Pranjal Malhotra 24BAI1110

animusaaa@gmail.com

RADAR DETECTION SYSTEM USING ARduino IDE

PRANJAL MALHOTRA -24BAI1110

Radar Detection System Using Arduino Uno, Ultrasonic Sensor, Servo Motor, and Processing 4

# Title Page

**Radar Detection and Visualization Using Arduino Uno and Processing 4**  
**Submitted By:**

* **Pranjal Malhotra** (Reg. No: 24BAI1110)
* **Prabhav Shukla** (Reg. No: 24BAI1051)
* **Dhananjay** (Reg. No: 24BAI1298)

**Course:** B.Tech – Artificial Intelligence and Machine Learning   
**Semester:** 2nd  
**Department of Computer Science and Engineering**  
**Academic Year:** 2024–2025

**PHYSICS PROJECT :**

**Slot : F1+TF1**

**Faculty : Prof Swaathi P**

Acknowledgement

We express our deepest gratitude to our faculty and the Department of Computer Science and Engineering for providing guidance and support throughout the duration of this project. This project allowed us to explore hands-on implementation of concepts in electronics, programming, and real-time data visualization.

We especially thank:

* **Pranjal Malhotra**, who contributed extensively to the C++ programming and hardware setup.
* **Prabhav Shukla**, who collaborated on coding and ensured robust wiring and circuit testing.
* **Dhananjay**, who worked on the software integration and the GUI development using Processing 4.

This project would not have been successful without the collaborative spirit and technical contributions of every team member.

Abstract

This project presents the design and implementation of a low-cost radar system using Arduino Uno, an ultrasonic sensor (HC-SR04), and a servo motor (SG90). The system mimics radar functionality by continuously scanning the environment in a semicircular arc and detecting obstacles using ultrasonic waves.

The system is integrated with Processing 4, a visual programming environment, to display the scanned data in a graphical radar-like interface. This enables users to visualize the presence and relative location of obstacles in real time. The rotating motion of the servo motor, controlled through Arduino, allows the ultrasonic sensor to measure distances at multiple angles, providing a 2D spatial mapping of nearby objects.

This project combines knowledge of microcontrollers, sensors, actuators, serial communication, and visual programming, offering a comprehensive introduction to embedded systems and real-time visualization.

Introduction

Radar systems have been pivotal in navigation, aviation, meteorology, and defense. The core principle of radar involves sending out signals (such as radio or ultrasonic waves), detecting their reflection from nearby objects, and calculating the distance based on the time taken for the reflection to return.

In this project, we attempt to simulate a simplified radar system using commonly available components. The Arduino Uno serves as the central microcontroller that controls both the ultrasonic sensor and the servo motor. The ultrasonic sensor emits sound waves and calculates the time it takes for them to reflect off objects, thus measuring distance. The servo motor rotates the sensor from 0° to 180°, enabling it to scan the environment.

All collected data is sent via serial communication to a laptop, where Processing 4 reads the angle and distance, and draws them in a radar-like interface. The real-time visualization gives the user a clear understanding of object placement and movement within the field of detection.

This project serves not only as a technical challenge but also as a demonstration of how embedded systems and software can be harmonized to create interactive, functional systems.

**Objectives**

The primary objectives of this project are:

1. **Design a functioning prototype** that mimics the working of a basic radar detection system using affordable and accessible components.
2. **Implement ultrasonic sensing** to measure the distance of obstacles accurately.
3. **Use a servo motor** to rotate the sensor and collect distance data across a semi-circular field.
4. **Enable serial communication** between the Arduino board and a computer to transmit sensor data.
5. **Develop a graphical user interface (GUI)** using Processing 4 to visualize the radar scan in real time.
6. **Enhance technical skills** in microcontroller programming, circuit design, and visual programming.

**Team Member Contributions**

| **Team Member** | **Reg. No** | **Area of Contribution** |
| --- | --- | --- |
| Pranjal Malhotra | 24BAI1110 | Wrote C++ code for Arduino, connected and tested the hardware, handled circuit setup. |
| Prabhav Shukla | 24BAI1051 | Co-authored the Arduino code, managed ultrasonic sensor testing, and validated logic. |
| Dhananjay | 24BAI1298 | Implemented Java-based GUI in Processing 4, established serial port communication. |

All three members took part in the **final testing and debugging**, ensuring that the system worked seamlessly and reliably.

Hardware Components and Pin Configuration

**1. Arduino Uno:**

* Acts as the brain of the system, managing input and output of sensor data and controlling the servo motor.

**2. Ultrasonic Sensor (HC-SR04):**

* Measures distance using the time of flight of ultrasonic pulses.
* **Pin Connections:**
  + VCC → 5V on Arduino
  + GND → GND on Arduino
  + Trig → Digital Pin 10
  + Echo → Digital Pin 11

**3. Servo Motor (SG90):**

* Rotates the sensor across a 180° field.
* **Pin Connections:**
  + Red (Power) → 5V on Arduino
  + Brown (Ground) → GND on Arduino
  + Orange/Yellow (Signal) → Pin 12

**4. Power Supply:**

* Powered via USB connection to computer or external 5V source.

**5. Breadboard and Jumper Wires:**

* For easy and safe electrical connections.

**Arduino Code Explanation**

The Arduino sketch controls the servo to rotate from 0° to 180° and back, measuring the distance at each step. The data (angle and distance) is sent via the serial port.

**Arduino Code:**

#include <Servo.h>

Servo myServo; // create servo object

int trigPin = 10;

int echoPin = 11;

long duration;

int distance;

void setup() {

myServo.attach(12); // attach servo to pin 12

Serial.begin(9600);

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

}

void loop() {

for (int angle = 0; angle <= 180; angle++) {

myServo.write(angle);

delay(15);

distance = getDistance();

Serial.print(angle);

Serial.print(",");

Serial.println(distance);

}

for (int angle = 180; angle >= 0; angle--) {

myServo.write(angle);

delay(15);

distance = getDistance();

Serial.print(angle);

Serial.print(",");

Serial.println(distance);

}

}

int getDistance() {

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

return duration \* 0.034 / 2; // distance in cm

}

GUI with Processing 4 (Java)

The Processing sketch reads the serial data, splits it into angle and distance, and plots it on a radar-style interface.

import processing.serial.\*;

Serial myPort;

String data;

float angle, distance;

void setup() {

size(800, 800);

myPort = new Serial(this, Serial.list()[0], 9600);

myPort.bufferUntil('\n');

}

void draw() {

background(0);

translate(width/2, height/2);

drawRadar();

drawLine();

}

void serialEvent(Serial myPort) {

data = myPort.readStringUntil('\n');

if (data != null) {

data = trim(data);

String[] parts = split(data, ',');

if (parts.length == 2) {

angle = float(parts[0]);

distance = float(parts[1]);

}

}

}

void drawRadar() {

stroke(0, 255, 0);

noFill();

for (int r = 100; r <= 400; r += 100) {

ellipse(0, 0, r \* 2, r \* 2);

}

}

void drawLine() {

float x = cos(radians(angle)) \* distance \* 2;

float y = sin(radians(angle)) \* distance \* 2;

stroke(255, 0, 0);

line(0, 0, x, y);

}

**Results and Observations**

* The radar system successfully scanned objects within a 180° field of view.
* Obstacles appeared accurately on the GUI, with real-time updates.
* The system could detect objects within a range of 2 cm to ~400 cm.
* Performance was stable, and servo motion was smooth.
* All components worked as expected under different lighting and environmental conditions.