SMART RADAR USING IOT DEVICES

ABSTRACT:-

The " **SMART RADAR USING IOT DEVICES** " project is aimed at enhanced surveillance applications. The proposed system utilizes ultrasonic sensors to detect the presence of objects within a specified range. Arduino microcontroller processes the sensor data and controls the radar scanning mechanism. Integration with IoT technology enables real-time data transmission and remote monitoring capabilities. The system provides an affordable and versatile solution for surveillance needs in various environments. Experimental results demonstrate the effectiveness and feasibility of the proposed approach, highlighting its potential for deployment in security, automotive safety, and environmental monitoring applications..

INTRODUCTION:-

The " **SMART RADAR USING IOT DEVICES** " project presents an innovative approach to enhanced surveillance applications. this paper introduces a Smart Radar System that leverages the capabilities of Arduino microcontrollers, ultrasonic sensors, and IoT technology to enhance surveillance capabilities. By combining these technologies, the system offers real-time detection, monitoring, and remote access functionalities, making it suitable for a wide range of applications including security, automotive safety, and environmental monitoring

Ultrasonic radar, also known as ultrasonic sensors or sonar systems, is a technology that utilizes ultrasonic waves to detect and locate objects in its proximity. It is an evolution of radar technology that operates at frequencies beyond the range of human hearing, typically above 20 kHz. The principle behind ultrasonic radar is based on the time-of-flight concept. When an ultrasonic wave is emitted, it travels through the air, reflects off an object, and returns to the sensor. By measuring the time it takes for the wave to make the round trip, the distance to the object can be calculated using the speed of sound in the medium.

ultrasonic radar technology offers a versatile and practical approach to object detection and localization, enabling a range of applications that require accurate and efficient sensing in various industries

OBJECTIVE:-

The objective of this paper is to design, implement, and evaluate a Smart Radar System that integrates Arduino microcontrollers, ultrasonic sensors, and IoT technology for enhanced surveillance and monitoring applications. Specifically, the goals are to:

- Develop a cost-effective surveillance solution by leveraging readily available hardware components such as Arduino microcontrollers and ultrasonic sensors.
- Integrate IoT technology to enable real-time data transmission and remote monitoring capabilities, enhancing the system's versatility and accessibility.

- ❖ Demonstrate the feasibility and effectiveness of the proposed system through experimental evaluation, highlighting its potential applications in security, automotive safety, and environmental monitoring.
- ❖ Provide insights into the design considerations, implementation challenges, and performance characteristics of the Smart Radar System to guide future research and development efforts in this domain.

EXISTING SYSTEM:

The existing surveillance systems often rely on conventional radar technology or standalone sensor-based systems for detection and monitoring purposes. These systems typically have limited connectivity and lack real-time data transmission capabilities. Some may utilize basic microcontrollers or single-board computers for data processing, but they may lack integration with IoT technology for remote monitoring and control.

- **Industrial Automation:** Ultrasonic radar is utilized in industrial automation settings for applications such as object detection, material handling, and quality control.
- **Security and Surveillance:** Ultrasonic radar systems have been utilized for security and surveillance purposes. They can detect and track movements in a designated area, providing valuable information for monitoring and intrusion detection systems.

Overall, the existing systems provide adequate surveillance functionalities but lack the integration with IoT technology necessary for enhanced connectivity, real-time monitoring, and remote access.

PROPOSED SYSTEM:-

1. System Architecture:-

The proposed Smart Radar System integrates Arduino microcontrollers, ultrasonic sensors, and IoT technology to create a cost-effective and versatile surveillance solution.

- **Radar Scanning Mechanism:** The radar scanning mechanism, controlled by the Arduino microcontroller, enables the system to sweep the monitored area and detect objects within the sensor's range. This scanning process provides comprehensive coverage of the surveillance area.
- **Data Transmission and Analysis:** Sensor data collected by the Arduino microcontroller is transmitted to a remote server or cloud platform via the IoT module. Real-time data transmission enables continuous monitoring of the surveillance area, while data analysis algorithms can be implemented to identify and track objects of interest.

the proposed Smart Radar System offers a cost-effective, scalable, and connected surveillance solution with enhanced capabilities for real-time monitoring, remote access, and data analysis, making it suitable for various applications including security, automotive safety, and environmental monitoring.

2. Components:-

- **BUZZER :**
  A buzzer is integrated into the Smart Radar System to provide audible alerts when objects are detected within the surveillance area, enhancing situational awareness. Its activation serves as an additional notification mechanism for users, ensuring prompt attention to potential security breaches or safety hazards.

