ECE DEPARTMENT

Arduino Radar Project – Ultrasonic Based Radar:

Project Done by: ECE A SECTION(B.TECH 2ND SEM)
21ECB0A45* Potnuri Sri Anjali Pravallika
21ECB0A34* Manepalli Alekhya
21ECB0A09* Geethika

Submission To **Dr. J. Ravi Kumar**

ABSTRACT:

The aim of this Arduino Radar project is to achieve a radar system prototype based on an Arduino board that detects stationary and moving objects.

Do you know? The first time, Radar was developed as a method of detecting enemy aircraft in World War 2. Along with

the developments in technology, these days it is used in a wide range of sectors. Over the last few decades, there have been significant advancements in the use of radar technology.

In this project, we are going to design an Arduino radar project using Ultrasonic Sensor for detection. An ***Arduino microcontroller*** *makes electronics more disciplined.*

This Arduino radar project aims to achieve a radar system prototype based on an Arduino board that detects stationary and moving objects. The radar system has different performance specifications, and it is also available in a variety of sizes.

COMPONENTS:

1. Arduino UNO- costs: Rs.850
2. Ultrasonic sensor-costs: RS.75
3. Servo Motor-costs: Rs.95
4. Jumper Pins-costs: Rs.50

Required tools:

- Cardboard-can find anywhere

ABOUT THE COMPONENTS:

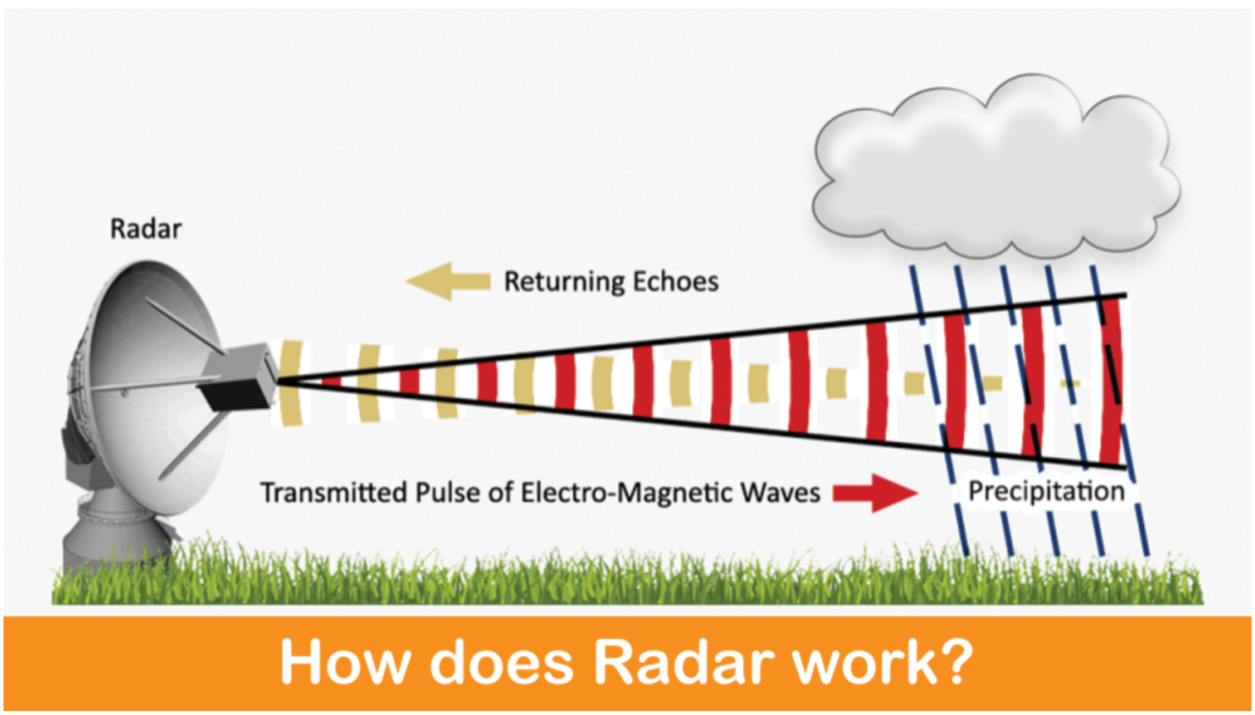
How does Radar Work?

