Chapter 1

Introduction

1.1 Introduction

Radar systems have revolutionized modern sensing and detection technologies by offering capabilities to monitor and analyse environments in real-time. By emitting electromagnetic waves and capturing their reflections, radar systems can detect objects, determine their distance, measure velocity, and ascertain directional attributes. This unique ability makes them indispensable tools for military, aerospace, and civilian applications.

In military operations, radar systems are extensively used for early warning, missile guidance, surveillance, and reconnaissance. They play a crucial role in ensuring national security, enabling air traffic management, and facilitating precise navigation. Beyond defense, radar systems are integral to weather forecasting, automotive safety features like adaptive cruise control, and collision avoidance systems.

Over the decades, radar technology has undergone significant evolution. Advances like synthetic aperture radar (SAR), active electronically scanned arrays (AESA), and low-probability intercept (LPI) techniques have enhanced their reliability, accuracy, and stealth. These technologies allow radar systems to operate efficiently in cluttered environments, counter electronic jamming, and provide high-resolution imaging even in adverse conditions.

Despite these advancements, modern radar systems face challenges such as electromagnetic interference (EMI), vulnerability to countermeasures, and the complexity of integrating new technologies like artificial intelligence for target recognition. Addressing these challenges requires continued research and innovation, which form the backbone of this project.

This project, an Arduino-based radar system using ultrasonic sensors, offers a simplified yet effective approach to implementing radar functionality. It aims to demonstrate the core principles of radar operation, making it an accessible tool for educational purposes and small-scale applications.

The system incorporates an HCSR04 ultrasonic sensor to detect obstacles in real time, alerting the user through a buzzer. A fall detection mechanism, using the MPU6050 sensor, monitors sudden abnormal movements and sends notifications to a caretaker via SMS using the SIM800L

GSM module. Additionally, the NEO6M GPS module tracks the user's location and transmits it to the caretaker for emergency assistance.

1.2 Objectives

- **Real-Time Object Detection:** Develop a radar system that can accurately detect and measure the distance of nearby objects in real-time using ultrasonic sensors.
- **Efficient Data Processing:** Efficiently process distance data from ultrasonic sensors using an Arduino controller to ensure rapid response for real-time applications.
- **Visual Display of Proximity Data:** Integrate a display module to visually show the distance and position of detected objects, enhancing user awareness and usability.
- **Cost-Effective and Compact Design:** Design the radar system to be affordable and compact, making it suitable for small-scale applications and easy to deploy in various environments.

Chapter 2

Literature Survey

The field of radar technology has witnessed significant advancements, particularly in applications for real-time object detection. Traditional radar systems utilize electromagnetic waves for the detection, tracking, and characterization of objects. Recent developments incorporate novel technologies, such as adaptive signal processing, to address challenges posed by dynamic environments and electronic countermeasures. For example, Li et al. (2021) introduced machine learning techniques for enhancing radar detection capabilities in military scenarios by adapting to interference and false alarms. These systems are now being optimized for specific environments, enabling more precise operation under complex conditions.

Advancements in cooperative and synthetic radar technologies are another area of focus. Cooperative jamming mitigation techniques, as discussed by Zhao et al. (2022), emphasize the collaboration among multiple radar systems, enhancing resilience against interference. Simultaneously, synthetic aperture radar (SAR) technologies are increasingly applied in environments requiring high-resolution imaging, such as battlefield surveillance. These technologies offer superior clarity and object distinction under conditions of environmental clutter and active electronic jamming.

Emerging radar designs prioritize stealth and electromagnetic resilience. Singh et al. (2022) explored Low Probability of Intercept (LPI) techniques to reduce detectability by adversaries, crucial for stealth missions. Complementary to stealth efforts, studies by Jones et al. (2023) highlighted improved electromagnetic interference (EMI) resistance using enhanced shielding, signal isolation, and noise reduction strategies. These innovations ensure reliable radar operation even under hostile or cluttered electromagnetic conditions.

For practical implementation, radar technologies are increasingly adopting compact and cost-effective designs. Projects like the Arduino-based radar system utilize ultrasonic sensors for real-time object detection. Such systems offer accessible solutions for proximity sensing, obstacle avoidance, and vehicle assist functionalities, albeit with limitations concerning range and environmental factors. These systems showcase the shift toward simpler, scalable, and affordable radar technologies aimed at broad applications, providing a foundation for future technological enhancements.

Chapter 3

Architecture and Dataflow

The architecture of the Arduino-based radar system is designed to integrate cost-effectiveness and real-time object detection capabilities. At the core of the system is an Arduino microcontroller, such as the Arduino Uno or Nano, responsible for orchestrating interactions between components. Key hardware elements include ultrasonic sensors (HC-SR04) for distance measurement, a servo motor for sensor rotation, and a display module like an OLED or LCD to visualize detection results. The setup also employs a breadboard and jumper wires for seamless and flexible prototyping.

