REAL-TIME OBJECT DETECTION

A PROJECT REPORT

Submitted by

PRASADH C (2116210701191)
KARTHICK RAJA K (2116210701505)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

ANNA UNIVERSITY, CHENNAI

MAY 2024

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Thesis titled **"REAL-TIME OBJECT DETECTION"** is the bonafide work of "**PRASADH C (210701191), KARTHICK RAJA K (210701505)**" who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Mr. . S. Gunasekar M.E.,

SUPERVISOR

Assistant Professor

Department of Computer Science and Engineering

Rajalakshmi Engineering College

Chennai - 602 105

Submitted to Project Viva-Voce Examination held on___________________

Internal Examiner **External Examiner**

ABSTRACT

This project presents the design and implementation of a real-time object detection system using an Arduino microcontroller, an ultrasonic sensor, and a servo motor. The system employs the HC-SR04 ultrasonic sensor to measure distances by emitting ultrasonic pulses and calculating the time it takes for the echoes to return. These distance measurements are processed by the Arduino, which subsequently controls a servo motor to adjust its position based on the detected object's proximity. When an object is detected within a predefined range, the servo motor is programmed to move to a specific angle, providing a visual and mechanical response to the detection. This setup can be utilized in various applications, such as obstacle avoidance in robotics, automated distance measurement tools, and smart automation systems. The integration of these components showcases the potential of simple electronic devices and programming to create effective real-time detection systems. The project not only highlights the feasibility of using readily available components for practical applications but also provides a foundation for future enhancements, such as incorporating multiple sensors for improved accuracy, integrating displays for real-time data visualization, and developing more sophisticated .

ACKNOWLEDGMENT

First, we thank the almighty god for the successful completion of the project. Our sincere thanks to our chairman **Mr. S. Meganathan B.E., F.I.E.,** for his sincere endeavor in educating us in his premier institution. We would like to express our deep gratitude to our beloved Chairperson **Dr. Thangam Meganathan Ph.D.,** for her enthusiastic motivation which inspired us a lot in completing this project and Vice Chairman **Mr. Abhay Shankar Meganathan B.E., M.S.,** for providing us with the requisite infrastructure.

We also express our sincere gratitude to our college Principal, **Dr. S. N. Murugesan M.E., PhD.,** and **Dr. P. KUMAR M.E., PhD, Director computing and information science , and Head Of Department of Computer Science and Engineering** and our project coordinator **Mr. S.Gunasekar M.E..,** for her encouragement and guiding us throughout the project towards successful completion of this project and to our parents, friends, all faculty members and supporting staffs for their direct and indirect involvement in successful completion of the project for their encouragement and support.

PRASADH C

KARTHICK RAJA K

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The capability to detect objects in real time is crucial for numerous applications across various domains, including robotics, automotive systems, industrial automation, and consumer electronics. Real-time object detection systems are designed to identify and respond to the presence of objects within a given environment promptly. These systems are integral to the development of autonomous vehicles, security systems, smart home devices, and many other technologies that require dynamic interaction with their surroundings.

This project focuses on creating a simple yet effective real-time object detection system using an Arduino microcontroller, an ultrasonic sensor, and a servo motor. The Arduino, a popular and versatile platform for prototyping and electronics projects, serves as the central processing unit, interfacing with the ultrasonic sensor and the servo motor to achieve real-time object detection and response.

The ultrasonic sensor (HC-SR04) is employed to measure the distance to objects by emitting ultrasonic waves and calculating the time it takes for the waves to reflect back. This time-of-flight data is processed by the Arduino to determine the proximity of objects. Based on the detected distance, the Arduino controls the servo motor to adjust its position, thus providing a mechanical response to the presence of objects

The project aims to demonstrate the feasibility and effectiveness of using basic electronic components and straightforward programming techniques to develop a functional real-time object detection system. By leveraging the capabilities of the Arduino platform, this project also serves as a foundation for more advanced systems that can incorporate additional sensors, more complex algorithms, and enhanced functionalities.