The word RADAR means Radio Detection And Ranging. Radar is an object detection system that uses microwaves to determine the *range, altitude, direction, and speed of objects within about a 100-mile radius of their location.*

The radar antenna transmits radio waves or microwaves that bounce off any object in their path. Due to this, we can easily determine the object in the radar range.

The basic principle of operation:

A radar is an electromagnetic sensor that is used to detect and locate an object.



Radio waves or microwaves are radiated out from the radar into free space. Some of these waves are intercepted by reflecting objects.

These intercepted radio waves hit the target and are reflected in many different directions. Some of these waves *can be directed back toward the radar, where they are received and amplified.*

If these waves are received again at their origin, then it means an object is in the propagation direction.

The modern radar system is very advanced and used in highly diverse applications such as Air traffic control, Air-defense system, radar Astronomy, Antimissile system, Outer space Surveillance system, and many more.

What is an Ultrasonic Sensor?

*An **ultrasonic sensor** is a proximity sensor that is used to measure the distance of a target or object. It detects the object by transmitting ultrasonic waves and converts the reflected waves into an electrical signal. These sound waves travel faster than the speed of the sound that humans can hear.*

It has two main components: the transmitter & receiver. The transmitter emits the sound using a piezoelectric crystal, and the receiver encounters the sound after it has traveled to and from the target.

For the calculation of the object distance, the sensor measures the time taken by the signal to travel between the transmission of the sound by the transmitter to the reflecting back towards the receiver.

The formula for this calculation is,

$D = \frac{1}{2} T \times C$

Where,

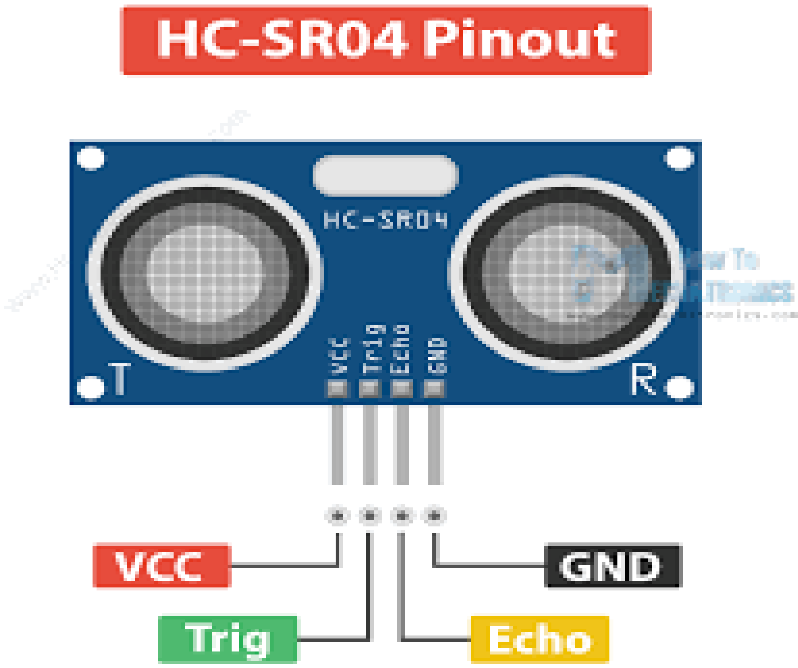
- *D = distance,*
- *T = time*

- *C = speed of sound which is 343 meters/second.*

These sensors are mostly found in automobile self-parking technology and anti-collision safety systems. Also, used in robotic obstacle detection systems, manufacturing technology, and many more. To know more about the ultrasonic sensor, refer to the ultrasonic sensor working principle.

Ultrasonic sensors are used primarily as proximity sensors. They can be found in automobile self-parking technology and anti-collision safety systems. Ultrasonic sensors are also used in robotic obstacle detection systems, as well as manufacturing technology. In comparison to infrared (IR) sensors in proximity sensing applications, ultrasonic sensors are not as susceptible to interference of smoke, gas, and other airborne particles (though the physical components are still affected by variables such as heat).

