A Project Report

on

IoT based Radar for Object Detection

Submitted By:

Roll	Name
2106060	Somya Jha
2106061	Palak Krishna
2106062	Abhishek Kumar
2106067	Harshit Kumar

CSE-1, A2

Computer Science and Engineering

National Institute of Technology Patna

Bihar-800005

December 2023

ABSTRACT

Radar is an object detection system which uses radio waves to determine the range, altitude, direction, or speed of objects. It can be used to detect aircraft, ships, spacecraft, guided missiles, motor vehicles, weather formations, and terrain. The radar dish or antenna transmits pulses of radio waves or micro waves which bounce off any object in their path. The object returns a tiny part of the wave's energy to a dish or antenna which is usually located at the same site as the transmitter.

The modern uses of radar are highly diverse, including air traffic control, radar astronomy, air-defense systems, antimissile systems ;marine radar start locate landmarks and other ships; aircraft anti-collision systems; ocean surveillance systems, outer space surveillance and rendezvous systems; meteorological precipitation monitoring; altimetry and flight control systems; guided missile target locating systems; and ground-penetrating radar for geological observations. High tech radar systems are associated with digital signal processing and are capable of extracting useful information from very high noise levels.

The Arduino based project requires a ultrasonic sensor, the sensor released the waves which we want to measure the distance of a object. The microcontrollers of the Arduino board can be programmed using C and C++ languages. When a code is written in Arduino UNO IDE software and connected to the board through a USB cable, Arduino boards have lot of applications in the present day scenario, so we have decided to do a small project on them.

INTRODUCTION

Defining Arduino: An Arduino is actually a microcontroller based kit which can be either used directly by purchasing from the vendor or can be made at home using the components, owing to its open source hardware feature. It is basically used in communications and in controlling or operating many devices.

1. Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.
2. Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.
3. Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is opensource, and it is growing through the contributions of users worldwide.

1.1. ULTRASONIC SENSOR

As the name indicates, ultrasonic sensors measure distance by using ultrasonic waves.The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception.
An optical sensor has a transmitter and receiver, whereas an ultrasonic sensor uses a single ultrasonic element for both emission and reception. In a reflective model ultrasonic sensor, a single oscillator emits and receives ultrasonic waves alternately. This enables miniaturization of the sensor head.
Distance calculation

The distance can be calculated with the following formula:

Distance $L = 1/2 \times T \times C$

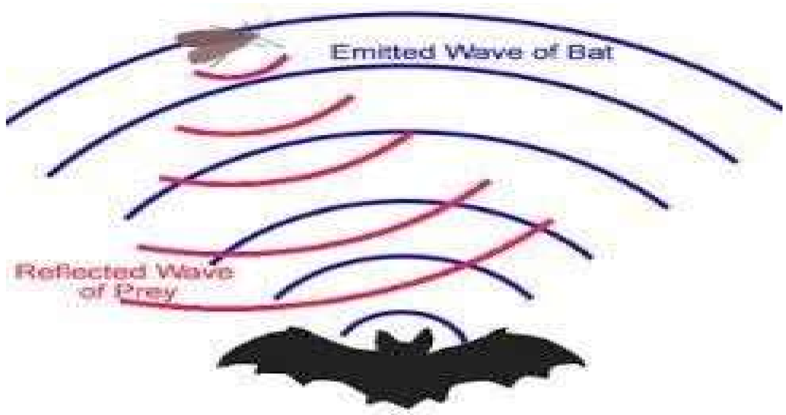
Where L is the distance, T is the time between the emission and reception, and C is the sonic speed. (The value is multiplied by 1/2 because T is the time for go-and-return distance.) Features
The following list shows typical characteristics enabled by the detection
system. [Transparent object detectable]
Since ultrasonic waves can reflect off a glass or liquid surface and return to the sensor head, even transparent targets can be detected.
[Resistant to mist and dirt]

Detection is not affected by accumulation of dust
or dirt. [Complex shaped objects detectable]
Presence detection is stable even for targets such as mesh trays or springs.