1.1 Motivation

The motivation behind this project stems from the growing need for efficient and cost-effective real-time object detection systems in various fields such as robotics, automotive safety, industrial automation, and home automation. As technology continues to advance, the demand for intelligent systems capable of interacting dynamically with their environment has increased significantly. Object detection plays a pivotal role in enabling these systems to perform tasks autonomously, safely, and efficiently.

One of the primary motivations is the potential application of real-time object detection in enhancing the capabilities of robotic systems. In robotics, the ability to detect and respond to obstacles in real-time is crucial for navigation and task execution in dynamic environments. This project aims to provide a foundational understanding of how simple components like ultrasonic sensors and servo motors can be integrated to create a responsive system, making it an excellent starting point for more complex robotic applications.

Objectives:

The primary objectives of this real-time object detection project are to design and develop a functional system using an Arduino microcontroller, an ultrasonic sensor, and a servo motor, ensuring seamless integration of these components. The system aims to achieve accurate distance measurement and dynamic response by utilizing the HC-SR04 ultrasonic sensor to detect objects in real-time and control the servo motor accordingly. Practical applications such as obstacle avoidance and automation are demonstrated to showcase the system's potential in robotics and automated tasks. The project provides a hands-on learning experience, promoting skill development in electronics and programming, while being low-cost and accessible for educational purposes. Comprehensive testing and validation ensure reliable performance, with detailed documentation facilitating knowledge sharing and troubleshooting.

CHAPTER 2

LITRETURE SURVEY

The literature survey for this real-time object detection project spans across a diverse array of sources encompassing research papers, academic texts, and online resources. Redmon and Davila’s seminal work on "Real-time Object Detection" introduces YOLO (You Only Look Once), an influential algorithm renowned for its high-speed and precision in directly predicting bounding boxes and class probabilities from images.

Cook's comprehensive exploration in "Ultrasonic Sensor Applications and Fundamentals" provides crucial insights into the underlying principles and practical applications of ultrasonic sensors, laying a solid foundation for understanding the sensor technology utilized in this project. Blum and Bresnahan's instructional guide, "Arduino Programming in 24 Hours," equips enthusiasts with essential programming skills vital for interfacing sensors and actuators with Arduino, while Scherz and Monk's "Practical Electronics for Inventors" delves deep into electronics, facilitating a nuanced understanding of sensor integration and motor control mechanisms. McKinnon’s "Robotics:

Everything You Need to Know About Robotics" offers a panoramic view of robotics, underlining the pivotal role of real-time object detection across a spectrum of applications. Research papers such as Dangi and Vinayak's "Object Detection for Robotics using Arduino" furnish pragmatic insights into implementing object detection with Arduino, while Thakur and Kumar's "Advanced Object Detection Techniques for Robotics" explores cutting-edge methodologies, including deep learning, for object detection in robotics applications.

2.1 EXISTING SYSTEM:

Existing systems for real-time object detection encompass a spectrum of approaches, from traditional sensor-based setups like ultrasonic and infrared sensors commonly used in proximity sensing, to sophisticated computer vision-based systems such as OpenCV and YOLO (You Only Look Once) algorithms, capable of real-time detection and tracking using cameras and deep learning techniques. Additionally, robotics platforms like ROS (Robot Operating System) and Arduino-based solutions offer customizable frameworks for developing object detection systems tailored to specific applications, while commercial solutions such as smart cameras and industrial automation systems provide ready-made solutions with advanced capabilities for industrial and commercial use. Each of these systems presents unique advantages and trade-offs in terms of cost, complexity, accuracy, and suitability for different applications, highlighting the diverse landscape of existing approaches in the field of real-time object detection.

2.1.1 Advantages of the existing system :

- Real-time object detection systems offer instantaneous response, enhancing safety, efficiency, and automation across diverse applications.
- These systems leverage advanced algorithms and sensor technologies to provide accurate and versatile detection capabilities, driving innovation and research in fields such as computer vision and artificial intelligence.
- With cost-effective solutions available, real-time object detection is accessible to hobbyists, educational institutions, and businesses, offering enhanced user experiences and paving the way for further advancements in technology.

