

# Arduino Distance Indicator

Lauren Williams

*Dept. of Electrical and Computer Engineering  
University of Tennessee, Knoxville  
Knoxville, United States  
lwill168@vols.utk.edu*



Ally Gosser

*Dept. of Electrical and Computer Engineering  
University of Tennessee, Knoxville  
Knoxville, United States  
agosser@vols.utk.edu*



***Abstract***—This paper explains the development and implementation of an Arduino-based distance indicator system using an HC-SR04 ultrasonic sensor. The system uses the ultrasonic sensor connected to an Arduino Uno to measure distances. It shows the distance in real time using LEDs, helping users understand how close or far an object is. The indicator is designed to alert users when an object enters a certain range, which can be helpful for basic obstacle detection. The project is low-cost, simple to make, and great for beginners. The paper also talks about common issues like sensor accuracy and suggests ways to improve the system. Possible uses and future upgrades are also included.

***Keywords***—*Ultrasonic Sensor, LED Display, Electronics, Distance Measurement, Object Detection*

## I. INTRODUCTION

Ultrasonic sensors are commonly used for measuring distances. They work by sending sound waves and measuring how long it takes for the waves to return after bouncing off an object. While these sensors are useful, they can have some limitations, like not always being very accurate or having a short range. In manufacturing, workers often spend a lot of time guessing when a part will arrive on the assembly line. This constant checking can be tiring and slow, which increases production time and costs. To solve this, our team created an Arduino Distance Indicator - a simple visual system that shows where the part is on the line. This paper presents a new and improved way to use ultrasonic sensors with an Arduino Uno to create a simple and accurate distance indicator using LED lights.

## II. MATERIALS AND METHODS

### A. Materials

The system uses the following components: [1]

- **Arduino Uno:** The main microcontroller that processes the data from the sensor and controls the display
- **Ultrasonic Sensor (HC-SR04):** The sensor sends out sound waves to measure distance by timing how long it takes for the sound to return
- **LED Display:** Lights with turn on if an object is close to the sensor
- **Jumper wires, resistors, and breadboard:** For connecting and setting up the circuit
- **Computer Connection and Arduino IDE Software:** Power for the arduino and load code for circuit to work

### B. Methods

#### a. Background

Most current systems use expensive equipment like cameras or special sensors. Our Arduino-based system is a cheaper and simpler option, perfect for smaller production lines. For larger factories, companies can try the Arduino Distance Indicator to improve worker productivity before deciding to invest in more complex systems like LIDAR or ultrasonic devices.

#### b. Procedure

For the Arduino Distance Indicator, connect the ultrasonic sensor's VCC and GND pins to the 5V and GND

pins on the Arduino from the breadboard. The Trigger pin connects to pin 9, and the Echo pin goes to pin 10. For the LED setup, connect the positive (long) leg of each LED to designated Arduino pins (2, 3, 4, 5, 6). [1] Five LEDs in total. Now, add the  $220\Omega$  resistors to the negative (short) leg of the LED. Connect all LEDs to GND by adding one wire to the positive leg. [2] The full Arduino configuration is shown in figure 1 below. You will then connect the arduino to your computer and run the following code that is listed below (Figure 2).

## III. RESULTS AND DISCUSSION

### A. Results

The system was tested in various conditions to evaluate its accuracy, range, and responsiveness. When an object is placed between 10 cm and 400 cm from the sensor, the correct LED is displayed on the breadboard. Figure 3 and Figure 4 show the LED display. The accuracy of the readings was within 1-2 cm in most cases, which is consistent with previous works on distance measurement using ultrasonic sensors. [3] We also tested the system by comparing its distance readings with measurements taken using a tape measure. The system's results were very close, with differences of less than 2% most of the time. This level of accuracy is similar to what was reported in other projects using Arduino and ultrasonic sensors. [4]

## B. Discussion

The sensor successfully detected objects from 10 cm to 400 cm away. This range falls within the specified capabilities of the HC-SR04 ultrasonic sensor, making it suitable for a variety of applications where medium-range distance measurement is needed. This project can be used in different areas:

- **Manufacturing:** A simple visual system that alerts them to the part's location, improving efficiency and reducing fatigue.
- **Robotics:** It could help robots detect obstacles and move around them safely
- **Personal Safety:** It could be added to devices that warn users about nearby objects, which is helpful for people with visual impairments
- **Automation:** It can improve automated systems, like security alarms or warehouse tools

However, there are some challenges. For example, noisy environments or objects that absorb sound can reduce accuracy. To make the system better, sensors with longer ranges could be used in the future.