PRINCIPLE OR MEDTHODOLOGY

A radar system has a transmitter that emits radio waves called a radar signals in predetermined directions. When these come into contact with an object they are usually reflected or scattered in many directions Example:- let us take example for bat

Bat released the eco sound while travelling .if any object came in middle and it reflect back to the bat



Applications and usages:-

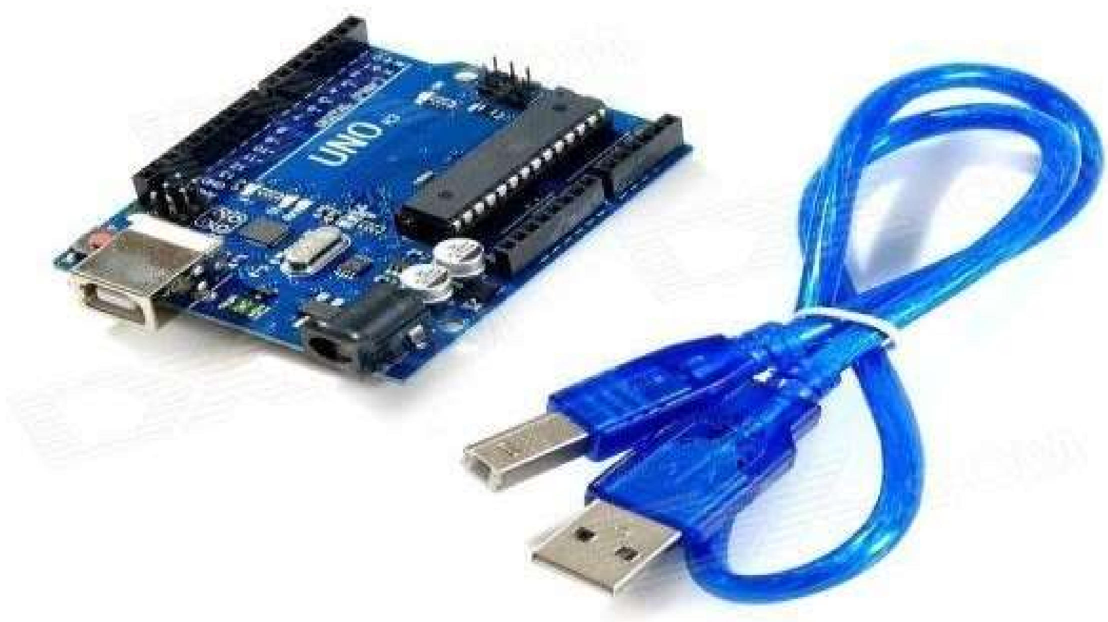
The development of the radar technology took place during the World War II in which it was used for detecting the approaching aircraft and then later for many other purposes which finally led to the development of advanced military radars being used these days. Military radars have a highly specialized design to be highly mobile and easily transportable, by air as well as ground. Military radar should be an early warning, altering along with weapon control functions. It is specially designed to be highly mobile and should be such that it can be deployed within minutes.

PROCEDURE

Components Required:

In this project we have used the arduino and ultrasonic sensor along with the jumping wires and the relay motors and details list of the hard ware components are

- Arduino board and arduino cable



- Jumper wires

We create a variable analog and assign it to 0.This is because the voltage value we are going to read is connected to the analog pin is A0. This voltage represents the voltage value falls across the resistor value we are measuring. Next we create a variable name raw, which we will use to read in the analog voltage value. This later is our code get assigned to the analogue read () function.

ADVANTAGES:-

1. **The cost effective** : our project below 1000rs only.

2. **Improvised accuracy**: The resistors with low value in milliohms are used in advanced cars with sensitive power steering and break circuits. Now a days these advancements have become the major cause for the severe accidents . Therefore the components used in such circuits must have accurate and precise value for smooth working of such circuits. Ultimately this refers to the accurate testing of the resistors used. Improvised accuracy is thus the second primary aim of the sensor.
3. **Reduced hardware complexity**: Hardware complexity is one of the reasons for the high cost of the ultrasonic sensor. The use of arduino Uno is to reduce the motherboard present in the conventional ohmmeter in arduino based ultrasonic sensor. The arduino acts as the central board. Since arduino are readily available in market it leads to the reduction in the complexity of the design. The automated range selection is also the objective in order to speedup the testing process. This will also reduce the faults in range selection in manually operated conventional sensor.

