Mechatronics Lab (ME69036)

Project report

on

Hand Gesture Controlled Car

Department of Mechanical Engineering

Indian Institute of Technology, Kharagpur

Submitted to: -

Prof. CS Kumar

Prof. AK Samantaray

Prof. Mihir Sarangi

Submitted by

Sambhav Prashit kaphle(21ME63A01)

R Lohit Raj (21ME63D01)

Mayank Lakhera (21ME63R07)

Pankaj Suhas Pandit(21ME63R08)

Kanhu Hansdah(21ME63R01)

ACKNOWLEDGEMENT

We would like to thank sincerely, our mentor Prof. C S Kumar, Prof. AK Samantaray and Prof. Mihir Sarangi, for his invaluable guidance, constant assistance, support, endurance, and constructive suggestions for the implementation and betterment of the project.

We would also like to thank Lab assistant (TA's) of Mechatronics lab for their help in completing this project.

Finally, we are also thankful to each other for our continued moral and material support throughout the preparation of this project.

Table of Contents:

Abstract

Hand Gesture Controlled Car is a robot which can be controlled by simple human gestures. The user just needs to wear a gesture device in which MPU6050 (Accelerometer cum Gyro) sensor is included. The sensor will record the movement of hand in a specific direction which will result in the motion of the robot in the respective directions. The robot and the Gesture instrument are connected wirelessly through radio waves. User can interact with the robot in a more friendly way due to the wireless communication. We can control the car using accelerometer sensors connected to a hand glove. The sensors are intended to replace the remote control that is generally used to run the car. It will allow user to control the forward, backward, leftward and rightward movements, while using the same accelerometer sensor to control the throttle of the car. Movement of car is controlled by the differential mechanism. The mechanism involves the rotation of both front & rear wheels of left or right side to move in the anticlockwise direction and the other pair to rotate in the clockwise direction which makes the car to rotate about its own axis without any kind of forward or backward motion. The main advantage of this mechanism is the car with this mechanism can take sharp turn without any difficulty.

Introduction

Nowadays, Robotics is becoming one of the most advanced in the field of technology. A Robot is an electro-mechanical system that is operated by a computer program. Robots can be autonomous or semi-autonomous. An autonomous robot is not controlled by human and acts on its own decision by sensing its environment. Majority of the industrial robots are autonomous as they are required to operate at high speed and with great accuracy. But some applications require semi-autonomous or human controlled robots. Some of the most commonly used control systems are voice recognition, tactile or touch controlled and motion controlled. A Gesture Controlled robot is a kind of robot which can be controlled by your hand gestures not by old buttons. You just need to wear a small transmitting device in your hand which included an acceleration meter. This will transmit an appropriate command to the robot so that it can do whatever we want. The transmitting device includes a nRF24L01 module which is (SPI protocol) used to send and receive data at ISM operating frequency from 2.4 to 2.5 GHz. This data is then processed by a microcontroller of Arduino Uno and finally our motor driver to control the motor's. Now it's time to break the task in different modules to make the task easy and simple any project become easy or error free if it is done in different modules. As our project is already divided into two different part transmitter and receiver. **The applications of robotics mainly involve in automobiles, medical, construction, defense and also used as a firefighting robot to help the people from the fire accident.** But, controlling the robot with a remote or a switch is quite complicated. So, a new project is developed that is, an accelerometer cum Gyroscopic based gesture control robot. The main goal of this project is to control the movement of the robot with hand gesture using MPU6050.The robot is usually an electro-mechanical machine that can perform tasks automatically. Some robots require some degree of guidance, which may be done using a remote control or with a computer interface. Robots have evolved so much and are capable of mimicking humans that they seem to have a mind of their own.

Major Components Required

Transmitter section

1. Arduino Pro Mini
2. MPU6050
3. nRF24L01
4. PCB
5. Battery

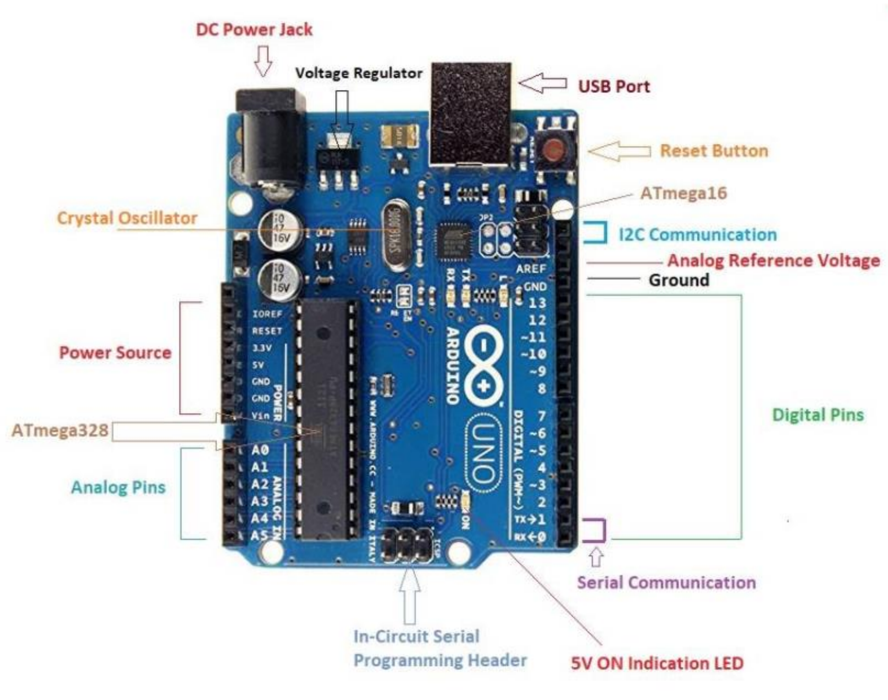
Receiver Section

1. nRF24L01

2. Arduino Uno
3. Battery
4. L298N Motor driver
5. Chassis and Wheel

Description

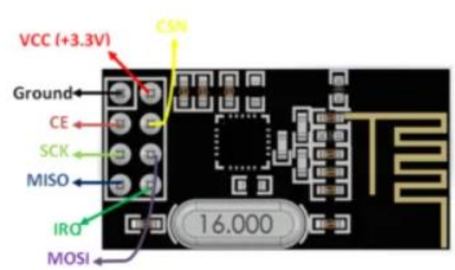
A. Arduino Uno



Arduino UNO

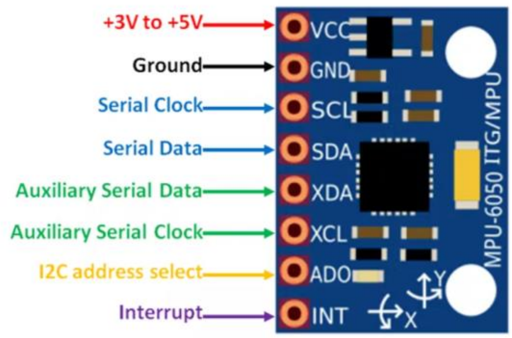
This is the brain of the car and is installed with some code. The Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ,206 quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. Arduino consists of both a physical programmable circuit board and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board. Arduino Uno can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators.

