|  |
| --- |
| Home | AIUB PROJECT REPORT **COURSE:**  HUMAN COMPUTER INTERACTION |

**INSTRUCTOR:**

PROF. DR. TABIN HASAN

**GROUP MEMBERS:**

|  |  |
| --- | --- |
| **NAME** | **ID** |
| OMI, NUSRAT ZANNAT | 18-38738-3 |
| JARIN, TASNIM | 19-39394-1 |
| RAFIQ, FAHIM BIN | 18-36800-1 |
| THAMID, MD. AHANAF | 18-36794-1 |

Department of Computer Science

Faculty of Science & Technology

American International University Bangladesh

**SUMMER, 2022**

* **Introduction**

Hand gestures are used as input signals to control the dynamics of robots in gesture controlled cars. A gesture device that is attached to a glove and has sensors to detect movements must be worn by the user in order to detect hand movements. The gesture device and the car are wirelessly linked. The sensors pick up on hand motions and relay commands to the car, which causes the arm or car to move in the desired direction.

* **Project background**
* **Identification of problem**

Traditional wired buttons control robots get exceedingly bulky and have a limited range of motion. Wearable hand gloves that may be utilized as input for the movement of the robot will enable the Wireless Hand operated Robot to move wirelessly. The main goal of our research is to create a robot that can recognize human interaction and carry out the activities that have been delegated to it. As part of our project, we will create a wearable hand glove with sensors installed on it to record hand motion and transform the mechanical data into electrical form. This information will then be further analyzed and transformed into a format that the lily pad installed on the Glove can understand. The objective of this lily pad is to serve as a data transmitter for wireless communication. The transmitted data will be processed and sent again to the microcontroller once it has been received by the receiver module, which will be attached to it. The robot's microcontroller will determine the commands and, in response, activate the motor drivers to operate the motors for specific tasks.

* **Background study**

The 9dof Accelerometer Gyroscope Magnetometer is used in Microcontroller Based Gesture Recognition and Angle Measurement without Image Processing. which can measure the rotation and acceleration of the X and Y axes. The project was composed of main electronics circuits and electronic components such as:

❖ Arduino Uno

❖ L293D Motor Controller Driver

❖ DC motor

❖ 433khz Wireless Radio transmitter and receiver.

❖ The system can give you both the angle and gesture of forward, backward, right and left.

* **Goals**

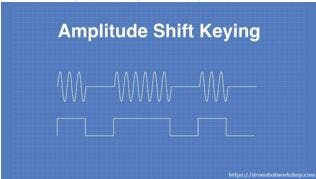
This project's main goal is to make it possible to drive via hand gestures. An accelerometer is used to recognize human hand motions. The necessary activities for the human gesture are performed as a result of the coding. Robots that can be controlled by hand gestures are referred to as gesture controlled robots. To give the robot the proper instructions so that it may perform whatever we want, all you need is a small transmitting device in your palm that includes an acceleration meter.

* **Methodology**
* **Methods & tools**

For our project, we have used the ASK method. Basically, ASK stands for Amplitude Shift Keying. This is a less complex technique, similar to AM radio or amplitude modulation. We have utilized "ASK" in our code since these modules are quite straightforward. In amplitude shift modulation, the carrier wave's amplitude (i.e., level) is modified in response to the incoming data signal. Given that there are only actually two levels to consider, this is also referred to as binary amplitude shift modulation

Digital 1 – This drives the carrier at full strength.

Digital 0 – This cuts the carrier off completely.



For the project the **tools**, we have used are discussed below:

1. **MPU-6050(x1):** This is a module with a 3-axis accelerometer (AX, AY, AZ) and 3-axis gyroscope (GX, GY GZ). We have used it because it was fulfilling our signal demands.

2. **Chassis(x1):** This is the robotic vehicle's frame

3. **Arduino Uno and Arduino Nano:** The transmitter and receiver's central processing unit. One Arduino is used for sending the data, while another Arduino is used for receiving it.

4. **433Mhz TX RX Module:** A 433 MHz TX/RX module will be utilized to broadcast and receive data from the MPU-6050.

5. **Other Necessary Components:** Jumper Wires, Mini Breadboard,4 Wheels, and 4 motors.

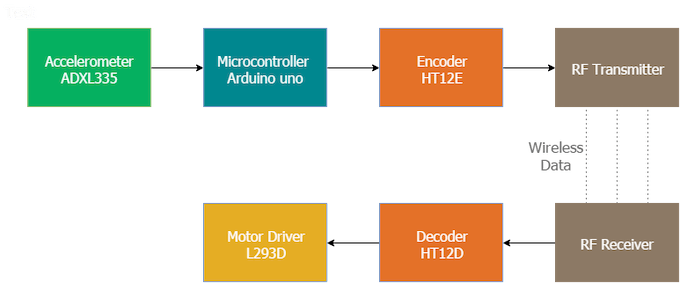
6. **L298N**: This is a 4-channel Motor Driver to drive the motors of the Car.

7. **HC-05 Wireless BT Module:** we can quickly add the Bluetooth feature to our Arduino project, and then you can use it to control some gadgets or send signals.