CONCLUSIONS:-

This project aims on the use of Ultrasonic Sensor by connected to the Arduino UNO R3 board and the signal from the sensor further provided to the screen formed on the laptop to measure the presence of any obstacle in front of the sensor as well as determine the range and angle at which the obstacle is detected by the sensor.

Arduino Radar Project : Source Code

Team

2106060 Somya Jha ; 2106061 Palak Krishna

2106062 Abhishek Kumar ; 2106067 Harshit Kumar

```
/********** Arduino Radar Project *********/
// Includes the Servo library
#include <Servo.h>.

// Defines Tirg and Echo pins of the Ultrasonic Sensor
const int trigPin = 5;
const int echoPin = 6;
// 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( 2); // 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<=180;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=180;i>15;i--){
  myServo.write(i);
  delay(30);
  distance = calculateDistance();
  Serial.print(i);
  Serial.print(",");
  Serial.print(distance);
  Serial.print(".");
  }
}
// Function for calculating the distance measured by the Ultrasonic sensor
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;
}
```

Arduino Radar Visualisation : Source Code

```
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;
```

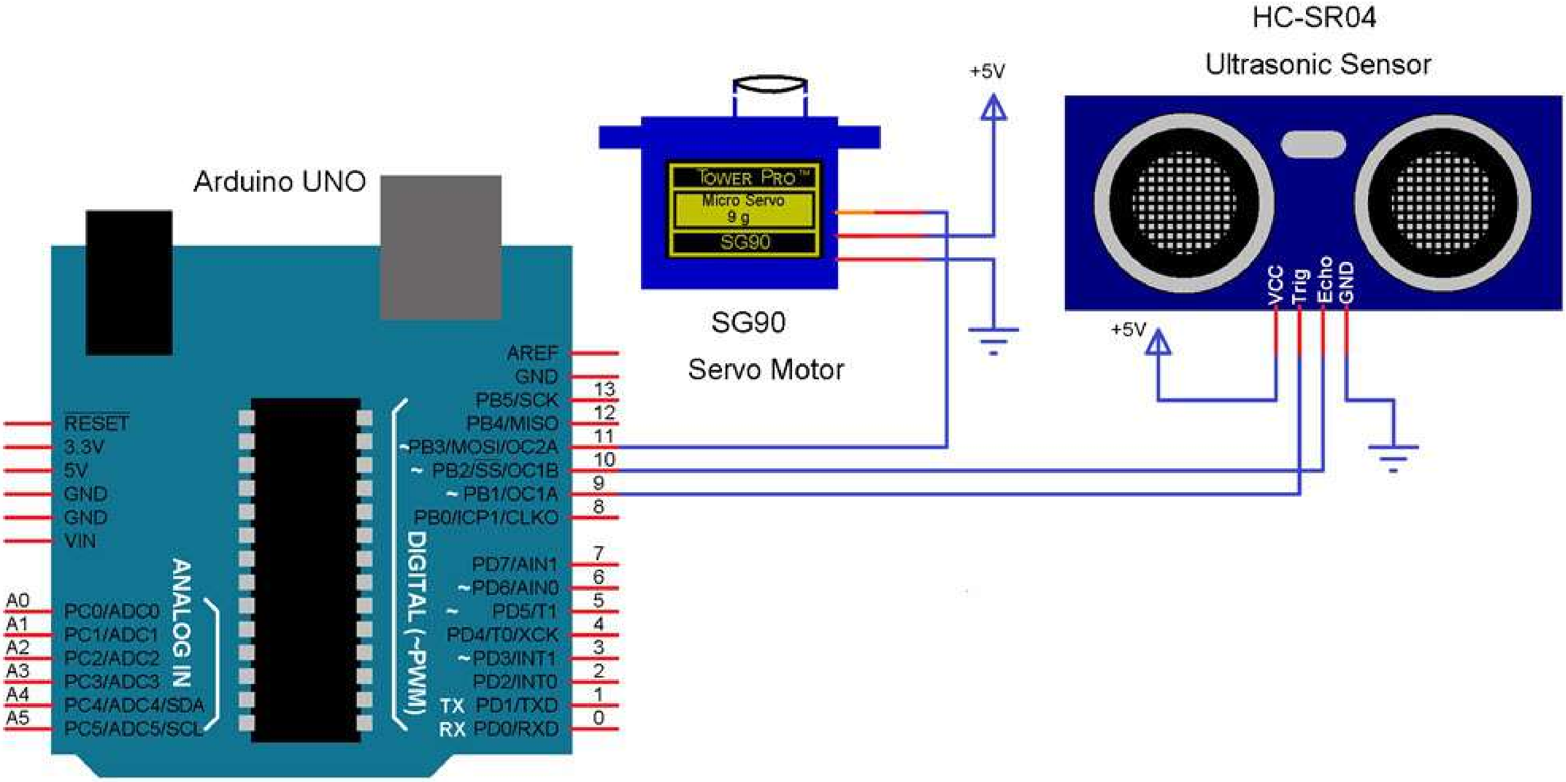
```
Serial myPort; // defines Object Serial
// defines variables
String angle="";
String distance="";
String data="";
String noObject;
float pixsDistance;
int iAngle, iDistance;
int index1=0;
int index2=0;
PFont orcFont;

void setup() {

 size (1200, 680);
 // Change this to your screen resolution
 // Size must be smaller than screen resolution
 //For best look, size must be (1200, 680);
 smooth();
 //You must change "COM5". To check the port name, go to Arduino
IDE>Tools>Port
 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();
```

