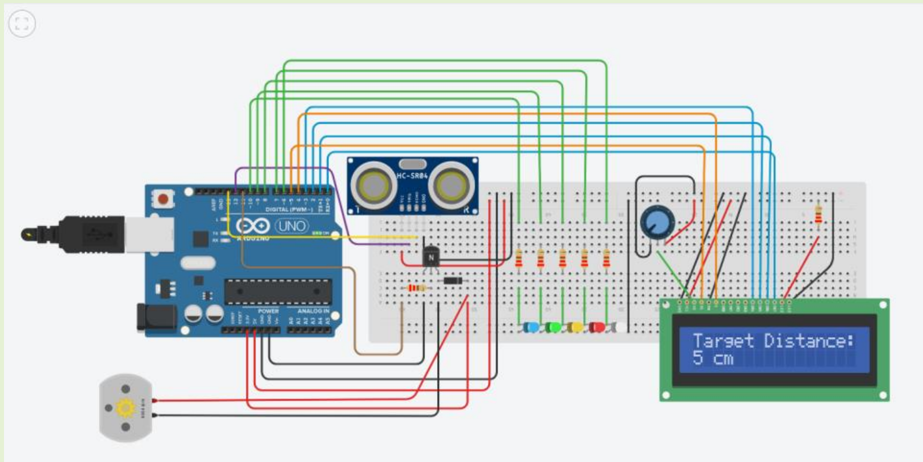
Distance Measuring and Fan Cooling LED Ultrasonic Sensor

Kevin Nguyen, 217228255

Abstract—A summary of my project, I created an LED ultrasonic sensor through various techniques. The application for my project would be used for measuring distances between objects and detecting if there is a presence in front of the ultrasonic sensor. Another application of my project would work as a hands-free device for example a fan that turns on just by being in front of the ultrasonic sensor. Some of the techniques that I used were prototyping through TINKERCAD. Using TINKERCAD I was able to make a schematic for my project and a diagram showing where each component would need to be placed. By using TINKERCAD I was able to code and test out if the components work and if there were any issues with the wiring and assembling of my project. Using TINKERCAD has helped managed my time more effectively because I did not have to assemble my project yet and was able to make some changes during the process. Another technique I used was using the 3.3V and 5V pin so that the Arduino DUE could power both the DC motor and LCD display. The result of the project was a success because I was able to fill all the criteria that was needed in this project such as using a DC motor, LCD display, and ultrasonic sensor. For the continuation of this project, I would improve it by optimizing the hardware and software. I could use less jump wires by using a different type of LCD display that only requires 4 output pins instead of 12. Its future application would be in presence detection, and measurements. Presence detection could be use in smart homes when the sensor in your home detects if there is a person in a room, if so then the light bulb turns on. Presence detection could also be used in self driving cars where the car can detect how far an object is from the car and would use that information to drive safer by stopping or slowing down the car when an object is in front of it. It could also be used in robotics for obstacle detection system. Measurement would be done much easier without using a ruler and just by using a sensor that can measure how far or close an object is. Thus, this project would show that there are many possibilities that it can apply to in the future.



Index Terms—Ultrasonic sensor, LCD display, DC motor, TINKERCAD, Arduino DUE

I. Introduction

THIS semester in EECS 2032 we were takes with creating an Arduino system using knowledge gained from the class and from outside sources. This was a passion project because it was my first-time using Arduino. The reason and motivation to why I decided to pick this project is because I was inspired from smart light bulbs, smart/self-driving cars and many other technologies related to using motion capturing sensor. I also wanted to make a hands-free device that allows me to turn on the fan or lights when my presence is in front of the sensor. I decided to create a Distance Measuring and Fan Cooling LED Ultrasonic Sensor that measures the distance of the object from the sensors and turns on the LEDs or fan base on the distance of the object. When the chance arrived to create a project using Arduino, I was excited and took it as a challenge for me as a computer engineer to branch out and do more projects like this. To add on, I was not familiar with Arduino and knew that I could not do a very complex project as I would not be able to successfully complete it, while attending other to other courses. This led me to take inspirations and ideas from previous projects and recognized that somebody had created a similar project using the HC-SR04 Ultrasonic Sensor that resonates with me. I chose to add more things to the project to make it different from the one that inspired me such as a DC motor and LEDs. A reference I used to start the project and learned about the ultrasonic sensor was, [1] Link 1, and this taught me how the ultrasonic sensor works, how to calculate distance using the ultrasonic sensor, and how to display values on the LCD display. Some of the challenges faced during the project were the ordering and receiving of components. Some of the components I wanted to order were either too expensive or out of order. When it came to receiving the components, the time duration of shipping of these components can be unpredictable and can range from three days to more than one week. While waiting for the components to arrive, I was able to make use of what I currently have. While dealing with the challenges of wiring and prototyping my project, I made sure to keep in check of the wiring for my device as it can get messy, especially for the LCD display because the LCD display had many pins to keep in mind. Another challenge I faced was in the conversion of the values from the ultrasonic sensors to an actual metric value. With my inexperienced with Arduino, I had to find a way to do some difficult tasks with the help of the internet. The potential market for my project is already implemented using smart light bulbs, smart/self-driving cars, home security systems, smart fans, robotics, and many more. As simple as it is to turn on the lights in your bedroom yourself, most people are very lazy and would rather if the light bulb turned on itself