Disadvantages of the existing system :

- These include the complexity of implementation, high costs associated with advanced sensors and computational resources, potential issues with false positives/negatives affecting reliability, environmental factors impacting performance, privacy concerns regarding data handling, ongoing maintenance requirements, and integration challenges with existing systems.
- Despite these limitations, careful consideration of design choices and continuous refinement can help mitigate these drawbacks and maximize the effectiveness of real-time object detection systems in various applications.

2.2 PROPOSED SYSTEM

The proposed real-time object detection system aims to leverage a combination of ultrasonic sensors and servo motors controlled by an Arduino microcontroller to detect and respond to objects in its vicinity. The system will utilize ultrasonic sensors to measure distances to nearby objects, with the Arduino processing this data to determine the presence and proximity of objects in real-time. Upon detecting an object within a predefined range, the Arduino will command the servo motor to adjust its position accordingly, providing a visual and mechanical response. This system will be designed with simplicity, affordability, and accessibility in mind, making it suitable for educational purposes, hobbyist projects, and potentially practical applications such as obstacle avoidance in robotics or automated surveillance systems. Through careful design and testing, the proposed system aims to demonstrate the feasibility and effectiveness of using basic electronic components and programming techniques to develop a functional real-time object detection system.

2.2.1 Advantages of the proposed system :

- The proposed real-time object detection system offers a cost-effective and accessible solution for detecting and responding to objects in real-time, leveraging ultrasonic sensors, servo motors, and Arduino microcontrollers.
- Its simplicity in design and programming enables easy implementation, making it suitable for beginners and educational purposes.

2.2.2 Disadvantages of the proposed system:

While the proposed real-time object detection system presents cost-effective and accessible solutions for object detection, its limitations include a finite detection range and potential accuracy issues with ultrasonic sensors, susceptibility to environmental factors affecting reliability, constrained processing power and mechanical limitations of Arduino microcontrollers and servo motors, dependency on physical components posing maintenance challenges, scalability issues, and a lack of advanced features compared to more sophisticated systems. Despite these drawbacks, the system remains a valuable educational and practical tool for basic object detection tasks.

CHAPTER 3
SYSTEM DESIGN

3.1 DEVELOPMENT ENVIRONMENT

3.1.1 Hardware Requirements

- Arduino Uno: The heart of the system, Arduino Uno serves as the main microcontroller to process data from sensors, control the relay and motor, and manage the LCD display interface.
- Servo Motor: The servo motor is used to physically open and close the parking gate. It should have enough torque and precision to handle the weight and movement of the gate effectively.
- Ultrasonic Sensor: The ultrasonic sensor detects the presence of vehicles approaching the parking gate. It measures the distance between the sensor and the vehicle, triggering the gate to open when a vehicle is within the specified range.
- Breadboard: The breadboard provides a platform for connecting and prototyping the various electronic components of the system, including the Arduino Uno, servo motor, and ultrasonic sensor. It allows for easy wiring and experimentation during the development process.

