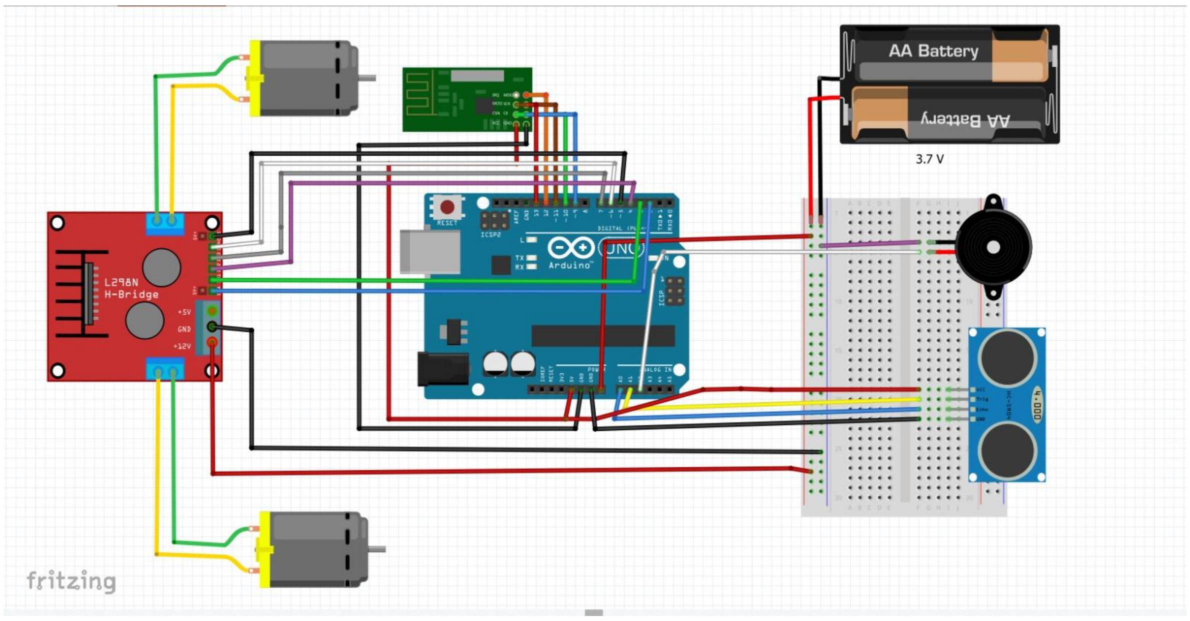
Here is the photo of the car:



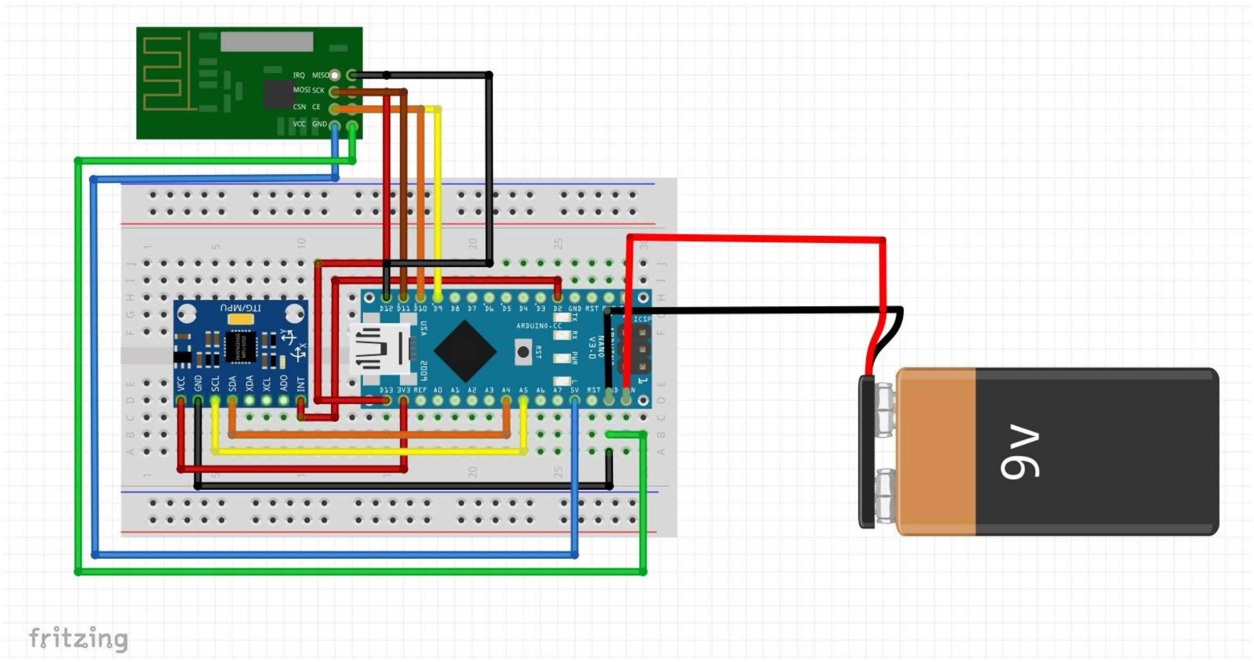
Here is the photo of the glove:



Design of the car:



Design of the glove:



Results and Discussion:

Firstly we chose the equipment for the project and then we began our process with designing an electric circuit in Fritzing program. Folllowing that, we wrote codes for Arduino for both car and glove. Somehow, it faced problems with downloading the libraries to the Arduino. Furthermore, we had power problems, so we used li-ion batteries, so we can recharge them. Moreover, we broke our Gyro Accelerometer sensor twice while we were soldering it, since it was heating up. Also, our left motor works slowlier than right motor, so chane the speed code according to it. That`s why we our left speed 145 and right 120.