Title: Hand Gesture Car with Obstacle Avoider

Course No: CSE 3104

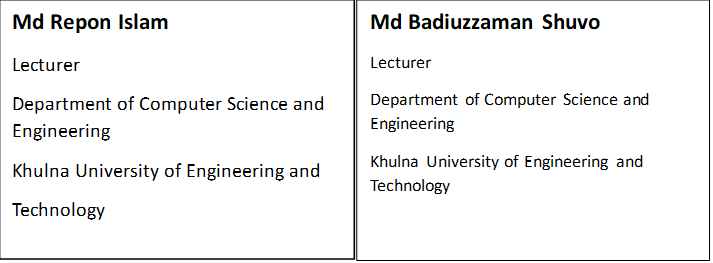
Course Title: Peripherals and Interfacing Laboratory

Submitted by:

Al Mahamud Hasibul Hasan Hasib

Roll:**1907082** Roll:**1907089**

Submitted To:



**Khulna University of Engineering & Technology , Khulna**

**Objectives:**

1. Hand gesture control: Enable the car to be controlled wirelessly through hand gestures. This involves using the gyro sensor to detect the movements of the hand and translating them into specific commands for the car's movements, such as forward, backward, left, and right.
2. Obstacle avoidance: Implement a system that allows the car to detect and avoid obstacles in its path.
3. Wireless communication: Utilize the NRF module to establish wireless communication between the hand gesture controller and the car.
4. Real-time responsiveness: Ensure that the communication between the hand controller and the car is fast and responsive. This requires optimizing the code and minimizing delays to achieve real-time control
5. Integration and coordination: Integrate the various components, including the Arduino Uno, gyro sensor, NRF module, motor driver, and obstacle detection system, to work together seamlessly

**Introduction:**

The hand gesture-controlled car with an obstacle avoider is an project that combines Arduino Uno, a gyro sensor, and an NRF module to create a remote-controlled car that responds to hand gestures while intelligently avoiding obstacles. This project showcases the fusion of robotics, wireless communication, and sensor integration, providing an interactive and dynamic experience for users.

In this project, we harness the power of hand gestures to control the movement of a car wirelessly. By simply moving your hand in specific ways, we can command the car to move forward, backward, turn left, or turn right.

The obstacle avoidance feature further enhances the functionality and safety of the car. By incorporating ultrasonic sensor, the car can detect obstacles in its path and stop its front movement.

To establish wireless communication between the hand controller and the car, we employ an NRF module. This module facilitates the seamless transmission of commands from the hand gestures to the car, eliminating the need for cumbersome wires or physical connections. The wireless nature of the communication provides flexibility and mobility, allowing users to control the car from a distance.

Throughout this project, we emphasize real-time responsiveness, ensuring that the car responds instantaneously to hand gestures and obstacle detection. By optimizing the code and minimizing delays, we aim to achieve a fluid and interactive user experience, making the car feel like a natural extension of the user's hand movements.

