***Gesture Controlled Robot***

*Submitted by:*

**Raiyan Ashraf Arpa Saha**

**Roll: 1907023 Roll: 1907014**

*Course No: CSE 3104*

*Course Title: Peripherals and Interfacing Laboratory*

*Under The Guidance Of:*

****

|  |  |
| --- | --- |
| **Md Repon Islam**  Lecturer  Department of Computer Science and Engineering.  Khulna University of Engineering and Technology. | **Md. Badiuzzaman Shuvo.**  Lecturer  Department of Computer Science and Engineering.  Khulna University of Engineering and Technology. |

Department of Computer Science and Engineering

**Khulna University of Engineering & Technology**

**Khulna, Bangladesh**

**Objectives:**

* Design and build a robot car that can be controlled using hand gestures.
* Implement gesture recognition techniques to interpret and translate hand gestures into car actions.
* Evaluate the accuracy and effectiveness of the gesture recognition system with various hand gestures.
* Assess the car's responsiveness and reliability in executing commands based on recognized hand gestures.
* Test the car's ability to navigate and interact with its surroundings using gesture-based control.
* Compare different gesture recognition algorithms in terms of accuracy and speed.
* Identify limitations and potential improvements for the gesture-controlled robot system.
* Provide recommendations for enhancing the gesture recognition system and the car's functionality based on the experiment's results.

**Introduction:**

This project report presents an exciting endeavor where we built a car robot that can be controlled using hand gestures. We used Arduino Uno and Nano microcontrollers, a Bluetooth module, an accelerometer, and a motor driver to bring this innovative idea to life. The main aim of the project was to create a car robot that responds to wireless hand gestures, making the interaction between humans and robots more intuitive and effortless.

The Arduino Uno and Nano microcontrollers acted as the brain of the robot, allowing us to process gesture recognition algorithms and control its movements. With these microcontrollers, we could capture sensor data, interpret gestures, and make the robot move accordingly. They are powerful and flexible tools that made our project possible.

The Bluetooth module enabled wireless communication between the robot and a controlling device like a smartphone or computer. This meant that we didn't need any physical connections to operate the robot. Instead, we could send gesture commands wirelessly, giving us more freedom and convenience in controlling the car robot.

The accelerometer played a vital role as a motion sensor. It detected the user's hand gestures and movements, providing us with real-time information about hand orientation and motion. This data was crucial for recognizing and mapping gestures to specific robot actions. The accelerometer's accuracy and precision made the robot respond smoothly and accurately to the user's gestures.

The motor driver was responsible for controlling the robot's movements. It acted as a bridge between the microcontrollers and the motors, converting the recognized gestures into motor control signals. This allowed the robot to move forward, backward, turn, or stop based on the user's gestures. The motor driver ensured that the robot moved precisely and responsively.

This project report describes the detailed steps we took to integrate the hardware components, develop software algorithms, and calibrate the system. We explored various gesture recognition techniques, including pattern recognition and machine learning algorithms, to accurately interpret and map hand gestures to robot actions.

We evaluated the performance of the gesture-controlled car robot by testing its responsiveness, accuracy, and reliability. We examined how well it recognized and differentiated different hand gestures and analyzed the effectiveness and efficiency of the gesture recognition algorithms, considering factors like accuracy and speed.

Throughout the project, we encountered limitations and areas for improvement. We discussed these in the report and provided recommendations for enhancing the hardware, refining the software, and optimizing the gesture recognition techniques based on our findings and observations.

