Instrumentation Laboratory:
DISTANCE MEASUREMENT AND MONITORING

Mazen Mohamed, Muhamad Aiman Fikri Bin Mohd Fadzli, Muhammad Azlan Bin Abdul Azim, Omar Awad Abdelhadi, Omar Walaa Hassan Mohamed Hassan Elmasri
Faculty of Electrical Engineering, Universiti Teknologi Malaysia
FKE, UTM, 81310 Skudai, Johor
mahrous20@graduate.utm.my, mafikri28@graduate.utm.my, mazlan237@graduate.utm.my, abdelhadi@graduate.utm.my, walaa.omar@graduate.utm.my

***Abstract*—This paper describes the design steps of a car braking apparatus using a distance measurement system involving the application of the ultrasonic sensor.**

***Index Terms*--Arduino, Distance, LED, Sensor, Ultrasonic**

I. Introduction

The usage of sensors has steadily increased in this ever-growing world. The field of electronic Instrumentation has seen major innovations throughout the last twenty years among which is the ultrasonic sensor. This device allows us to detect the presence of an object through the calculation of the distance incurred. It essentially sends an ultrasonic signal from its emitter and gets back its reflection through the receiver. The time taken for the signal to travel from the emitter to the receiver is then used to approximate the distance with an appropriate calibration.

We decided to use this clever device to create an automatic car braking system to be installed in commercial vehicles in order to improve the general road safety.

II. Procedures

1. As a beginning, we conceived the hardware circuit according to the design requirements, essentially having the sensor at the front of the vehicle for optimal operation (see Figure 1). Since the whole system is smaller than the real-life application, we realized this experience using an LCD screen as our dominant scale.

2. The programming part became the next step as we needed to specify the working and limiting conditions for the apparatus (see Figure #). When the distance between the car and an object is above the range of 25cm, the car continues to move. As the object reaches below 25 cm away from the obstacle, the car stops moving to prevent accidents.

3. Lastly, we connected the microcontroller to the Arduino IDE to test its functioning. After multiple trials, we managed to make the system work as planned.

Hardware

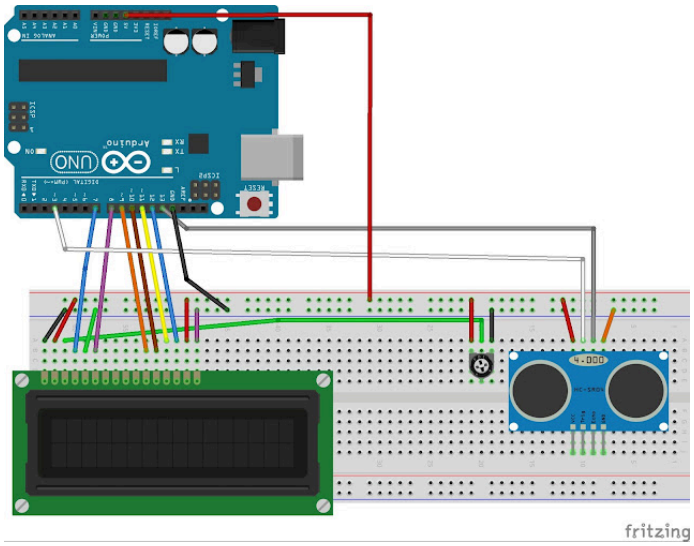

Figure 1: TinkerCad Simulation

For the reading we used LCD screen as our scale to display the distance between the car and the object

Figure 2: LCD screen

An LCD typically requires a ground connection, labeled as VSS, which connects to the ground of the power supply. The power supply pin, known as VDD, connects to a positive voltage source, usually 5V or 3.3V, depending on the specific requirements of the LCD. The Enable (E) pin is used to initiate the processing of data or commands sent to the LCD, and the Register Select (RS) pin is critical for determining whether the data sent to the LCD is a command—such as a command to clear the screen or position the cursor—or actual data meant to be displayed.

Figure 3:HC-SR04 Ultrasonic sensor

The HC-SR04 has a maximum range of 4 meters and a minimum distance of 2 cm which was suitable for the project since we conducted our tests in a controlled environment. It has a ranging accuracy of 3mm. It operates on 5 volts and 15mA of current that can easily be powered with batteries.

Figure 4: HC-SR04 Pin Layout

The HC-SR04 has 4 pins, the 1st and 4th being VCC and Ground. The 2nd pin is the trigger pin that is used to trigger the ultrasonic sound pulses. Lastly, the Echo pin produces a pulse when the reflected signal is received. The length of the pulse is proportional to the time taken for the transmitted signal to be detected.