* **Implementation planning**

We have decided to divide our project into two different parts the transmission station, which is mounted on the hand, and the receiving station, which is the automobile, are the two main components of this project.

The transmission unit: For the transmission unit we wanted to use Arduino nano,mpu6050 sensor, Bluetooth HC-05 module(master), and battery 9v and mount them on a breadboard. Because the MPU6050 module is made of two built-in sensors which are the gyroscope and the accelerometer, each of these sensors gives data and gets processed by the Arduino nano in order to get the appropriate angle, then according to the output data we'll send special characters that specify the direction the car will choose via Bluetooth module to the Arduino UNO. As the functionality matches our requirements step by step we wanted use this component to give a suitable result to our project.

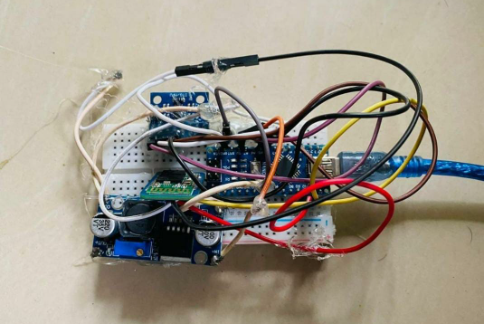


**Block Diagram of the car**

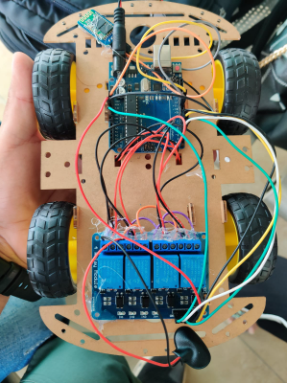
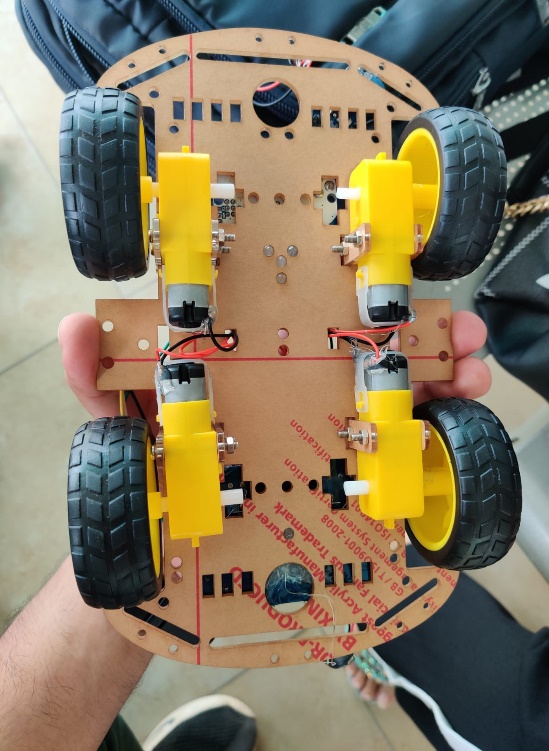
**The reception unit**: we basically use an Arduino UNO for the same purpose as Arduino Nano we didn’t have to use bug moduler as it is built into the Arduino UNO, Bluetooth module for receiving the data, at first we thought we will run the Dc motors directly from the Arduino UNO but then we have seen it cannot provide the power the 4 DC motors wheels needed so we use the L298N motor driver to control the direction and speed of motors and 9v battery to power the Arduino UNO and other components.

* **Implementation**

After implanting all the components the final design of our project will look like this:



**The Controller**

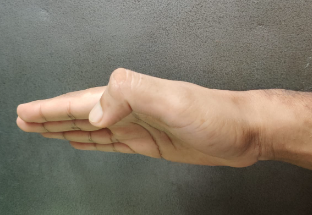
 ****

**The Care and the Recipient**

Now let’s talk about the hand gesture and how it works:

**Going Forward Going Backwoods**

**Going Left Going Right**



The gestures we made to control the car are illustrated in the image above.

The MPU-6050 values are mapped by the Arduino nano when the user makes a gesture (Transmitter). This value is encoded by the transmitter controller and transmitted as a string to the receiving Arduino using a 433 MHz transmitter module. Data is transmitted to the receiving Arduino through a 433 MHz receiver module. Data is received in String Format when it is received. This "String" is transformed into an "INT" by the controller, which then, in compliance with the program, regulates the direction of the vehicle forward, backward, left, or right.