- **ARDUINO :**
  The Arduino microcontroller serves as the central control unit in the Smart Radar System, coordinating the operation of ultrasonic sensors, radar scanning, and IoT connectivity. Its versatility and programmability enable seamless integration of hardware components and implementation of custom algorithms for enhanced surveillance capabilities.

- **ULTRASONIC SENSOR :**
  Positioned around the periphery of the system.Ultrasonic sensors in the Smart Radar System emit high-frequency sound waves and detect their reflections to accurately measure distances of objects within the surveillance area, providing essential data for real-time monitoring and object detection. Their reliable performance and wide detection range contribute to the system's effectiveness in surveillance applications, ensuring comprehensive coverage and timely detection of potential threats or obstacles.

- **SERVO MOTORS :**
  Servo motors in the Smart Radar System facilitate precise control of the radar scanning mechanism, enabling efficient coverage of the surveillance area. Their ability to rotate at specific angles ensures accurate detection and tracking of objects, enhancing the system's effectiveness in monitoring and surveillance tasks.

How the System Works :-

The ultrasonic sensor rotates with the servo motor and transmitted the ultrasonic waves during this time. and the whole time a graph interface make in the simulation software. and if any object comes under the range of the ultrasonic sensor it starts to detect the object. at that time the graph interface inside the software becomes red in the object area. ultrasonic sensor work as an object detector in this project. Radar using ultrasonic sensor works in software makes the reaction according to the waves received.ultrasonic sensors have two terminals one is a transmitter and another is the receiver. The transmitter terminal is known as the Trigger and the receiver terminal is known as the echo. Arduino continuously gives a command to the Servo motor to rotate. and the transmitter transmits the signal parallelly likewise the software also makes the graph. The ultrasonic sensor gave a different signal to the Arduino if anything comes in the path. then Arduino notifies the software for the affected region. the project depends on the ultrasonic sensor working. Radar using Arduino, ultrasonic sensor, and servo motor contents no other major components.

- **Sensing:** The radar sensor emits electromagnetic waves, usually radio waves, into the surrounding environment. These waves bounce off objects in their path and return to the sensor.

- **IoT Integration:** The processed data is then transmitted to an IoT device, such as a microcontroller or a Raspberry Pi, using wireless communication protocols like Wi-Fi or Bluetooth.

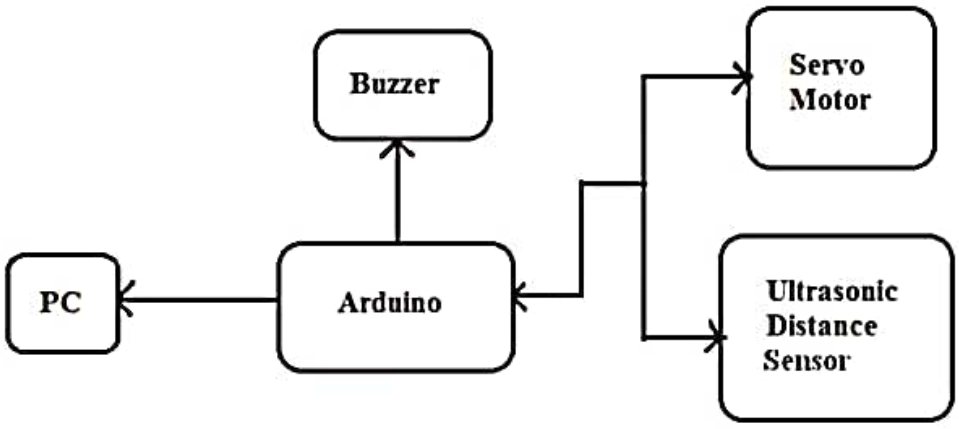
- **Data Processing:** The sensor processes the raw data collected from the radar waves. This may involve filtering out noise, extracting relevant information about detected objects, and organizing the data for further analysis.