B. nRF24L01



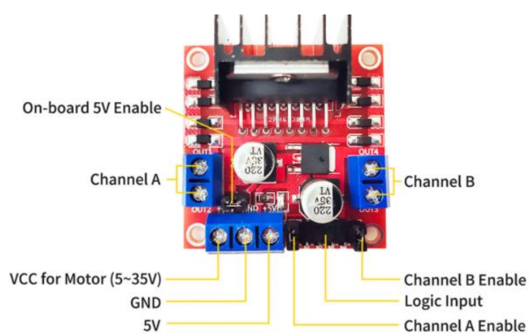
The nRF24L01 is wireless device commonly known as which receives or transmits data when interfaced with Arduino. It is based on SPI communication protocol with operating voltage of 3.3V and 50mA of current. It carries the band rate of 250kpds with 125channel range.There are many other RF module like 433 RF module. Modules are having 8 pins i.e. Ground, VCC, Chip Enable, Chip select Not, Serial clock, Master out slave in, Master in slave out.

C. MPU6050



MPU6050 is a Micro Electro-mechanical system (MEMS), it consists of three-axis accelerometer and three-axis gyroscope. It helps us to measure velocity, orientation, acceleration, displacement and other motion like features. MPU6050 consists of Digital Motion Processor (DMP), which has property to solve complex calculations. MPU6050 consists of a 16-bit analog to digital converter hardware. Due to this feature, it captures three-dimension motion at the same time. This module has some famous features which are easily accessible, due to its easy availability it can be used with a famous microcontroller like Arduino. Friend if you are looking for a sensor to control a motion of your Drone, Self-Balancing Robot, RC Cars and something like this, then MPU6050 will be a good choice for you.

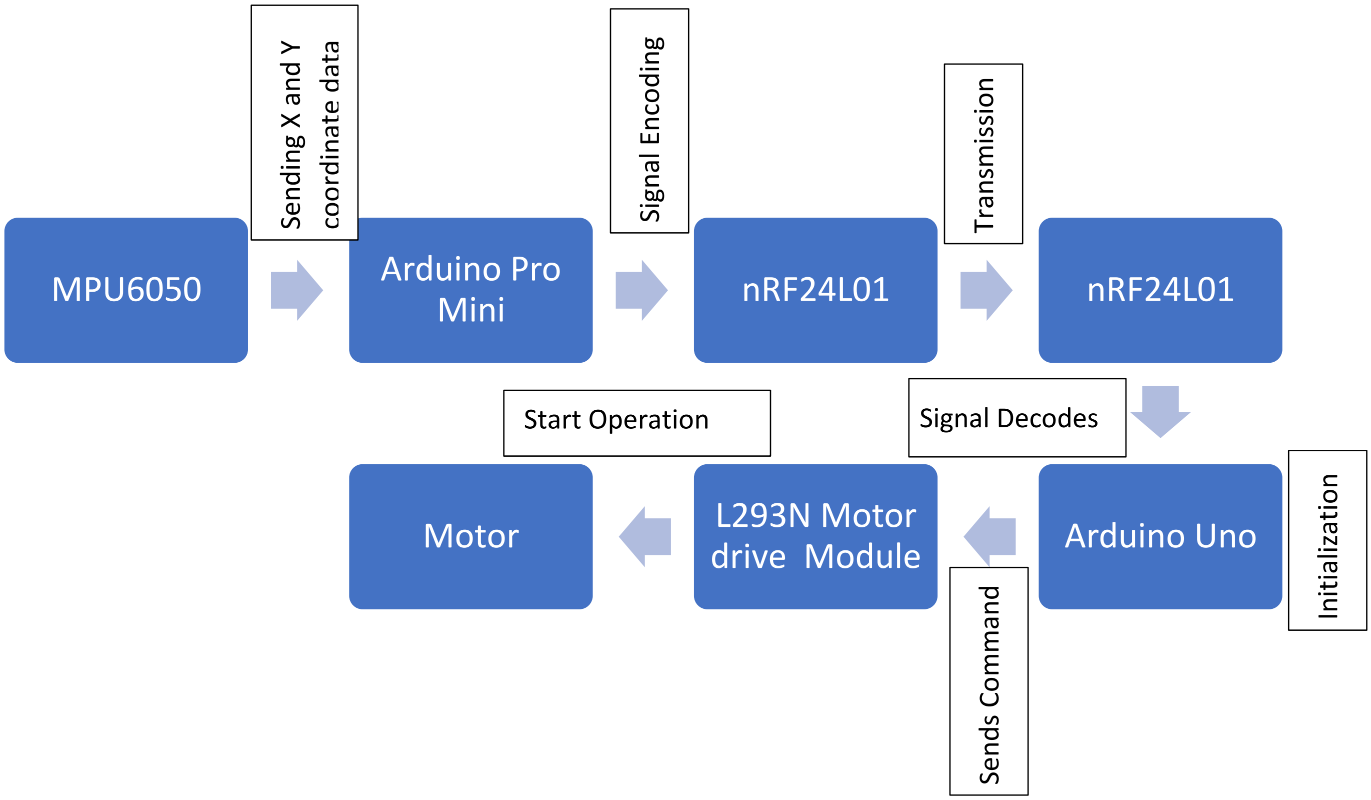
D. L293N Motor Driver module



This L298N Motor Driver Module is a high-power motor driver module for driving DC and Stepper Motors. This module consists of an L298 motor driver IC and a 78M05 5V regulator. L298N Module can control up to 4 DC motors, or 2 DC motors with directional and speed control.

Block Diagram

In this block diagram, referring to figure below, first Arduino is initialized and hand gesture (x and Y coordinates) are collected by Arduino using Gyroscopic and data encoder and transmitted by transmitter. On the receiving side, data received by receiver decodes it and send it to Arduino for interpretation. After interpretation, Arduino sends the necessary command to motor drive control for movement of Motors.