**The CODE we have used for the transmitter is:**

#include <SoftwareSerial.h>

#include "Wire.h"

#include <MPU6050\_light.h>

SoftwareSerial mySerial(10, 11);

char s;

MPU6050 mpu(Wire);

unsigned long timer = 0;

String state;

void setup() {

Serial.begin(57600);

Wire.begin();

mySerial.begin(9600);

byte status = mpu.begin();

Serial.print(F("MPU6050 status: "));

Serial.println(status);

while(status!=0){ }

Serial.println(F("Calculating offsets, do not move MPU6050"));

delay(1000);

mpu.calcOffsets();

Serial.println("Done!\n");

}

void loop() {

mpu.update();

if(mpu.getAngleX() > 60){

s = 'f';

delay(10);

}

else if(mpu.getAngleX() < -60){

s = 'b';

delay(10);

}

if(mpu.getAngleY() > 60){

s = 'l';

}

if(mpu.getAngleY() < -60){

s = 'r';

delay(10);

}

Serial.println(s);

mySerial.write(s);

}

**The CODE we have used for the Receiver is**:

int m1a = 9;

int m1b = 10;

int m2a = 11;

int m2b = 12;

char val;

void setup()

{

pinMode(m1a, OUTPUT);

pinMode(m1b, OUTPUT);

pinMode(m2a, OUTPUT);

pinMode(m2b, OUTPUT);

Serial.begin(9600);

}

void loop()

{

while (Serial.available() > 0)

{

val = Serial.read();

Serial.println(val);

}

if( val == 'F')

{

digitalWrite(m1a, HIGH);

digitalWrite(m1b, LOW);

digitalWrite(m2a, HIGH);

digitalWrite(m2b, LOW);

}

else if(val == 'B')

{

digitalWrite(m1a, LOW);

digitalWrite(m1b, HIGH);

digitalWrite(m2a, LOW);

digitalWrite(m2b, HIGH);

}

else if(val == 'L')

{

digitalWrite(m1a, LOW);

digitalWrite(m1b, LOW);

digitalWrite(m2a, HIGH);

digitalWrite(m2b, LOW);

}

else if(val == 'R')

{

digitalWrite(m1a, HIGH);

digitalWrite(m1b, LOW);

digitalWrite(m2a, LOW);

digitalWrite(m2b, LOW);

}

else if(val == 'S')

{

digitalWrite(m1a, LOW);

digitalWrite(m1b, LOW);

digitalWrite(m2a, LOW);

digitalWrite(m2b, LOW);

}

else if(val == 'I')

{

digitalWrite(m1a, HIGH);

digitalWrite(m1b, LOW);

digitalWrite(m2a, LOW);

digitalWrite(m2b, LOW);

}

else if(val == 'J')

{

digitalWrite(m1a, LOW);

digitalWrite(m1b, HIGH);

digitalWrite(m2a, LOW);

digitalWrite(m2b, LOW);

}

else if(val == 'G')

{

digitalWrite(m1a, LOW);

digitalWrite(m1b, LOW);

digitalWrite(m2a, HIGH);

digitalWrite(m2b, LOW);

}

else if(val == 'H')

{

digitalWrite(m1a, LOW);

digitalWrite(m1b, LOW);

digitalWrite(m2a, LOW);

digitalWrite(m2b, HIGH);

}

}

* **Result**

We are already aware of this idea that uses hand motion or hand gesture technologies to remotely operate an automated car. Every time we tilt our hands in whatever direction, the proper movements occur.

|  |  |  |  |
| --- | --- | --- | --- |
| **Action** | **X-axis** | **Y-axis** | **Z-axis** |
| Stable | 0-15 | 0-15 | 0-15 |
| Forward | 0-15 | 20-50 | 0-15 |
| Backward | 0-15 | -(10)-(-30) | 0-15 |
| Right | 20-50 | 0-15 | 0-15 |
| Left | (-10)-(-30) | 0-15 | 0-15 |

Fig: Display of xyz axis values in different positions

This table displays values along the x, y, and z axes. So in this x, y,z axis that is in different positions namely stable position, forward, backward, right and left. Here we took stable position as a starting point. X-axis and Y – axis value changeable but the z axis value always same(0-15).

* **Discussion**
* **Findings**

The first problem that we faced during our project came from trying to send signals wirelessly. As it happens, most wireless module receivers face a lot of noise in the signal. As we didn't use rechargeable batteries that's why we faced Repurchasing batteries trouble.

* **Analysis**

A Gesture Controlled car is a kind of hand device which can be controlled by hand gestures and not the old-fashioned way by using gear and steering. Our expectation is this hand device will work when we will make hand movements like left, right, front, and back. The controller sends signals and the receiver read the signal and works as commands. In the automotive industry, this ability allows drivers and passengers to interact with the car**.**

* **Conclusion**

Making something easier and less expensive than it was before is a constant challenge for engineers. Additionally, we wanted to keep the cost down, so we were more careful while selecting the components. Even though we made an effort to make things less expensive, we did not skimp on quality. We made an effort to make it more dependable and clear. Our project's greatest benefit is that it is incredibly simple to control. Future applications for the gesture-controlled robot created in this work are numerous. Using an ARDUINO microcontroller and an Android smartphone, the design and development of a gesture-controlled robot are demonstrated in this work. A full explanation of how an algorithm works has been provided. The system's updating has been left as a future objective because there are countless updating alternatives. The constructed object is lightweight and inexpensive to transport. It will be more productive if there are more sensors or cameras added.

**Our Project Video Link**:

[**https://drive.google.com/file/d/1woghnduiuFG8ppEGUjOh-cR8NAXlf\_uU/view?usp=sharing**](https://drive.google.com/file/d/1woghnduiuFG8ppEGUjOh-cR8NAXlf_uU/view?usp=sharing)