3.1.2 Software Requirements

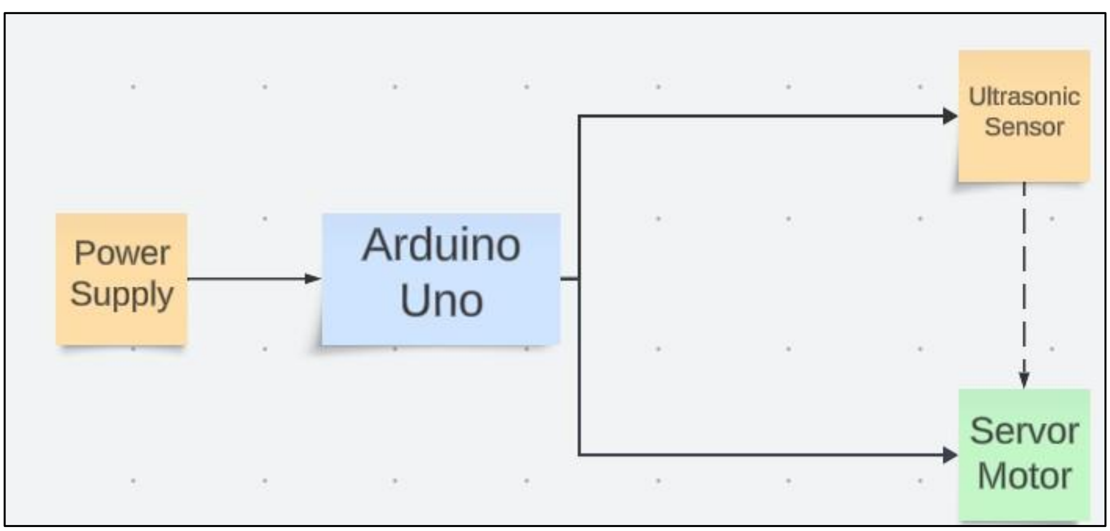
Arduino IDE: The Arduino Integrated Development Environment (IDE) is used for programming the Arduino Uno microcontroller, allowing users to write and upload code to control the smart plant watering system.

CHAPTER 4

PROJECT DESCRIPTION

The proposed project aims to develop a real-time object detection system using an Arduino microcontroller, ultrasonic sensors, and a servo motor. The system will utilize ultrasonic sensors to measure the distance to objects by emitting sound waves and calculating the time it takes for the waves to reflect back. The Arduino will process this data to determine the presence and proximity of objects in real-time. Based on the detected distance, the Arduino will control a servo motor to adjust its position, providing a mechanical response to the presence of objects. This setup can be applied to various practical applications, such as obstacle avoidance in robotics, automated door openers, and distance measurement tools. The project will focus on simplicity, affordability, and accessibility, making it suitable for educational purposes and hobbyist projects. Detailed documentation and testing will ensure the system's reliability and performance, demonstrating the feasibility of using basic electronic components and straightforward programming techniques to create an effective real-time object detection system..

4.1 SYSTEM ARCHITECTURE:



4.2 METHODOLOGY:

The methodology for developing the real-time object detection system involves several steps, beginning with the selection and acquisition of key components, including the HC-SR04 ultrasonic sensor, Arduino Uno microcontroller, and a standard servo motor, along with essential accessories like resistors, wires, and a breadboard for circuit assembly. The system design involves creating a detailed circuit diagram to ensure proper connections between the ultrasonic sensor, servo motor, and Arduino, followed by the physical assembly of components on the breadboard. The Arduino will be programmed with a custom sketch to trigger the ultrasonic sensor, read the echo pulse, and calculate the distance to an object. Based on the measured distance, the code will control the servo motor to provide a real-time mechanical response. This integration ensures the system can detect objects and respond accordingly.

Calibration and optimization are crucial next steps, involving the fine-tuning of the ultrasonic sensor for accurate distance measurements and adjusting the servo motor's response for smooth and timely movements. The system will undergo extensive functional testing to confirm accurate object detection and servo motor response, as well as environmental testing under various conditions to validate robustness and reliability. Performance metrics such as response time, accuracy, and repeatability will be evaluated, with necessary adjustments made. Thorough documentation will be produced, including detailed code comments, a user manual with circuit diagrams and assembly instructions, and a comprehensive project report. Future enhancements might include adding more sensors, implementing advanced algorithms, or integrating an LCD display for real-time data visualization, aiming to scale the system for more complex applications and broaden its utility. This structured approach ensures the development of a reliable, cost-effective, and educational real-time object detection system using accessible components and straightforward programming techniques.