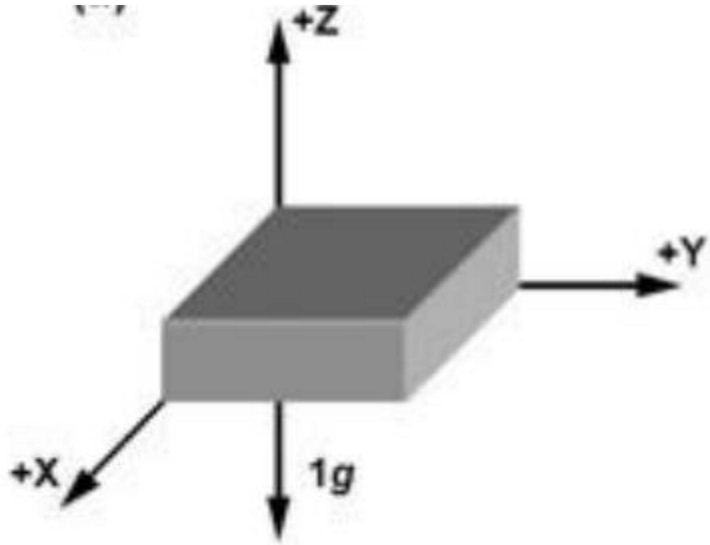


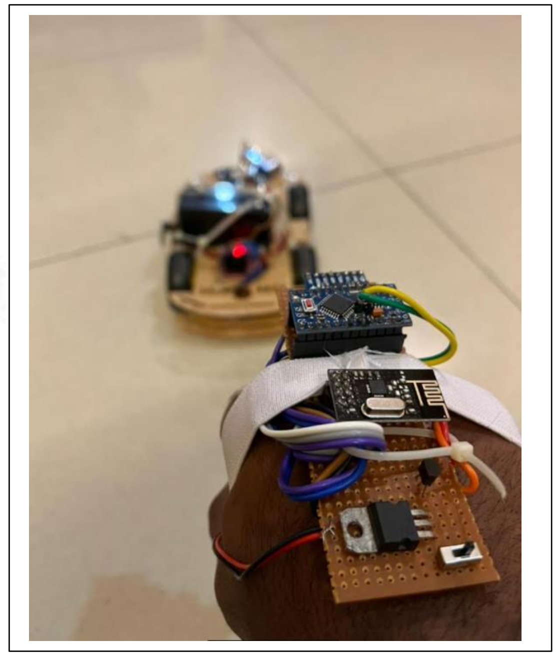
Methodology

The MPU6050 is kept on the palm of the user using gloves and the robot moves in steps with the palm movement. Here we have explained about 5 distinctive gestures role of the persons hand i.e. halt or stop condition, front moving, backward moving and turns towards right and left Stop condition

1. **Stop condition**

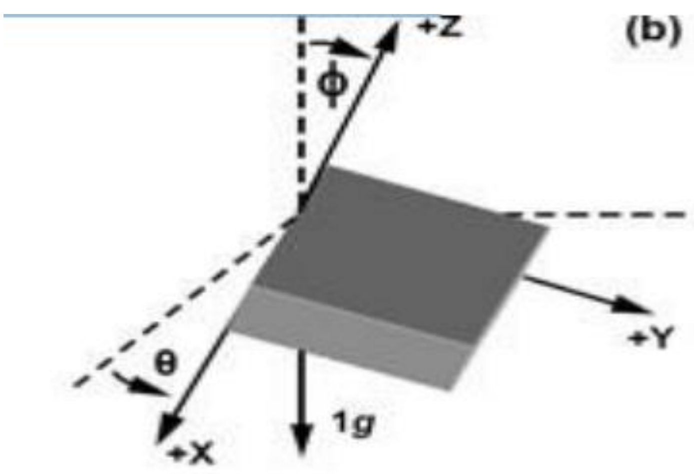
   The user holds the MPU6050 towards parallel to the ground. At that time the signal from the Mpu6050 is sent to the Arduino and the robot stop moving. This state is referred here as stop condition.

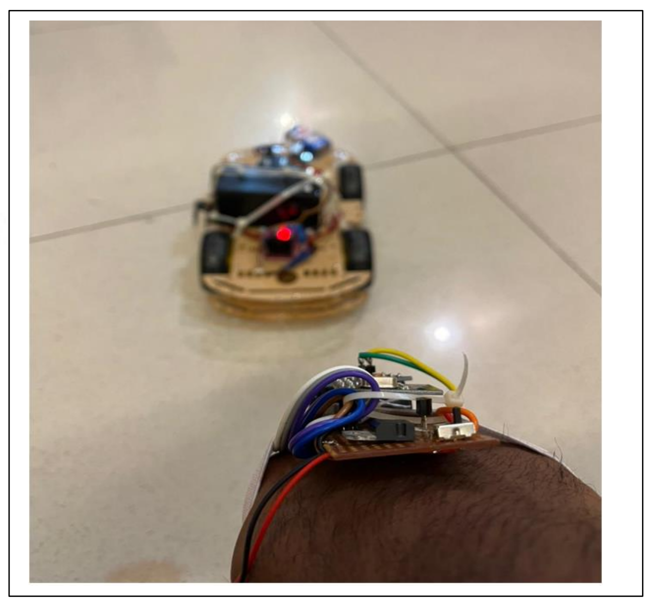




2. **Forward Tilt**

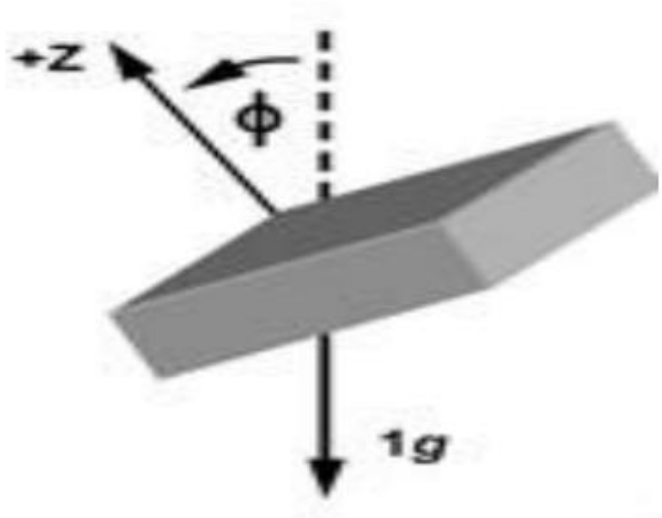
   4.3 FORWARD TILT: The user holds the accelerometer and the accelerometer tilted towards the front the X Y Z axis is sent to the Arduino. If the X, Y, Z axes satisfies the condition X<=-8000, the robot moves front.