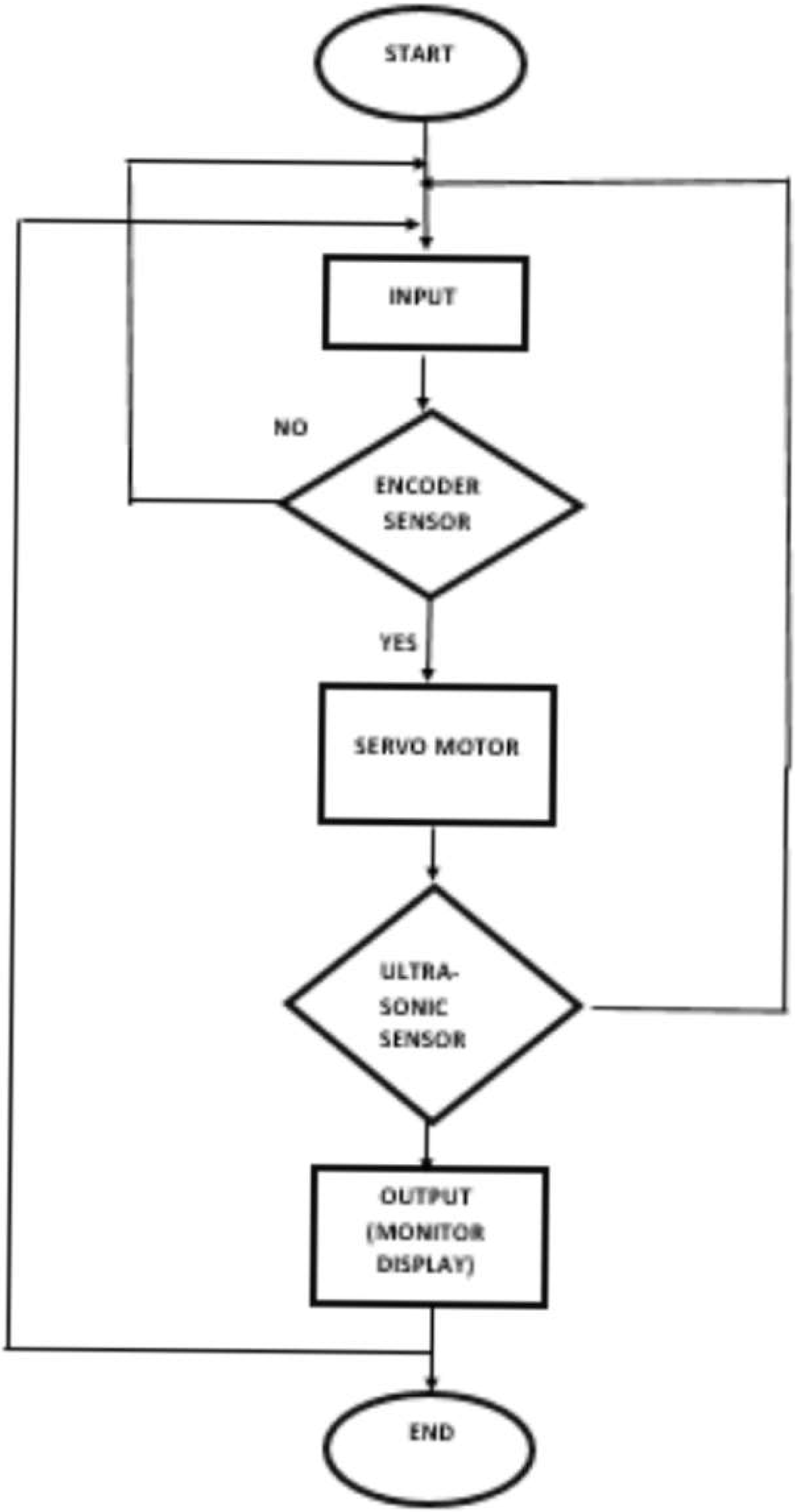
```
  drawObject();
  drawText();
}

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 thats 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 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);
  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)*s
in(radians(iAngle))); // draws the line according to the angle
  popMatrix();
}

void drawText() { // draws the texts on the screen
```

```
  pushMatrix();
  if(iDistance>40) {
  noObject = "Nothing";
  }
  else {
  noObject = "Detect";
  }
  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(32);
  text("Object: " + noObject, width-width*0.93, height-height*0.0277);
  text("Angle: " + iAngle +" °", width-width*0.5735,
height-height*0.0277);
  text("Distance: ", width-width*0.277, height-height*0.0277);
  if(iDistance<40) {
  text("        " + iDistance +" cm", width-width*0.225,
height-height*0.0277);
  }

  translate(width-width*0.98,height-height*0.95);
  text(“Radar",0,0);