When a pulse of at least 10 µS (10 microseconds) in duration is applied to the Trigger pin, the sensor transmits a sonic burst of eight pulses at 40 KHz. This 8-pulse pattern makes the "ultrasonic signature" from the device unique, allowing the receiver to differentiate the transmitted pattern from the ambient ultrasonic noise.

The eight ultrasonic pulses travel through the air away from the transmitter. Meanwhile, the Echo pin goes HIGH to start forming the beginning of the echo-back signal.

Lastly, we used Arduino Nano as our microcontroller

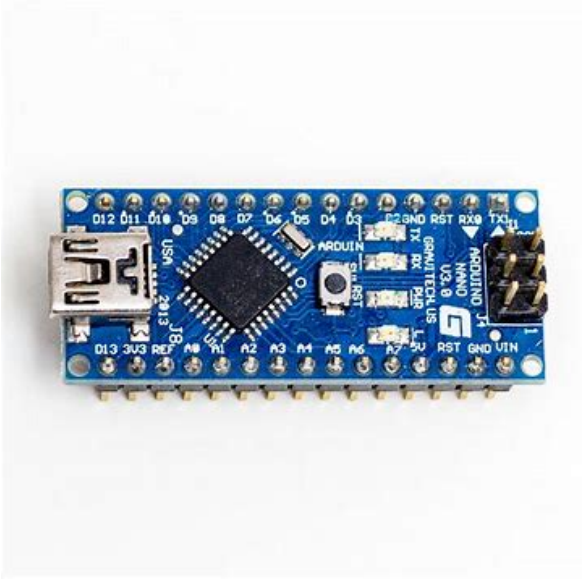

Figure 5: Arduino

Starting with the power options, the Arduino Nano can be powered through the VIN pin when connected to an external power source, which typically ranges from 6 to 12 volts. Alternatively, if a regulated 5V power supply is available, it can be directly supplied to the 5V pin, bypassing the need for external voltage regulation. For reading analog signals, the Arduino Nano is equipped with eight analog pins (A0 to A7), the Nano provides several pins capable of Pulse Width Modulation (PWM), specifically D3, D5, D6, D9, D10, and D11. PWM enables the control of devices like servos, LEDs, and other items requiring precise proportional control by emulating an analog output through digital signals.

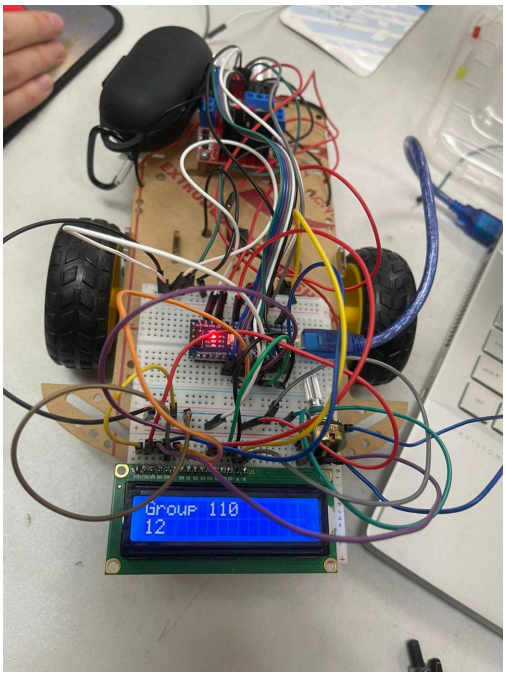

Figure 6: Hardware Setup for Testing

The circuit is assembled as shown above. It worked perfectly after uploading the program into the microcontroller via Arduino IDE. The system was able to measure the distance between the object and the sensor accurately after some calibration.

Program

The Arduino was programmed for the system to accomplish the stated system function using Arduino as a programming medium. The ultrasonic sensor, which acts as a distance measuring sensor, becomes triggered when the echo emitted through is reflected by any obstacle. Emitted ultrasounds travel forward until they get reflected by the object and then travel backward. The reflected ultrasound is then detected by the receiver.