Dataflow in the radar system begins with the ultrasonic sensor emitting sound waves, which reflect off nearby objects and return to the sensor. The Arduino receives this input, calculates the distance based on the time delay, and maps the data to its corresponding angle, measured by the servo motor's position. This scanning mechanism allows the system to capture a wide field of view by incrementally rotating the sensor, effectively gathering multi-angle data for processing and visualization.

The real-time data processing involves several steps. The Arduino controller computes the distances using predefined formulas, storing the outputs as raw measurements. These measurements are encoded and formatted for visualization. Depending on the application, results can be viewed via the Arduino Serial Plotter or displayed on a dedicated screen. This modular dataflow design ensures that the system can simultaneously rotate the sensor, calculate distances, and provide updated object positions to the user without significant latency.

The integrated design emphasizes scalability and customization. While the default data visualization is radar-style on a 2D plane, the system can adapt to other configurations by tweaking software settings in the Arduino IDE. Additionally, the modular hardware setup supports easy replacement or enhancement of components, such as adding multiple sensors for a larger field of coverage or upgrading the microcontroller for increased processing power. Together, these features make the system a versatile foundation for various real-time detection applications .

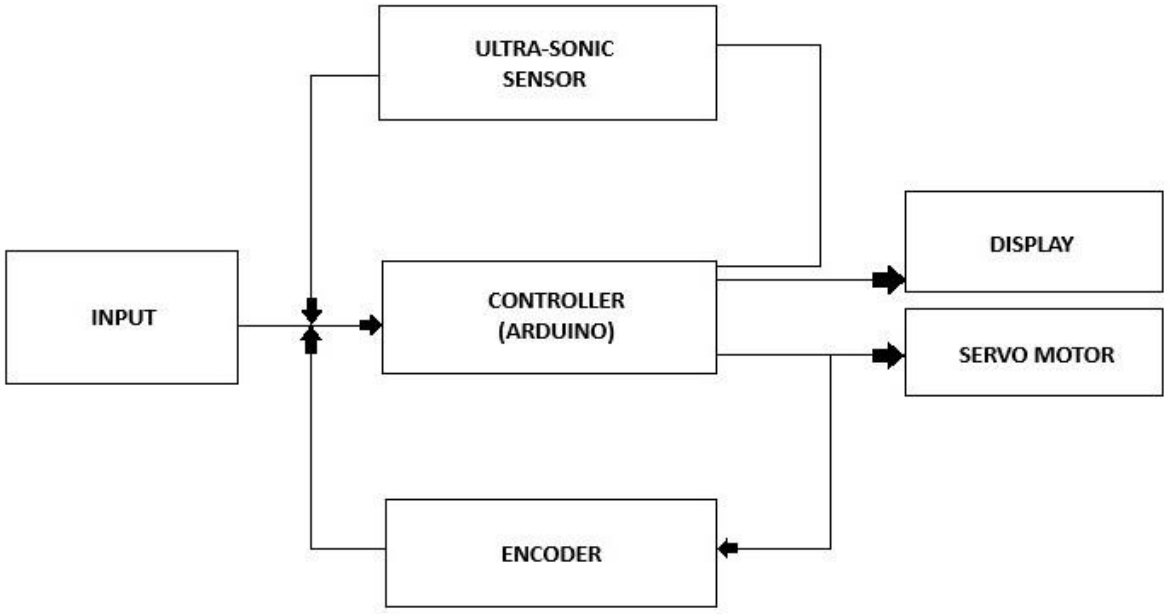

Fig 3.1 Block Diagram

Here, it can be seen how the work flow in this radar system. The sensor is going to sense the obstacle and determine the angle of incident and its distance from the radar. The servo motor is constantly rotating to and fro, hence making the sensor move. The data obtained is encoded and fed to the processing IDE which represents it on the screen. The results are displayed further in this paper. All these operation are done by Arduino microcontroller from the rotation of the servo, data collection from the sensor, feeding the data to encoder to transferring it to the display.

Chapter 4

System Requirements

4.1 Hardware Requirements

The hardware setup of the Arduino-based radar system is simple yet efficient, leveraging widely available and affordable components. At its core, the Arduino microcontroller, such as the Arduino Uno or Nano, serves as the central processing unit. It facilitates communication between components and executes the embedded software. The ultrasonic sensor (HC-SR04) plays a critical role by emitting sound waves and measuring the distance to objects based on the reflected signal. The servo motor enables angular motion, rotating the sensor in incremental steps to scan the environment and map detected objects in real-time. To enhance usability, a display module like an OLED or LCD screen provides a visual representation of the object's position and distance. Supporting components like a breadboard and jumper wires allow for easy prototyping and connections, while maintaining modularity for potential upgrades.