Fig. 1. Motion Sensor light bulbs that turn on when there is some sort of motion during the night. Which acts as a security light bulb.

when a person enters a room. My project does differ from this product as it uses an ultrasonic sensor instead of a PIR motion sensor that product uses. While the [2] product example provides more functions and quality than the device from my project. I would say that the device from my project most closely resembles the product in terms of usage and practicality.

II. Proposed Project

This is a detail analysis of my project. My project can be broken down into two parts software and hardware components.

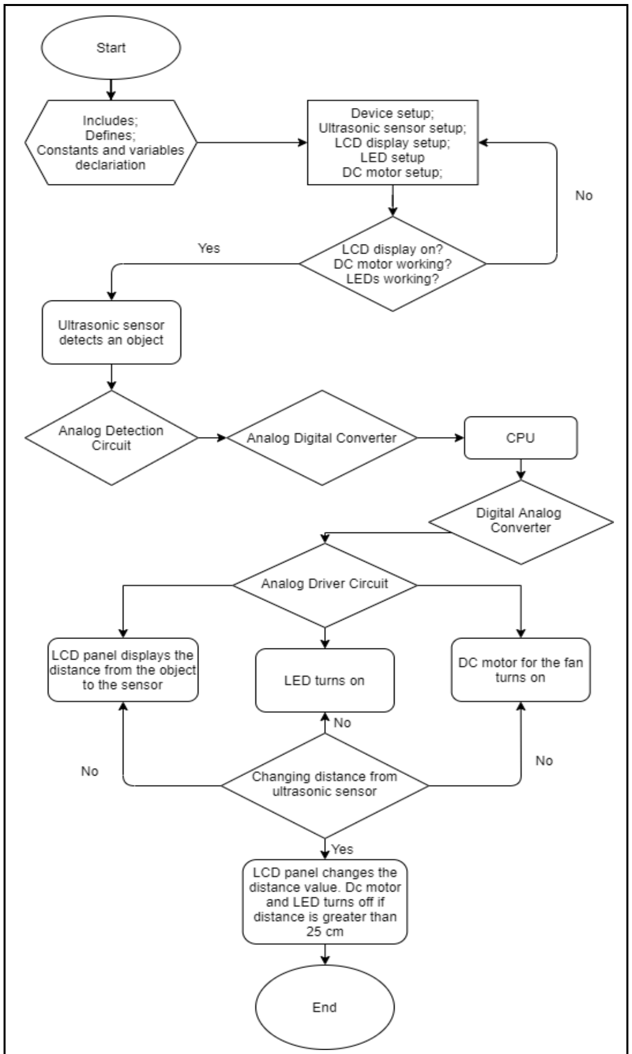


Fig. 2. Program flowchart depicting how the program works.

A. Program

Base on the Figure 2, we can see how the program works and the decision-making process in the program. To start off, the code begins by defining the variables, constants, and libraries. Next, the device is set up making sure all the components are functioning. Then, as the device proceeds to work, the ultrasonic sensor would sense an object. The ultrasonic sensor would transfer the analog signal of the sound waves back to the Arduino DUE where it would be converted to a digital signal and calculates the distance of the object from the sensor. Once the calculation has been completed the value of the distance would be display on to the LCD display. If the distance is greater than 25 cm then the LEDs and DC motor would not turn on. If the distance is 25 cm or less, then the blue LED would turn on. If the distance is 20 cm or less, then the green LED would turn on. If the distance is 15 cm or less, then the yellow LED and DC motor would turn on. If the distance is 10 cm or less, then the red LED would turn on. If the distance is 5 cm or less, then the white LED would turn on. Base on this flowchart the LEDs and DC motor turns on or off depending on the distance of the object from the sensor as the whole process would be repeated.

```
// includes the LiquidCrystal Library
#include <LiquidCrystal.h>

// Creates an LCD object. Parameters: (rs, enable, d4, d5, d6, d7)
LiquidCrystal lcd(5, 4, 3, 2, 1, 0);

const int trigPin = 12;
const int echoPin = 13;
long duration;
int distance;
int motorPin = 11;
```

