ARDUINO MILITARY RADAR MODEL USING ULTRASONIC SENSOR FOR DISTANCE, DETECTION AND ANGLE FINDING FOR ZIMBABWE DEFENCE FORCES

DEPARTMENT OF COMPUTER SCIENCE

HCSE 236 COMPUTER PROJECT

MUSHIPE FARAI P.J (R2118159E)

MANYERE SOLOMON (R2111997J)

ARDUINO MILITARY RADAR MODEL USING ULTRASONIC SENSOR FOR DISTANCE, DETECTION AND ANGLE FINDING

BY

Mushipe Farai P.J (R2118159E)

Manyere Solomon (R2111997J)

Submitted in partial fulfilment of the requirements for the degree of

BSc Honors Computer Systems Engineering

DEPARTMENT OF COMPUTER SCIENCE

In the

Faculty of Science and Technology

At the

MIDLANDS STATE UNIVERSITY

GWERU

Supervisor: **Mr. P Mupfiga**

ABSTRACT

This project aims to develop an Arduino-based military radar model for the Zimbabwe Defense Forces. The model uses an ultrasonic sensor to determine distance and determine angle. The system is designed to detect and track objects within a specific range and angle of view. An ultrasonic sensor is used to measure the distance to the target and a servo motor is used to rotate the sensor to determine the angle of the target. Arduino boards are used to process data and control servo motors. The system can detect objects up to 4m away and at an angle of 180 degrees. This model is economical and easily integrated into existing equipment.

DECLARATION

We, **Mushipe Farai P.J** and **Manyere Solomon,** hereby declare that we are the sole author of this mini project. We authorize the **Midlands State University** to lend this project to other institutions or individuals for the purpose of scholarly research.

Signature: …………………………………………….

Signature: …………………………………………….

Date: ………………………………..……………….

APPROVAL

The project, entitled **“Arduino Military Radar Model Using Ultrasonic Sensor For Distance, Detection and Angle Finding”** by **Mushipe Farai P.J and Manyere Solomon** meets the regulations governing the award of the degree of **BSc Honors Computer Systems Engineering** of the **Midlands State University,** and is approved for its contribution to knowledge and literary presentation.

Supervisor’s Signature: [illegible]

Date: 09 - 05 - 2023

ACKNOWLEDGEMENTS

We would like to thank everyone who contributed to the completion of this project. First of all, our project supervisor, Mr. P. Mupfiga for his valuable advice, support and feedback throughout the project. His wisdom and insight has been instrumental in shaping the direction and performance of our work.

We would also like to thank the faculty and staff of the School of Computer Science for providing us with the resources and facilities we need to conduct our research and experiments. We are grateful to all the participants who generously gave their time and input, without whom this project would not have been possible.

We thank our colleagues and friends for their cooperation, who provided support, advice and moral support. Finally, we would like to express our heartfelt appreciation to our families, who have been a constant source of love and encouragement throughout our academic journey. Thank you all for your support and contributions. Lastly we thank the almighty for his continuously gracing us with his mercies. Thank you for your inspiration and guidance. This project is dedicated to you with heartfelt gratitude and admiration.

DEDICATION

We dedicate this project to our parents, whose inspiration and support have played a significant role in our academic and personal journey. They have been a driving force behind our passion for this project. Their unwavering commitment has inspired us to pursue this project with dedication, perseverance, and enthusiasm. Their values have served as a guiding light for us, and we are grateful for their mentorship, encouragement, and support throughout our academic journey.

TABLE OF CONTENTS

Table of Contents

LIST OF FIGURES

LIST OF ACRONYMS

LCD-Liquid Crystal Display

CHAPTER 1: INTRODUCTION

1.1 Introduction

During World War 2, radar was initially created as a technique for locating enemy aircraft .As technology has developed, it is currently used in many different businesses. Over the past few decades, radar technology has made significant advancements.