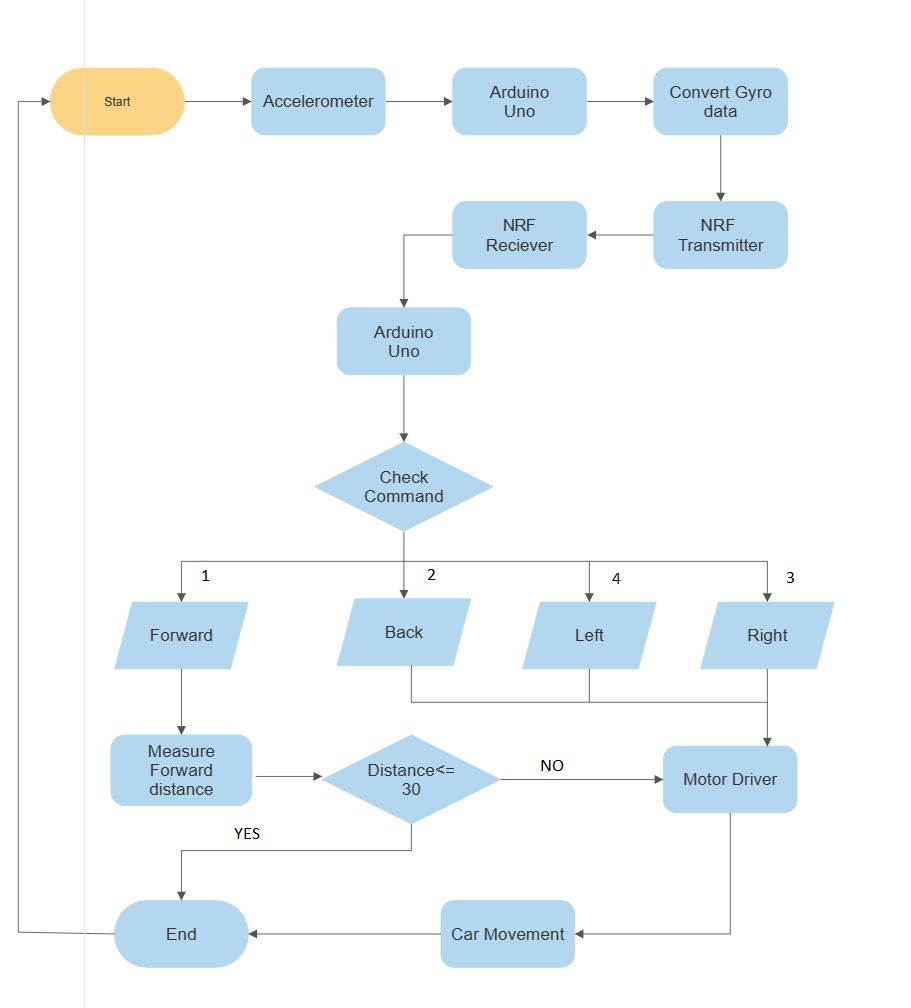
**SOFTWARE SPECIFICATION:**

* Arduino IDE: The Arduino Integrated Development Environment (IDE) is the primary software tool used for programming and uploading code to the Arduino Uno board.
* Arduino Libraries &Sensor Libraries:
* Programming Language: The Arduino IDE uses a simplified version of C++ for programming the Arduino boards.
* Code Editor: The Arduino IDE provides a basic code editor.