Fig. 3. Libraries, LCD object, and initial variables used in program

As you can see in the above figure, I selected the

LiquidCrystal library to be able to use the function to program the LCD display. I would then use const variable trigPin and echoPin to initialized where the ultrasonic sensor would be on the Arduino DUE which is Digital (PWM) pin 12 and 13. I had to initialize the duration and distance variables which would be needed to calculate the distance of the object from the sensor. I also initialize motorPin where the DC motor would be on the Arduino DUE which is Digital (PWM) pin 11.

```
void setup()
{
  //LCD display
  lcd.begin(16,2); // Initializes the interface to the LCD screen

  //HC-SR04 Ultrasonic Sensor
  pinMode(trigPin, OUTPUT); // trigPin is OUTPUT
  pinMode(echoPin, INPUT);  // echoPin is INPUT

  //DC motor
  pinMode(motorPin, OUTPUT);// motorPin is OUTPUT

  //LEDs
  for (int i = 6; i <= 10; i++)
  {
    pinMode(i, OUTPUT);    // // LEDs pin is OUTPUT
  }
}
```

Fig. 4. In void setup() all the components are setup which includes LCD display, ultrasonic sensor, DC motor, and LEDs.

The figure above shows us the void setup() for all the components on the device which includes the LCD display, ultrasonic sensor, DC motor, and LEDs. For the setup of the LEDs, I had to setup the LEDs by making a for loop. In the for loop the variable “i” represents the Digital (PWM) of 6 to 10. The DC motor and LEDs are outputs, while for the HC-SR04 ultrasonic sensor the trigPin would be the output while the echoPin would be the input.

```
void loop()
{
  // Write a pulse to the HC-SR04 Trigger Pin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Measure the response from the HC-SR04 Echo Pin
  duration = pulseIn(echoPin, HIGH);

  // Determine distance from duration
  // Use 343 m/s as speed of sound
  distance= duration*0.034/2;

  // Prints "Target Distance: " on the first line of the LCD
  // Set cursor to first column of first row
  lcd.setCursor(0,0);
  lcd.print("Target Distance: ");
  //Print blanks on the second line of the LCD to clear the line
  // Set cursor to first column of second row
  lcd.setCursor(0,1);
  lcd.print("                "); // to clear the distance value
  // Prints "<value>" on the second line of the LCD
  // Set cursor to first column of second row
  lcd.setCursor(0,1);
  lcd.print(distance);
  lcd.print(" cm");
  delay(800);       // longer delay would cause less flickering
```

Fig. 5. In void loop() is the code for the function of the HC-SR04 ultrasonic sensor. The code for the LCD display to display the distance value is also there.

The figure above shows us the void loop() for the HC-SR04 ultrasonic sensor and LCD display. For the code of the ultrasonic sensor, I had to write a pulse to the HC-SR04 Trigger Pin and with that measure the response from the HC-SR04 Echo Pin. Finally, I determine the distance from the duration by using the formula distance = duration*0.034/2. Sound travels at about 343 m/s in dry air at 20 °C. HC-SR04 gives you the total time t in microseconds. So, if we wish to calculate the distance in centimeter then we would need to make the following two conversions. First, convert speed of sound in cm/s. Next, convert seconds to microseconds since the received value is in microseconds. Finally, after the two conversion you calculate the distance by taking your value after the conversion, 0.034 and multiply by the duration and then divide by 2 because this is the total time taken by the sound waves from transmitter to the object and back to the

HC-SR04's receiver. Because I wanted to learn the math behind calculating distance using Ultrasonic sensor HC-SR04, I used this [3] website. As for the LCD display, it was simple to implement inside the code. All I had to do was set the cursor at which row and column I wanted to print on the display. I wanted to print "Target Distance: " at the first row and first column of the display and print the distance value at the second row and first column of the display. I would clear the distance value in the loop so that new distance value can be updated constantly. I also added a delay of 800 so that the LCD display would not flicker when updating the new distance values.

```
//For DC motor
if (distance <= 15){   // Turn DC motor on if distance is 15cm or less
  digitalWrite(motorPin, HIGH);
}
else{
  digitalWrite(motorPin, LOW);
}
```

Fig. 6. Code for DC motor. If the distance is 15 cm or less, then the DC motor turns on.

