MRE 320 Sensors and Actuators Individual Project

Milestone #1: Get familiar with your sensor and propose testing plans

Name: Zhirui Sun

Fall 2022 MRE 320

**Introduction:**

**Ultrasonic sensor:** Ultrasonic sensor is an electronic device that emits ultrasonic waves and converts the reflected sound into electrical signals to determine the distance of the object. Ultrasound travels faster than audible sound (that is, the sound that humans can hear). The transmitter that uses piezoelectric crystals to generate sound and the receiver that encounters sound after the sound travels back and forth to the target are the two main components of the ultrasonic sensor.

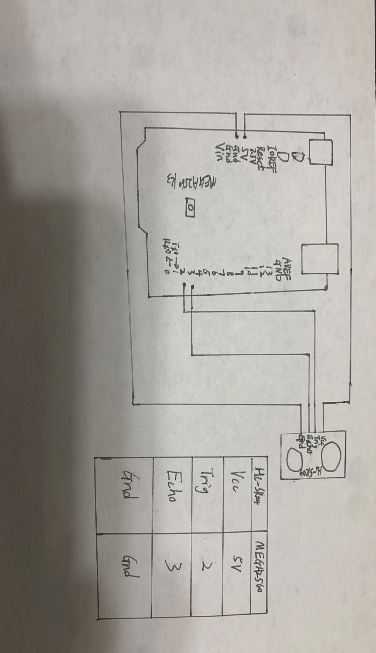
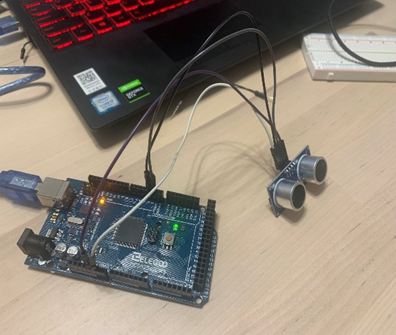
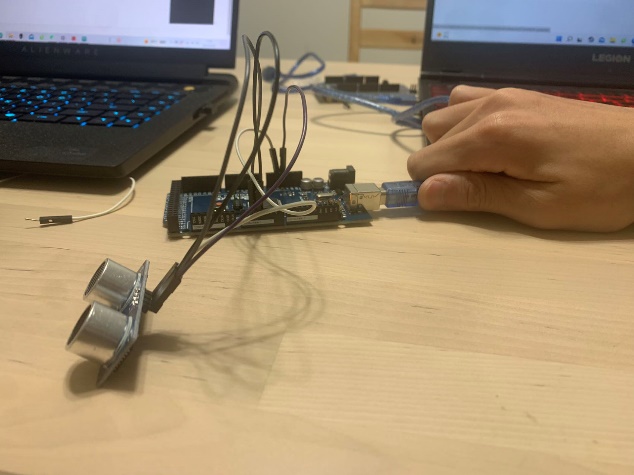
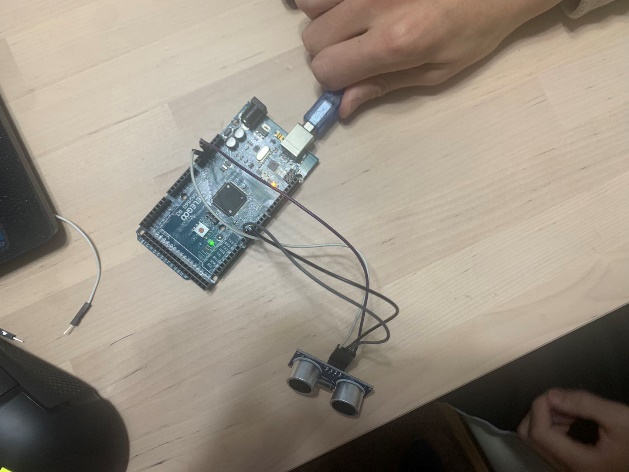
**Working principle of ultrasonic sensor：**The basic physical principle of ultrasonic sensor is that the sensor sends ultrasonic pulse and receives it back. Using the time difference between the transmitted signal and the received signal, the distance to the object can be determined. A common design is to build the transmitter and receiver into the same physical housing, although they can also be installed in separate units, such as some photoelectric sensors with separate transmitters and detectors.

**Procedure**

Part 1:

**Assignment:**

**(1)Wiring diagram of how you connect the sensor to Arduino, or a picture of your connected hardware**

**(2)Arduino code**

// ---------------------------------------------------------------- //

// Arduino Ultrasoninc Sensor HC-SR04

// Re-writed by Arbi Abdul Jabbaar

// Using Arduino IDE 1.8.7

// Using HC-SR04 Module

// Tested on 17 September 2019

// ---------------------------------------------------------------- //

#define echoPin 3 // attach pin D2 Arduino to pin Echo of HC-SR04

#define trigPin 2 //attach pin D3 Arduino to pin Trig of HC-SR04

// defines variables

long duration; // variable for the duration of sound wave travel

int distance; // variable for the distance measurement

void setup() {

  pinMode(trigPin, OUTPUT); // Sets the trigPin as an OUTPUT

  pinMode(echoPin, INPUT); // Sets the echoPin as an INPUT

  Serial.begin(9600); // // Serial Communication is starting with 9600 of baudrate speed