We will create an Arduino military RADAR model for the Zimbabwe Defense Forces that makes use of an ultrasonic sensor for detection, distance measurement, and angle calculation. It will use radio waves to detect objects and determine their range, altitude, direction, and speed. Any item in its path will reflect the radio wave pulses that the radar antenna transmits. A component of the wave that was received by the receiver and returned by the object is in line with.

1.2 Background of Study

According to the recommendations of the Civil Aircraft Authority of Zimbabwe (CAAZ), Zimbabwe is now using out-of-date, hazardous aviation technology that was meant to have been urgently updated nine years ago. Zimbabwe's airspace is unstable, dangerous, and unstable due to its outdated technology.

The risks of using an outdated aviation radar system, such as the likelihood of aircraft collisions, the inability to quickly identify troubled aircraft, delays and elevated operational expenses for airlines, were highlighted by the International Civil Aviation Organization. Other dangers of operating a dysfunctional aviation radar system include financial losses from planes avoiding the area and a negative image of the nation.

Nowadays, people use current technology for a variety of criminal operations, including the smuggling of unidentified flying objects like drones, endangered animals, and minerals. The adoption of a new radar system reflects the need for the Zimbabwe Defense Forces to mechanize and modernize existing equipment. There are too many airports, and some small, stealthy aircraft fly at extremely low altitudes. This is because the outdated radar system needs to be replaced immediately.

By maintaining a safe airspace, the Zimbabwe Defense Forces will continue to fulfill their duty of protecting the country from external threats.