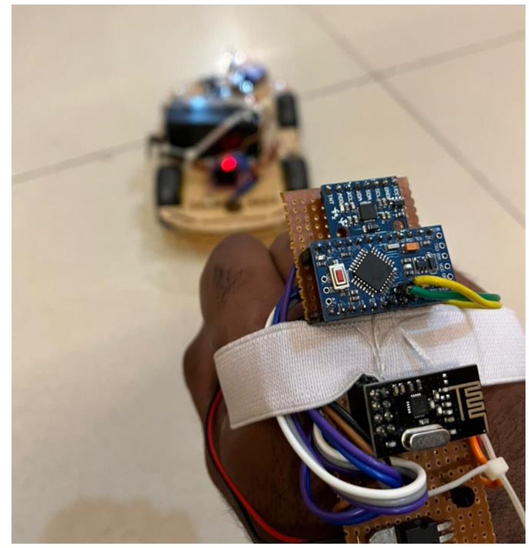




3. **Backward Tilt**

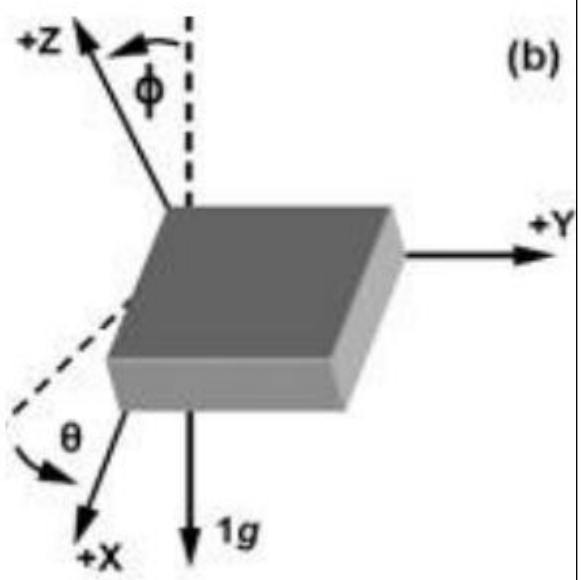
   The user holds the accelerometer and the accelerometer tilted towards the back the X Y Z axis is sent to the Arduino. If the X Y Z axes satisfies the condition x>=8000, the robot moves back.

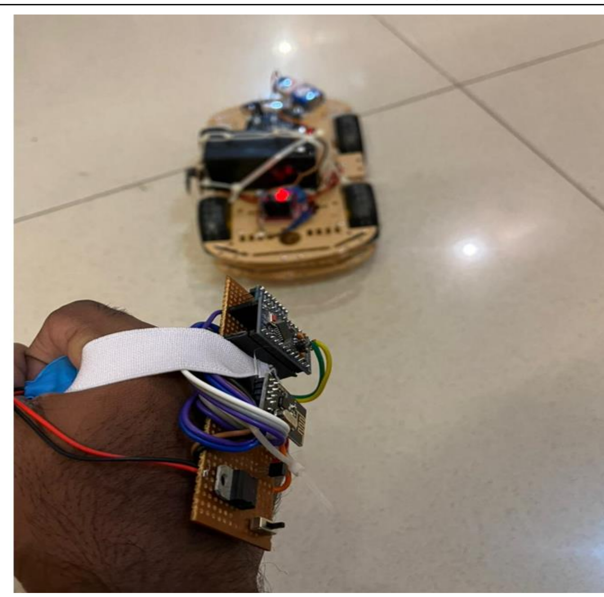




4. Right tilt

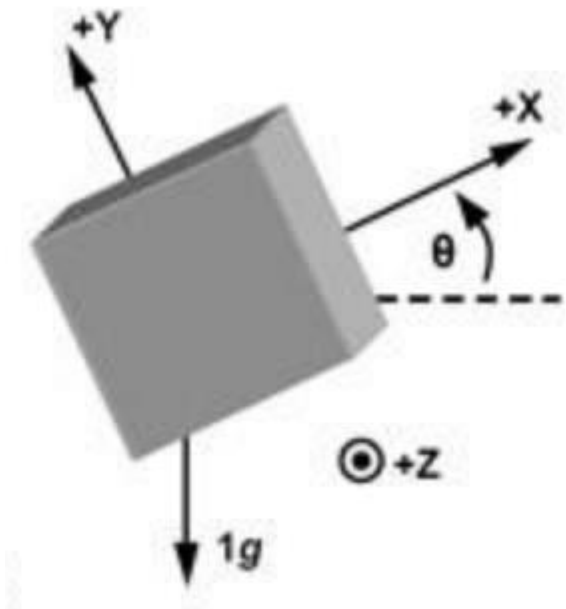
The user holds the accelerometer and the accelerometer tilted towards the right the X Y Z axis is sent to the Arduino. If the X Y Z axes satisfies the condition x>=8000 the robot moves right.





5. Left tilt

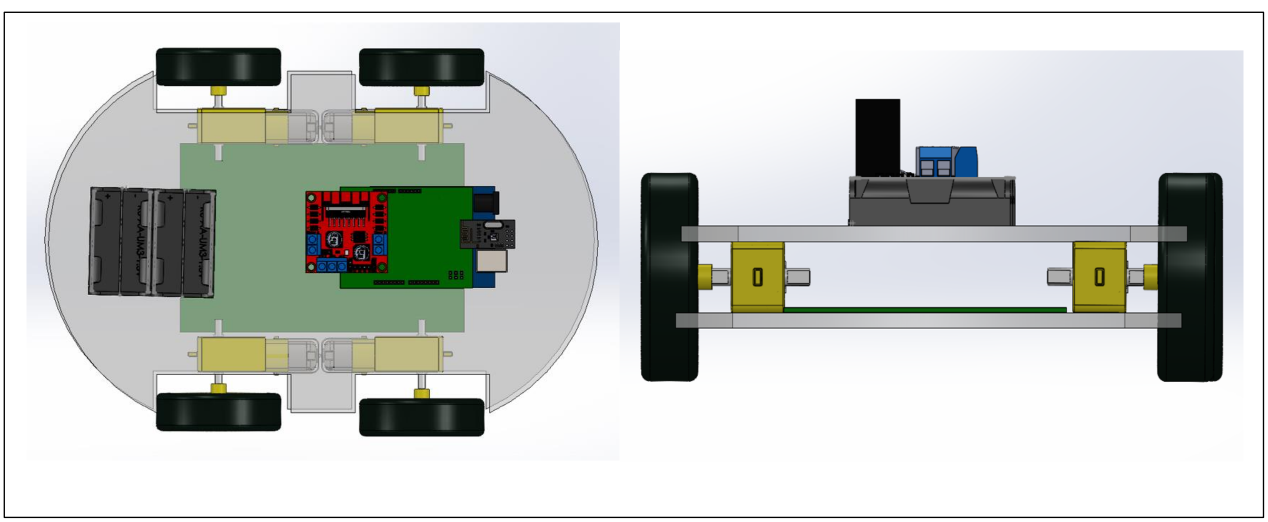
The user holds the accelerometer and the accelerometer tilted towards the left the X Y Z axis is sent to the Arduino. If the X Y Z axes satisfies the condition x<-8000, the robot moves left.