Ultrasonic sensors are also used as level sensors to detect, monitor, and regulate liquid levels in closed containers (such as vats in chemical factories). Most notably, ultrasonic technology has enabled the medical industry to produce images of internal organs, identify tumors, and ensure the health of babies in the womb.



What is a Servo Motor?

*The **servo motor** is a simple DC motor that can be controlled for specific angular rotation with the help of additional servomechanism. This motor will only rotate as much as we want and then stop. The servo motor is a closed-loop mechanism that uses positional feedback to control the speed and position.*

This closed-loop system includes a control circuit, servo motor, shaft, potentiometer, drive gears, amplifier, and either an encoder or resolver.

A servo motor (or servo motor) is ***a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity, and acceleration****. It consists of a suitable motor coupled to a sensor for position feedback. Servo Motors are used in applications such as robotics, CNC machinery, or automated manufacturing.*

The servo motor is unlike a standard electric motor which starts and stops according to the power input. According to the signal, the servo motor will work.

Nowadays, servo motors are widely used in industrial and robotics applications. They are also commonly seen in remote-controlled toy cars, RC planes, and in the CD or DVD player. Besides these, we see hundreds of applications in our daily life that use a *servo motor. To know more about the servo motor, refer to the servo motor working principle.*

Mechanism Of Servo Motor

A servomotor is a closed-loop servomechanism that uses position feedback to control its motion and final position. The input to its control is a signal (either analog or digital) representing the position commanded for the output shaft.

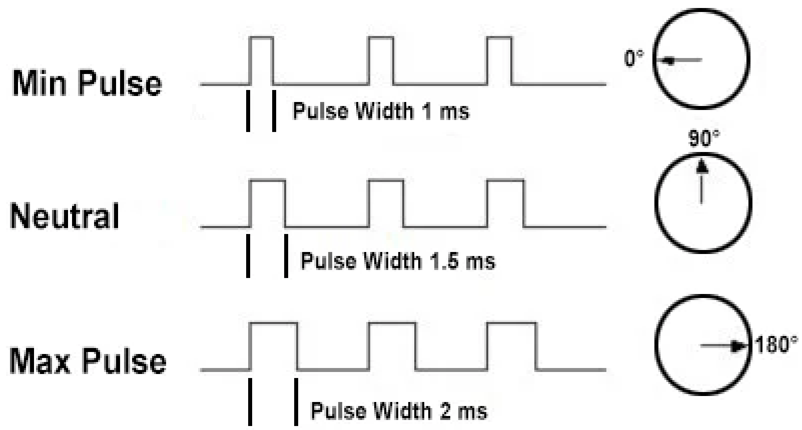
The motor is paired with some type of position encoder to provide position and speed feedback. In the simplest case, only the position is measured. The measured position of the output is compared to the command position, the external input to the controller.

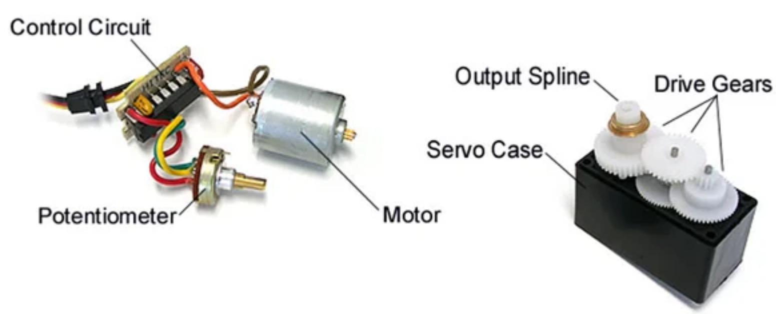
If the output position differs from that required, an error signal is generated which then causes the motor to rotate in either direction, as needed to bring the output shaft to the appropriate position. As the positions approach, the error signal reduces to zero, and the motor stops.