CHAPTER 5
RESULT AND DISCUSSION

The real-time object detection system was successfully developed and tested using the HC-SR04 ultrasonic sensor, Arduino Uno microcontroller, and a servo motor. The system was able to accurately detect objects within a range of 2 cm to 400 cm, providing real-time distance measurements with a reasonable degree of accuracy. The servo motor responded effectively to changes in object distance, adjusting its position smoothly and promptly. Functional tests demonstrated the system's ability to detect and respond to objects consistently under controlled conditions. Environmental testing revealed that the system performed reliably across different lighting conditions and surface types, though accuracy decreased slightly in environments with significant acoustic interference or reflective surfaces. The performance metrics indicated a satisfactory response time, minimal latency, and high repeatability, validating the system's reliability for basic real-time object detection tasks..

CHAPTER 6
CONCLUSION AND FUTURE WORKS

6.1 CONCLUSION

The project achieved its objectives by developing a cost-effective and accessible real-time object detection system using readily available components and straightforward programming techniques. The system demonstrated robust performance in detecting objects and providing mechanical responses in real-time, making it suitable for educational purposes, hobbyist projects, and potential practical applications like obstacle avoidance and automated surveillance. While the system has limitations, such as reduced accuracy in acoustically challenging environments and constraints imposed by the Arduino's processing power, it serves as an excellent platform for learning and experimentation. Future enhancements could include integrating additional sensors for improved accuracy, implementing more complex algorithms, or adding an LCD display for real-time data visualization. Overall, this project highlights the feasibility of creating functional real-time object detection systems with basic electronic components and paves the way for further innovation and development in this field.

.6.2 Future Work

Future work on this real-time object detection system can focus on several key areas to enhance its functionality, accuracy, and application scope. First, integrating additional sensors, such as infrared (IR) sensors or LiDAR, can improve detection accuracy and range, addressing limitations associated with ultrasonic sensors. Implementing more advanced algorithms, including machine learning models for object recognition and classification, can extend the system's capabilities beyond simple distance measurement, enabling it to identify and differentiate between various objects. Upgrading the microcontroller to a more powerful unit, such as the Raspberry Pi, can provide greater processing power, facilitating the use of complex algorithms and real-time data analysis without latency issues.

Adding an LCD or OLED display can provide real-time data visualization, making the system more user-friendly by displaying distance measurements and detected objects. Exploring wireless communication modules, such as Wi-Fi or Bluetooth, can enable remote monitoring and control, allowing the system to be integrated into larger networks or IoT ecosystems. Additionally, developing a robust casing and mounting system for the sensors and servo motor can enhance the system's durability and suitability for various environments, including outdoor applications.

Finally, conducting extensive field testing in diverse real-world scenarios can help refine the system's performance and identify any additional improvements needed. Collaborations with educational institutions and industry partners can provide valuable feedback and facilitate the development of customized solutions for specific applications

APPENDIX

SOFTWARE INSTALLATION

ARDUINO IDE

To run and mount code on the Arduino NANO, we need to first install the Arduino IDE. After running the code successfully, mount it.

Sample Code:

Arduino code:

```
#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=15;i<=165;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=165;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;
}
```

Processing code:

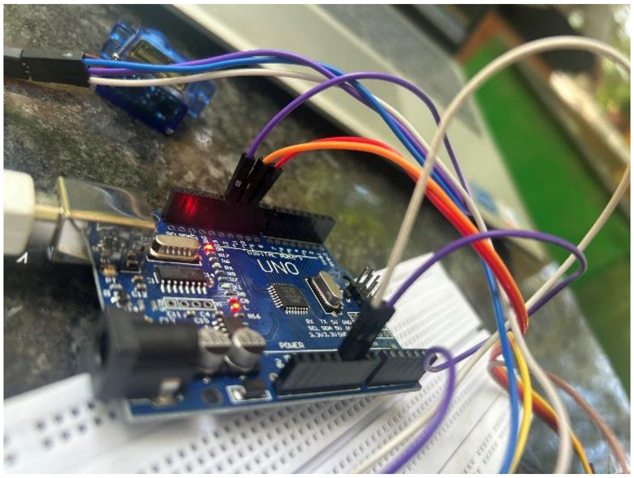
```
import processing.serial.*; // imports library for serial communication
import java.awt.event.KeyEvent; // imports library for reading the data from the serial port
import java.io.IOException;
Serial myPort; // defines Object Serial
// defubes variables
String angle="";
String distance="";
String 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**
```