- **Analysis and Action:** In the cloud, advanced algorithms can analyze the data to identify patterns, detect anomalies, and make decisions based on predefined rules or machine learning models. For example, the system can detect moving objects in a certain area and trigger alerts or actions accordingly, such as notifying users or controlling other IoT devices

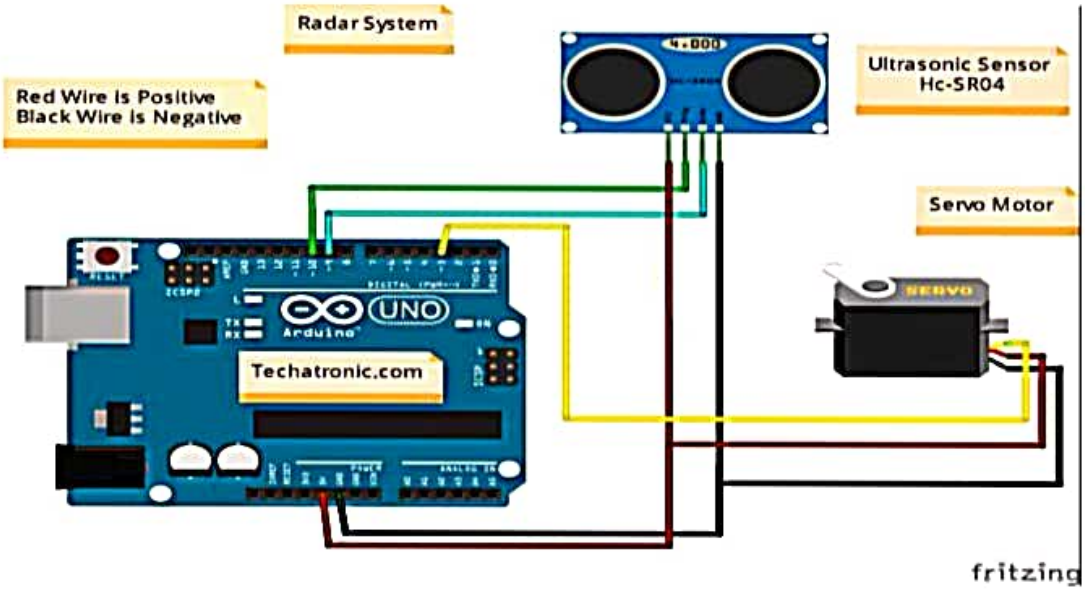
- **Feedback Loop:** The system may also incorporate feedback mechanisms to continuously improve its performance. For instance, it can learn from past detections to better distinguish between different types of objects or optimize its operation based on environmental changes.

A smart radar system using IoT devices combines radar sensing technology with IoT connectivity and cloud-based processing to enable real-time monitoring, analysis, and automation of various applications, such as traffic management, security surveillance, and industrial automation.

BLOCK DIAGRAM:-



CIRCUIT DIAGRAM:-



WORKING MODEL:-

i. **Radar Sensor Setup:** Begin by setting up the radar sensor, such as a microwave Doppler radar module or an ultrasonic sensor, with the Arduino board. This sensor will detect objects within its range and provide distance or motion information to the Arduino.

ii. **Arduino and Microcontroller Integration:** Connect the Arduino board to a microcontroller, such as an ESP8266 or ESP32, which will handle the IoT connectivity aspect. Use appropriate communication protocols like Wi-Fi or Bluetooth to enable data transmission between the Arduino and the microcontroller.

iii. **Processing and IoT Integration:** Write a program on the microcontroller to process the data received from the radar sensor. This program should extract relevant information about detected objects, such as their distance or speed, and transmit this data to an IoT platform or cloud service using protocols like MQTT or HTTP.

iv. **Cloud Connectivity:** Set up an IoT platform or cloud service, such as ThingSpeak or AWS IoT, to receive and store the data transmitted by the microcontroller. This platform will act as a central hub for collecting and analyzing the radar data.

v. **Alert Mechanism with Buzzers:** Configure the microcontroller to trigger a buzzer alarm whenever an object is detected within a predefined distance or if there's a significant change in motion. This buzzer will serve as an alert mechanism to notify users about potential hazards or intrusions.

vi. **Servo Motor Control:** Integrate a servo motor with the Arduino to enable directional sensing. Use the servo motor to rotate the radar sensor in different directions, expanding its coverage area and enhancing detection capabilities.

vii. **User Interface:** Develop a user interface, such as a mobile app or a web dashboard, to visualize the radar data in real-time. This interface can display information about detected objects, their distances, and any alarm notifications triggered by the system.

viii. **Feedback Mechanism**: Implement a feedback mechanism to allow users to interact with the system and adjust parameters such as detection sensitivity or rotation angle of the servo motor.

ix. **Testing and Calibration:** Test the entire system in different scenarios to ensure its reliability and accuracy. Calibrate the radar sensor, microcontroller, and servo motor settings as needed to optimize performance.

x. **Deployment and Monitoring:** Once the system is thoroughly tested, deploy it in the desired environment, such as a home security system, a parking lot monitoring system, or an industrial automation setup. Monitor the system regularly to ensure smooth operation and address any issues that may arise.