  Serial.println("Ultrasonic Sensor HC-SR04 Test"); // print some text in Serial Monitor

  Serial.println("with Arduino UNO R3");

}

void loop() {

  // Clears the trigPin condition

  digitalWrite(trigPin, LOW);

  delayMicroseconds(2);

  // Sets the trigPin HIGH (ACTIVE) for 10 microseconds

  digitalWrite(trigPin, HIGH);

  delayMicroseconds(10);

  digitalWrite(trigPin, LOW);

  // Reads the echoPin, returns the sound wave travel time in microseconds

  duration = pulseIn(echoPin, HIGH);

  // Calculating the distance

  distance = duration \* 0.034 / 2; // Speed of sound wave divided by 2 (go and back)

  // Displays the distance on the Serial Monitor

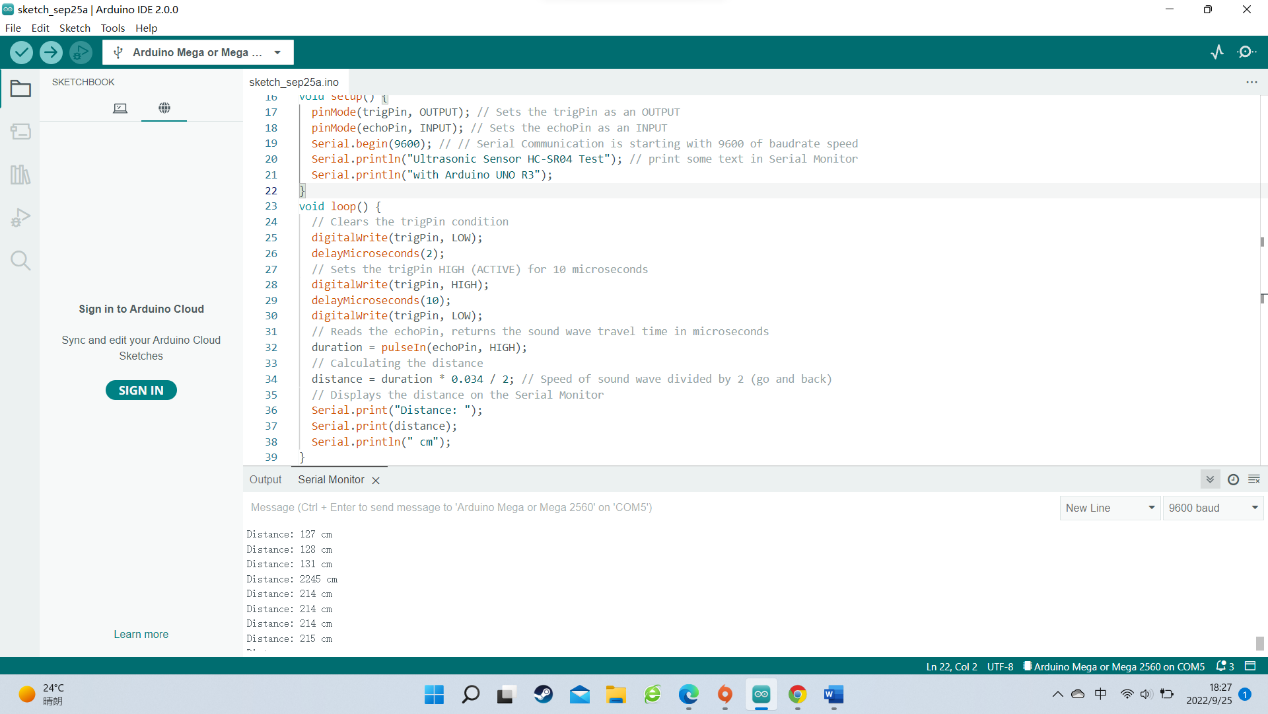
  Serial.print("Distance: ");

  Serial.print(distance);

  Serial.println(" cm");

}

**(3)Screenshot of the sensor reading**



**(4)The summary of the main findings from your sensor study**

In the study of ultrasonic sensors, when doing experiments, the connection circuit is a necessary procedure, but also the most important, and the connection circuit is the most important is careful. We both started doing the experiment, did not pay more attention to, not careful at the beginning of the circuit board into the short circuit, so we put the circuit connection good found that we cannot get the data after the power supply, no matter how to adjust the wrong, afterwards just know is wrong, our circuit connection in the end we only in a serious examination, see you where is wrong. With this experience, I will be more careful next time. This experiment let us understand how the ultrasonic sensor works, and how it is at this distance, and understand his test range, there are certain gains.