**Connecting Components:**

**Software tools:**

1. Programming Languages: Arduino.
2. Operating Systems: Windows 11
3. IDE: Arduino IDE

**Hardware Tools:**

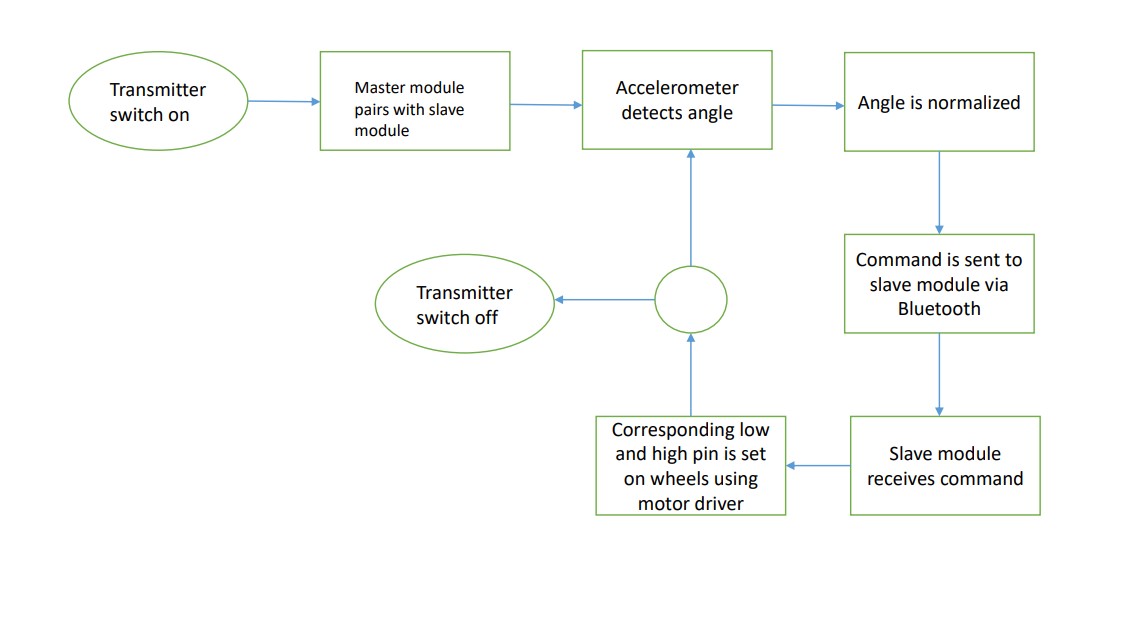
**Table 1: Table for Hardware Tools.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Serial** | **Product Name** | **Rating** | **Quantity** |
| 1 | Arduino Uno | Micro-controller: ATmega328.  Operating Voltage: 5V.  Input Voltage (recommended):7-12V.  Digital I/O Pins: 14 (of which 6 provide PWM output).  Analog Input Pins: 6.  Weight: 28(without Cable)  54(with cable)  Flash Memory: 32 KB | 01 |
| 2 | Arduino Nano | Operating Voltage (logic level): 5V  8 analog inputs ports: A0 ~ A7  14 Digital input / output ports: TX, RX, D2 ~ D13  1 pair of TTL level serial transceiver ports RX / TX  Using Atmel Atmega328P-AU MCU  There is a bootloader installed in it  Standard 0.1” spacing DIP (breadboard friendly).  Manual reset switch. | 01 |
| 3 | MPU6050 Accelerometer | Supply Voltage: 3.3 V/ 5V.  Interface Type: I²C, SPI.  Sensing Range: ±2g, ±4g, ±8g, ±16g.  Sensitivity X: 28.6 LSB/g.  Sensitivity Y: 31.2 LSB/g.  Sensitivity Z: 34.5 LSB/g.  Ultra-Low Power: 40µA in measurement mode, 0.1µA | 01 |
| 4 | HC-05 Bluetooth Master  Module | Bluetooth protocol: Bluetooth Specification v2.0+EDR.  Frequency: 2.4GHz ISM band.  Modulation: GFSK(Gaussian Frequency Shift Keying).  Emission power: =4dBm, Class 2. | 01 |
| 5 | HC 05 Bluetooth Slave Module | Bluetooth protocol: Bluetooth Specification v2.0+EDR.  Frequency: 2.4GHz ISM band.  Modulation: GFSK(Gaussian Frequency Shift Keying).  Emission power: =4dBm, Class 2. | 01 |
| 6 | L298 Motor Driver | Heatsink for better performance.  Power-On LED indicator.  Double H bridge Drive Chip: L298N.  Operating Voltage(VDC): 5~35  Peak Current (A): 2  Continuous Current (A): 0-36mA  No. of Channels: 2  Over-Current Protection (A): Yes  Thermal Protection: Yes | 01 |
| 7 | DC Gear Motor | Operating Voltage(VDC): 3~12  Shaft Length (mm): 8.5  Shaft Diameter (mm): 5.5 (Double D-type)  No Load Current: 40-180mA.  Rated Speed(After Reduction): 180 RPM  Rated Torque: 0.35 Kgcm | 04 |
| 8 | Smart robot car Wheels | -------------------------- | 04 |
| 9 | On/Off switch | --------- | 02 |
| 10 | Battery Cell Holder | 4 cell capacities | 01 |
| 11 | Battery Cell | 3.7 Volt each battery. | 06 |
| 12 | Velcro Tape | --------------- | O1 |
| 13 | Nuts and Bolts | ----------------- | As req |
| 14 | Male to Male Jumper Wires | -------------------- | As req |
| 15 | Male to Female Jumper Wires | --------------- | As req |