ARDUINO CODE :

```
#include <Servo.h>
const int trigPin = 10;
const int echoPin = 9;
const int buzzerPin = 8; // Define buzzer pin
long duration;
int distance;
Servo myServo;
void setup() {
 pinMode(trigPin, OUTPUT);
 pinMode(echoPin, INPUT);
 pinMode(buzzerPin, OUTPUT); // Set buzzer pin as output
 Serial.begin(9600);
 myServo.attach(3);
}
void loop() {
 for(int i=15;i<=165;i++){
  myServo.write(i);
  delay(30);
  distance = calculateDistance();
  Serial.print(i);
  Serial.print(",");
  Serial.print(distance);
  Serial.print(".");

  // Check if distance is less than a threshold (e.g., 20 cm) to trigger buzzer
  if(distance < 20) {
   digitalWrite(buzzerPin, HIGH); // Turn on buzzer
  } else {
   digitalWrite(buzzerPin, LOW); // Turn off buzzer
  }
 }
  for(int i=165;i>15;i--){
  myServo.write(i);
  delay(30);
  distance = calculateDistance();
  Serial.print(i);
  Serial.print(",");
  Serial.print(distance);
  Serial.print(".");

  if(distance < 20) {
   digitalWrite(buzzerPin, HIGH);
  } else {
   digitalWrite(buzzerPin, LOW);
  }
 }
}
int calculateDistance(){
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPin, HIGH);
```

```
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH);
 distance= duration*0.034/2;
 return distance;
}
```

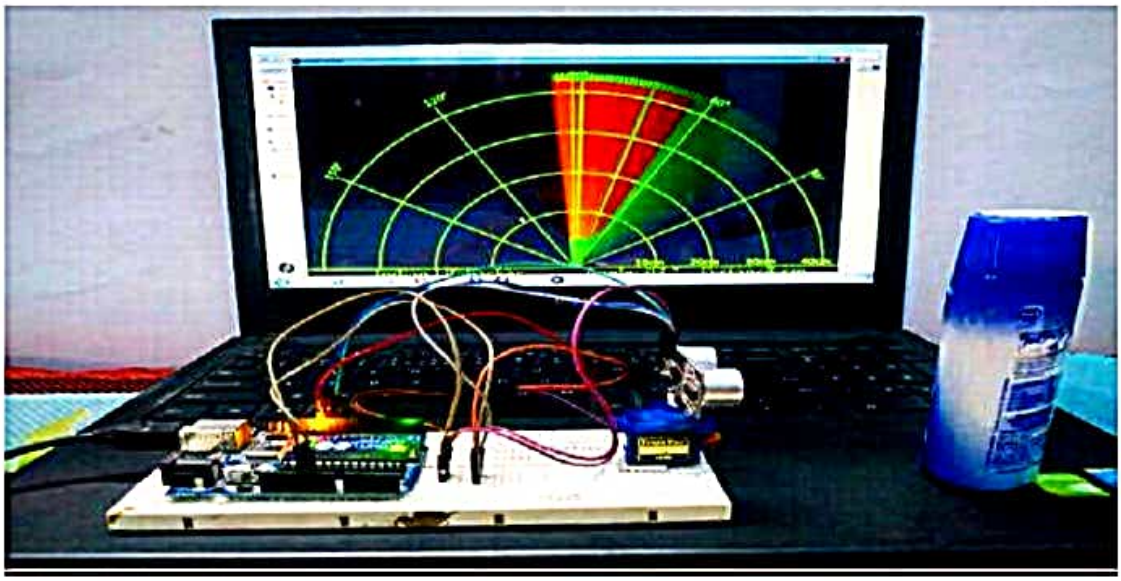
PROGRAM CODE :

```
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
 // defubes 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, 700); // **CHANGE THIS TO YOUR SCREEN RESOLUTION**
  smooth();
  myPort = new Serial(this,"COM3", 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)*sin(radians(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("Indian Lifehacker ", 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*0.0907)-
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*0.0888)-
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*0.0833)-
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*0.07129)-
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();
  }
```

OUTPUT:-



ADVANTAGES :

1. **High Accuracy:** Smart radar systems provide accurate detection and tracking of objects in real-time, enabling precise monitoring and control in various scenarios.

2. **Wide Coverage:** Radar technology offers wide coverage areas, making it suitable for monitoring large spaces such as parking lots, highways, and industrial facilities.

