

## **Title:-** DIY Radar With Ultrasonic Sensor.

### **Introduction:-**

A radar system using an Arduino Nano is an exciting project that involves building a basic radar-like system for object detection and tracking. Here's a general introduction to such a project:

#### **Project Overview:-**

A radar system is a device that uses radio waves to detect and locate objects in its vicinity. While professional radar systems are complex and expensive, a DIY radar project using an Arduino Nano can provide a simple and educational introduction to radar concepts.

#### **Components Needed:-**

To build a basic radar system, you'll require the following components:

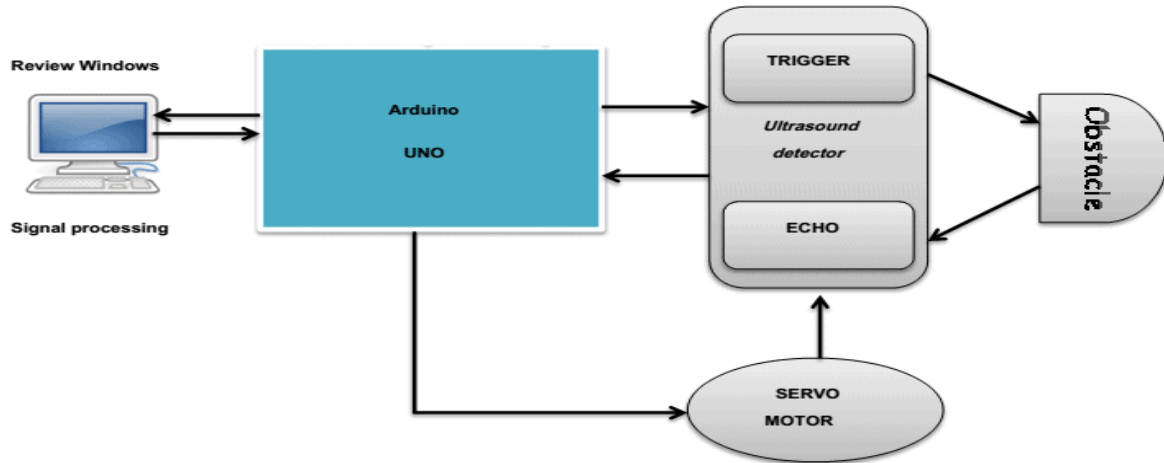
1. **Arduino Nano:** This microcontroller serves as the brain of the radar system, controlling the operation and processing data.
2. **Ultrasonic or Microwave Sensor:** Typically, an ultrasonic or microwave sensor is used to transmit and receive signals. These sensors emit a signal and measure the time it takes for the signal to bounce back from an object. The speed of sound or radio waves is used to calculate the distance.
3. **Servo Motor:** A servo motor is used to move the sensor, allowing it to scan its surroundings.
4. **Display:** An LCD or LED display can be used to visualize the detected objects or distances.

### **Applications:-**

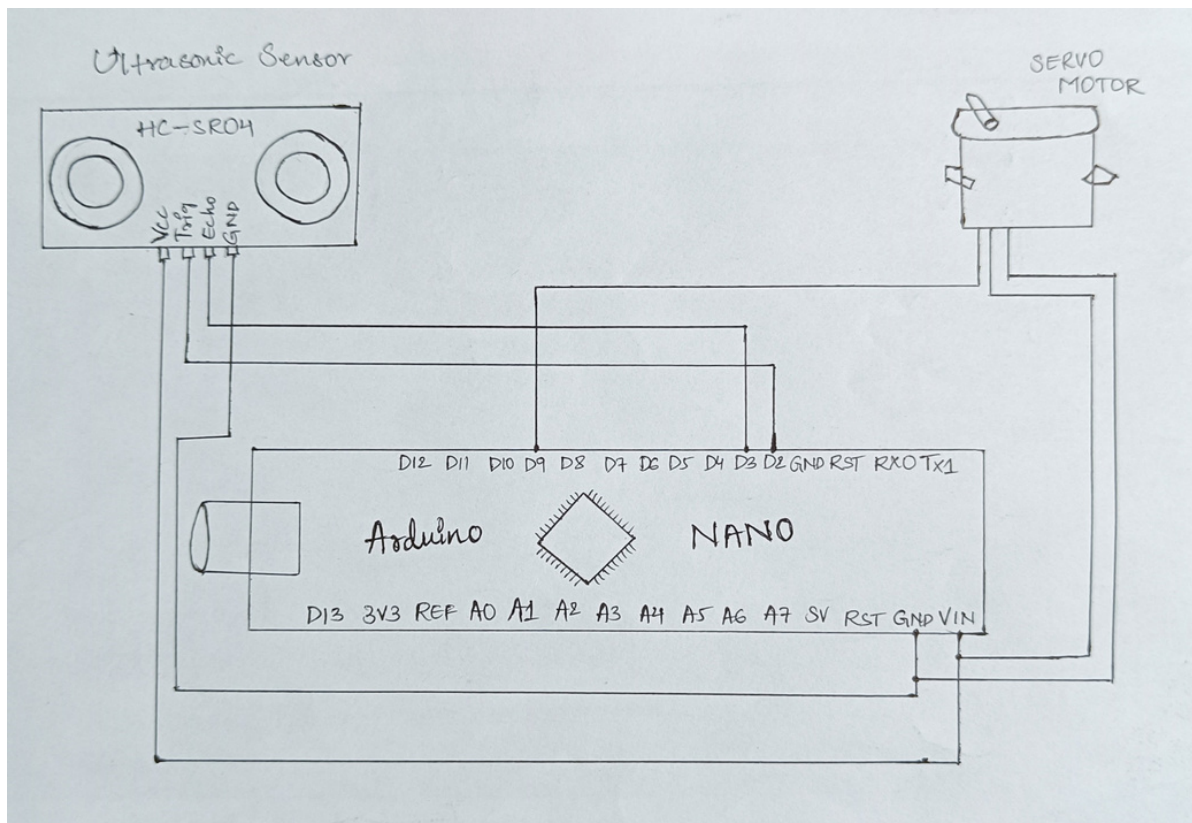
While this DIY radar system is basic compared to professional radar technology, it can serve as a learning platform for understanding the principles of radar. You can use it for fun applications like object detection in a room or as part of a security system.