**Methodology:**

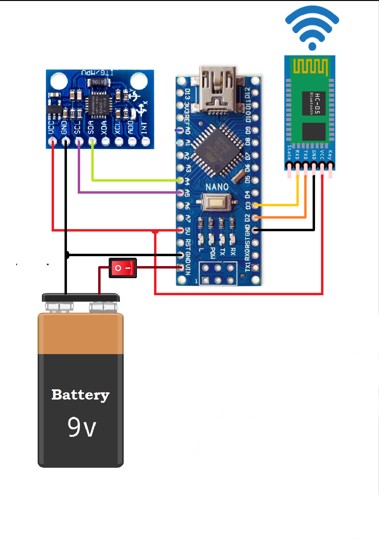
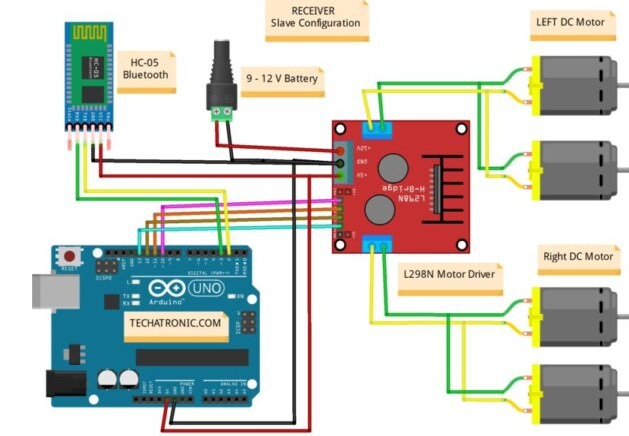
1. At first, we connect the transmitter and receiver components separately. In either case, caution should be taken so that no more than 3.3v passes through the Bluetooth module.
2. 4 dc motors are connected to the motor driver, for the car, and 10,11,12,13 pins of Arduino UNO are connected with the IN1, IN2, IN3, and IN4 pins of motor driver after testing the car’s wheel direction.
3. Then the Bluetooth module (slave) has to be connected. It’s RX,TX pin goes to digital pin 2,3.VCC to 3.3v and GND to GND of Arduino.
4. Then the battery’s connection is to be given accordingly.
5. While uploading the code to the receiver’s Arduino, the vcc of Bluetooth module should be kept open.
6. Then we build the transmitter, the Bluetooth module’s Rx , Tx pin goes to 2,3 of Arduino NANO, VCC to 3.3v and GND to GND of Arduino.
7. Then the mpu6050 is to be connected, SCL, SDA pin A5 and A4 respectively, VCC to 5v and GND to GND.
8. Now that the connection is all set up. We connect battery and switch.
9. We upload respective codes to the receiver and transmitter.
10. We turn on switch of both car and transmitter, and wait around 5 sec for the master and slave module to connect properly.
11. After the connection has been established the car can be moved according to our gesture oh hand that carries the transmitter.

**Flow Chart:**

****

**Figure 1:** Flow chart for Gestured Controlled Robot.

**Circuit Diagram:**

** **

**Figure:** Circuit Diagram of Transmitter **Figure:** Circuit Diagram of Receiver.

**Pseudocode:**

***Transmitter Code:***

Start:

Accelerometer setup

Bluetooth module connection setup

Loop:

Read Accelerometer

Normalize reading

If(x<60)

Set m= 'F'

If(x>130)

Set m= 'B'

If(y<60)

Set m= 'L'

If(x>130)

Set m= 'R'

If(x>70 AND x<120) AND (Y<70 AND y>120) AND flag==1)

Set m='N'

Transmit m;

***Receiver Code:***

//Receiver code

Start:

Serial monitor setup

Bluetooth connection setup

Motor pin setup

Loop:

Read transmitter

If(m='F')

Move forward

If(m='B')

Move Backwards

If(m='L')

Turn left

If(m='R')

Turn Right

If(m='N')

Stop

End

**Conclusion and Discussion:**

The entire project of gesture-controlled robot was exciting to exercise and acquire practical knowledge about Arduino and various hardware components. But in time of production, we had to face many challenges. The main challenge of the project was the transmitter part. In this case, we had to select and set up a pair of master slave Bluetooth module. It was quite challenging as we had to change our Bluetooth module several times due to defects. After pairing the module, they were rigorously checked by sending signals and checking it out. Then using the mpu 6050, the whole circuit of transmitter was developed. In case of building the receiver, or our main car, motors were carefully and cautiously wired and tested with wheels. But the challenge was the motor driver. At first, we tested our motors by connecting them with

Motor driver and batteries, perfectly. But after several times, the motor driver was burned out and stopped working. The entire set up has to be changed and modified and new motor driver was installed. While testing, there were some problems too. In spite of setting up everything perfectly after tests, the receiver was not receiving. Then we had to modify our code. We had to change the baud rate in code and also had to make some modification, then, after uploading, the car was perfectly controlled by hand gesture.

In conclusion, the development of the gesture controlled car, gave us a practical knowledge about Arduino UNO and NANO as well as providing us a clear and concise idea of wireless communication between devices. The entire project was an exciting endeavor to exhibit our knowledge about Arduino as well as interfacing devices.

**References:**

* <https://techatronic.com/how-to-make-gesture-control-robot-using-arduino/>
* <https://docs.arduino.cc>
* https://www.instructables.com/Arduino-Bluetooth-Master-and-Slave-Using-Any-HC-05