The figure above shows us the code for the DC motor. This code is simple, I made sure the DC motor turns on only when the distance value is 15 cm or less and turns off when the distance value is greater than 15 cm.

```
// For LED
if (distance <= 5) {   //Turn LED on if distance is 5cm or less
  digitalWrite(6, LOW);
}
else{
  digitalWrite(6, HIGH);
}

if (distance <= 10) {  //Turn LED on if distance is 10cm or less
  digitalWrite(7, LOW);
}
else{
  digitalWrite(7, HIGH);
}

if (distance <= 15) {  //Turn LED on if distance is 15cm or less
  digitalWrite(8, LOW);
}
else{
  digitalWrite(8, HIGH);
}

if (distance <= 20) {  //Turn LED on if distance is 20cm or less
  digitalWrite(9, LOW);
}
else{
  digitalWrite(9, HIGH);
}

if (distance <= 25) {  //Turn LED on if distance is 25cm or less
  digitalWrite(10, LOW);
}
else{
  digitalWrite(10, HIGH);
}
}
```

Fig. 7. Code for LEDs. I can turn LEDs on or off base on the distance of an object.

The figure above shows us the code for the LEDs. The LEDs turn on or off depending on the distance of the object from the sensor. All the LEDs would be off if the object is greater than 25 cm and all the LEDs would be on if the object is within 5 cm or less. The blue LED would be the first LED to light up if the object is within the range of 25 cm or less, while the white LED would be the last LED to light up if the object is within the range of 5 cm or less. As for the other LEDs they would turn on within their own respective range. The main software design technique mostly came from learning how to calculate the distance of the object from the sensor using the formula distance = duration*0.034/2. Lastly, the reference for my code came from, [1] Link 1, and using examples from Arduino.

B. Hardware

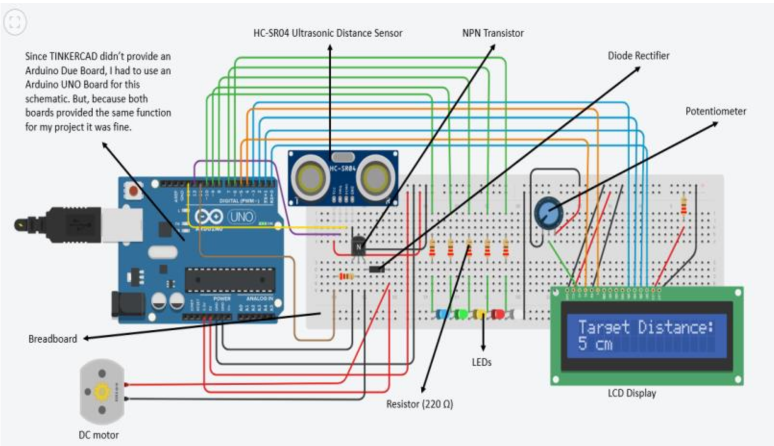


Fig. 8. Schematic of System.

My system consists of several components which includes Arduino DUE, HC-SR04 ultrasonic sensor, LCD display, DC motor, 5 colored LEDs, 7 resistors (220 Ω), potentiometer, diode rectifier, NPN transistor, breadboard, and many jumper wires. The ultrasonic sensor measures the distance of the object from the sensor and the LCD display, displays the value of distance. The LEDs and DC motor uses the information that the ultrasonic sensor receives and turns on or off depending on the distance of the object. The potentiometer is for adjusting the brightness of the LCD display. The diode rectifier, NPN transistor, and a resistor (220 Ω) are needed for the DC motor for it to function. This was one of the main hardware design tricks because without the 3 components the DC motor would not be able to be run and be controlled by the Arduino DUE. One hardware problem I had was sharing the 5V pin with the LCD display and DC motor because the display would glitched out and not display the values whenever I shared the 5V between them. My solution was using the main hardware design tricks by using a 3.3V pin for the DC motor and the 5V pin for the LCD display. Overall, the hardware part was fun to assemble.