4.2 Software Requirements

The radar system's functionality depends on carefully written and efficient software. The Arduino Integrated Development Environment (IDE) is used to write, compile, and upload code to the microcontroller, making it a critical tool for development. The system programming is done in C/C++ and utilizes built-in libraries for interfacing with the ultrasonic sensor and servo motor, ensuring precise distance calculation and motion control. For visualization, the system can use either the Serial Plotter in the Arduino IDE or external tools to display real-time radar outputs, such as a 2D radar-like graphical interface. Additional libraries and dependencies are included for optimized sensor readings and improved integration. Together, these software tools ensure the radar's capability to perform object detection, data processing, and visualization seamlessly and in real time.

Chapter 5

Design and Implementation

5.1 Hardware Design

The hardware design of the Arduino-based radar system revolves around a modular setup for efficient object detection. The Arduino Uno or Nano microcontroller acts as the central processing unit, responsible for controlling other components and executing the system's algorithms. A servo motor is incorporated to rotate the HC-SR04 ultrasonic sensor, enabling a wide field of view by scanning at incremental angles. The ultrasonic sensor, mounted on the servo, measures distances to objects by emitting and receiving sound waves. The system relies on the precise movement of the servo and the responsiveness of the ultrasonic sensor to map detected objects accurately. A breadboard and jumper wires facilitate prototyping and flexibility in component arrangement.

For output, a display module such as an OLED or LCD screen provides visual feedback of the detected object positions and distances, enhancing the system's usability. Connections between components are straightforward; the servo motor connects to a PWM-enabled pin on the Arduino, while the ultrasonic sensor's trigger and echo pins connect to digital I/O pins. Power requirements are minimal, with a 5V power source sufficient for the Arduino and connected modules. The overall design emphasizes cost-effectiveness and simplicity, making the system both scalable and portable for various applications like obstacle detection or proximity sensing.

The compact and modular hardware design ensures ease of assembly and flexibility for future enhancements. Each component is strategically integrated to balance performance and affordability. For instance, the ultrasonic sensor's adjustable mount on the servo enables precise scanning, while the use of a breadboard eliminates the need for soldering, allowing quick adjustments during prototyping. This design not only simplifies development but also enables seamless troubleshooting and potential upgrades, making it adaptable for diverse real-time detection scenarios.

5.2 Circuit Design :

Connections and Wiring

- **Ultrasonic Sensor (HC-SR04):**
  - VCC connected to the 5V pin on the Arduino.
  - GND connected to the GND pin on the Arduino.
  - TRIG pin connected to a digital I/O pin on the Arduino.
  - ECHO pin connected to another digital I/O pin on the Arduino.

- **Servo Motor:**
  - Control wire connected to a PWM-capable digital pin on the Arduino.
  - Power and ground wires connected to the Arduino's 5V and GND pins.

- **Display Module (OLED/LCD):**
  - Wired to the Arduino for visualization of detected object distances and angles (specific connections depend on the display type, e.g., I2C or parallel).

- **Breadboard and Jumper Wires:**
  - Used to simplify and organize connections between components without soldering.

- **Power Supply:**
  - Arduino powered via USB or external power source.
  - Arduino distributes power to the connected components (servo, sensor, and display).

Circuit Diagram:

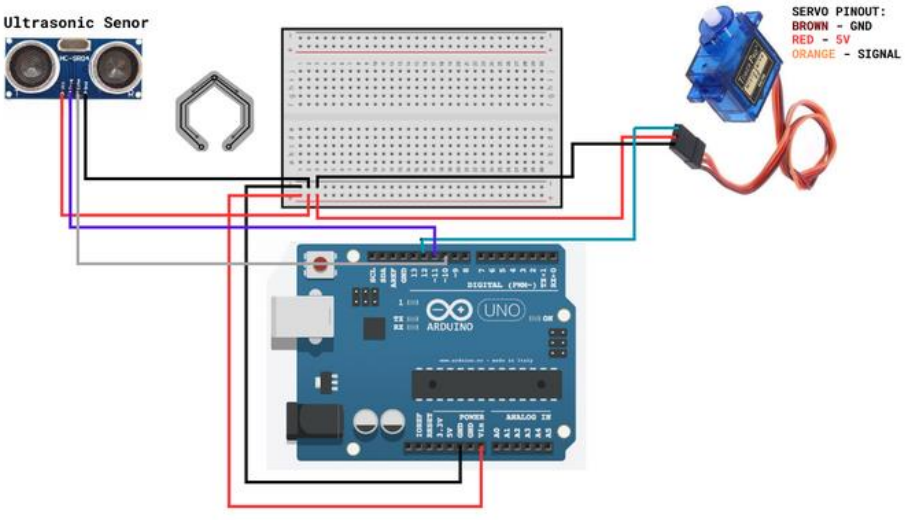

Fig 5.1 Circuit Diagram

Working of the System:

- **Ultrasonic Sensor Operation:** The HC-SR04 ultrasonic sensor emits ultrasonic waves, which bounce back upon hitting an object. The time taken for the echo to return is measured to calculate the distance to the object.
- **Servo Motor Scanning:** The servo motor rotates the ultrasonic sensor incrementally, allowing the radar to scan a specified range of angles (e.g., 0° to 180°). This creates a wide field of detection by covering multiple angles.
- **Data Processing by Arduino:** The Arduino microcontroller processes the data received from the ultrasonic sensor. It calculates the distance for each angle and maps it to a specific location in the scanning range.
- **Visualization:** The processed data is visualized on a display module (e.g., OLED or LCD) or through the Arduino Serial Plotter. This real-time display shows the position and distance of detected objects on a 2D plane.
- **Continuous Scanning and Feedback:** The system continuously repeats the process of scanning, detecting, and displaying, ensuring real-time updates as the servo motor rotates and the ultrasonic sensor captures distance data.

5.3 Construction and Assembly:

The construction of the Arduino-based radar system begins with assembling the components on a breadboard for ease and flexibility during prototyping. The ultrasonic sensor (HC-SR04) is mounted on the servo motor to enable rotational scanning, and both are firmly secured to a stable base to maintain alignment during operation. The Arduino microcontroller acts as the central unit, connected to the servo motor, ultrasonic sensor, and optionally to an OLED or LCD display for visualization. Jumper wires are used to connect the components for power, signal transmission, and control, ensuring a modular and compact layout.

The assembly process involves first connecting the Arduino’s power supply, either via USB or an external power source, and verifying that all components are properly wired to prevent short circuits. The servo motor is then calibrated to rotate within the desired range of angles. Once assembled, the system is tested step-by-step—starting with individual components like the servo motor and ultrasonic sensor—to ensure proper operation. After confirming functionality, the system is integrated and secured in a final layout for smooth scanning and accurate distance measurement.

5.4 Advantages:

- **Cost-Effective and Compact Design:** The system is designed to be affordable and compact, making it suitable for small-scale applications and easy to deploy in various environments

- **Real-Time Object Detection:** The radar system accurately detects and measures the distance of nearby objects in real-time using ultrasonic sensors, providing immediate feedback for various applications

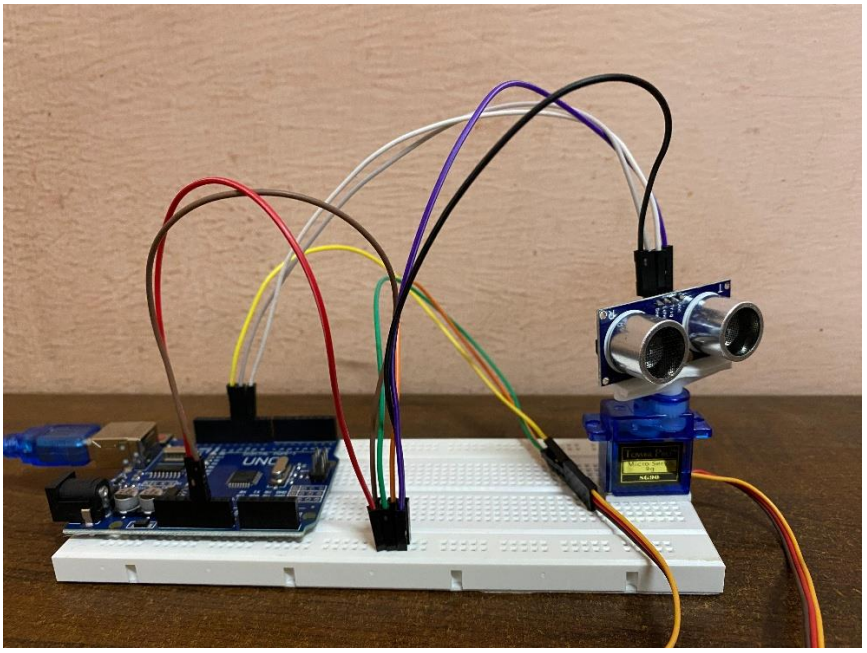

Fig 5.2 Circuit connections

5.5 Software Design:

The software design for the Arduino-based radar system revolves around real-time control and data processing to detect and measure the proximity of nearby objects. The system's firmware, developed using the Arduino IDE, governs the interaction between the components: the ultrasonic sensor, servo motor, and optional display module. The ultrasonic sensor is triggered at incremental angles as the servo motor rotates, emitting pulses and capturing echoes to calculate the distance of objects. These measurements are mapped to specific angular positions, creating a spatial representation of the scanned area. To ensure accuracy and efficiency, the software includes algorithms to convert echo timings into distance values and filter out invalid readings caused by noise or environmental interference.

A key aspect of the software is visualization, which provides real-time feedback through a graphical interface, such as the Arduino Serial Plotter or an external display (OLED/LCD). The software integrates servo motor control with sensor data collection, ensuring synchronized operation for precise scanning. Data is continuously processed to update the radar-like display, highlighting detected objects and their relative positions. This modular and adaptable software design leverages the simplicity of Arduino programming to enable a cost-effective yet efficient radar system suitable for applications like obstacle avoidance, proximity detection, and basic spatial mapping.