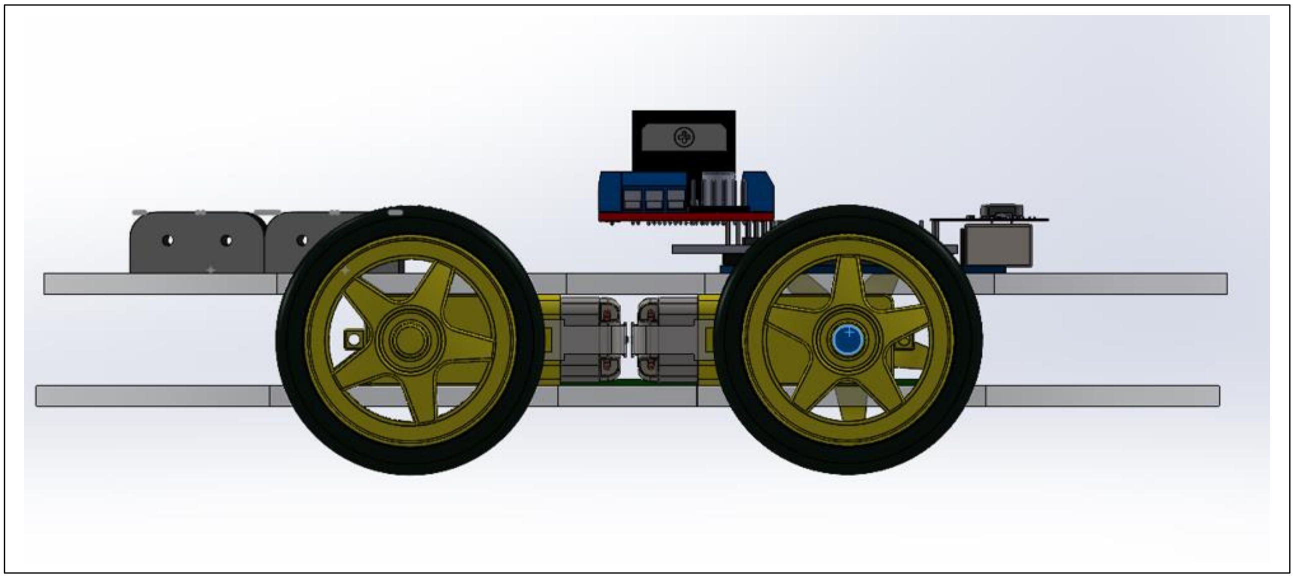


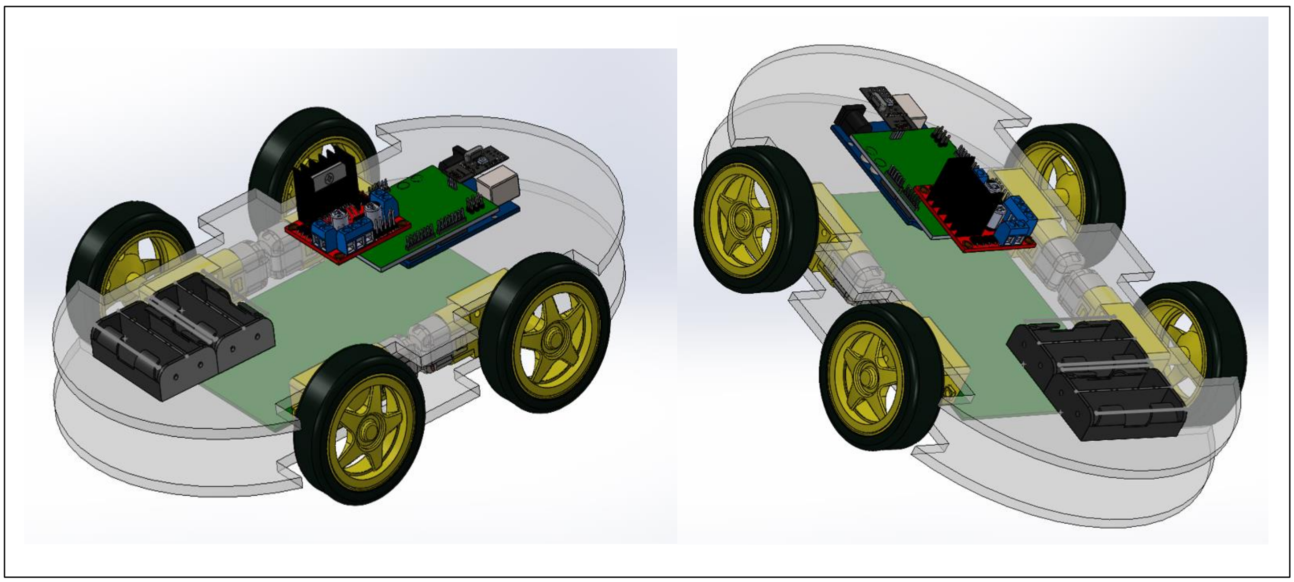
Working of the Car

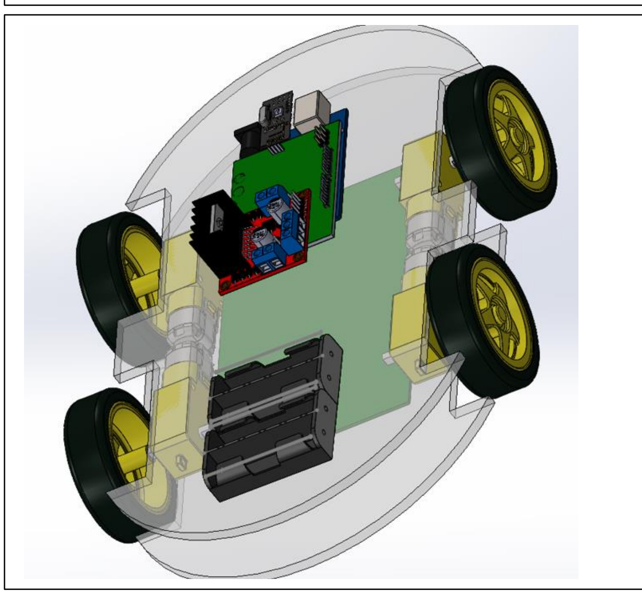
1. Here we use MPU6050 to transmit the axis value to the Arduino. The axis is changed according to the hand tilt of the user and the axis value is sent to the Arduino repeatedly.
2. If the user turns the MPU6050 towards front simultaneously the robot moves forward. If the user turns the MPU 6050 towards backward, simultaneously the robot moves backward.
3. If the user turns the MPU6050 toward left, simultaneously the robot turns left. If the user turns the MPU6050 towards right, simultaneously the robot turns right.
4. If the user keeps the MPU6050 parallel to the ground, simultaneously the robot stops moving. The movement of the robot is based on the tilting of the MPU6050.\