The very simplest servo motors use position-only sensing via a potentiometer and bang-bang control of *their motor; the motor always rotates at full speed (or is stopped). This type of servomotor is not widely used in industrial motion control, but it forms the basis of the simple and cheap servos used for radio-controlled models.*

More sophisticated servo motors use optical rotary encoders to measure the speed of the output shaft and a variable-speed drive to control the motor speed. Both of these enhancements, usually in combination with a PID control algorithm, allow the servo motor to be brought to its commanded position more quickly and more precisely, with less overshooting.

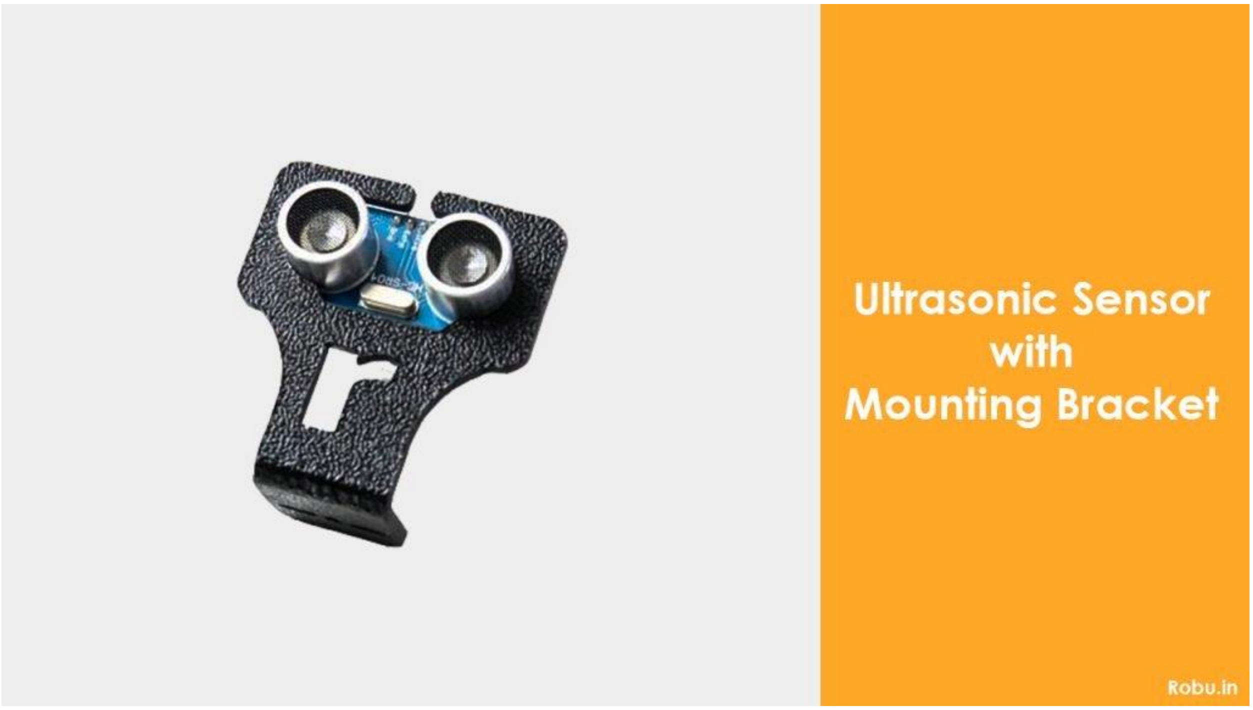
A servo motor is an electromechanical device that generates torque and velocity based on the supplied current and voltage. A servo motor operates as part of closed-loop control, providing torque and velocity as commanded by a servo controller which uses a feedback device to close the loop.