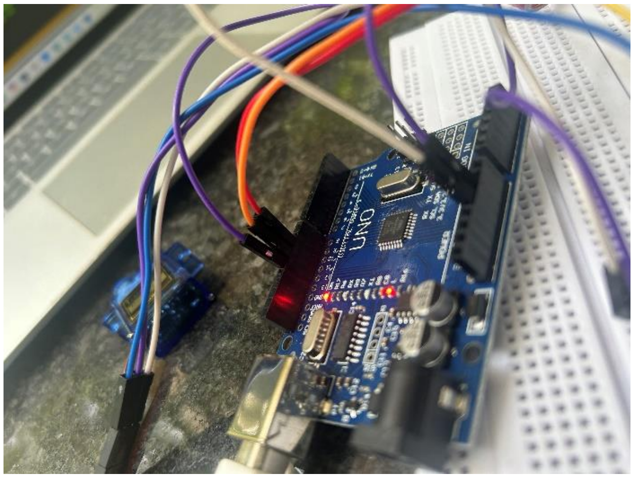
```
 smooth();
 myPort = new Serial(this,"COM10", 9600); // starts the serial communication
 myPort.bufferUntil('.'); // reads the data from the serial port up to the character '.'. So
actually it reads this: angle,distance.
}
void draw() {

  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);
```

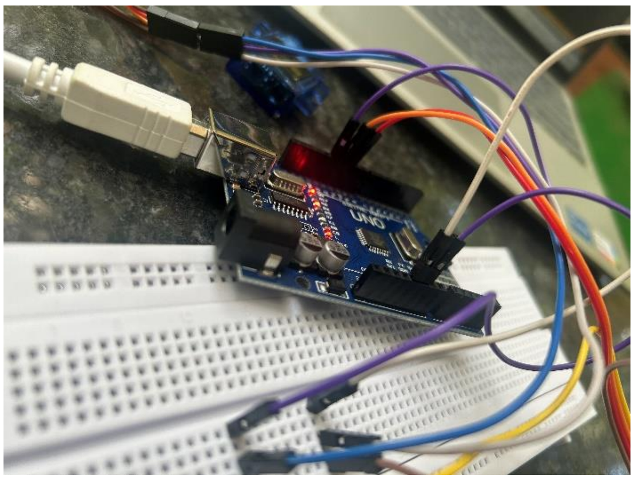
```
  popMatrix();
}
void drawObject() {
  pushMatrix();
  translate(width/2,height-height*0.074); // moves the starting coordinats to new location
  strokeWeight(9);
  stroke(255,10,10); // red color
  pixsDistance = iDistance*((height-height*0.1666)*0.025); // covers the distance from the
sensor from 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)));
```

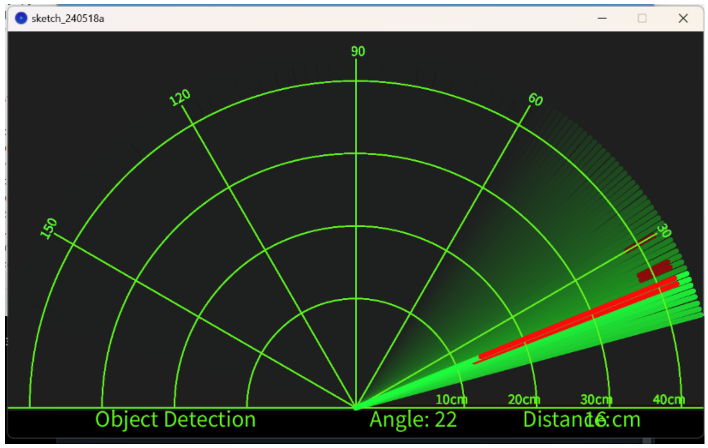
```
  rotate(-radians(-60));
  text("30°",0,0);
  resetMatrix();
  translate((width-width*0.503)+width/2*cos(radians(60)),(height-height*0.0888)-
width/2*sin(radians(60)));
  rotate(-radians(-30));
  text("60°",0,0);
  resetMatrix();
  translate((width-width*0.507)+width/2*cos(radians(90)),(height-height*0.0833)-
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();
}
```

