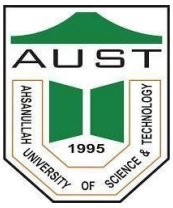


Ahsanullah University of Science & Technology
Department of Electrical & Electronic Engineering

Project: Detection System: RADAR

Course No : EEE 3210
Course Name : Microprocessor, Interfacing and System design Lab
Year : 3rd
Semester : 2nd
Section : C2
Group : 04

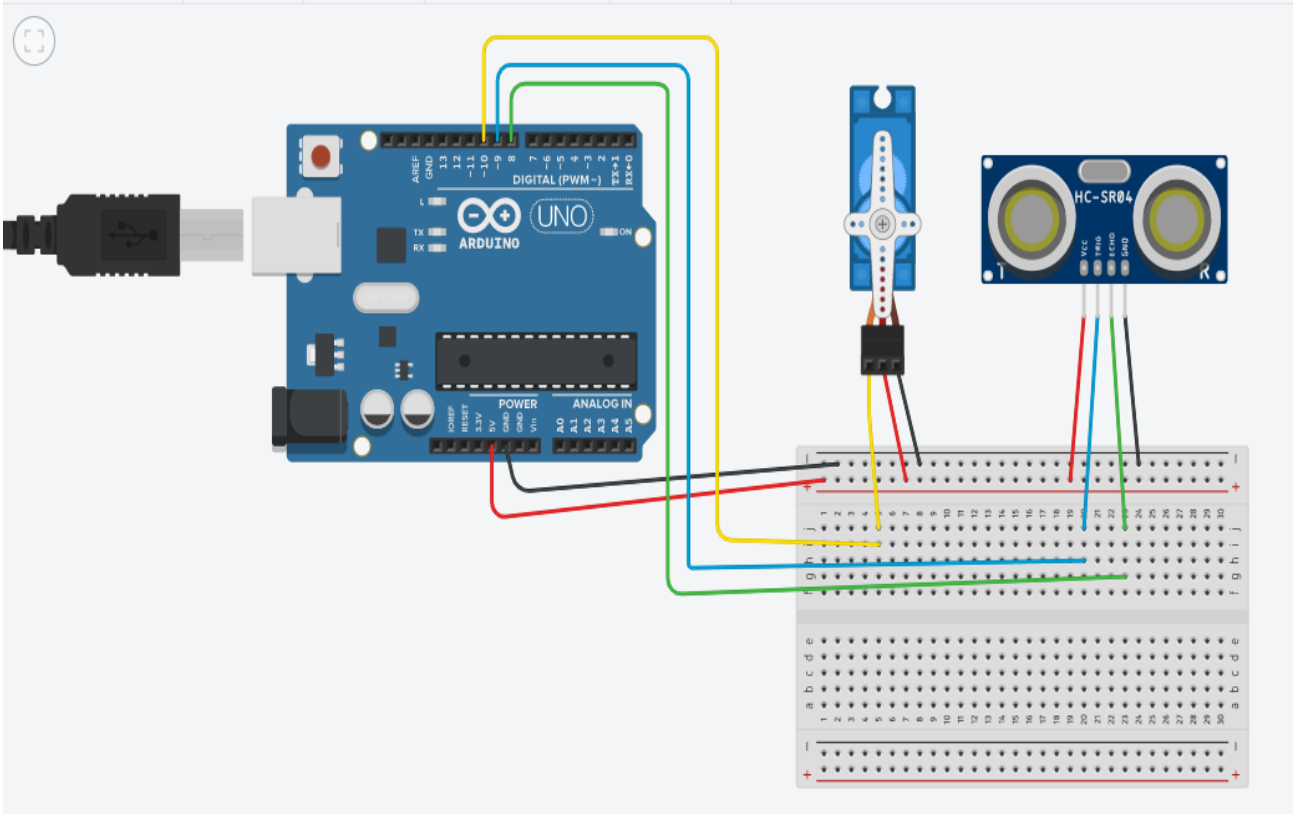
Submitted by:

ID	Name	Email
190205180	Sha Syed Taki	190205180@aust.edu
190205181	Ananta Barua	190205181@aust.edu
190205182	Rifat Bin Arif	190205182@aust.edu
190205183	Md. Ekramul Hasan Sharon	190205183@aust.edu
190205184	**Hashin Israq**	**190205184@aust.edu**