Preparing the Fixture

*After making the connections, to build the fixture of this construction, I have used a **mounting bracket** as shown in the below image to fix the ultrasonic sensor.*



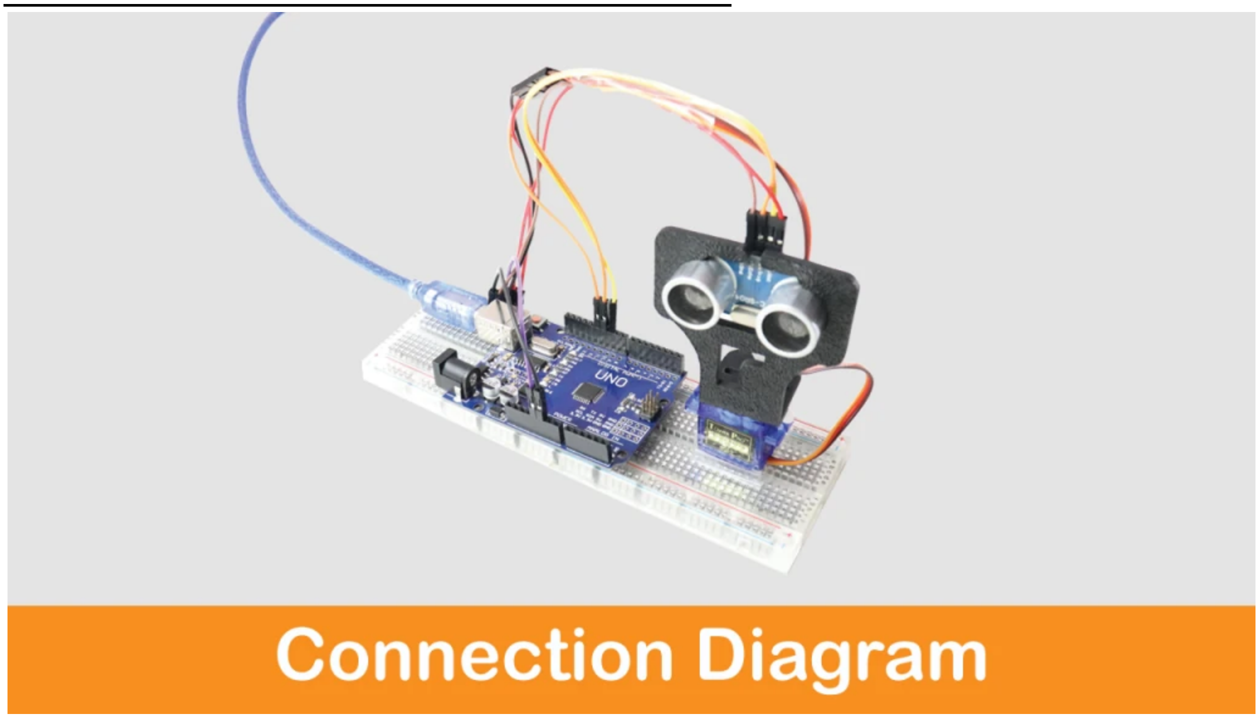
After fixing it, the mounting bracket is screwed to the servo motor. Kindly use the DST (double-sided) tape to fix the servo motor firmly with the surface so that it can easily handle the weight of the bracket and the ultrasonic Sensor.

This step is optional. You can also make a simple structure with cardboard to mount the ultrasonic sensor with a Servo motor.