III. Results

My process of fabrication started at the beginning of the winter term. During the beginning I would find inspirations and ideas from the internet and would try to find one that resonates with me. After finding what my project topic would be, I search on the internet to get an idea of what components are needed for this project, an idea on the code, and the difficulty of it. The most irritating and tedious part was trying to understand the components and how to wire it together, when that was finally completed, the software was sort of difficult and needed to learn the basic coding component for the ultrasonic sensor. After learning the basics, I had to find a way to calculate the distance of an object from the sensor using this formula distance = duration*0.034/2. Overall, the software component was not that complex and easy to understand. I was able to do what I intended using if and else statements, loops, and some functions. The functionality of my program, hardware and general function of the project is related to many applications of similarity sold motion light sensor light bulbs, sensors in smart/self- driving cars, home security systems, and robotics for obstacle detection system, although less sophisticated and less complex. Overall, the results were satisfying, and the project was a 100% successful. I feel very happy about this project because I knew that I did my very best to make it as exciting as possible.

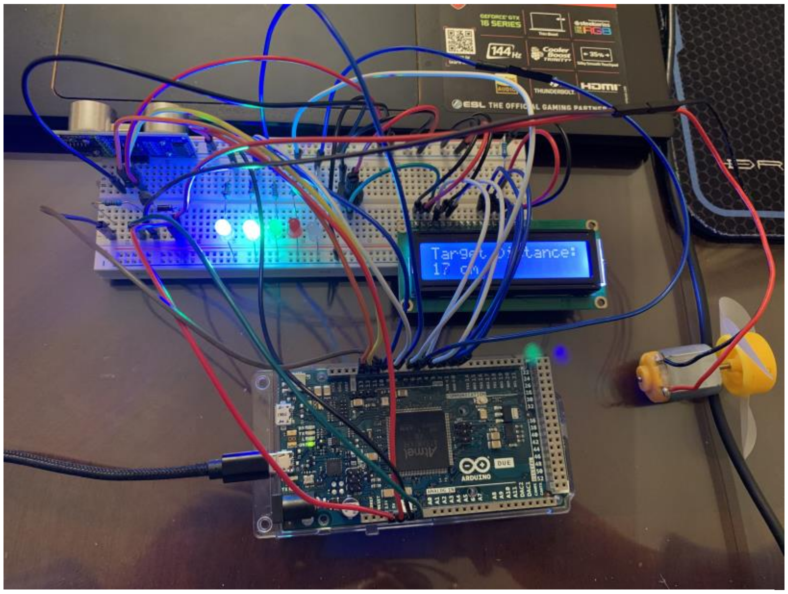


Fig. 9. The result of my Project.

IV. Discussion

For the hardware portion, the DC motor would not be able to be run or controlled by Arduino DUE. So, to troubleshoot and fix the problem I had to use a diode rectifier, NPN transistor, and a resistor (220 Ω) so that I can control and turn the DC motor on or off using the Arduino DUE. Another hardware technical issue I have was sharing the 5V pin with the LCD display and DC motor because the display would glitched out and not display the values whenever I shared the 5V between them. So, to troubleshoot and fix the problem I used a 3.3V pin for the DC motor and the 5V pin for the LCD display. For the software portion of the project, I did not know how to code the Ultrasonic sensor, so I went and got some online help to teach me the basics of the HC-SR04 sensor.

V. Conclusion

The main object for this project was to measure distance and turn on the fan using the HC-SR04 ultrasonic sensor. I achieved a successful outcome as my project was able to complete these tasks. Despite being stuck at home during the COVID-19 issue, I was able to manage my time and keep myself on track. Obtaining the components that I bought was tough because shipping took longer than expected. My secret to my success was mainly asking for online help when I need it, for example I did not know much about the ultrasonic sensor, so I went online and look for additional help to learn the basics of it. By getting peoples' opinion, tips, and online examples, I was able to get a general outlook on what my project would look like and how to build it. The success of my project helps pave a way for my future success as it helps improve my experience with Arduino, C/C++, and many other programing languages. It helps develop my skills in project and time management. This successful experience can help me developed a career in the future because many of the skills associated with the project such as brainstorming, rapid prototyping, design, and electronics were improved through the many smaller tasks I have completed. In conclusion, this project was a lot of fun to do and was probably one of the best part of the whole EECS 2032 course.