Part 2

**Sensor parameter table：**

|  |  |
| --- | --- |
| **Specifications** | **Measurement** |
| Working Voltage | DC 5V |
| Working Current | 15mA |
| Working Frequency | 40Hz |
| Max Range | 4m |
| Min Range | 2cm |
| Measuring Angle | 15 degree |
| Trigger Input Signal | 10µS TTL pulse |
| Echo Output Signal | Input TTL lever signal and the range in proportion |
| Dimension | 45 \* 20 \* 15mm |

Analysis of static characteristics

**(1)Linearity:**

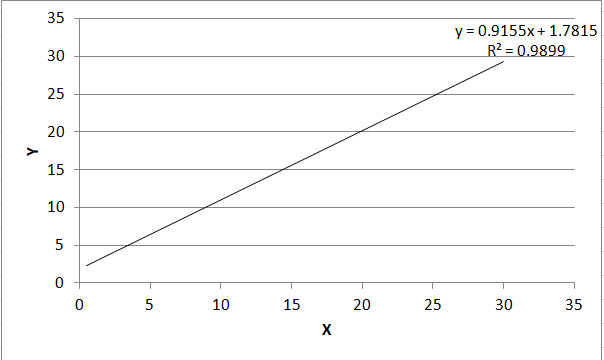
**Equipment:**A ruler(0-30cm), A book(or any flat item that can reflect sound waves),

Arduino UNF, HC-SR04 Ultrasonic Distance Sensor

**Methods:** First, a flat surface book is used as the test object. The ultrasonic sensor is placed 0 cm from the ruler, and the book is gradually moved away from the ultrasonic sensor. Start at 0 centimeters and move away until you reach 30 centimeters.

**Explanation of experimental procedures:** X is the actual distance from the book to the ultrasonic sensor, and Y is the measured distance from the ultrasonic sensor. Because x and y form a function to see if it's linear. If the function of x and y tends to a straight line, then they're linear, and if they're not, then they're not linear.

|  |  |  |
| --- | --- | --- |
| X=Move the object | Y=Serial output [Distance(cm)] | Error(%) |
| 0.0 | 0 | INF |
| 0.5 | 4.15 | 730 |
| 1.0 | 3.43 | 243 |
| 1.5 | 3.07 | 104.7 |
| 2.0 | 4.01 | 100.5 |
| 4.0 | 4.10 | 2.5 |
| 6.0 | 6.05 | 0.83 |
| 8.0 | 8.66 | 8.25 |
| 12.0 | 12.04 | 0.33 |
| 18.0 | 18.14 | 0.78 |
| 24.0 | 24.15 | 0.625 |
| 30.0 | 29.76 | 0.8 |



**Summary:** It is nonlinerity between X and Y when distance below to 4cm, because the distance is too close, so the sound basically does not need the propagation speed, but the sensor is based on the influence of sound speed, which leads to the non-linear relationship between X and y. At 4 to 30 centimeters, the function is linear

**(2)resolving power**

**Methods:** duration = pulseIn(echoPin, HIGH);

distance = duration \* sound\_speed / 2;

**Explanation of experimental procedures:** It seems that the resolution of this sensor is dependent on the resolution given by the pulseIn() function.

**Summary:** yes, pulseln()=2distance/sound\_speed

**(3)Sensitivity**

**Equipment:** A ruler(0-30cm), A book(or any flat item that can reflect sound waves),

Arduino UNF, HC-SR04 Ultrasonic Distance Sensor

**Methods:** The ratio of the output increment to the corresponding input increment that causes the increment

|  |  |  |
| --- | --- | --- |
| X=Move the object | Y=Serial output [Distance(cm)] | Error(%) |
| 0.0 | 0 | INF |
| 0.5 | 4.15 | 730 |
| 1.0 | 3.43 | 243 |
| 1.5 | 3.07 | 104.7 |
| 2.0 | 4.01 | 100.5 |
| 4.0 | 4.10 | 2.5 |
| 6.0 | 6.05 | 0.83 |
| 8.0 | 8.66 | 8.25 |
| 12.0 | 12.04 | 0.33 |
| 18.0 | 18.14 | 0.78 |
| 24.0 | 24.15 | 0.625 |
| 30.0 | 29.76 | 0.8 |

**Summary:** Due to the limited range of the equipment, the sensitivity value is within the acceptable error range within a reasonable range.

**(4)****measuring range:**

**Equipment:** Tape measure, A book(or any flat item that can reflect sound waves),

Arduino UNF, HC-SR04 Ultrasonic Distance Sensor

**Methods:** First, use a plane book as the test object. Secondly, the ultrasonic sensor is placed at a distance of 0 cm from the ruler, and the book is gradually away from the ultrasonic sensor.

**Explanation of experimental procedures:** Move from 0 cm and observe the measured value until there is a significant error in the measured value. In this way, we can judge the approximate value of the measurement range.

**Summary:** Through measurement, we know that after 0 to 4 cm and 200 cm, the value is obviously wrong, so we know that range is 4-200 cm. The measuring range is 196cm.

**(5)Stability:**

**Equipment:** A ruler(0-30cm), A book(or any flat item that can reflect sound waves),

Arduino UNF, HC-SR04 Ultrasonic Distance Sensor

**Methods:** Fix the equipment and turn it on for a long time to observe whether the measured value change76s.

**Explanation of experimental procedures:** Through this method, we can know whether the ultrasonic sensor can work normally for a long time.

**Summary:** After a long time of testing, we found that it has high stability and can work for a long time.