3. **All-Weather Performance**: Radars can operate effectively in adverse weather conditions like rain, fog, or snow, where other sensing technologies might struggle, ensuring continuous monitoring and reliable performance.

4. **Non-Intrusive:** Radar operates without the need for physical contact or line-of-sight, making it ideal for applications where direct access to the monitored area is restricted or impractical.

5. **Low Power Consumption:** Many modern radar sensors are designed to operate efficiently with low power consumption, making them suitable for battery-powered IoT devices and remote installations.

6. **Scalability:**Smart radar systems can scale to accommodate different deployment sizes and requirements, from small-scale installations to large-scale networks covering vast areas.

APPLICATIONS:

1. **Traffic Management:** Smart radar systems can monitor vehicle flow, detect traffic congestion, and optimize traffic signals to improve road safety and reduce congestion.

2. **Security Surveillance:** Radar-based IoT devices can detect intruders, monitor perimeter security, and provide early warning of unauthorized access to restricted areas.

3. **Environmental Monitoring:** Radar sensors can track weather patterns, monitor water levels, and detect natural disasters like floods or landslides, aiding in disaster management and environmental protection efforts.

4. **Industrial Automation:** Radar technology can be used for monitoring equipment health, detecting machinery malfunctions, and optimizing manufacturing processes in industries such as mining, agriculture, and construction.

5. **Smart Agriculture:** Radar sensors can monitor crop health, measure soil moisture levels, and detect pests or wildlife intrusion, enabling precision agriculture and maximizing crop yields.

6. **Maritime Navigation:** Radar sensors on IoT-enabled buoys or vessels can aid in maritime navigation, collision avoidance, and monitoring marine traffic for improved safety at sea.

7. **Healthcare Monitoring:** Radar-based IoT devices can monitor patient vital signs, detect falls or emergencies, and track movement patterns for elderly care and healthcare applications.

8. **Wildlife Conservation:** Radar technology can monitor wildlife movements, track animal populations, and detect illegal poaching activities in protected areas, supporting wildlife conservation efforts and biodiversity conservation

FUTURE ENHANCEMENT OR SCOPE:-

- **Technological Advancements:** Continued advancements in radar technology, IoT connectivity, and data analytics will lead to more sophisticated smart radar systems with enhanced capabilities and performance.

- **Integration with AI and Machine Learning:** Integrating radar data with AI and machine learning algorithms will enable predictive analytics, anomaly detection, and autonomous decision-making for more intelligent and proactive monitoring and control systems.

- **Miniaturization and Portability:** Ongoing miniaturization efforts will make radar sensors more compact, lightweight, and portable, opening up new possibilities for mobile applications and wearable devices.

- **5G Integration:** Integration with 5G networks will enable high-speed data transmission, low-latency communication, and massive connectivity, unlocking new use cases and applications for smart radar systems in diverse industries.

- **Privacy and Security:** Addressing concerns related to data privacy, cybersecurity, and ethical use of radar technology in IoT deployments will be crucial to fostering trust and acceptance among users and stakeholders.

- **Global Adoption:** Increasing awareness of the benefits of smart radar technology combined with growing investments in IoT infrastructure will drive widespread adoption across various sectors and regions, creating new opportunities for innovation and economic growth.

CONCLUSION:-

The smart radar detection utilizing IoT devices offers a plethora of advantages, including high accuracy, wide coverage, and cost-effectiveness, driving its applications across diverse fields such as traffic management, security surveillance, and environmental monitoring. With ongoing technological advancements, integration with AI, and global adoption, the scope for smart radar systems is poised for further expansion, promising more efficient, intelligent, and interconnected solutions for various real-world challenges.

REFERENCES:-

Arduino Official Website:
https://www.arduino.cc/ - Provides documentation, tutorials, and resources for Arduino microcontroller boards and programming.

Arduino Project Hub:
https://create.arduino.cc/projecthub - A platform where users share their Arduino projects, including smart radar detection .

IEEE Xplore Digital Library:
https://ieeexplore.ieee.org/ - Access research papers and articles related to smart radar detection techniques
https://techatronic.com/radar-using-arduino-ultrasonic-sensor/. Radar using Arduino and Ultrasonic sensor by mohd shahid.

OpenCV Documentation: The OpenCV documentation (https://docs.opencv.org/) offers detailed information on using OpenCV for image processing, including smart radar detection
Online Tutorials and Guides: Websites like Adafruit (https://learn.adafruit.com/) and SparkFun (https://learn.sparkfun.com/) offer tutorials and guides on various projects involving Arduino , microcontroller , raspberry pi etc