1.3 Organizational Organogram

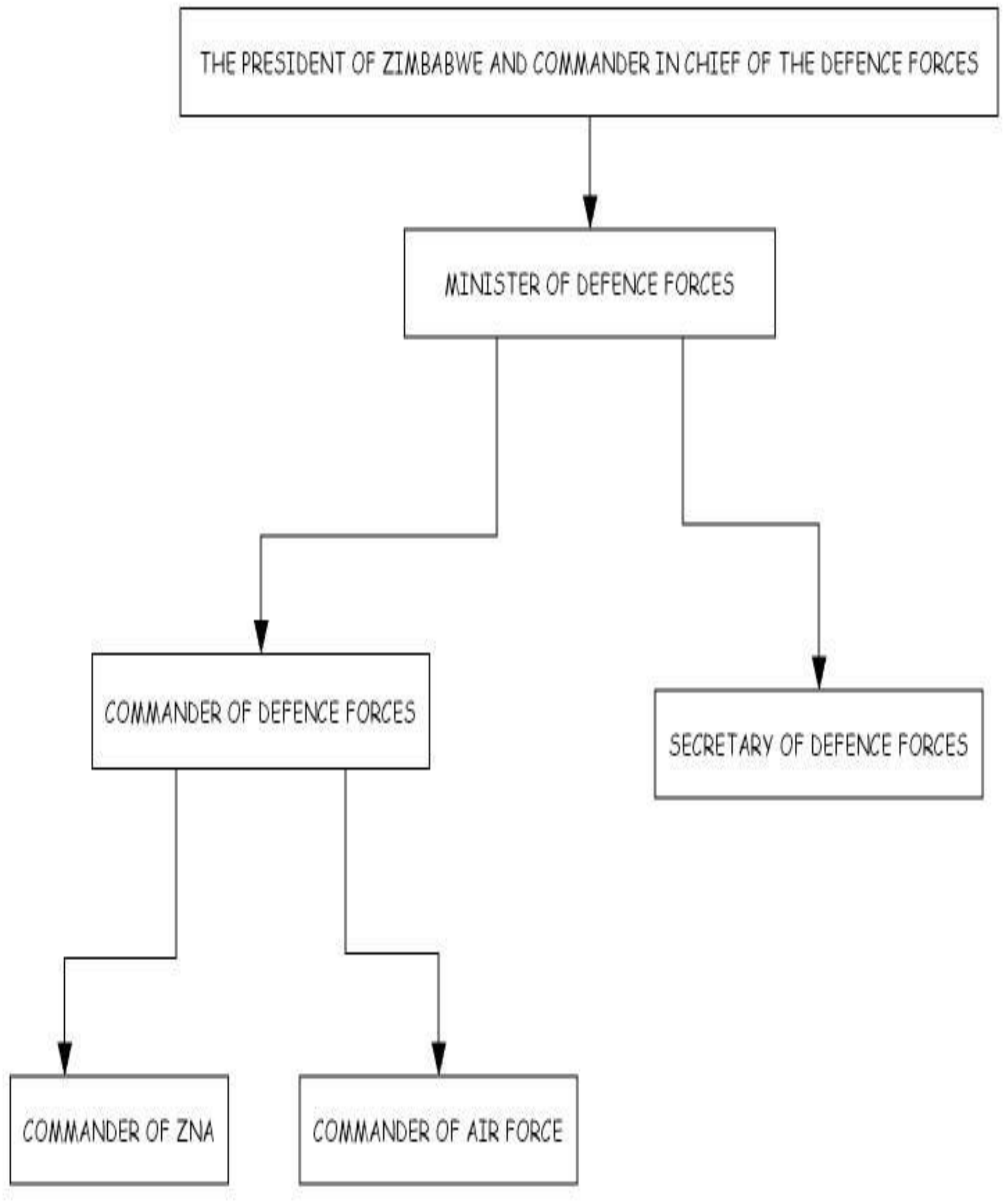

THE PRESIDENT OF ZIMBABWE AND COMMANDER IN CHIEF OF THE DEFENCE FORCES
MINISTER OF DEFENCE FORCES
COMMANDER OF DEFENCE FORCES
SECRETARY OF DEFENCE FORCES
COMMANDER OF ZNA
COMMANDER OF AIR FORCE

Figure 1.0 Organogram

1.4 Problem Definition

Zimbabwe's old and dysfunctional aviation radar control system makes the country's airspace insecure and undermines national security. Unmonitored aircraft are now flying in the nation's airspace, raising fears that some of them may be used to smuggle resources out of the country. According to reports, sophisticated syndicates are responsible for leaks costing the country up to $100 million USD each month. Additionally, there are the hazards of an aircraft crash, failing to see a troubled aircraft in time, delays, and rising airline operating costs.

1.5 Aim

This project intends to develop an accurate ultrasonic RADAR measuring system prototype for the Zimbabwe Defense Forces, using dependable and economical sensors and actuators for object identification, distance calculation, and angle triangulation.

1.6 Objectives

- Program an Arduino controlled ultrasonic radar in the IDE to display servo distance and angle readings on the Processing IDE.
- Keep track of the values you extracted from the prototype and perform thorough testing to ensure their accuracy.
- Examine and record the ultrasonic sensor's measurements, then contrast them with actual scale readings made with various materials having various physical characteristics (from 0-40 cm, obstacle placed at every 10 cm).
- When the barrier is positioned at intervals of 5 mm, examine and record the ultrasonic readings of the ultrasonic sensor to actual scale measurements.
- Examine and record the ultrasonic sensor readings that were acquired when it was placed in a symmetric rectangular box with the sensor covering all three walls and rotating 180 degrees.
- Examine and record the ultrasonic sensor's measurements when it is rotated 180 degrees and placed against a straight wall without any edges.
- To locate low-lying airborne items

1.7 Instruments and Methods

- The fundamental and all-purpose programming software for every form of Arduino programmable board is called Arduino IDE.
- The project is written in the C and C++ programming languages, which are also the best and most well-known for creating hardware-based projects for Arduino since they are simple to use with the Arduino IDE software.
- Processing IDE It offers a potent collection of features and capabilities that can greatly speed up the design process and assist programmers in creating fascinating, captivating applications.