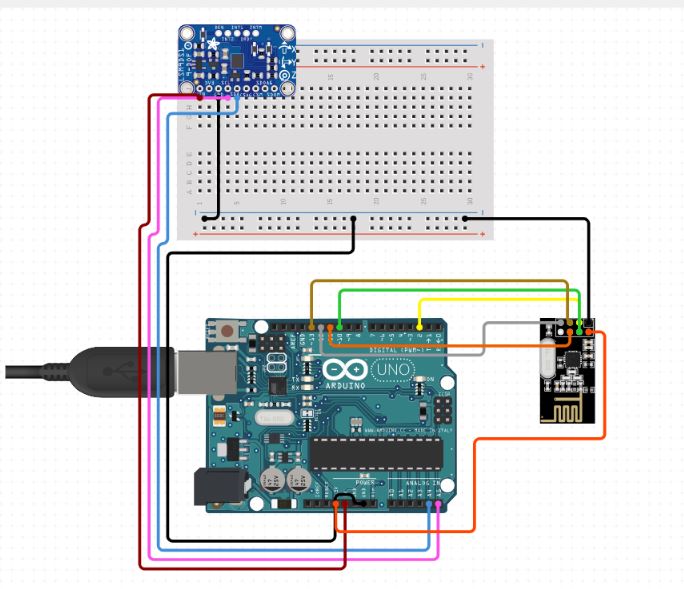
**Table 1: HADRWARE CONFIGURATION**

|  |  |  |  |
| --- | --- | --- | --- |
| **Serial No.** | **Equipment Name** | **Rating** | **Quantity** |
| 1 | Arduino Uno | Micro-controller: ATmega328.  Operating Voltage: 5V.  Input Voltage :7-12V.  Input Voltage (limit):6-20V.  Digital I/O Pins: 14 (6 provide PWM output).  Analog Input Pins: 6.  Clock Speed: 16MHz  SRAM :2 Kbit  Flash Memory: 32 KB | 02 |
| 2 | MPU6050 Accelerometer | Supply voltage:  VDD supply voltage: 2.375V to 3.46V  VLOGIC supply voltage: 1.8V to VDD  Accelerometer:   * Sensing element: MEMS (Micro Electro-Mechanical System) capacitive accelerometer * Full-scale range: ±2g, ±4g, ±8g, ±16g (configurable)   Sensitivity:  Sensitivity X: 28.6 LSB/g.  Sensitivity Y: 31.2 LSB/g.  Sensitivity Z: 34.5 LSB/g. | 01 |
| 3 | nRF24L01: | Wireless Communication: 2.4GHz proprietary protocol (not Bluetooth)  Wireless Range: Up to 100 meters (depending on environmental conditions and settings) (with antenna up to 1Km)  MCU: 8-bit microcontroller with integrated radio transceiver  Data Rate: Up to 2 Mbps  Power Consumption: Low power consumption for battery-operated applications  Interfaces: SPI (Serial Peripheral Interface)  GPIO: Available for connecting sensors, actuators, and other peripheral devices  Antenna Options: Integrated PCB antenna or external antenna with u.FL connector  Package: 20-pin QFN or 16-pin TSSOP package options | 02 |
| 4 | Robot Car Chassis | --- | 01 |
| 5 | Car Wheels | . | 04 |
| 6 | L298 Motor Driver | Motor Compatibility: Can control two DC motors or a single bi-polar stepper motor.  Voltage Range: 4.5V to 46V.  Peak Current: Up to 3A per channel.  Logic Compatibility: TTL and CMOS.  Logic Voltage: 2.3V to VSS (supply voltage).  Enable and Control Inputs: IN1, IN2, IN3, IN4 for direction control; EN for motor enable.  Current Sensing: Built-in current sensing with OUTA and OUTB pins.  Diode Emulation: Included for protection against back EMF.  Package Options: 15-lead Multiwatt, 15-lead Zip, and 15-lead DIP.  Heat Sink: Recommended for heat dissipation. | 01 |
| 7 | DC Gear Motor | Voltage: 3V.  RPM: 125r / minute.  Current: 80 - 100mA.  Reduction: 48:1.  Output Torque: 0.8kg.cm. | 04 |
| 8 | Ultrasonic Sensor | * Range : 40 cm to 300 cm. * The response time :50ms-200ms * The Beam angle: around 50. * Operating voltage: 20 - 30 VDC. * Preciseness I: ±5% * Frequency :120 kHz. * Resolution : 1mm | 01 |
| 9 | Battery Cell | 3.7 Volt | 05 |
| 10 | Switch (On/Off) | --- | 02 |
| 11 | Male to Male &  Male to Female Jumper Wires | -------- | As Required |
| 12 | Battery Cell | 3.7 Volt | 05 |

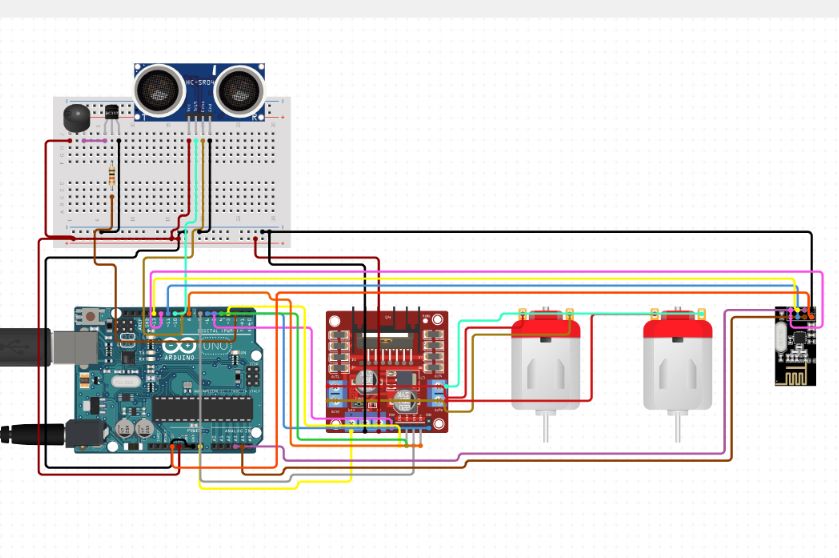
**Flow Chart:**



**Figure 1:** Flow chart for Hand Gesture Car with obstacle avoider.

**Circuit Diagram:**

**Figure 2:** Circuit Diagram of Transmitter



**Figure 3:** Circuit Diagram of Receiver

**Pseudo Code for Transmitter:**

Initialize the necessary variables and pins

Setup:

- Initialize the serial communication

- Set the pin mode for the TX\_ENABLE\_KEY

- Initialize the radio communication

- Configure the radio module

- Stop listening and write the initial Tx\_command value to the radio module

- Initialize the Wire library and MPU6050 sensor

- Test the connection with the MPU6050 sensor

- Initialize the Tx\_Array with default values

Loop:

Get the motion values from the MPU6050 sensor

Check if the TX\_ENABLE\_KEY is pressed:

- If it is pressed for the first time (TX\_Key\_Pressed = 0):

- Set TX\_Key\_Pressed to 1

- If Tx\_Enable\_Flag is 0:

- Set Tx\_Enable\_Flag to 1

- Set the corresponding PORTD bit

- Else:

- Set Tx\_Enable\_Flag to 0

- Clear the corresponding PORTD bit

- Print "Pressed"

- If it is released (PIND & 0x04) and TX\_Key\_Pressed is 1:

- Set TX\_Key\_Pressed to 0

- Print "Released"

Check the accelerometer values:

- If ay is outside the threshold (-4000 to 4000) or ax is outside the threshold:

- If ax is within the threshold (-4000 to 4000) and ay is within the threshold:

- If ay is less than or equal to -4000:

- Set Tx\_command to 1 (forward)

- Calculate the Speed\_index based on ay

- If ay is greater than or equal to 4000:

- Set Tx\_command to 2 (backward)

- Calculate the Speed\_index based on ay

- If ax is within the threshold (-4000 to 4000) and ay is within the threshold:

- If ax is less than or equal to -4000:

- Set Tx\_command to 4 (left)

- Calculate the Speed\_index based on ax

- If ax is greater than or equal to 4000:

- Set Tx\_command to 3 (right)

- Calculate the Speed\_index based on ax

- Else (both ax and ay are outside the threshold):

- Set Tx\_command and Speed\_index to 0 (stop)

Print the Tx\_command and Speed\_index values

If Tx\_Enable\_Flag is true:

- Update the Tx\_Array with Tx\_command and Speed\_index

- Write the Tx\_Array to the radio module

Delay for a short period Receiver Code:

**Pseudo Code for Receiver:**

Initialize the necessary variables and pins

Setup:

- Initialize the serial communication

- Set the pin modes for motor control, ultrasonic sensor, and LED

- Initialize the radio communication

- Configure the radio module

- Start listening for incoming data

- Print "START"

Loop:

Check if data is available from the radio module

- Read the received data into variables

- Print the received command and speed index

Speed Control Logic:

If Run\_Stop\_Mode is 0 (Stop):

- Turn off all motor directions

- Increment Run\_Stop\_Counter

- If Run\_Stop\_Counter reaches a threshold based on Speed\_index, set Run\_Stop\_Mode to 1

If Run\_Stop\_Mode is 1 (Run):

- Increment Run\_Stop\_Counter

- If Run\_Stop\_Counter reaches a threshold based on Speed\_index, set Run\_Stop\_Mode to 0

Check the Received\_Command:

If it is 1 (Move forward):

- Measure the distance using the ultrasonic sensor

- If the distance is less than or equal to 30:

- Turn on the LED

- Set sound to 250

- Set Run\_Stop\_Mode to 0

- Else:

- Move all motors forward

If it is 2 (Move reverse):

- Move all motors backward

If it is 4 (Turn left):

- Turn left by controlling the motors

If it is 3 (Turn right):

- Turn right by controlling the motors

If it is 0 (Stop):

- Turn off all motor directions

If it is any other command:

- Turn off all motor directions

Delay for a short period

**Discussion:**

The project aims to develop a hand gesture-controlled car with obstacle avoidance using an Arduino Uno, an NRF module for wireless communication, and a gyro sensor for hand gesture recognition. The car is designed to move in different directions based on hand gestures and navigate around obstacles in its path. The project involves hardware setup, sensor integration, wireless communication, motor control, and programming. Through systematic development and testing, the hand gesture-controlled car successfully recognizes gestures and avoids obstacles.

**Conclusion:**

In conclusion, the hand gesture-controlled car with obstacle avoidance, utilizing the NRF module, gyro sensor, Arduino Uno, and sonar sensor, has been a challenging and rewarding project. It showcases the possibilities of gesture-based control and autonomous obstacle detection in robotics and automation. The project serves as a foundation for future advancements in the field, encouraging further exploration and innovation in the realm of interactive robotics.

**Applications:**

**1.Industrial Automation**: Gesture control can be integrated into industrial automation processes, allowing workers to control machines or robotic systems in a hands-free manner. This can improve efficiency, productivity, and safety in manufacturing or assembly line operations.

**2.Security Systems:** The gesture-controlled car's obstacle avoidance capabilities can be utilized in security systems to enhance surveillance and monitoring. The car can navigate autonomously, detecting and avoiding obstacles in restricted areas or patrolling environments where physical access is challenging.

**3.Healthcare and Rehabilitation**: Gesture-based control can be applied in healthcare and rehabilitation settings. It can assist in physical therapy exercises or rehabilitation programs, providing a more engaging and interactive experience for patients

**4. Gaming and Virtual Reality:** Gesture-based control can enhance the immersive experience in gaming and virtual reality environments.

**References:**

* <https://lastminuteengineers.com/nrf24l01-arduino-wireless-communication/>
* <https://www.youtube.com/watch?v=-0HCNMU6KrM>
* <https://sensorpartners.com/en/knowledge-base/everything-about-the-operation-principles-of-ultrasonic-sensors/>