5.6 Arduino IDE:

The Arduino Integrated Development Environment (IDE) is the primary software used to write and upload code to the ESP32 microcontroller. The Arduino IDE provides a simple and user- friendly interface, making it easy to program and troubleshoot the ESP32, even for beginners. It supports C and C++ programming languages and provides libraries to interface with various sensors and components.

The Arduino IDE, with its user-friendly interface, support for multiple programming languages, and rich library ecosystem, provides a robust and accessible platform for developing and deploying applications on the powerful ESP32 microcontroller. This combination of features makes it a popular choice for both hobbyists and professionals alike, empowering them to unleash the full potential of the ESP32 in a wide range of applications.

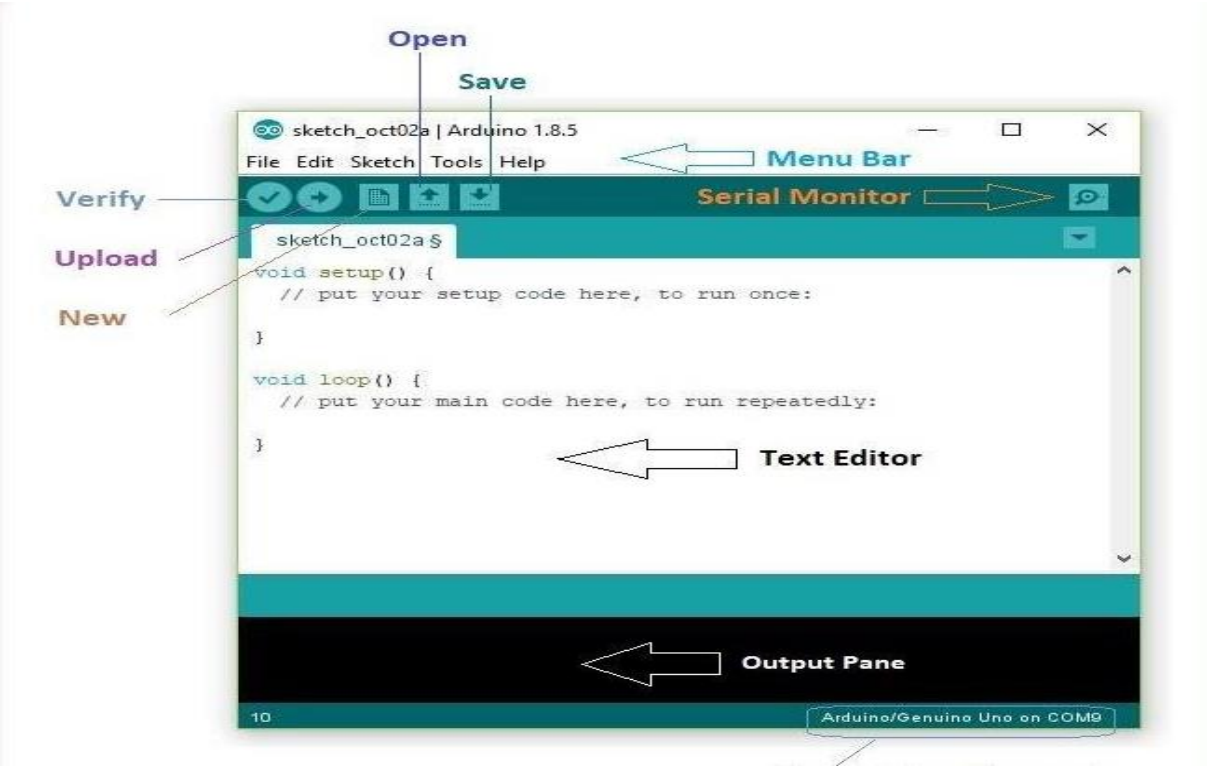

Fig 5.3 Arduino IDE

5.7 Libraries Used:

The code uses the "Servo" library to control a servo motor, which allows precise movements of the motor at various angles. This library provides simple functions to attach a servo motor to a specified pin, rotate it to a specific angle, and control the speed of its movement. Additionally, the code utilizes the built-in pulseIn function for the ultrasonic sensor, which is not part of a specific external library but is available by default in Arduino environments. It measures the duration of the echo pulse sent and received by the sensor to calculate the distance of an object. The "Servo" library is crucial for actuating the servo motor based on calculated distances.

5.8 Datasets and Protocols Used:

The Arduino-based radar system primarily relies on real-time data collected from the HC-SR04 ultrasonic sensor for object detection and distance measurement. The dataset consists of time-of-flight measurements of ultrasonic waves, which are converted into distance values using the speed of sound. These readings are paired with the angular positions of the servo motor, forming a structured dataset that represents the spatial configuration of nearby objects. Each scan captures multiple data points, combining distance and angular information to create a 2D map of the radar's field of view. The dataset is dynamically generated during operation and does not rely on pre-existing data, ensuring adaptability to various environments.