Remember that building a radar system from scratch can be complex, and safety precautions should be taken into consideration, especially when dealing with microwave sensors. Always follow safety guidelines and consider the local regulations for using such devices.

## Block Diagram:-



## Circuit Diagram:-



## **Working:-**

To build the project, follow these steps:

### **Step 1**

First, identify these components.

Arduino Nano board

Ultrasonic sensor

Servo motor

Breadboard

Jumper wires

### **Step 2**

OK, Now attach the Arduino Nano board to the Breadboard.

### **Step 3**

Then connect the servo motor to one side of the breadboard. For that, I'm using a glue gun.

### **Step 4**

Next, attach the ultrasonic sensor to the top of the servo motor.

### **Step 5**

Now, wire these components using the circuit diagram below.

### **Step 6**

Well, next we will create the required Arduino program for this.

### **Step 7**

Well, now upload this code to the Arduino board. For that, select the correct board and port. After, upload this code.

### **Step 8**

OK, Now Create the processing ide program.

### **Step 9**

Well, now let us finally run this processing code.

## Coding: -

### Arduino Program

```
#include <Servo.h>//include library
Servo myServo;//create object your own name
int dis;
void setup() {
  pinMode(2, OUTPUT);//define arduino pin
  pinMode(3, INPUT);//define arduino pin
  Serial.begin(9600);//enable serial monitor
  myServo.attach(9);//servo connect pin
}
void loop() {
  for (int x = 0; x <= 180; x++) { //servo turn left
    myServo.write(x);//rotete servo
    dis=distance();
    Serial.print(x);//print servo angle
    Serial.print(",");
    Serial.print(dis);//print ultrasonic readings
    Serial.print(".");
    delay(50);
  }
  for (int y = 179; y > 0; y--) { //servo turn right
    myServo.write(y);//rotete servo
    dis=distance();
    Serial.print(y);///print servo angle
    Serial.print(",");
    Serial.print(dis);//print ultrasonic readings
    Serial.print(".");
    delay(50);
  }
}
//ultrasonic sensor code
int distance() {
  digitalWrite(2, LOW);
  delayMicroseconds(4);
  digitalWrite(2, HIGH);
  delayMicroseconds(10);
  digitalWrite(2, LOW);

  int t = pulseIn(3, HIGH);
  int cm = t / 29 / 2; //time convert distance
  return cm;//return value
}
```