When the reflected ultrasound is received by the receiver, *echoPin* is made low. Now we have the time taken by the ultrasound to reach the object and to reach the source that is also equal to the duration for which the *echoPin* was high. This recorded time is stored in the microcontroller. Therefore, the travel time of ultrasounds between the source and the object is half the time taken to travel the distance source-object-source (Figure 2). The formula is

written below:

Distance=Speed x Total Time Distance=(Time/2) x Speed Distance=(Time/2) x 29.1 **Where 29.1 is the speed of ultrasound in centimeters/microseconds.**

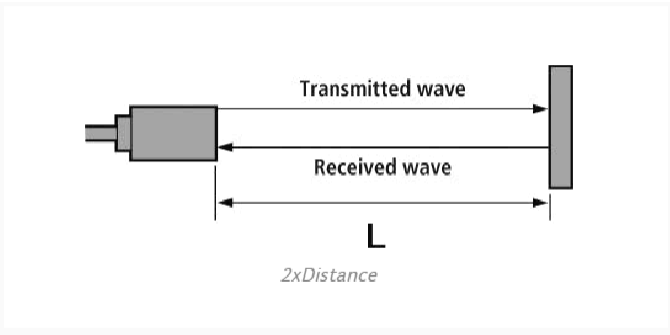

Figure 7: Ultrasonic sensor working principle

The programming flowchart is provided below:

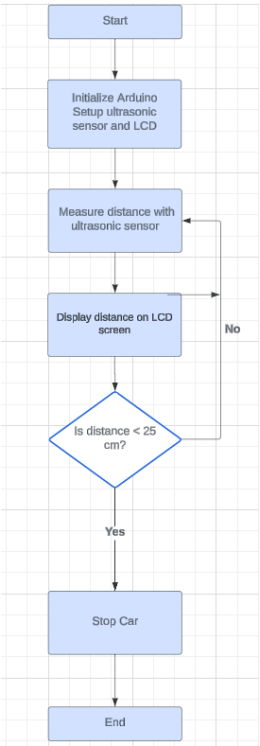

Figure 8: Flowchart of the program uploaded

The Arduino sketch provided controls a car using an ultrasonic sensor, an LCD screen, and an LED indicator. The program initiates by setting up the necessary components and defining pin modes for inputs and outputs. It uses the ultrasonic sensor to measure the distance to objects and displays this information on the LCD. If an object is detected within 25 centimeters, the car stops, indicated by the activation of a red LED. If the object is farther than 25 centimeters, the car continues to move forward. This loop of measuring and reacting runs continuously, allowing the car to dynamically respond to its surroundings by stopping to avoid collisions and moving when the path is clear. The integration of serial communication aids in debugging by providing real-time distance data through the serial monitor.

```
#include <LiquidCrystal.h>

#define TRIG_PIN A3
#define ECHO_PIN A2
#define RED_LED_PIN A4

int ENA = 10;
int IN1 = 6;
int IN2 = 5;
int ENB = 9;
int IN3 = 3;
int IN4 = 2;

LiquidCrystal lcd(12, 4, A5, 7, 8, 11);

void setup() {
 Serial.begin(9600);
 lcd.begin(16, 2); // Initialize the LCD
```

Figure 9: Arduino code 1 (Initialisation)

```
// Motor control based on distance
if (distance < 25) {
 // If distance is less than 25 cm, stop the car
 digitalWrite(IN1, LOW);
 digitalWrite(IN2, LOW);
 digitalWrite(ENA, LOW);

 digitalWrite(IN3, LOW);
 digitalWrite(IN4, LOW);
 digitalWrite(ENB, LOW);

 digitalWrite(RED_LED_PIN, HIGH); // Turn on red LED to indicate the car has stopped
} else {
 // If distance is greater than or equal to 25 cm, resume moving forward
 digitalWrite(IN1, LOW);
 digitalWrite(IN2, HIGH);
 analogWrite(ENA, 100);

 digitalWrite(IN3, HIGH);
 digitalWrite(IN4, LOW);
 analogWrite(ENB, 103);

 digitalWrite(RED_LED_PIN, LOW); // Turn ed LED
}
```

Figure 10: Arduino code 2 (Mechanism)

III. Data and Results

In our system wy had set the distance for the car to stop at 25 cm, but after trying many times the car doesn't stop exactly at 25 cm , due to the speed of the car it needs some and response to the signal coming from the ultrasonic sensor it needs some time and will completely stop at a distance less than 25 cm .

we have set the right motor speed to 100 and the left motor speed to 103, there is a difference in speed to make the car move straight instead of tilting to one side. Below is the data for the distance when the car fully stops.