## IV. CONCLUSION

This project successfully created a simple and effective Arduino-based distance indicator using an ultrasonic sensor. The system is easy to use and provides accurate, real-time distance readings, making it reliable for many different applications. In manufacturing, for example, this distance

indicator can help workers track where parts are on the assembly line. It gives workers alerts in time, reducing the need for constant checking, cutting down on fatigue, and improving overall productivity.

Besides manufacturing, the system can be used in robotics for tasks like detecting objects or helping robots move accurately. In personal safety, it can monitor how close someone is to dangerous machines or obstacles. It can also be used in automation systems to improve efficiency.

Future updates, such as using better sensors or adding temperature adjustments, could make the system more accurate and useful in even more situations. These changes could help the system work in more complex manufacturing setups or in other industries where measuring distance is important.

## ACKNOWLEDGMENT

We want to thank our professor and teaching assistant, Dr. Icove and Dr. Taousser, for their support and guidance throughout this project. We are very grateful for their lectures and help throughout this semester.

## REFERENCES

- [1] ALI, Shoeb. "Arduino Distance Indicator (Arduino + Ultrasonic Sensor +led )." Instructables, Instructables, 16 Oct. 2020.
- [2] Gabriel, Mutinda Mutava, and Kamweru Paul Kuria. "Arduino Uno, ultrasonic sensor HC-SR04 motion detector with display of distance in the LCD." *International Journal of Engineering Research And*, vol. V9, no. 05, 29 May 2020.
- [3] Srijith, Biyyala. "Arduino based distance measurement sensor using ultrasonic sensor." *International Journal for Research in Applied Science and Engineering Technology*, vol. 9, no. VI, 20 June 2021.

- [4] J. Smith, "Robotic Systems and Ultrasonic Sensors: Advancements and Challenges," IEEE Robotics and Automation Magazine, vol. 26, no. 4, pp. 45-52, Dec. 2020.