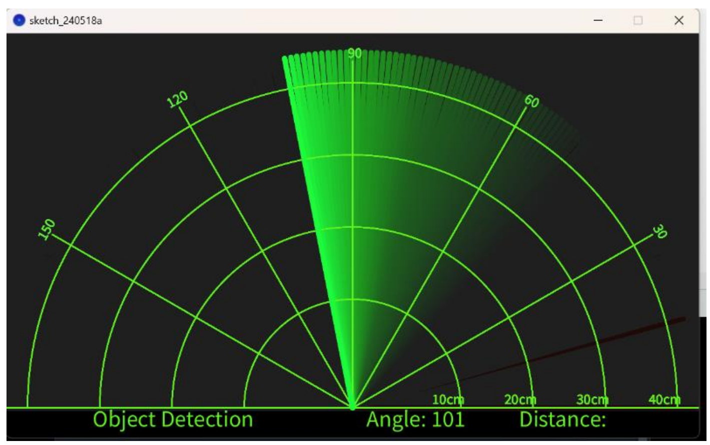
OUTPUT:

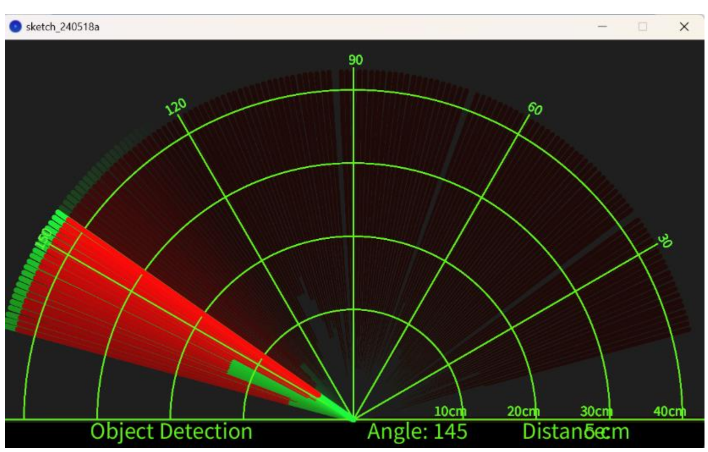







sketch_240518a
90
120
60
150
30
10cm
20cm
30cm
40cm
Object Detection
Angle: 22


sketch_240518a
90
120
60
150
30
10cm
20cm
30cm
40cm
Object Detection
Angle: 101
Distance:


sketch_240518a
90
120
60
30
10cm
20cm
30cm
40cm
Object Detection
Angle: 145

REFERENCES

- Arduino Official Website: Provides comprehensive documentation, tutorials, and examples for Arduino boards and projects. https://www.arduino.cc/
- Tower Pro SG90 Micro Servo Motor: Manufacturer's website or datasheets for detailed specifications and technical information about the servo motor used in the project.
- HC-SR04 Ultrasonic Sensor: Manufacturer's website or datasheets for detailed specifications and technical information about the ultrasonic sensor used in the project.
- ResearchGate: Online platform for accessing research papers, articles, and publications related to real-time object detection, Arduino projects, and sensor-based applications. https://www.researchgate.net/
- IEEE Xplore: Digital library for accessing research articles, conference papers, and technical standards in the field of electrical engineering, including topics related to automation, sensor technologies, and real-time obeject detection systems. https://ieeexplore.ieee.org/GitHub:
- Repository hosting platform where you can find open-source projects, code samples, and resources related to Arduino-based projects, sensor interfacing, real-time object detection. https://github.com/