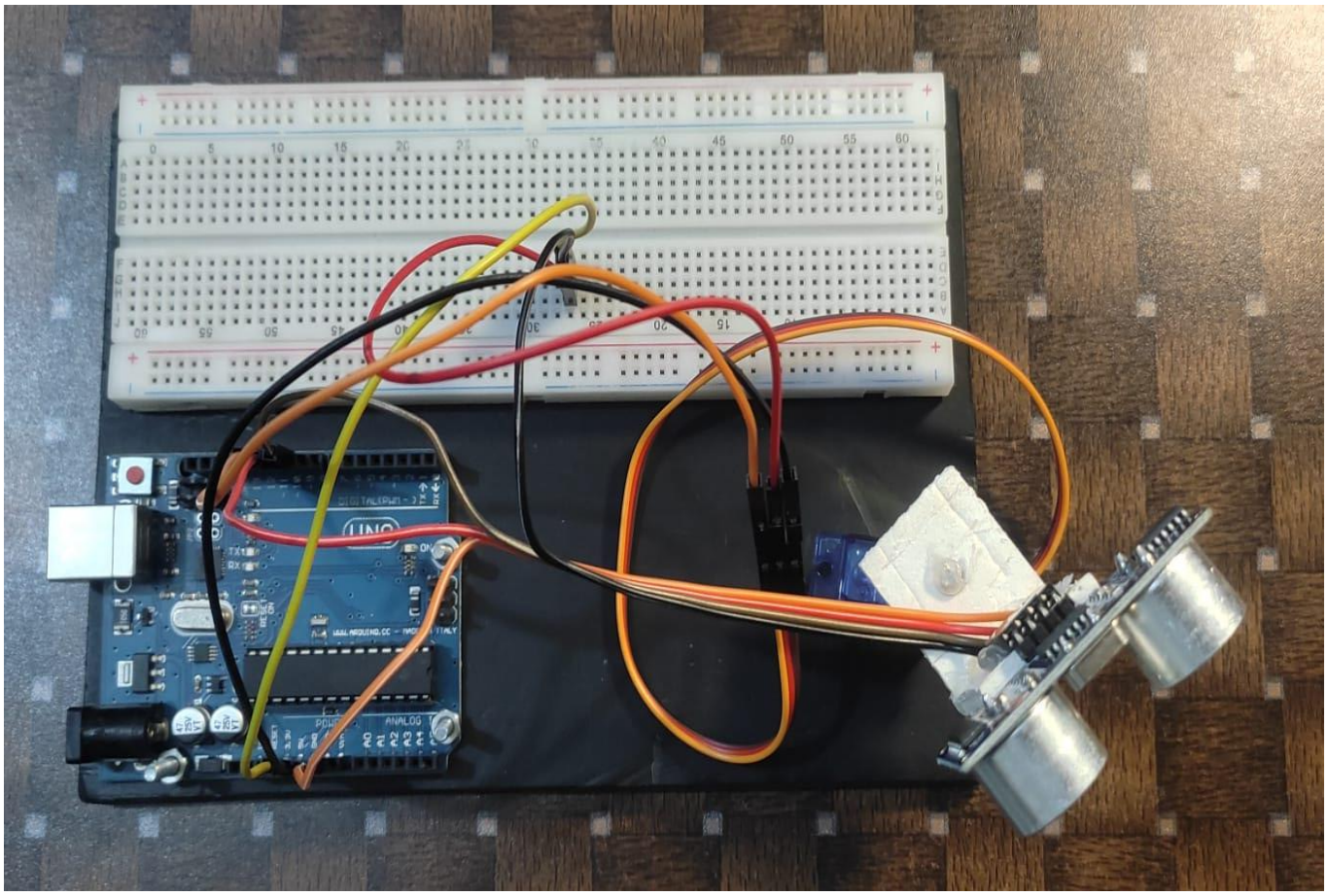
Overview:

Radar is a sensing technology used to detect and learn about objects at a wide variety of distances. In this project, three main components have been used which are Arduino Uno, servo-motor, and ultrasonic sensor. This project is working as a detection system that can identify an object in a limited range. If any object is placed in a limited range, we can visualize a red light in the processing application.

Circuit Diagram:



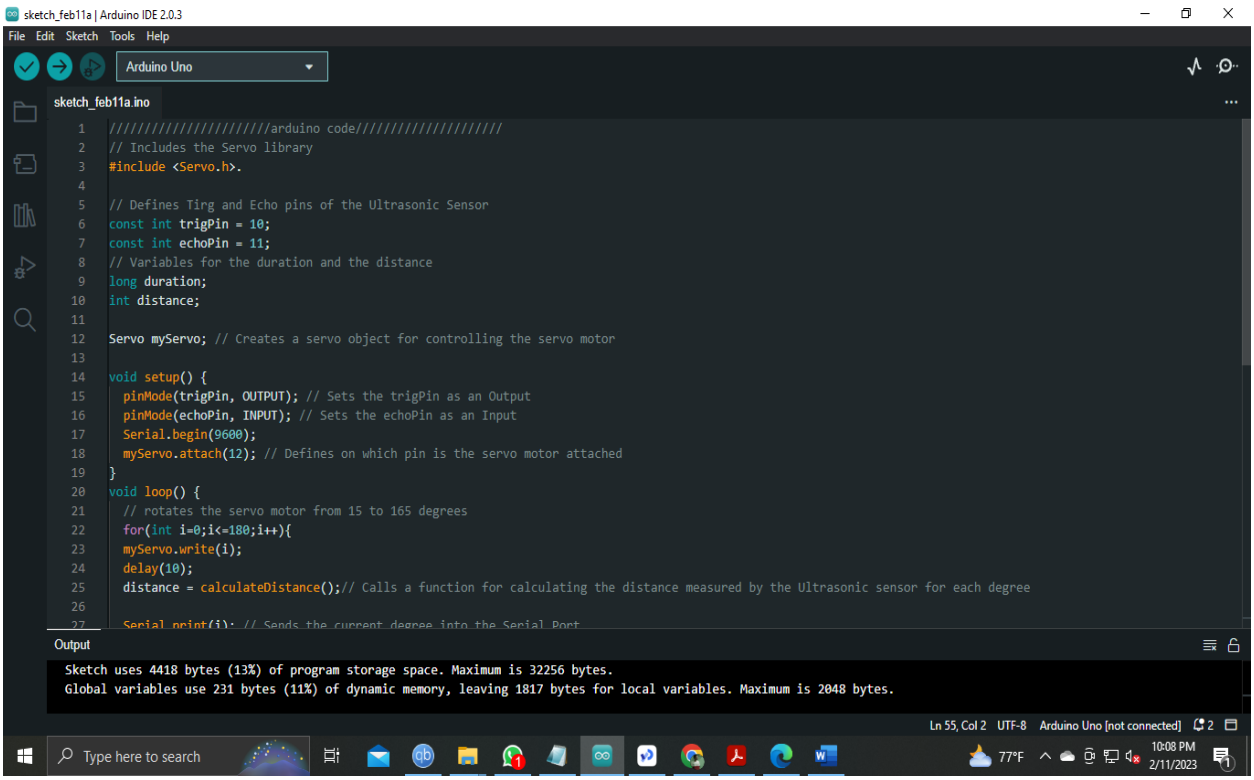
Hardware Setup:

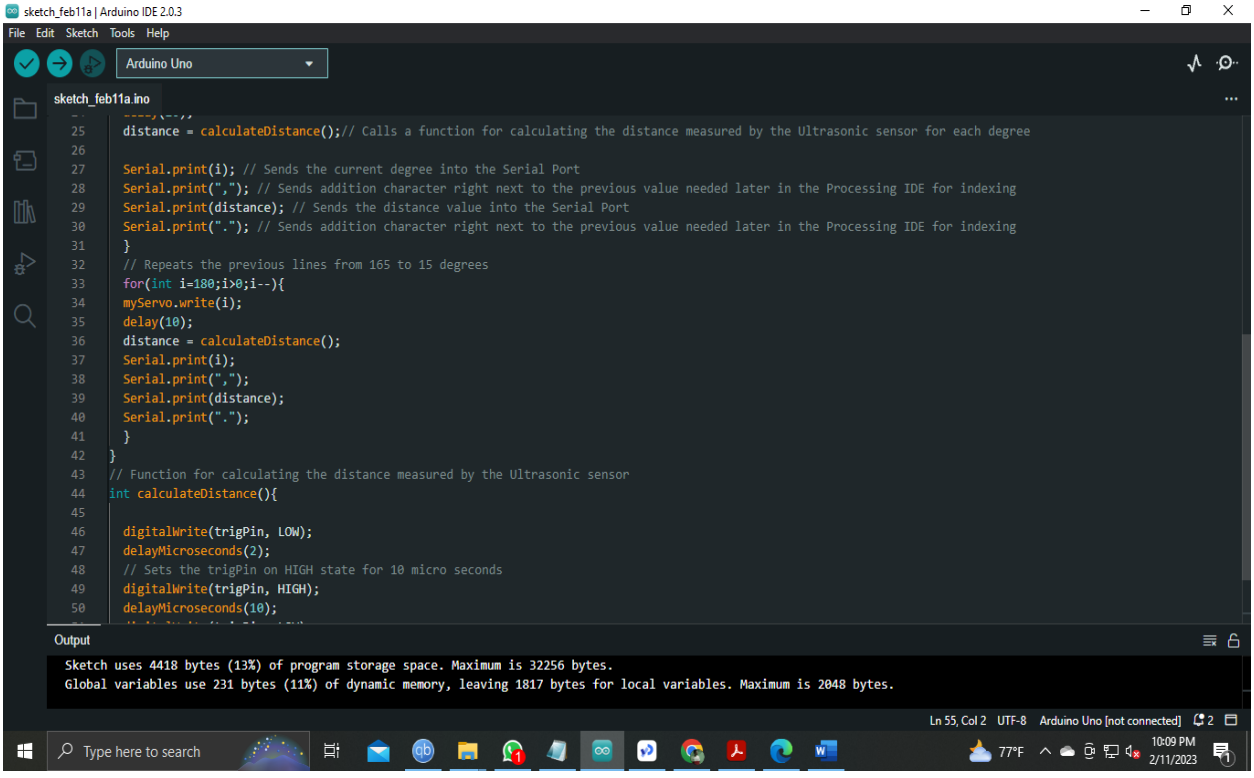


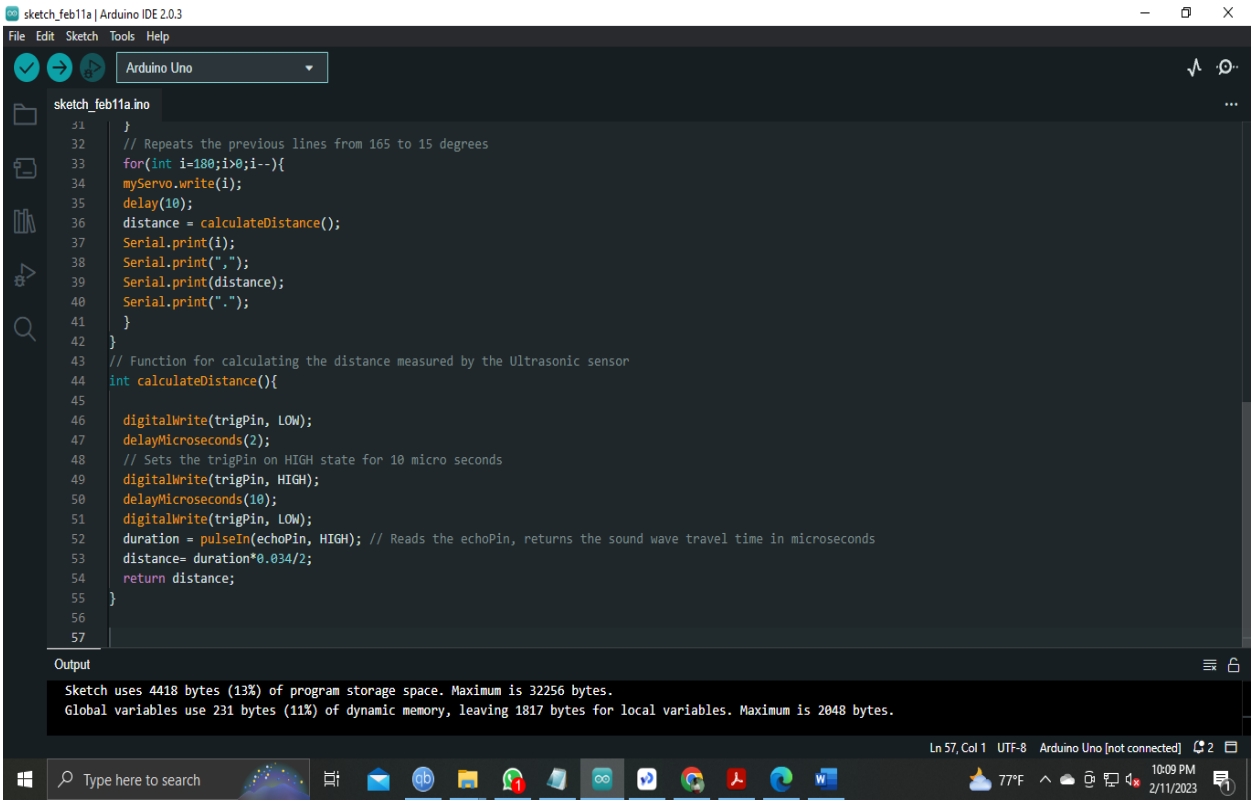
Working Principle:

The word RADAR means Radio Detection and Ranging. Radar is a technology that uses radio waves to detect, locate, and measure the speed of an object. It works by transmitting a radio signal in the direction of an object and then measuring the time it takes for the signal to be reflected a receiver. The time it takes to receive the signal is used to calculate the distance of the object, which can then be used to determine its location. The signal can also be used to measure the speed of the object, as the signal will be reflected at a different frequency depending on how fast the object is moving. The frequency of the signal can then be used to determine the speed of the object. The radar antenna transmits radio waves or microwaves that bounce off any object in its path. Due to this, we can easily determine the object in the radar range. The modern radar system is very advanced and used in highly diverse applications such as Air traffic control, Air defense system, radar Astronomy, Anti-missile system, Outer space Surveillance system, and many more.

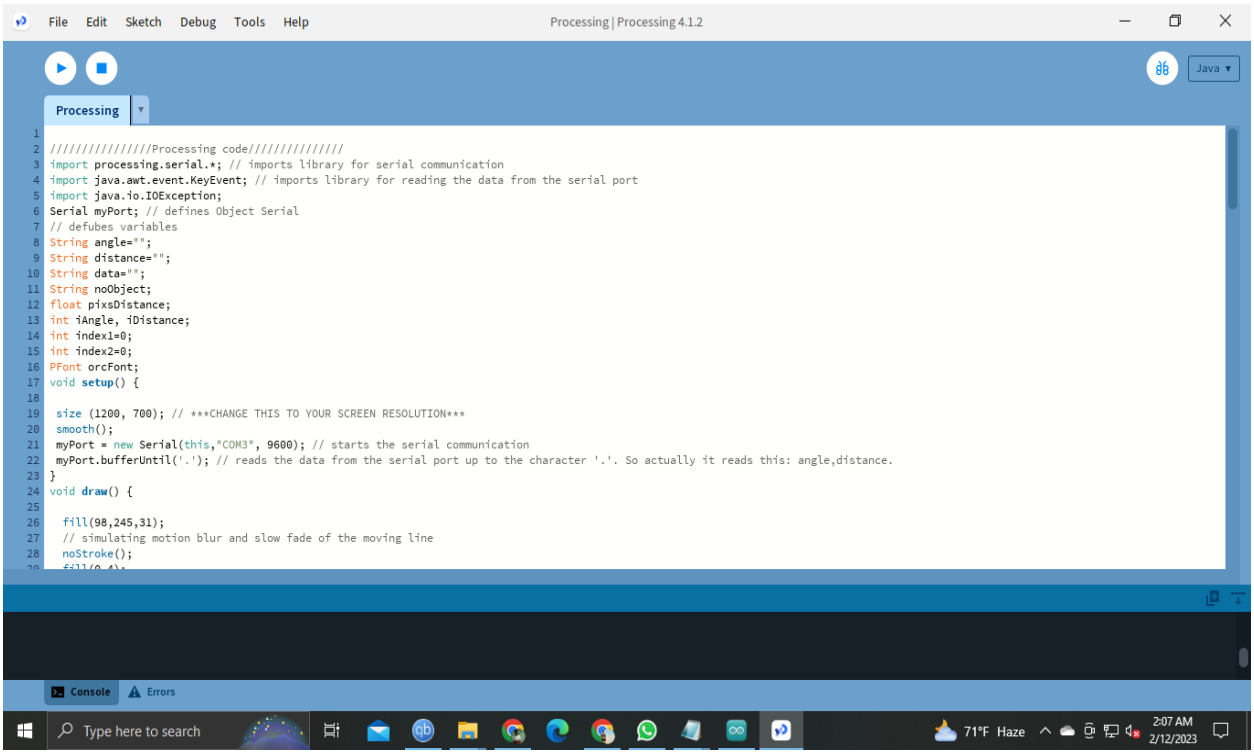
Arduino Codes:

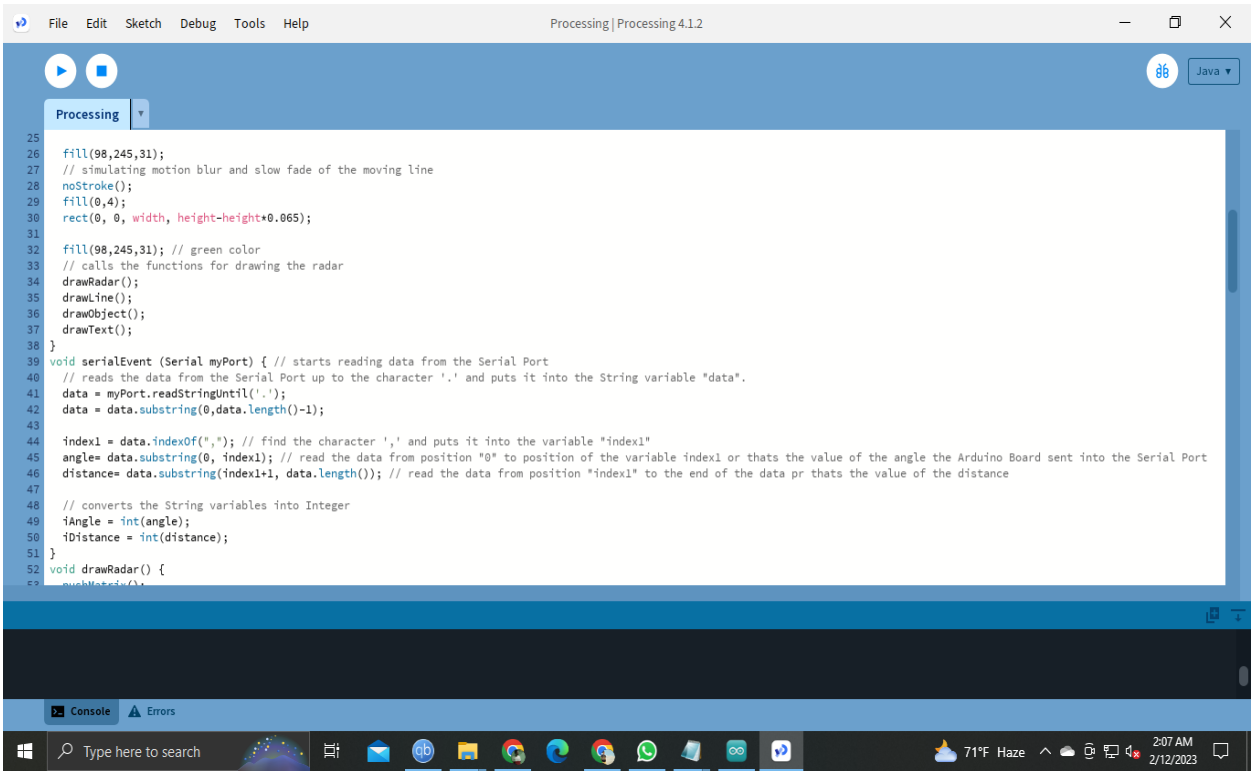




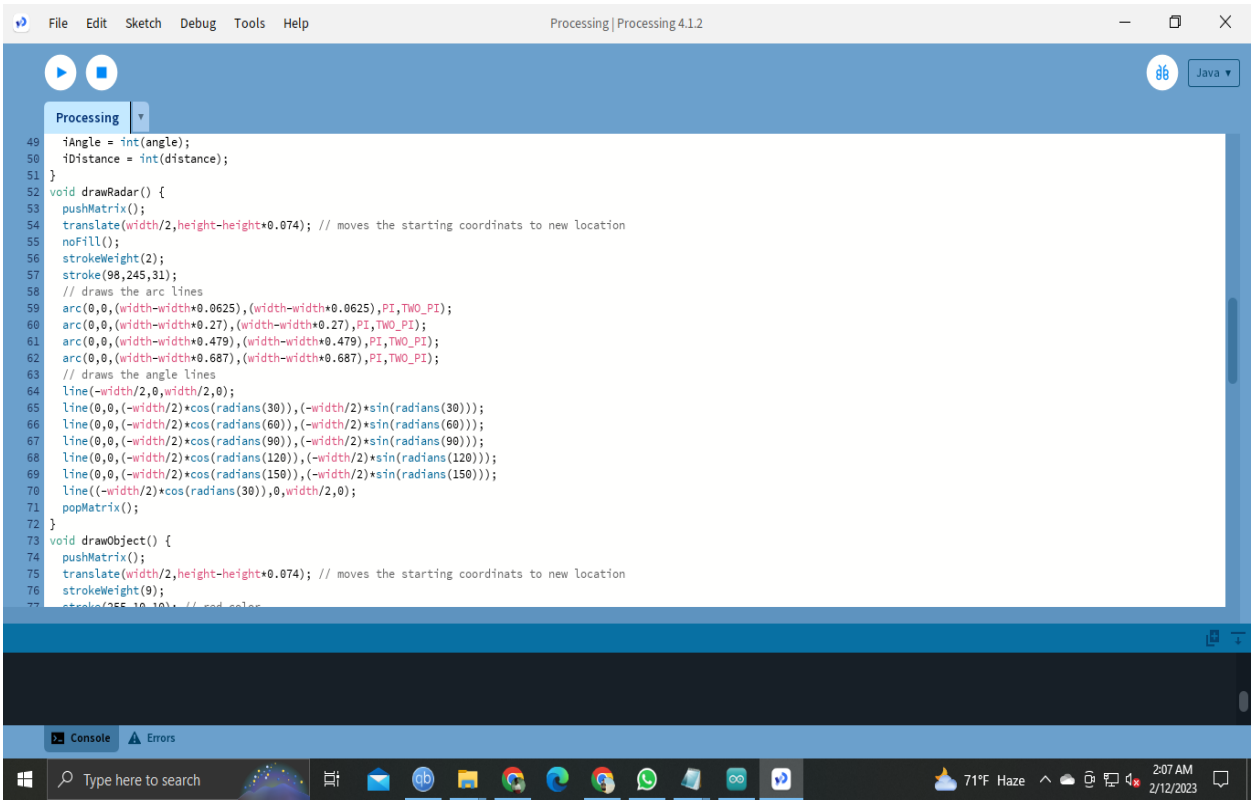


Processing Codes:

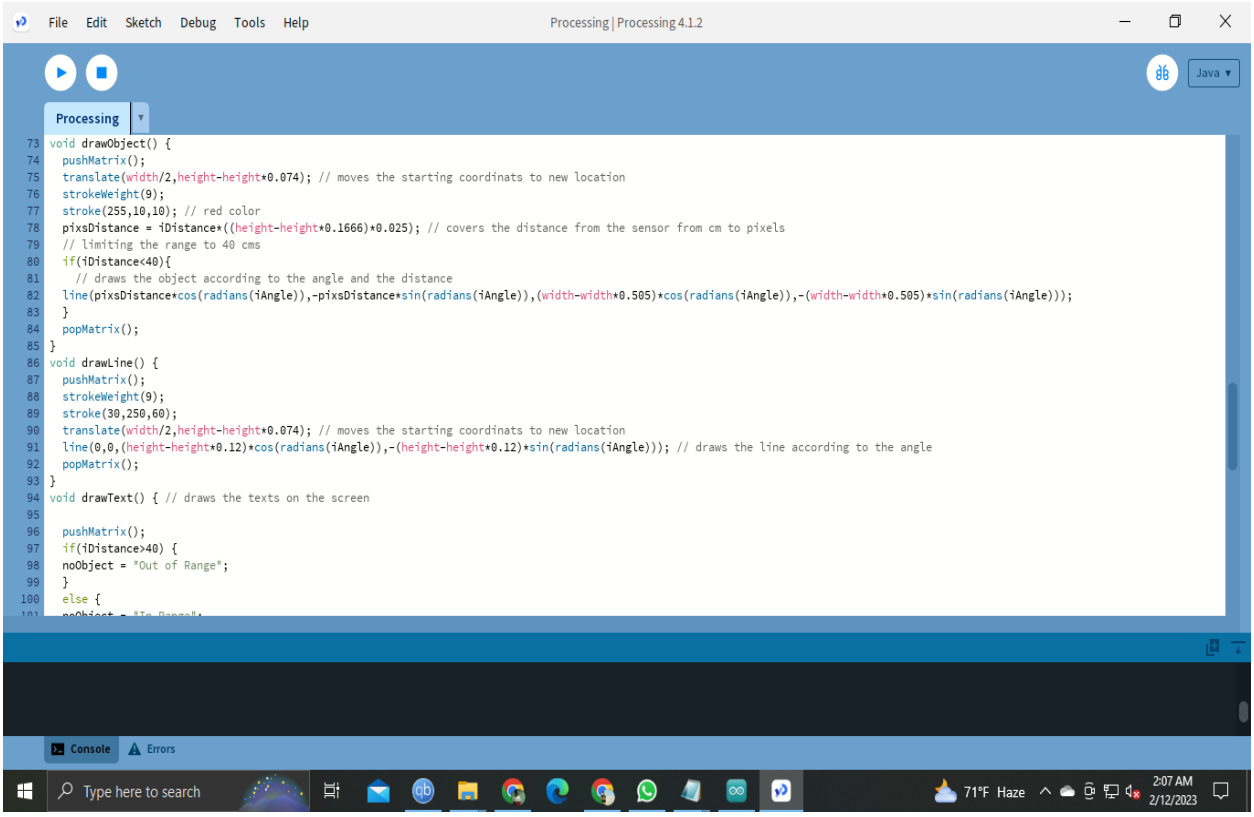




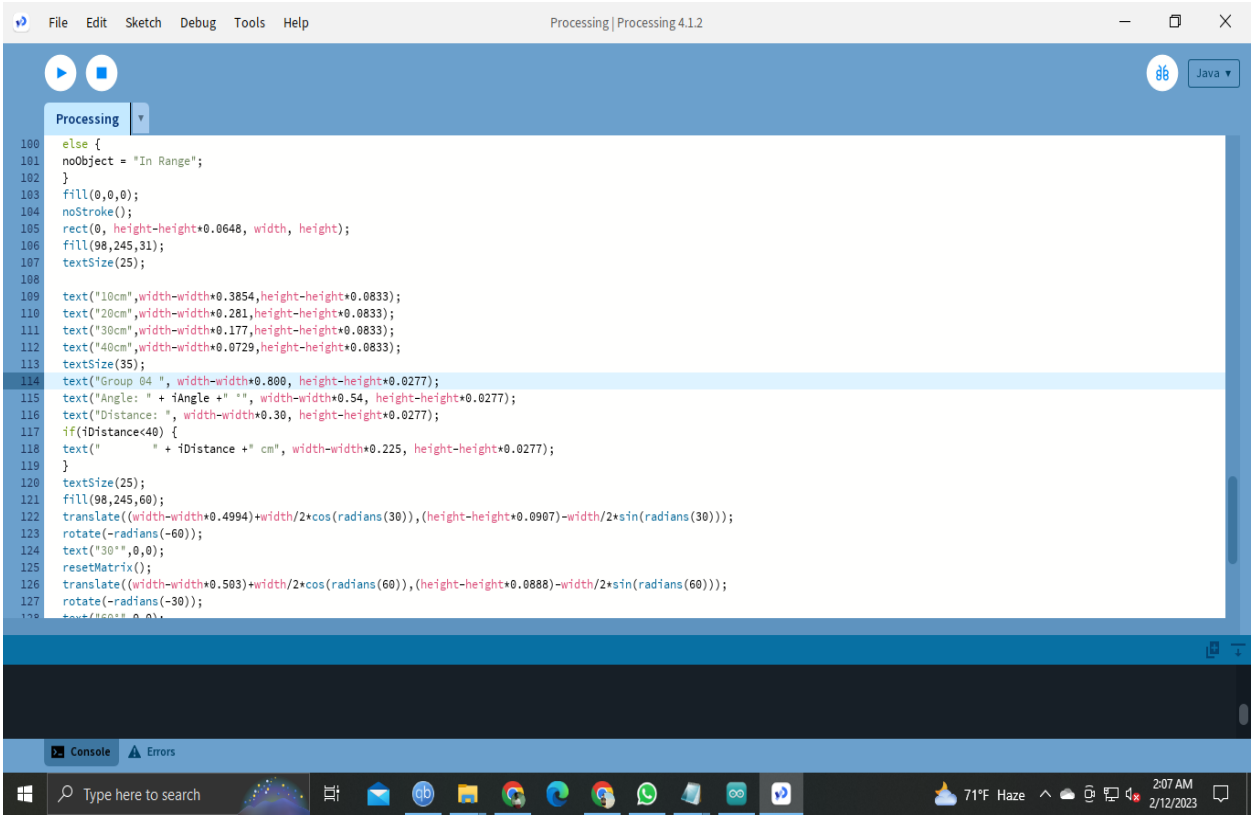
```
  fill(98,245,31);
  // simulating motion blur and slow fade of the moving line
  noStroke();
  fill(0,4);
  rect(0, 0, width, height-height*0.065);

  fill(98,245,31); // green color
  // calls the functions for drawing the radar
  drawRadar();
  drawLine();
  drawObject();
  drawText();
}
void serialEvent (Serial myPort) { // starts reading data from the Serial Port
  // reads the data from the Serial Port up to the character '.' and puts it into the String variable "data".
  data = myPort.readStringUntil('.');
  data = data.substring(0,data.length()-1);

  index1 = data.indexOf(","); // find the character ',' and puts it into the variable "index1"
  angle= data.substring(0, index1); // read the data from position "0" to position of the variable index1 or thats the value of the angle the Arduino Board sent into the Serial Port
  distance= data.substring(index1+1, data.length()); // read the data from position "index1" to the end of the data pr thats the value of the distance

  // converts the String variables into Integer
  iAngle = int(angle);
  iDistance = int(distance);
}
void drawRadar() {
```