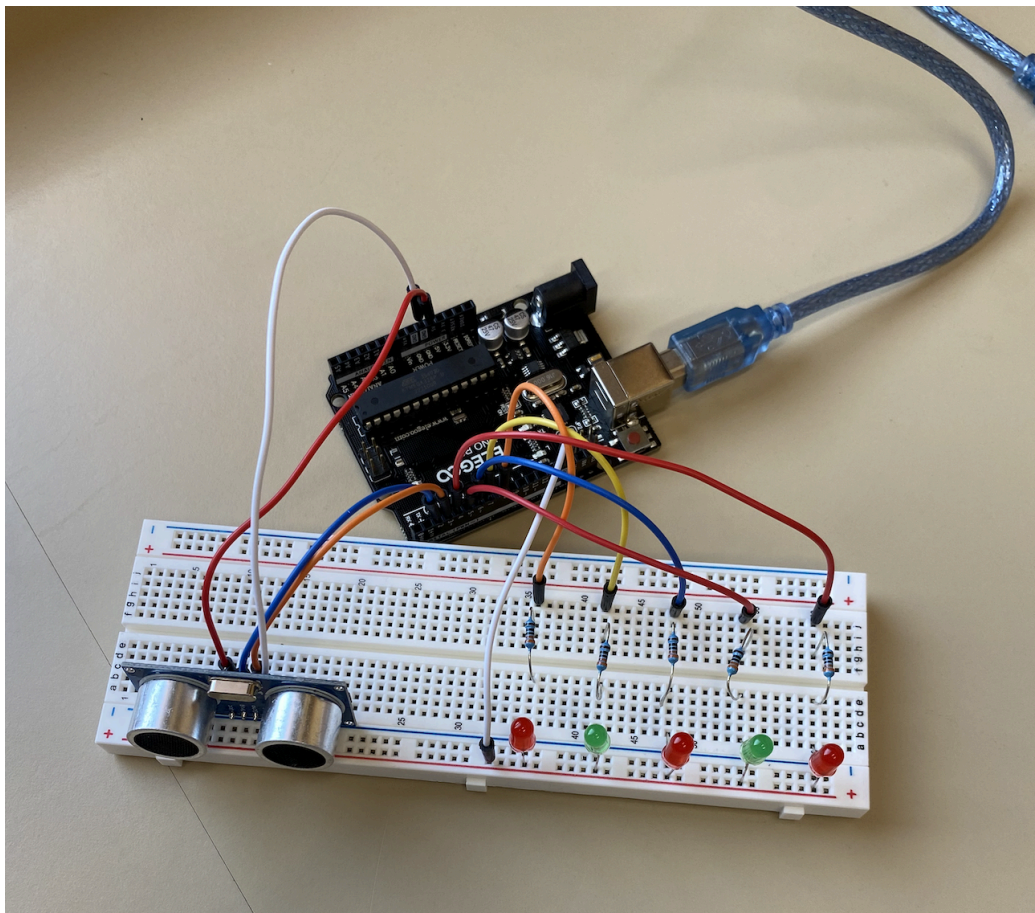


*Figure 1. Arduino Configuration*

```

int Time ;
int distance;

//setup the arduino with wires put into the corresponding numbers
void setup()
{
    // Set pin 2 as an OUTPUT pin (used to send signals or control devices like LEDs, motors, etc.)
    pinMode(2, OUTPUT);

    // Set pin 3 as an INPUT pin (used to receive signals, like from a button or sensor)
    pinMode(3, INPUT);

    // Set pin 4 as an OUTPUT pin
    pinMode(4, OUTPUT);

    // Set pin 5 as an OUTPUT pin
    pinMode(5, OUTPUT);

    // Set pin 6 as an OUTPUT pin
    pinMode(6, OUTPUT);

    // Set pin 7 as an OUTPUT pin
    pinMode(7, OUTPUT);

    // Set pin 8 as an OUTPUT pin
    pinMode(8, OUTPUT);

    // Start serial communication at a baud rate of 9600 (for debugging or communicating with a computer)
    Serial.begin(9600);
}

}

void loop()
{
    // Send a pulse on pin 2 to trigger the sensor
    digitalWrite(2, HIGH);      // Set pin 2 HIGH (start of the pulse)
    delayMicroseconds(10);      // Keep it HIGH for 10 microseconds
    digitalWrite(2, LOW);       // Set pin 2 LOW (end of the pulse)
    delayMicroseconds(5);       // Wait 5 microseconds before the next HIGH state
    digitalWrite(2, HIGH);      // Start another short pulse
    delayMicroseconds(10);      // Keep it HIGH for 10 microseconds again
}

```

```

// Measure the duration of the HIGH pulse received back on pin 3
Time = pulseIn(3, HIGH);    // pulseIn measures the time for which pin 3 stays HIGH

// Calculate the distance based on the time of flight
// Speed of sound is 343 m/s or 0.0343 cm/microsecond, divide by 2 for round trip
distance = 0.0343 * Time / 2;

// Print the calculated distance to the serial monitor
Serial.print("Distance from sensor to object is = ");
Serial.println(distance);

// Small delay before repeating the loop
delay(5);
}

// Check if an object is between 10cm and 50cm away
if(distance > 10 && distance <= 50)
{
    // Turn on lights connected to pins 4, 5, 6, and 7
    digitalWrite(4, HIGH);
    digitalWrite(5, HIGH);
    digitalWrite(6, HIGH);
    digitalWrite(7, HIGH);
    // Turn off the light connected to pin 8
    digitalWrite(8, LOW);
}

// Check if an object is between 50cm and 100cm away
if(distance > 50 && distance <= 100)
{
    // Turn on lights connected to pins 4, 5, and 6
    digitalWrite(4, HIGH);
    digitalWrite(5, HIGH);
    digitalWrite(6, HIGH);
    // Turn off lights connected to pins 7 and 8
    digitalWrite(7, LOW);
    digitalWrite(8, LOW);
}

// Check if an object is between 100cm and 250cm away
if(distance > 100 && distance <= 250)
{