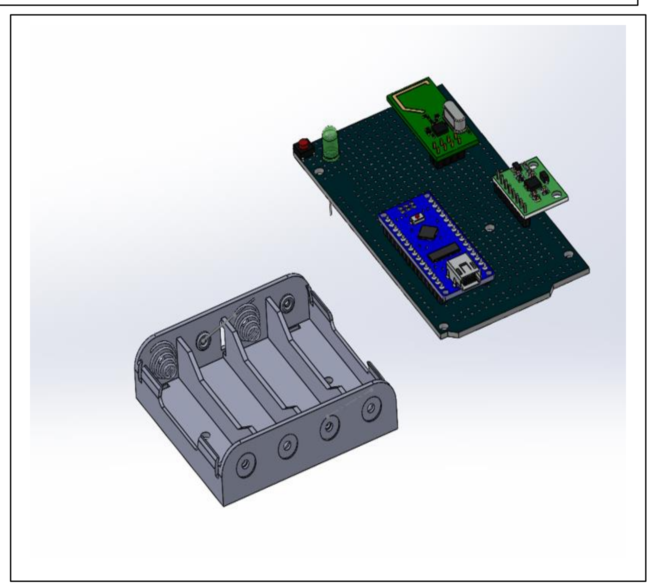
CAD model





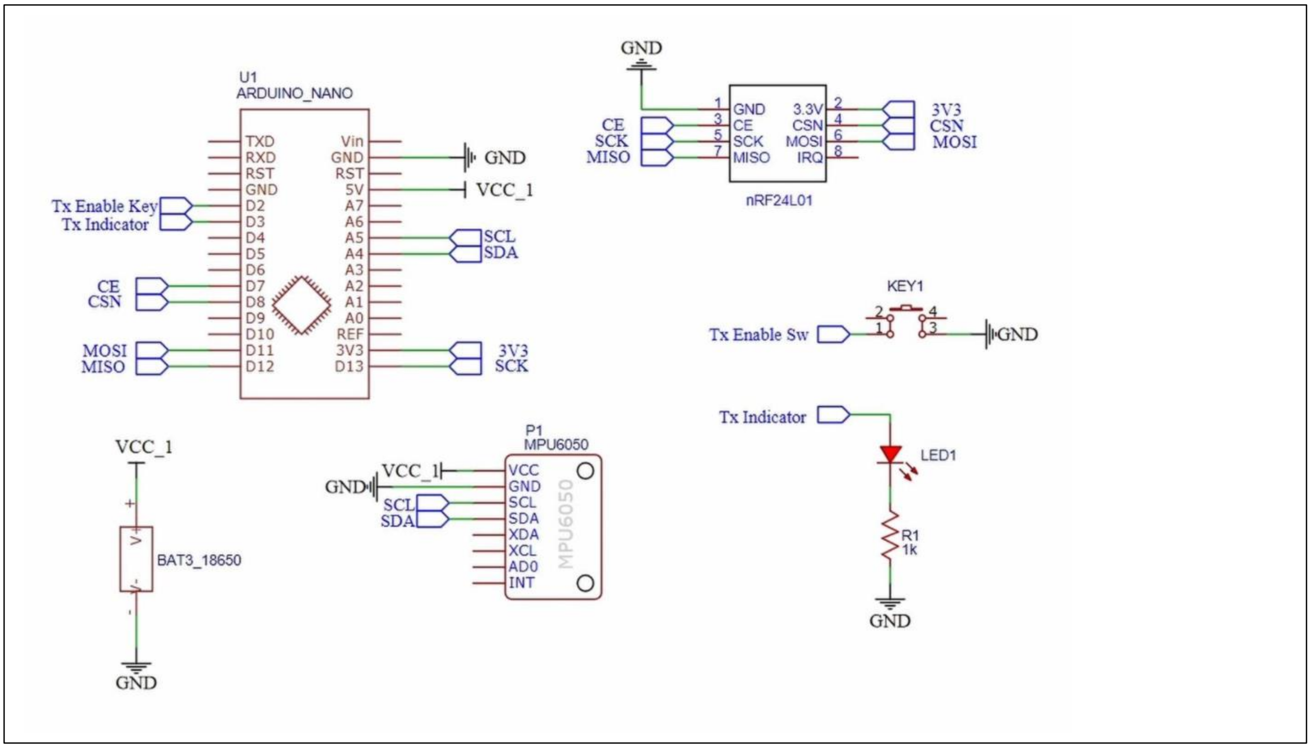




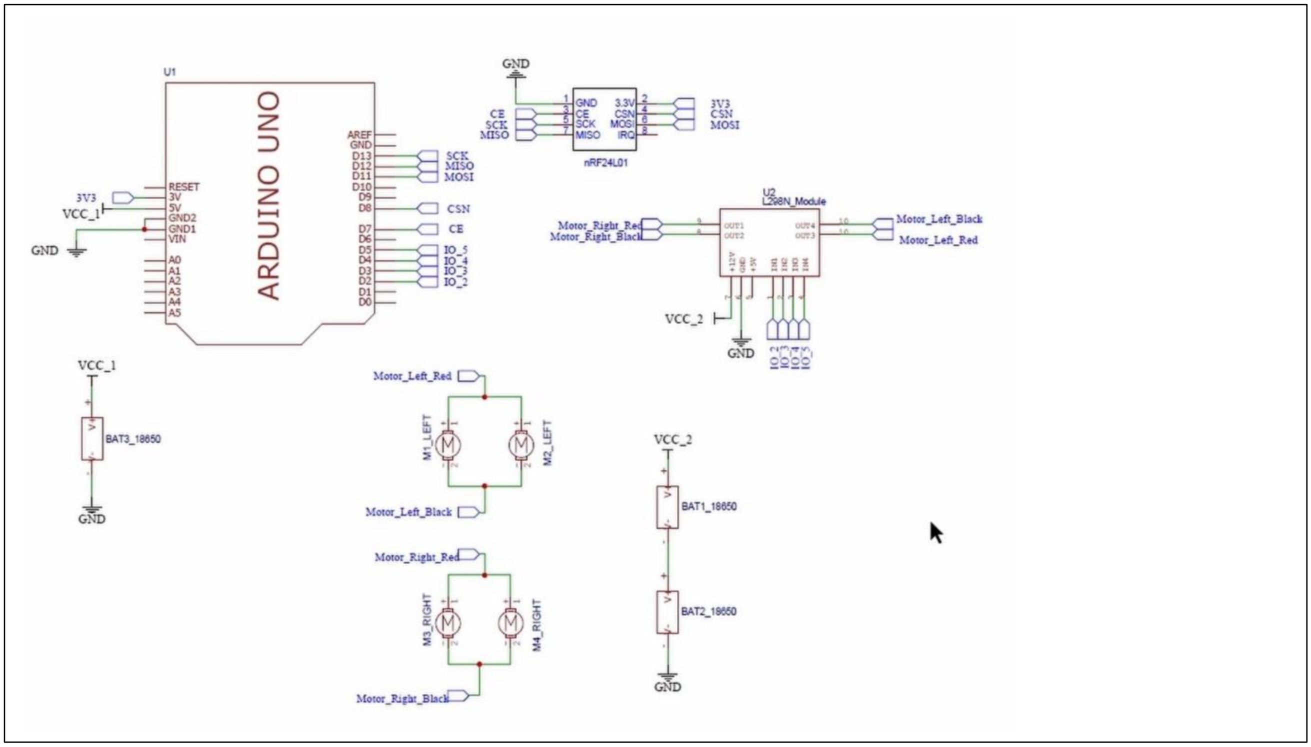


Circuit diagram

Transmitter section



Receiver Circuit



Code

Transmitter Code

```
//Include the libraries

#include <Wire.h>

#include <TinyMPU6050.h>

#include <SPI.h>

#include <NRFLite.h>

MPU6050 mpu (Wire);

int message;

const static uint8_t RADIO_ID = 1;            // Our radio's id.

const static uint8_t DESTINATION_RADIO_ID = 0; // Id of the radio we will transmit to.

const static uint8_t PIN_RADIO_CE = 7;

const static uint8_t PIN_RADIO_CSN = 8;

struct RadioPacket { // Any packet up to 32 bytes can be sent.

  uint8_t FromRadioId;

  uint32_t Data;

  uint32_t FailedTxCount;

};

//Create NRF24 object

NRFLite _radio;

RadioPacket _radioData;

void setup() {

  // Initialization

  mpu.Initialize();

  // Calibration (wait for about 20s to calibrate)

  mpu.Calibrate();

  //start up

  Serial.begin(9600);

  Serial.println("Done Caliberation");
```

```
if (!_radio.init(RADIO_ID, PIN_RADIO_CE, PIN_RADIO_CSN)) {
    Serial.println("Cannot communicate with radio");
    while (1); // Wait here forever.
  }
  _radioData.FromRadioId = RADIO_ID;
}
void loop() {
  mpu.Execute();
  while (mpu.GetRawAccX() <= -8000) {
    //send msg to move front
    message = 1;
    _radioData.Data = message;
    sendData();
    Serial.println("front");
    mpu.Execute();
  }
  while (mpu.GetRawAccX() >= 8000) {
    //send msg to move back
    message = 2;
    sendData();
    _radioData.Data = message;
    Serial.println("back");
    mpu.Execute();


  }
```

```
while (mpu.GetRawAccY() <= -8000) {