Hardware Connection

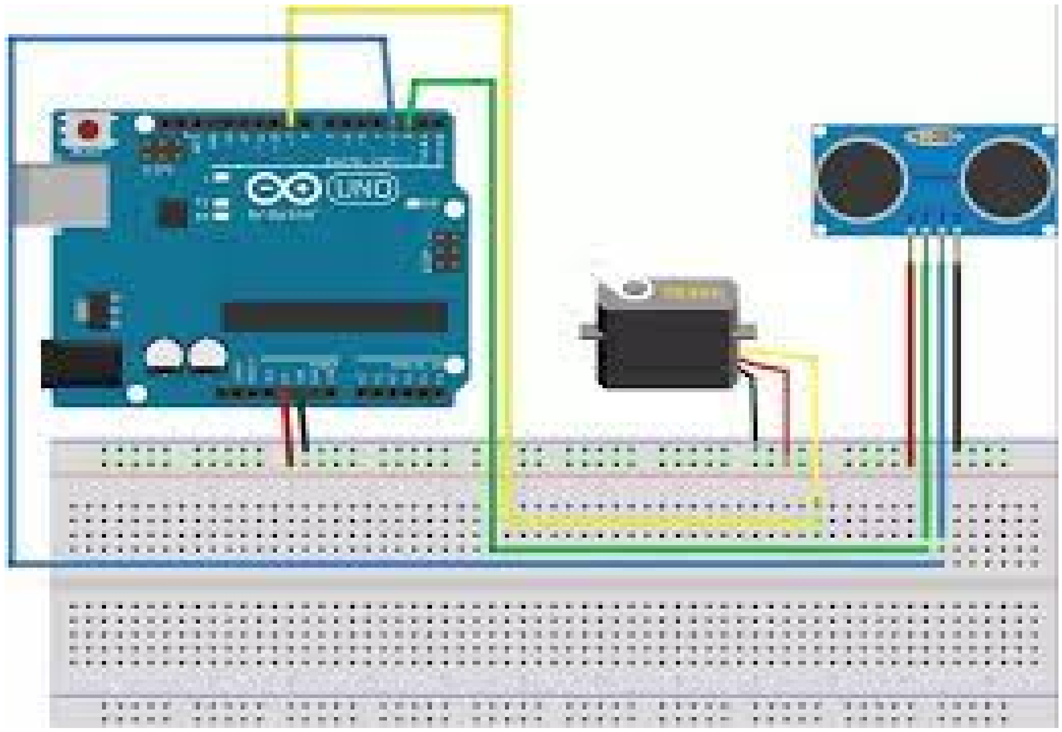
The below image shows the connection of the Arduino radar project. The connections are very simple. Here, we have interfaced the ultrasonic sensor and servo motor with an Arduino Uno. The connection details are,

Arduino Uno	*Ultrasonic Sensor*	*Servo Motor*
Vcc	*Vcc*	*Vcc*
Gnd	*Gnd*	*Gnd*
D10	*Trig*	-
D11	*Echo*	-
D12	-	*Signal*



Circuit Diagram:

(using arduino)



Software Installation and Code

We need two pieces of software to complete this Arduino radar project. One is an Arduino IDE, and the other is a Processing IDE. Download both Software from the below links,

- ***Arduino IDE 1.8.13***
- ***Processing IDE***

Processing application is visual arts-based software for learning to code. After downloading, extract the Zip file, and you will get the processing application (.exe file).

In this project, we are using two codes: Arduino UNO and the other for Processing.

- ***Here is the Arduino source code:***

```
// Includes the Servo library
#include <Servo.h>.
// Defines Trig and Echo pins of the Ultrasonic Sensor
const int trigPin = 10;
const int echoPin = 11;
// Variables for the duration and the distance
long duration;
int distance;
Servo myServo; // Creates a servo object for controlling the servo motor
void setup() {
 pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
 pinMode(echoPin, INPUT); // Sets the echoPin as an Input
 Serial.begin(9600);
 myServo.attach(12); // Defines on which pin is the servo motor attached
}
void loop() {
 // rotates the servo motor from 15 to 165 degrees
 for(int i=15;i<=165;i++){
 myServo.write(i);
 delay(30);
 distance = calculateDistance();// Calls a function for calculating the distance measured by the Ultrasonic sensor for each degree

 Serial.print(i); // Sends the current degree into the Serial Port
 Serial.print(","); // Sends addition character right next to the previous value needed later in the Processing IDE for indexing
 Serial.print(distance); // Sends the distance value into the Serial Port
 Serial.print("."); // Sends addition character right next to the previous value needed later in the Processing IDE for indexing
 }
```

```
 // Repeats the previous lines from 165 to 15 degrees
 for(int i=165;i>15;i--){
 myServo.write(i);
 delay(30);
 distance = calculateDistance();
 Serial.print(i);
 Serial.print(",");
 Serial.print(distance);
 Serial.print(".");
 }
}

int calculateDistance(){

 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 // Sets the trigPin on HIGH state for 10 micro seconds
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH); // Reads the echoPin, returns the
sound wave travel time in microseconds
 distance= duration*0.034/2;
 return distance;
}
```