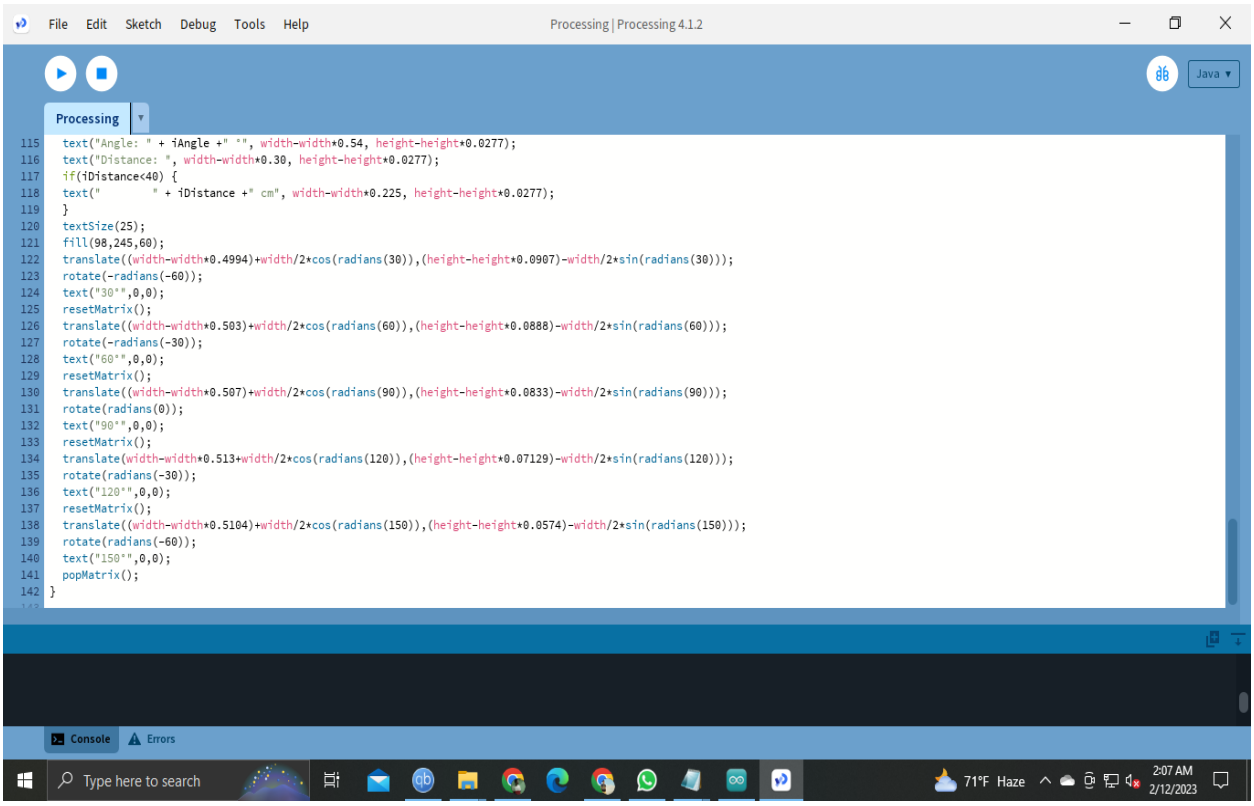
```
  iAngle = int(angle);
  iDistance = int(distance);
}
void drawRadar() {
  pushMatrix();
  translate(width/2,height-height*0.074); // moves the starting coordinats to new location
  noFill();
  strokeWeight(2);
  stroke(98,245,31);
  // draws the arc lines
  arc(0,0,(width-width*0.0625),(width-width*0.0625),PI,TWO_PI);
  arc(0,0,(width-width*0.27),(width-width*0.27),PI,TWO_PI);
  arc(0,0,(width-width*0.479),(width-width*0.479),PI,TWO_PI);
  arc(0,0,(width-width*0.687),(width-width*0.687),PI,TWO_PI);
  // draws the angle lines
  line(-width/2,0,width/2,0);
  line(0,0,(-width/2)*cos(radians(30)),(-width/2)*sin(radians(30)));
  line(0,0,(-width/2)*cos(radians(60)),(-width/2)*sin(radians(60)));
  line(0,0,(-width/2)*cos(radians(90)),(-width/2)*sin(radians(90)));
  line(0,0,(-width/2)*cos(radians(120)),(-width/2)*sin(radians(120)));
  line(0,0,(-width/2)*cos(radians(150)),(-width/2)*sin(radians(150)));
  line((-width/2)*cos(radians(30)),0,width/2,0);
  popMatrix();
}
void drawObject() {
  pushMatrix();
  translate(width/2,height-height*0.074); // moves the starting coordinats to new location
  strokeWeight(9);
```



```
void drawObject() {
  pushMatrix();
  translate(width/2,height-height*0.074); // moves the starting coordinats to new location
  strokeWeight(9);
  stroke(255,10,10); // red color
  pixsDistance = iDistance*((height-height*0.1666)*0.025); // covers the distance from the sensor from cm to pixels
  // limiting the range to 40 cms
  if(iDistance<40){
    // draws the object according to the angle and the distance
  line(pixsDistance*cos(radians(iAngle)),-pixsDistance*sin(radians(iAngle)),(width-width*0.505)*cos(radians(iAngle)),-(width-width*0.505)*sin(radians(iAngle)));
  }
  popMatrix();
}
void drawLine() {
  pushMatrix();
  strokeWeight(9);
  stroke(30,250,60);
  translate(width/2,height-height*0.074); // moves the starting coordinats to new location
  line(0,0,(height-height*0.12)*cos(radians(iAngle)),-(height-height*0.12)*sin(radians(iAngle))); // draws the line according to the angle
  popMatrix();
}
void drawText() { // draws the texts on the screen

  pushMatrix();
  if(iDistance>40) {
  noObject = "Out of Range";
  }
  else {
```



```
  else {
  noObject = "In Range";
  }
  fill(0,0,0);
  noStroke();
  rect(0, height-height*0.0648, width, height);
  fill(98,245,31);
  textSize(25);

  text("10cm",width-width*0.3854,height-height*0.0833);
  text("20cm",width-width*0.281,height-height*0.0833);
  text("30cm",width-width*0.177,height-height*0.0833);
  text("40cm",width-width*0.0729,height-height*0.0833);
  textSize(35);
  text("Group 04 ", width-width*0.800, height-height*0.0277);
  text("Angle: " + iAngle +" °", width-width*0.54, height-height*0.0277);
  text("Distance: ", width-width*0.30, height-height*0.0277);
  if(iDistance<40) {
  text("        " + iDistance +" cm", width-width*0.225, height-height*0.0277);
  }
  textSize(25);
  fill(98,245,60);
  translate((width-width*0.4994)+width/2*cos(radians(30)),(height-height*0.0907)-width/2*sin(radians(30)));
  rotate(-radians(-60));
  text("30°",0,0);
  resetMatrix();
  translate((width-width*0.503)+width/2*cos(radians(60)),(height-height*0.0888)-width/2*sin(radians(60)));
  rotate(-radians(-30));
```