   //send msg to move left

   message = 3;

   sendData();

   _radioData.Data = message;

   Serial.println("left");

   mpu.Execute();

 }

 while (mpu.GetRawAccY() >= 8000) {

//send msg to move right

   message = 4;

   sendData();

   _radioData.Data = message;

   Serial.println("right");

   mpu.Execute();

 }

 while (mpu.GetRawAccX() < 8000 and mpu.GetRawAccX() > -8000 and mpu.GetRawAccY() < 8000 and
mpu.GetRawAccY() > -8000) {

   //send msg to stop

   message = 0;

   sendData();

   _radioData.Data = message;

   Serial.println("none");

   mpu.Execute();

 }

}

void sendData()
```

```
  if (_radio.send(DESTINATION_RADIO_ID, &_radioData, sizeof(_radioData))) { // Note how '&' must be placed
in front of the variable name.
  }
  else {
    Serial.println("Failed");
    _radioData.FailedTxCount++;
  }
  delay(500);
  mpu.Execute();
}
```

```
//Include the libraries
#include <SPI.h>
#include <NRFLite.h>
//Initializing the variables
boolean x = 0;
int directionOfMovement = 0;
int leftMotorForward = 2;
int leftMotorBackward = 3;
int rightMotorForward = 4;
int rightMotorBackward = 5;
String message;
const static uint8_t RADIO_ID = 0;       // Our radio's id.  The transmitter will send to this id.
const static uint8_t PIN_RADIO_CE = 7;
const static uint8_t PIN_RADIO_CSN = 8;
struct RadioPacket { // Any packet up to 32 bytes can be sent.
  uint8_t FromRadioId;
  uint32_t Data;
  uint32_t FailedTxCount;
};
//Create NRF object
NRFLite _radio;
RadioPacket _radioData;
void setup() {
 Serial.begin(9600);
  //Set the pin modes
  pinMode(9, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(12, OUTPUT);
```