- <u>*Here is the processing code for the Arduino radar project:*</u>

```
import processing.serial.*; // imports library for serial communication
import java.awt.event.KeyEvent; // imports library for reading the data from
the serial port
import java.io.IOException;
Serial myPort; // defines Object Serial
```

```
// defines variables
String angle="";
String distance="";
String data="";
String noObject;
float pix Distance;
int iAngle, iDistance;
int index1=0;
int index2=0;
PFont orcFont;
void setup() {

 size (1200, 700); // ***CHANGE THIS TO YOUR SCREEN RESOLUTION***
 smooth();
 myPort = new Serial(this,"COM5", 9600); // starts the serial communication
 myPort.bufferUntil('.'); // reads the data from the serial port up to the
character '.'. So actually it reads this: angle,distance.
}
void draw() {

  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 that's the value of the distance

  // converts the String variables into Integer
  iAngle = int(angle);
  iDistance = int(distance);
}
void drawRadar() {
  pushMatrix();
  translate(width/2,height-height*0.074); // moves the starting coordinates 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 coordinates to
new location
  strokeWeight(9);
  stroke(255,10,10); // red color
  pix Distance = Distance*((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)),(wid
th-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 coordinates to
new location

line(0,0,(height-height*0.12)*cos(radians(iAngle)),-(height-height*0.12)*sin(ra
dians(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 {
  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(40);
 text("Robu.in", width-width*0.875, height-height*0.0277);
 text("Angle: " + iAngle +" °", width-width*0.48, height-height*0.0277);
 text("Distance: ", width-width*0.26, 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*0.09
07)-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*0.088
8)-width/2*sin(radians(60)));
 rotate(-radians(-30));
 text("60°",0,0);
 resetMatrix();

translate((width-width*0.507)+width/2*cos(radians(90)),(height-height*0.083
3)-width/2*sin(radians(90)));
 rotate(radians(0));
 text("90°",0,0);
 resetMatrix();

translate(width-width*0.513+width/2*cos(radians(120)),(height-height*0.0712
9)-width/2*sin(radians(120)));
```

```
 rotate(radians(-30));
 text("120°",0,0);
 resetMatrix();

translate((width-width*0.5104)+width/2*cos(radians(150)),(height-height*0.0
574)-width/2*sin(radians(150)));
 rotate(radians(-60));
 text("150°",0,0);
 popMatrix();
}
```

Output:

After uploading the code, the servo motors start running from 0 to 180 degrees and again back to 0 degrees. An ultrasonic sensor also rotates along with the servo as it is mounted on the motor.

Now, open the processing application and paste the above code. In this code, update the COM port number where your Arduino board is connected.

Now, run this processing code. If your code is right then, you will get a new window. This is the graphical representation of data from the Ultrasonic Sensor is represented in a radar type display.

If an ultrasonic sensor detects any object within its range, you can see the same on the graphical representation. The below gif shows the output of the Arduino radar project.

For surveillance tasks in the safety and security field, radar sensors are often essential instruments.

The benefits of radars are evident: radar is capable of detecting motion, measuring speed, distance and the angle of arrival as well as the direction of movement.

Radar can work in adverse conditions like rain, fog and dust and is able to cover long range as well as close distance. It preserves anonymity due to the fact that it is not able to deliver high-resolution pictures from faces or license plates, which ensures its acceptance.

Motion sensors for anti-burglary protection systems – indoor or outdoor – are very inexpensive components, which take advantage of the doppler effect. In this application, radar replaces the Passive Infrared Sensors, which have been established in the past because of their low price.