## IDE Program

```
import processing.serial.*;
import java.awt.event.KeyEvent;
import java.io.IOException;

Serial myport;//create serial object
PFont f;
int Angle, Distance;
String angle="";
String distance="";
String data;
int index1=0;
int index2=0;
float pixsDistance;
void setup() {
size(1250, 700);//screen size
smooth();
printArray(PFont.list());//show font list your computer
f = createFont("David Bold", 30);//font name and size
textFont(f);

String portName = Serial.list()[0];//set COM port
myport = new Serial(this, portName, 9600);
myport.bufferUntil('.');
}
void draw() {
noStroke();
fill(0, 10);
rect(0, 0, width, 700);
fill(98, 245, 31);
greenmesh();
radararea();
words();
greenLine();
redline();
}
//get arduino board serial values
void serialEvent (Serial myport) {
data = myport.readStringUntil('.');
data = data.substring(0, data.length()-1);
```

```

index1 = data.indexOf(",");
angle= data.substring(0, index1);
distance= data.substring(index1+1, data.length());

// converts the String variables into Integer
Angle = int(angle);
Distance = int(distance);
}

//Half circle and lines
void radararea() {
pushMatrix();
translate(625, 680);
noFill();
strokeWeight(2);
stroke(98, 245, 31);
// draws the arc lines
arc(0, 0, 1150, 1150, PI, TWO_PI);
arc(0, 0, 850, 850, PI, TWO_PI);
arc(0, 0, 550, 550, PI, TWO_PI);
arc(0, 0, 250, 250, PI, TWO_PI);
// draws the angle lines
line(-450, 0, 450, 0);
line(0, 0, -600*cos(radians(30)), -600*sin(radians(30)));
line(0, 0, -600*cos(radians(60)), -600*sin(radians(60)));
line(0, 0, -600*cos(radians(90)), -600*sin(radians(90)));
line(0, 0, -600*cos(radians(120)), -600*sin(radians(120)));
line(0,0,-600*cos(radians(150)), -600*sin(radians(150)));
line(-960*cos(radians(30)), 0, 960, 0);
popMatrix();
}
//Green box net
void greenmesh() {
stroke(98, 245, 31);
strokeWeight(0.1);
for (int x=0; x<=700; x+=5) {
line(0, x, width, x);
}
for (int y=0; y<=1250; y+=5) {
line(y, 0, y, height);
}
}
} //print text

```

```

void words() {
fill(98, 245, 31);
text("180", 10, 670);
fill(98, 245, 31);
text("0", 1210, 670);
fill(98, 245, 31);
text("30", 1160, 380);
fill(98, 245, 31);
text("60", 940, 160);
fill(98, 245, 31);
text("90", 615, 70);
fill(98, 245, 31);
text("120", 310, 150);
fill(98, 245, 31);
text("150", 80, 370);
fill(255);
text("SriTu Tech Radar system", 20, 30);
fill(255);
text("Angle -- "+Angle+" ", 20, 60);
fill(255);
text("Distance -- "+Distance+" cm", 20, 90);
}
//Drawing green lines
void greenLine() {
pushMatrix();
strokeWeight(7);
stroke(30, 250, 60);
//green color
translate(625, 680);
line(0,0,600*cos(radians(Angle)), -600*sin(radians(Angle)));
popMatrix();
}
//Drawing red lines
void redline() {
pushMatrix();
translate(625, 680);
strokeWeight(7);
stroke(255, 10, 10); //red color
pixsDistance = Distance*22.5;
// limiting the range to 40 cm
if (Distance<40) {
line(pixsDistance*cos(radians(Angle)), -pixsDistance*sin(radians(Angle)),
600*cos(radians(Angle)), -600*sin(radians(Angle)));
}
popMatrix();
}
}

```

### Bill Material:-

Sr. No.	Items Discription	Quantity	Amount
1	Arduino Nano	1	400
2	Servo Motor	1	150
3	Ultrasonic sensor	1	120
4	Breadboard	1	60
5	Jumper Wire	10	30
Total:-			760

### Conclusion:-

In conclusion, the DIY Radar System using Arduino Nano is an educational and exciting project that offers valuable insights into radar technology and embedded systems. Here's a summary of the project:

**1. Educational Value:** Building a radar system with an Arduino Nano is a great way to learn about various concepts, including sensor technology, microcontroller programming, and signal processing.

**2. Hands-On Experience:** The project provides hands-on experience in assembling hardware components, programming the Arduino, and troubleshooting issues, enhancing your practical skills.

**3. Object Detection:** The radar system enables object detection and tracking, making it suitable for applications like security systems, obstacle avoidance, or even recreational purposes.

**4. Flexibility:** DIY radar systems can be customized to meet specific project requirements, offering flexibility in terms of sensor selection, scanning patterns, and data visualization.

**5. Challenges and Problem-Solving:** As with any DIY project, you may encounter challenges during the build, helping you develop problem-solving and critical thinking skills.

**6. Safety Considerations:** It's important to follow safety guidelines when working with sensors, especially those emitting radiation, to avoid potential hazards.

**7. Expandability:** Once you've built a basic radar system, you can expand and improve it with additional features, like data logging, wireless connectivity, or integration with other devices.

Overall, a DIY radar system using an Arduino Nano is a rewarding project that combines engineering, electronics, and programming. It encourages creativity and exploration in the field of sensor technology, making it a valuable learning experience for enthusiasts, students, and hobbyists.

**Submitted by: -**

Sr.no.	Name	Roll No
1.	Ishwari Dhote	0104
2.	Yashika Chaudhary	0122
3.	Ankit Ganvir	0123

Section:- A

Sem/Year:- 5th sem/ 3rd year

Branch:- Electronics and Communication Department

Academic Year :- 2023-2024