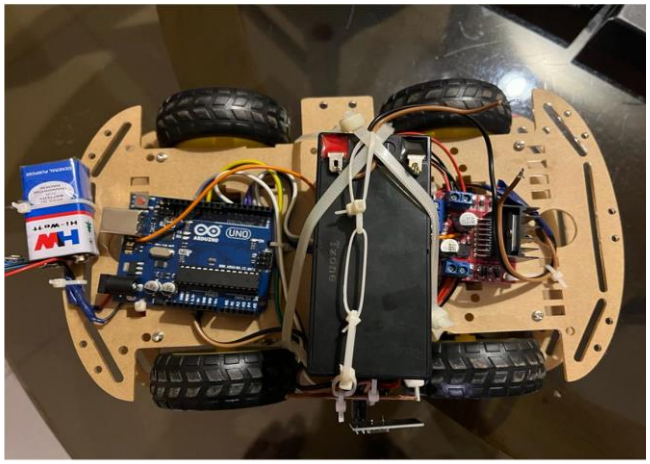
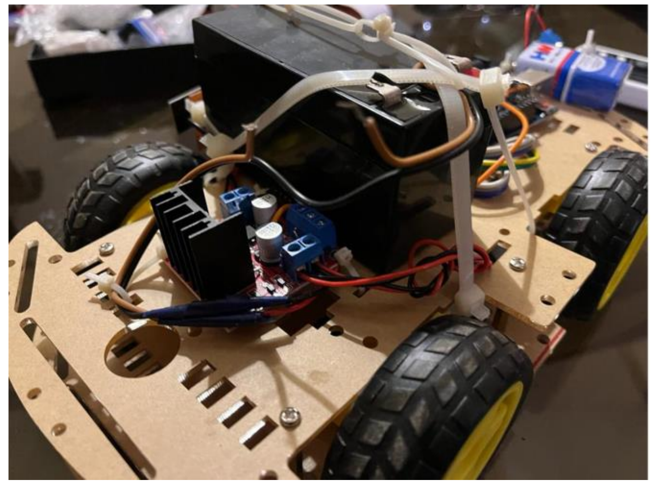
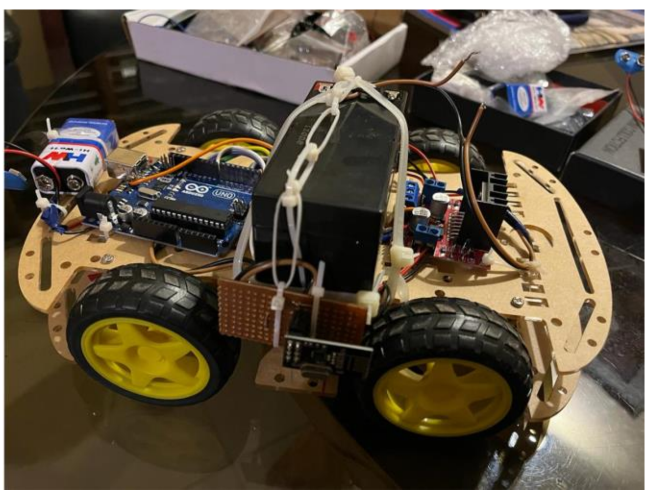
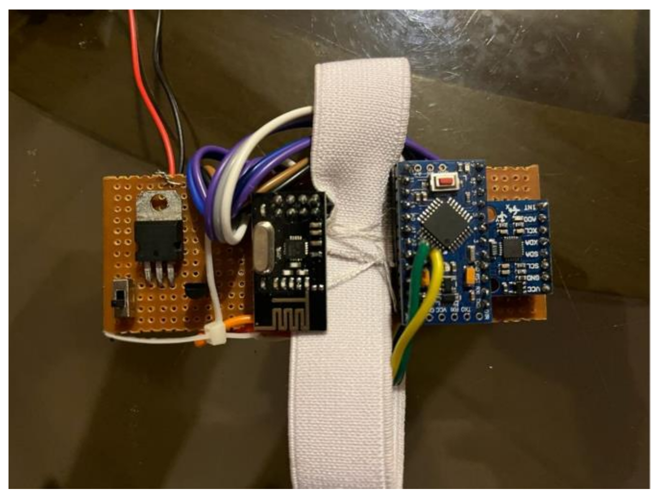
```
  pinMode(13, OUTPUT);
  if (!_radio.init(RADIO_ID, PIN_RADIO_CE, PIN_RADIO_CSN)) {
    Serial.println("Cannot communicate with radio");
    while (1); // Wait here forever.
  }
}
void loop() {
  while (_radio.hasData()) {
    _radio.readData(&_radioData); // Note how '&' must be placed in front of the variable name.
    message = _radioData.Data;
    Serial.println(message);
    directionOfMovement = message.toInt();
    moveAccordingly();
  }
}
//this function moves the car according to the message
void moveAccordingly() {
  if (directionOfMovement == 1) {
    front();
    Serial.println("front");
  }
  else if (directionOfMovement == 2) {
    back();
    Serial.println("back");
  }

  else if (directionOfMovement == 3) {
    left();
    Serial.println("left");
```

```
else if (directionOfMovement == 4) {
    right();
    Serial.println("right");
  }
  else if (directionOfMovement == 0) {
    none();
    Serial.println("none");
  }
}
void front() {
  digitalWrite(leftMotorForward, HIGH);
  digitalWrite(rightMotorForward, HIGH);
  digitalWrite(leftMotorBackward, LOW);
  digitalWrite(rightMotorBackward, LOW);
}
void back() {
  digitalWrite(leftMotorBackward, HIGH);
  digitalWrite(rightMotorBackward, HIGH);
  digitalWrite(leftMotorForward, LOW);
  digitalWrite(rightMotorForward, LOW);
}
void left() {
digitalWrite(leftMotorForward, LOW);
  digitalWrite(rightMotorForward, HIGH);
  digitalWrite(leftMotorBackward, LOW);
  digitalWrite(rightMotorBackward, LOW);
```

```
void right() {
  digitalWrite(leftMotorForward, HIGH);
  digitalWrite(rightMotorForward, LOW);
  digitalWrite(leftMotorBackward, LOW);
  digitalWrite(rightMotorBackward, LOW);
}
void none() {
  digitalWrite(leftMotorForward, LOW);
  digitalWrite(rightMotorForward, LOW);
  digitalWrite(leftMotorBackward, LOW);
  digitalWrite(rightMotorBackward, LOW);
}
```

Real Hardware Model

Conclusion

In spite of lot of improvements in Hand Gesture Controlled Robot using MPU6050 Module, there are many things which can be done. First improvement can be made is Camera Interfacing, which will be used to detect the position of the Robot, when it is out of eye sight or if any obstacle comes in front of the Robot. Also, we can increase the operating range of the Robot by introducing a GPS. Due to this, the range will be increased by worldwide. A gesture-controlled communication aid for elderly and disabled people can be a significant task for future. The two important aims will be to identify the different gestures of elderly and disabled people for communication and to design a rich augmented-reality interface for communication via ubiquitous device such as a television set. Touch less user interface is an emerging type of technology in relation to gesture control. TUI is the process of commanding the computer via body motion and gestures without touching a keyboard, mouse, and screen.

Applications

• Through the use of gesture recognition, remote Control with the wave of a hand of various devices is possible.

• Gesture controlling is very helpful for handicapped and physically disabled people to achieve certain tasks, such as driving a vehicle.

• Wireless controlled robots are very useful in many applications like remote **surveillance, military** etc.

• Hand gesture-controlled robot can be used by **physically challenged** in wheelchairs.

Team Members

S.N	Name	Roll No.	Role and Responsibility
1.	Sambhav Prashit Kaphle	21ME63A01	Coding and Documentation
2.	R Lohit Raj	21ME63D01	Hardware Procurement and fabrication
3.	Kanhu Hansdah	21ME63R01	Design and Animation
4.	Mayank Lakhera	21ME63R07	Coding and documentation
5.	Pankaj suhas Pandit	21ME63R08	Design and Animation

Bill of Materials

Components	Quantity	Cost (INR)
Arduino Uno	1	560
Arduino Pro mini	1	190
nRF24L01	2	2*70=140
MPU6050	1	110
L298N Motor driver	1	100
Batteries	4	60
Chassis	1	650
Miscellaneous		600
Total	------------------------------------	2410

References

1. https://www.ijert.org/research/hand-gesture-control-car-IJERTCONV8IS05066.pf
2. https://www.academia.edu/44867617/Hand_Gesures_Controlled_Robot_using_Arduino
3. Gesture controlled car by Sethu Institute of Technology, Kariapatti
4. https://www.electronicsforu.com/electronics-projects/hardware-diy/wireless-gesture-controlled-robot
5. https://www.ijitee.org/wp-content/uploads/papers/v9i2/B7185129219.pdf
6. https://www.academia.edu/43321690/Project_on_Gesture_Control_car
7. https://lastminuteengineers.com/mpu6050-accel-gyro-arduino-tutorial/
8. https://www.instructables.com/DIY-Hand-Gestures-Controlled-Car