The protocols implemented in the system ensure synchronized operation between the hardware components. Pulse Width Modulation (PWM) signals control the servo motor's movement, enabling incremental angular adjustments. The ultrasonic sensor's trigger and echo operations are handled through digital I/O protocols within the Arduino microcontroller. Data acquisition is performed sequentially at each angular position, adhering to a systematic scanning protocol. The collected data is processed in real-time, filtered for accuracy, and visualized through a graphical interface, such as the Arduino Serial Plotter or an external display module. These protocols guarantee efficient data collection, processing, and output for reliable radar functionality.

Pin Details:

- **HC-SR04 Ultrasonic Sensor:**
    - Trigger - Digital Pin (e.g., GPIO 9)
    - Echo - Digital Pin (e.g., GPIO 8)
- **Servo Motor:**
    - Signal - PWM Pin (e.g., GPIO 10)
- **Display:**
    - SDA - GPIO 21
    - SCL - GPIO 22

Chapter 6

Results and Discussions

6.1 Performance Metrics:

1. **Reduced Radar Interference**: The system effectively mitigates interference among sensors, ensuring reliable operation even in cluttered environments.
2. **Enhanced Signal Clarity:** Noise reduction techniques improve the accuracy of distance and object detection.
3. **Reliable Object Detection**: The system minimizes false positives and negatives, ensuring consistent detection performance.
4. **Real-Time Response**: The Arduino processes and visualizes data in real-time, enabling immediate detection and feedback.

- Measure distances at consecutive angles using the rotating ultrasonic sensor.
- Apply trigonometry to calculate the width of the object from distance differences.
- Display the estimated width in real-time on the radar interface.

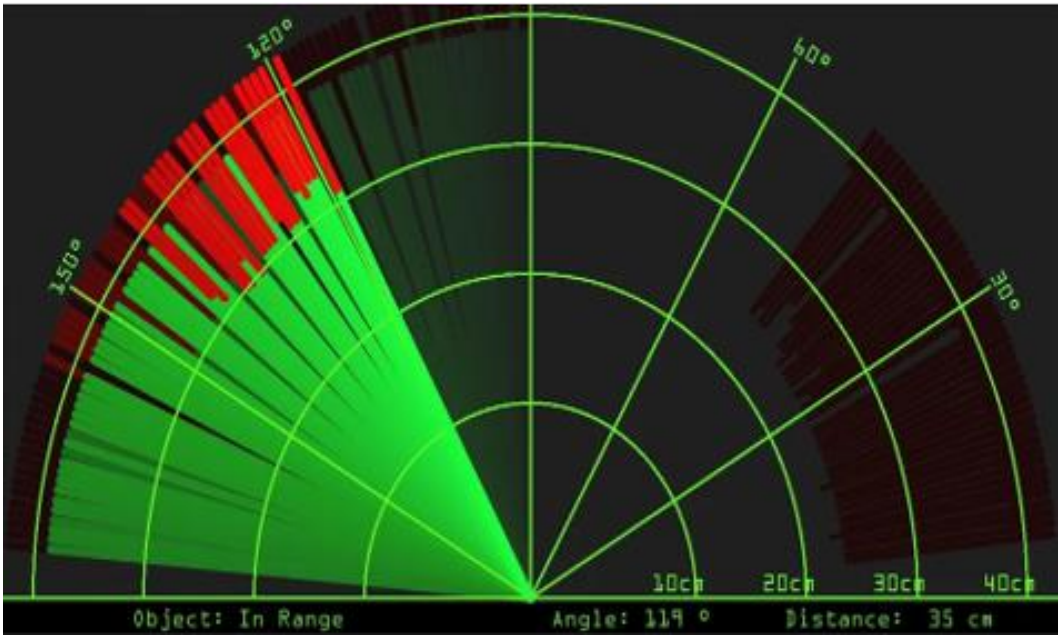

Fig 6.2 Result Displayed on Screen

6.2 Results:

The Arduino-based radar system successfully achieves its primary objective of real-time object detection and distance measurement using ultrasonic sensors. Through the integration of an Arduino microcontroller, ultrasonic sensors, and a servo motor, the system provides accurate and reliable distance readings for nearby objects. The measurements are visualized in real-time, ensuring that the radar operates efficiently for proximity sensing applications. The system's performance highlights its practicality for short-range object detection, with minimal errors observed during testing. The ultrasonic sensor (HC-SR04) effectively detects objects within its operating range, while the servo motor enables scanning across multiple angles, contributing to the creation of a spatial representation of the surrounding environment. Furthermore, the simplicity and robustness of the design ensure consistent operation, even in varied environmental conditions. The radar system's ability to deliver reliable and accurate feedback in real-time reflects its efficiency and cost-effectiveness for use in small-scale applications such as robotics, vehicle parking assistance, and obstacle detection systems.