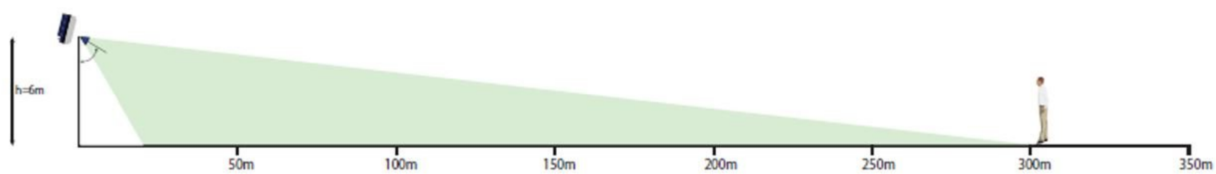
Reliable

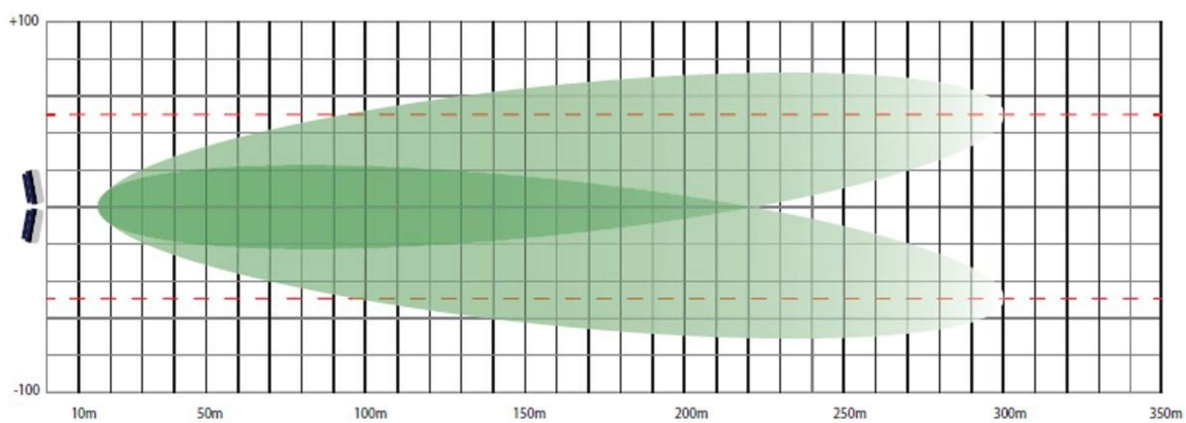
Radar, however, is much more reliable, because it is not affected by thermal variations and it offers additional information like direction of movement and the actual speed, making detection safer and more robust.

Broader view

If a broader view is required, two sensors can be combined in a Master-Slave configuration. An object list as well as tracks of detected targets with range, speed and angle information is accessible via Ethernet interface.

Due to the sensor´s architecture, multiple objects can be captured, and they can be separated as long as they differ in speed or range.





How did we get this idea?

Advantages of RADAR

1. RADAR can penetrate mediums such as clouds, fogs, mist, and snow. *The signals used by RADAR technology are not limited or hindered by snow, clouds, or fog. This means that even in the presence of these adverse conditions, data will still be collected.*

2. RADAR signals can penetrate insulators. *Materials that are considered insulators such as rubber and plastic do not hinder RADAR signals from collecting data. The signals will penetrate the materials and capture the necessary data required.*

3. It can give the exact position of an object. *RADAR systems employ the use of electromagnetic radiation to calculate the distance of an object and its exact position on the earth's surface or space.*

4. It can determine the velocity of a target. *RADAR systems have the capability of calculating the velocity of an object in motion. Besides knowing its location,* *you will also have data regarding the velocity of the object.*

5. It can measure the distance of an object. *RADAR systems work by measuring the exact distance of an object from the transmitter.*

6. It can tell the difference between stationary and moving targets. *The data collected by RADAR systems is enough to tell whether the object was in motion or it was stationary.*

7. RADAR signals do not require a medium of transportation. *RADAR employs the use of radio signals that can travel in air or space. They do not require any medium to be transported.*

8. RADAR signals can target several objects simultaneously. *The radio signals used by RADAR operate on a wider area and can target more than one object and return data regarding all the objects targeted.*

9. It allows for 3D Imaging based on the various angles of return. *The data captured by RADAR* *systems can be used to map an area and provide 3D images of the area based on the varying angles of return.*

10. It is cheaper as compared to other systems.
RADAR systems are relatively cheaper especially if used for large-scale projects.

These all advantages made me think, how it would be if we can make one! So we took on this project. We, on our own came up with the idea(we followed some references for further development)