Component problems:

HC-SR04 is a commonly used ultrasonic sensor that measures the distance between the sensor and an object by using ultrasonic waves. The HC-SR04 sensor has a limited range, usually between 2cm and 40cm. If the sensor is being used beyond this range, it might give incorrect readings or no reading at all. Sometimes the sensor can give false readings due to reflections of surrounding objects which is not in the range of the ultrasonic sensor. If the sensor readings are inconsistent, it could be due to incorrect wiring or a faulty sensor. The sensor readings can be affected by electrical noise in the surrounding environment.

Servo motors require a stable and sufficient power supply to operate properly. If the power supply voltage is too low, the servo motor might not have enough power to rotate through the full 180 degrees. Servo motors can also fail to rotate

through the full 180 degrees due to mechanical issues such as a damaged gear train or a binding potentiometer.

Wires can come loose from the breadboard if they are not inserted properly. Poor wiring can cause problems such as intermittent connections, short circuits, and, incorrect readings. Breadboard connections can corrode over time, especially if the breadboard is exposed to moisture or other corrosive elements. Power supply problems can also cause issues with the connections on a breadboard. It can give inconsistent results if the power supply is not providing a stable voltage.

Discussion:

In this project, a lab-scaled radar system was designed and implemented using an Arduino, a servomotor, and an ultrasonic sensor. The developed system can read the distance of obstacles and the angle of incident and convert this data into visually represented information. The system performance measures up with other systems at its level as it adequately reports any obstacle it finds in its path and provides an estimated range of the object. A very handy application for this system would be in the area of object detection and avoidance systems for robotics or maybe in intrusion detection systems for location sizes where it may not be economical to use multiple units to provide adequate coverage. The system's range is dependent on the range of the ultrasonic sensor that is used. In this system, the HC-SR04 ultrasonic sensor was used which has a range between 2cm to 40cm.

Group Contribution:

190205180	**-**
190205181,190205182	**Overview, Working Principle, Components Problem, Discussion**
190205183	**Hardware Setup**
190205184	**Codes**