  resetMatrix();

  textSize(25);
  fill(98,245,60);

translate((width-width*0.4994)+width/2*cos(radians(30)),(height-height*0.0
907)-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*0.08
88)-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*0.08
33)-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*0.071
29)-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.
0574)-width/2*sin(radians(150)));
  rotate(radians(-60));
  text("150°",0,0);
  popMatrix();
}
```


HC-SR04
Ultrasonic Sensor
+5V
Arduino UNO
TOWER PRO™
Micro Servo
9 g
SG90
SG90
Servo Motor
VCC
Trig
Echo
GND
+5V
AREF
GND
PB5/SCK
PB4/MISO
~PB3/MOSI/OC2A
~ PB2/SS/OC1B
~ PB1/OC1A
PB0/ICP1/CLKO
13
12
11
10
9
8
RESET
3.3V
5V
GND
GND
VIN
DIGITAL (~PWM)
ANALOG IN
PD7/AIN1
~PD6/AIN0
~ PD5/T1
PD4/T0/XCK
~PD3/INT1
PD2/INT0
TX PD1/TXD
RX PD0/RXD
7
6
5
4
3
2
1
0
A0 PC0/ADC0
A1 PC1/ADC1
A2 PC2/ADC2
A3 PC3/ADC3
A4 PC4/ADC4/SDA
A5 PC5/ADC5/SCL


START
INPUT
NO
ENCODER SENSOR
YES
SERVO MOTOR
ULTRA-SONIC SENSOR
OUTPUT (MONITOR DISPLAY)
END