Acknowledgment

Thank you to the people online for helping me by giving their opinions and tips on some hardware, and ideas on my project. I would also like to say thank you to my father for buying and supplying me all the components necessary for this project. I would also like to say thank you to my TA who supported me and gave me some tips and ideas about my

project and how I could improve on the project.

References

Add references using the below example.

[1] Last Minute Engineers - How HC-SR04 Ultrasonic Sensor Works & Interface It with Arduino. Retrieved April 21, 2021, from https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/

[2] Motion Sensor Light Bulbs, Aukora 12W (100-Watt Equivalent) E26 Motion Activated Dusk to Dawn Security Light Bulb Outdoor/Indoor for Front Door Porch Garage Basement Hallway Closet (Cold White 2 Pack). Retrieved April 21, 2021, from https://www.amazon.com/Aukora-100-Watt-Equivalent-Activated-Security/dp/B07DXMF23S?th=1

[3] Simple math behind calculating distance using Ultrasonic sensor HC-SR04. Retrieved April 21, 2021, from https://medium.com/@adityavijaynarkar/simple-math-behind-calculating-distance-using-ultrasonic-sensor-hc-sr04-66ed5a6aa214

Kevin Nguyen (S'2021) is a second-year student studying in York University in the program Computer Engineering B.S degree. From 2019 to 2021, He studied in York's Lassonde School of Engineering.

VI. The Project Code

```
/* EECS 2032 Final Project
 * By Kevin Nguyen, 217228255
 * This code will measure distance and turns on the fan and LEDs depending on the distance of the object.
 */

// includes the LiquidCrystal Library
#include <LiquidCrystal.h>

// Creates an LCD object. Parameters: (rs, enable, d4, d5, d6, d7)
LiquidCrystal lcd(5, 4, 3, 2, 1, 0);

const int trigPin = 12;
const int echoPin = 13;
long duration;
int distance;
int motorPin = 11;

void setup()
{
  //LCD display
  lcd.begin(16,2); // Initializes the interface to the LCD screen

  //HC-SR04 Ultrasonic Sensor
  pinMode(trigPin, OUTPUT); // trigPin is OUTPUT
  pinMode(echoPin, INPUT);  // echoPin is INPUT

  //DC motor
  pinMode(motorPin, OUTPUT);// motorPin is OUTPUT

  //LEDs
  for (int i = 6; i <= 10; i++)
  {
    pinMode(i, OUTPUT);    // // LEDs pin is OUTPUT
  }
}

void loop()
{
  // Write a pulse to the HC-SR04 Trigger Pin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Measure the response from the HC-SR04 Echo Pin
  duration = pulseIn(echoPin, HIGH);

  // Determine distance from duration
  // Use 343 m/s as speed of sound
  distance= duration*0.034/2;

  // Prints "Target Distance: " on the first line of the LCD
  // Set cursor to first column of first row
  lcd.setCursor(0,0);
  lcd.print("Target Distance: ");
  //Print blanks on the second line of the LCD to clear the line
  // Set cursor to first column of second row
  lcd.setCursor(0,1);
  lcd.print("                "); // to clear the distance value
  // Prints "<value>" on the second line of the LCD
  // Set cursor to first column of second row
  lcd.setCursor(0,1);
  lcd.print(distance);
```

```
  lcd.print(" cm");
  delay(800);       // longer delay would cause less flickering

  //For DC motor
  if (distance <= 15){   // Turn DC motor on if distance is
15cm or less
    digitalWrite(motorPin, HIGH);
  }
  else{
    digitalWrite(motorPin, LOW);
  }

  // For LED
  if (distance <= 5) {   //Turn LED on if distance is 5cm or less
    digitalWrite(6, LOW);
  }
  else{
    digitalWrite(6, HIGH);
  }

  if (distance <= 10) {  //Turn LED on if distance is 10cm or
less
    digitalWrite(7, LOW);
  }
  else{
    digitalWrite(7, HIGH);
  }

  if (distance <= 15) {  //Turn LED on if distance is 15cm or
less
    digitalWrite(8, LOW);
  }
  else{
    digitalWrite(8, HIGH);
  }

  if (distance <= 20) {  //Turn LED on if distance is 20cm or
less
    digitalWrite(9, LOW);
  }
  else{
    digitalWrite(9, HIGH);
  }

  if (distance <= 25) {  //Turn LED on if distance is 25cm or
less
    digitalWrite(10, LOW);
  }
  else{
    digitalWrite(10, HIGH);
  }
}
```