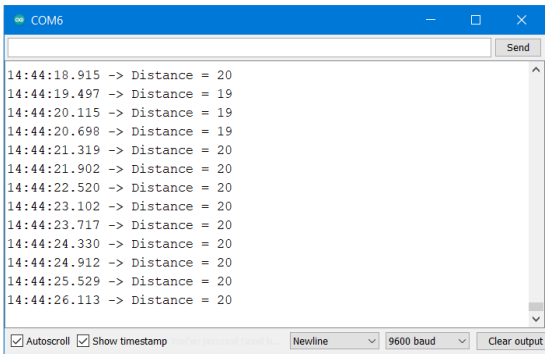

Figure 14: distance measured when car fully stops

we can see that generally the car stops completely at a distance of 20 cm, which means that it will take 5 cm for the car to stop with our specified speed.

IV. Discussion

To ensure accurate sensor readouts, we proceeded to perform a sensor characterization. Sensor characterization is the process of taking measurements from a sensor under controlled conditions in order to gather data related to its functioning in different circumstances and therefore allow for the most optimum application.

In this experiment, two testing scenarios were created in order to assess the prior reading accuracy of the HC-SR04 ultrasonic sensor, one before calibration and another after calibration as shown in Figure 4.

Figure 22: Sensor Readouts before and after calibration

For calibration, we had to tweak the reading formula in the programming code such as to add a value of 'one' to the detected distance, hence achieving the desired reading.

It can be inferred from the above experiment that the sensor is a fairly reliable and accurate device for distance measurements as long as it is being used according to the specifications mentioned on its respective datasheet. However, some other factors that may affect the accuracy of the readings should be kept in mind. For Instance, this experiment was conducted indoors and under relatively constant temperatures whereas in real life, this system would have to withstand various weather conditions that are often extreme alongside different types of noise which could inherently affect the accuracy of the readings. This provides some room for future studies and experiments to perform.

V. Conclusion

The purpose of this laboratory session was to use the ultrasonic sensor integrated in a distance measurement system in an automatic car braking set-up. The sensor was an adequate choice when it relates to distance measurements as the extracted data was relatively reliable and stable in our environment.

VI. References

[1] COMPONENTS 101. (22 April, 2020). NodeMCU ESP8266. Retrieved from components101: https://bit.ly/3NyKYxH

[2] HC-SR04 Ultrasonic Sensor: Working, Pin Diagram. https://bit.ly/3zv30wH

[3] (Anon n.d.).Lab 9: Sensor Characterization Lab (Digital). Retrieved from https://bit.ly/3NMjKDO

[4] Corp, M. (n.d.). Distance Measurement Sensors. Retrieved from Megatron Corp: https://bit.ly/3O1XfdV

[5] Engineer, L. M. (n.d.). How HC-SR04 Ultrasonic Sensor Works & Interface It With Arduino. Retrieved from last-minute engineers: https://bit.ly/3O30L7X (n.d.). NodeMCU ESP8266 Detailed Review. Retrieved on 03/06/202

VII. Appendix

```
#include <LiquidCrystal.h>

#define TRIG_PIN A3
#define ECHO_PIN A2
#define RED_LED_PIN A4

int ENA = 10;
int IN1 = 6;
int IN2 = 5;
int ENB = 9;
int IN3 = 3;
int IN4 = 2;

LiquidCrystal lcd(12, 4, A5, 7, 8, 11);

void setup() {
  Serial.begin(9600);
  lcd.begin(16, 2); // Initialize the LCD

  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(RED_LED_PIN, OUTPUT);

  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(ENB, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);

  // Set the initial motor direction and speed
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  analogWrite(ENA, 100);

  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  analogWrite(ENB, 103);

  digitalWrite(RED_LED_PIN, HIGH);
}

void loop() {
  // Ultrasonic sensor measurement
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  long duration = pulseIn(ECHO_PIN, HIGH);
  long distance = duration * 0.034 / 2;

  Serial.print("Distance: ");



  Serial.println(distance);

  // Display distance on LCD
  lcd.clear();
  lcd.setCursor(0, 0); // Start at the top-left corner
  lcd.print("Distance:");
  lcd.setCursor(0, 1); // Move to the second line
  lcd.print(distance);
  lcd.print(" cm");

  // Motor control based on distance
  if (distance < 25) {
    // If distance is less than 25 cm, stop the car
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, LOW);
    digitalWrite(ENA, LOW);

    digitalWrite(IN3, LOW);
    digitalWrite(IN4, LOW);
    digitalWrite(ENB, LOW);

    digitalWrite(RED_LED_PIN, HIGH); // Turn on red LED
to indicate the car has stopped
  } else {
    // If distance is greater than or equal to 25 cm, resume
moving forward
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
    analogWrite(ENA, 100);

    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
    analogWrite(ENB, 103);

    digitalWrite(RED_LED_PIN, LOW); // Turn off red LED
  }
}
```