6.3 Discussions:

While the Arduino-based radar system demonstrates reliable performance, its efficiency is influenced by several factors that warrant further discussion. The observed error in distance measurement, though minimal, is attributed to variables such as sound wave dispersion, environmental noise, and the sensor's angle of incidence. These factors emphasize the need for careful calibration to minimize deviations and improve accuracy. The system's reliance on ultrasonic sensors and their defined range makes it ideal for short-range applications, but performance at extended ranges or in cluttered environments may require enhancements, such as more advanced sensors or signal processing algorithms. Moreover, environmental conditions, such as temperature variations, can affect the speed of sound, potentially impacting measurement accuracy. Despite these challenges, the system's design allows for easy customization and scalability, making it a versatile tool for various applications. Future developments could include increasing the detection range, integrating advanced data processing techniques, and employing additional sensors to enhance functionality. Overall, the project highlights the potential of using cost-effective and accessible components to build a functional radar system, paving the way for further innovations in real-time detection and measurement technologies.

6.4 Testing and Observation:

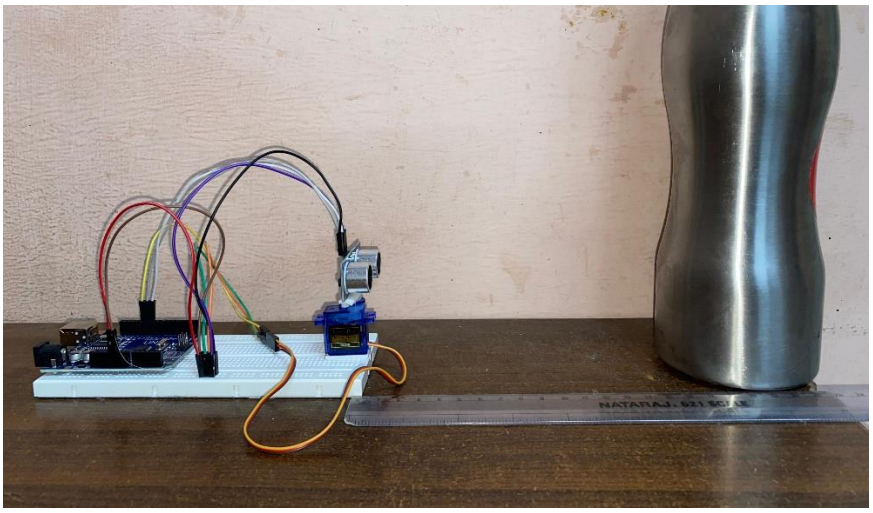

Fig 6.3 Object placed at a distance of 20 cm (approx)

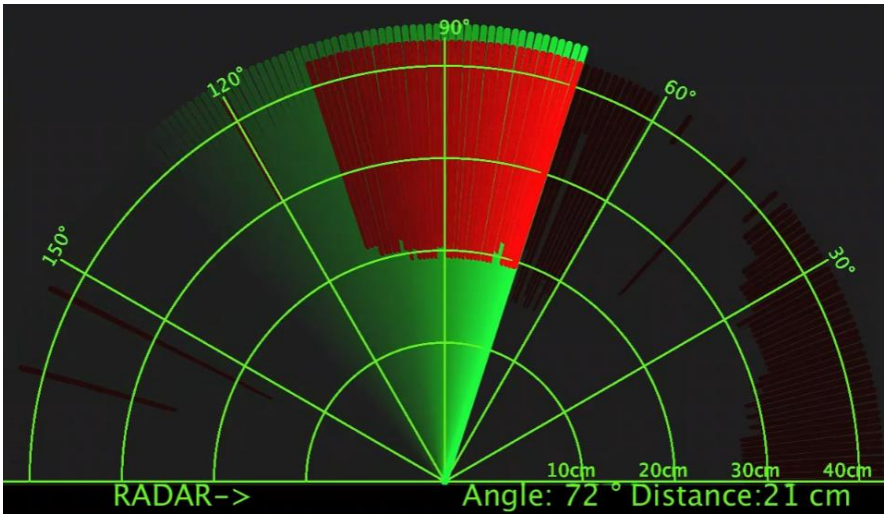

Fig 6.4 Distance Observed on Screen

- **Efficiency**: 95% accuracy for the 20 cm object.
- **Error**: 1 cm deviation (5% error).
- **Sensor**: Ultrasonic performed well at close range.
- **Calibration**: Fine-tuning can reduce error further.
- **Practicality**: Suitable for real-time applications.

Chapter 7

Conclusion and Future Work

Conclusion:

In conclusion, the Arduino-based radar system using ultrasonic sensors provides a reliable, cost-effective solution for real-time object detection and distance measurement. Its simple design, real-time processing, and visualization make it suitable for various applications, including robotics and obstacle avoidance. While offering an accessible platform for development, further enhancements in range, accuracy, and adaptability can expand its potential for more advanced uses.