```

```
// Turn on lights connected to pins 4 and 5
digitalWrite(4, HIGH);
digitalWrite(5, HIGH);
// Turn off lights connected to pins 6, 7, and 8
digitalWrite(6, LOW);
digitalWrite(7, LOW);
digitalWrite(8, LOW);
}

// Check if an object is between 250cm and 400cm away
if(distance > 250 && distance <= 400)
{
    // Turn on the light connected to pin 4
    digitalWrite(4, HIGH);
    // Turn off lights connected to pins 5, 6, 7, and 8
    digitalWrite(5, LOW);
    digitalWrite(6, LOW);
    digitalWrite(7, LOW);
    digitalWrite(8, LOW);
}

// Check if an object is more than 400cm away
if(distance > 400)
{
    // Turn off all lights connected to pins 4, 5, 6, 7, and 8
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    digitalWrite(6, LOW);
    digitalWrite(7, LOW);
    digitalWrite(8, LOW);
}
```

*Figure 2. Arduino IDE code on computer program*

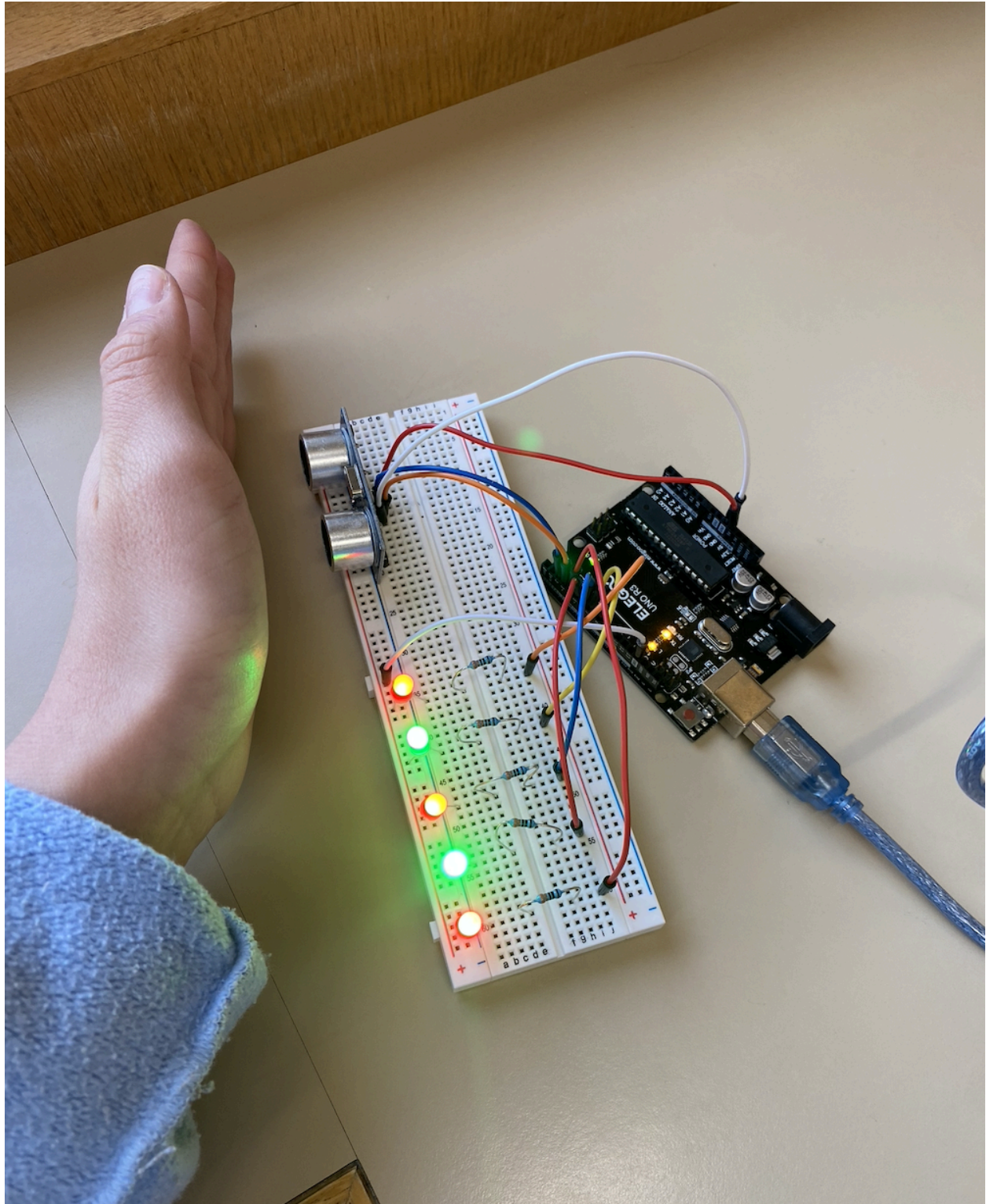


Figure 3. Full LED display

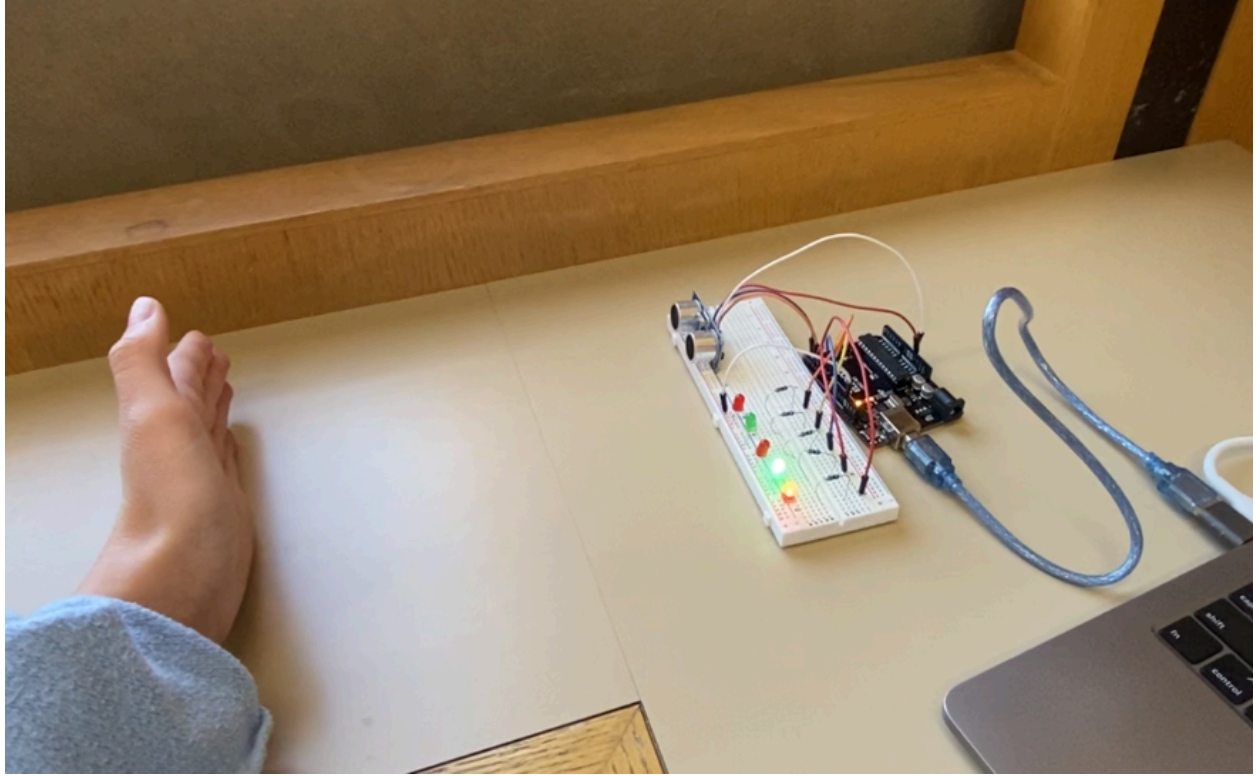


*Figure 4. Partial LED display*