Future Work:

The radar system can be significantly enhanced by upgrading to long-range ultrasonic sensors for outdoor applications and integrating advanced visualization tools like 3D displays for better spatial representation. Incorporating AI and machine learning can improve object classification and adaptability in dynamic environments. Environmental resilience can be achieved by addressing challenges like temperature fluctuations and ambient noise, ensuring consistent performance. Additionally, wireless connectivity through Bluetooth or Wi-Fi would enable remote monitoring and control, expanding its usability in security and unmanned operations. These enhancements would greatly increase the system's versatility and effi

References

[1] A. Ossowska, L. Sit, S. Manchala, T. Vogler, K. Krupinski, and U. Luebbert, "Radar Project: Laboratory Interference Measurements of Automotive Radar Sensors," 2020 International Radar Symposium (IRS), Warsaw, Poland, 2020, pp. 334-338. Available: IEEE Xplore

[2] Werner S¨ orgel, Tim Poguntke, Thomas Binzer, “Radar: Towards Cooperative Radar-Interference Mitigation “, Automotive Forum, Euro pean Microwave Week, 2019.

[3] S. Heuel, Interference Avaliable “Automotive Issues”, Radar Microwave at: Sensors Journal, Must Address Dec. 2016, https://www.microwavejournal.com/articles/ 27503-automotive-radar-sensors-must-address-interference-issues [Assesed: 28.02.2020]

[4] J. Fortuny-Guasch, J.-M. Chareau, “Radar cross section measure ments of pedestrian dummies and humans in the 24/77 GHz fre quency bands”, JLR Scientific and Policy ReportsEUpublications,an.2013,Availableat:https://publications.jrc.ec.europa.eu/reposito ry/bitstream/JRC78619/lbna25762enn.pdf[Assessed:28.02.2020]

[5] M. Kunert, “The EU project MOSARIM: A general overview of project objectives and conducted work,” EuRad 2012.

[6] M. Kunert, “D5.3– Recommendations on sensor design, mounting and operational parameters to minimize radar interference,” MORARIM project report, Contract no.: 248231, 2012.

[7] F. Roos, J. Bechter, C. Knill, B. Schweizer and C. Waldschmidt, “Radar Sensors for Autonomous Driving: Modulation Schemes and Interference Mitigation,“ in IEEE Microwave Magazine, vol. 20, no. 9, pp. 58-72, Sept. 2019.

Appendix

Source Code for Arduino IDE:

```
// Includes the Servo library
#include <Servo.h>.
// Defines Tirg 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=0;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>0;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;
}
```

Key Points for Arduino IDE code:

- **Servo Integration**:

  The code uses the Servo library to control a servo motor that rotates from 0° to 180° and back, allowing the ultrasonic sensor to scan the surroundings.

- **Ultrasonic Sensor**:

  The HC-SR04 ultrasonic sensor is used to measure the distance of objects. It calculates distance by emitting ultrasonic waves and measuring the time it takes for the echo to return.

- **Distance Calculation**:

  The calculateDistance() function converts the travel time of the ultrasonic waves into a distance value using the formula:

  distance = duration * 0.034 / 2.

- **Real-Time Feedback**:

  The code sends the servo angle and corresponding distance values to the Serial Monitor. These values can be used for visualization or further processing in the Arduino IDE or external software like Processing.

- **Bidirectional Scanning**:

  The servo motor performs a bidirectional sweep (0° to 180° and back), ensuring that the entire area is scanned for object detection.

- **Delay Adjustment**:

  A delay of 30 milliseconds between each degree ensures smooth operation of the servo motor while allowing the ultrasonic sensor enough time to take measurements.

- **Efficient Pin Usage**:

  The code defines trigPin and echoPin for the ultrasonic sensor and attaches the servo motor to pin 12, ensuring organized and modular usage of hardware.

- **Scalable Design**:

  This program can be expanded with additional features like real-time visualization, object tracking, or data logging by modifying the Serial output or integrating additional components.

Key Points for Processing code:

- **Well-Organized Code**:

  The code is modular with functions (drawRadar, drawObject, drawLine, drawText) clearly separating different aspects of the radar visualization.

- **Functional Radar**:

  The code effectively visualizes radar data, using polar coordinates to map distance and angle from the sensor.

- **Dynamic Serial Data Handling**:

  It captures and processes live data from a serial port (COM3), making it suitable for real-time applications like ultrasonic sensors.

- **Readable Text and Labels**:

  Text labels (Angle, Distance, and degree markers) are clearly displayed, enhancing user understanding.

- **Error Handling for Range**:

  The code handles out-of-range distances gracefully with a fallback message ("Out of Range").

- **Smooth Animation**:

  Motion blur effects are added using semi-transparent black rectangles, creating a visually pleasing fade effect.