Hardware components and purpose:

- Arduino Uno is a microcontroller, that serves as the project's central nervous system and controls all of the other components, is suitable for balancing power consumption and processing speed)
- 1 x Servo motor (is used to rotate the sonar sensor.)
- Breadboard
- Connecting Wires
- Buzzer
- Ultrasonic Sensor
- LCD Display

1.8 Delimitations and Limitations of the Project

- Sensors are extremely sensitive to temperature variations.
- Difficulties to understand the effects of small, curved and soft objects.
- These sensors have a minimum distance measurement.
- A professional or programmer is required when interfacing sensors with microcontrollers or other controllers.
- These sensors may be water-resistant when used for inspection; otherwise, they risk being damaged.

1.9 Feasibility analysis

Finding out how smoothly the system can be constructed and implemented is the aim of this examination. We can design the system, but analysts need to have the right abilities to assess how well it works and whether people can utilize it because the system will be very sophisticated technologically, the risk could be substantial.

On the other hand, we have economic viability, which leads to cost-benefit analysis, which is determined by analyzing the system's costs and benefits. The creation is great because it will be a locally produced good that can be fixed within the company through system calibration and has a tendency to be affordable yet expensive upon installation. When comparing the cost the proposed program tends to cost the same to manufacture as the existing one to be more inexpensive, but the old system usually needs to be replaced since it is broken and cannot be maintained. The system will be employed if it is created since the security of the airspace is in jeopardy and the Zimbabwe Defense Forces are having difficulties.

1. **Economic Feasibility**

   This analysis covers more issues than financial analysis usually covers. The financial sustainability of a project is the economic value associated with the project financial cost of the project. Balances are expressed as present value (net economic profit).

ITEM/S	COSTS/USD($)
Hardware	130.78
Software	30.00
Total	160.78

Table 1. 0 Economic Feasibility

The goal of a thorough evaluation known as technical feasibility is to determine which hardware and software will best satisfy user needs. It assesses the technical specifications of the system and analyzes whether the information and materials required to build the system are acquirable. The Hardware and Software Needs paper provides specifics on the design and development phase, testing, and implementation requirements.

1.10 Justification and Rationale

We encountered circumstances where we had to keep an eye on restricted areas, such military sites, to prevent trespassing and attacks, as well as look out for flying items like airplanes. Now, using human labor for this task is expensive and also unreliable for maintaining a watch over an area around-the clock. An ultrasonic radar project for object identification, distance computation, and angle triangulation system is therefore being developed for this purpose. The technology can keep an eye on a wide area and notify the controllers with a buzzer. We accomplish this by connecting a microcontroller circuitry to a surveillance ultrasonic sensor installed on a servo motor. To monitor the status of the detection, a LCD display and buzzer are also connected. The radar continuously monitors the ultrasonic sensor echo as it scans the area and also displays readings on Processing.

1.11 Work Plan

ID	TASK NAME	START	FINISH	DURATION	COMPLETE
1	PLANNING	14 NOV 2022	3 DEC 2022	19 DAYS	100%
2	SYSTEM ANALYSIS	5 DEC 2022	30 DEC 2022	25 DAYS	100%
3	SYSTEM DESIGN	2 JAN 2023	14 JAN 2023	12 DAYS	100%
4	DEVELOPMENT	15 JAN 2023	25 JAN 2023	10 DAYS	100%
5	INTEGRATING AND TESTING	28 JAN 2023	8 FEB 2023	11 DAYS	100%
6	IMPLEMENTATION	8 FEB 2023	18 FEB 2023	10 DAYS	100%
7	OPERATIONS AND RESEARCH	22 FEB 2023	8 MAR 2023	14 DAYS	100%
8	DOCUMENTATION	3 JAN 2023	30 APRIL 2023	106 DAYS	100%
9	SUPERVISOR’S REVIEW	14 DEC 2022	30 APRIL 2023	135 DAYS	100%

Table 1.1 Work Plan

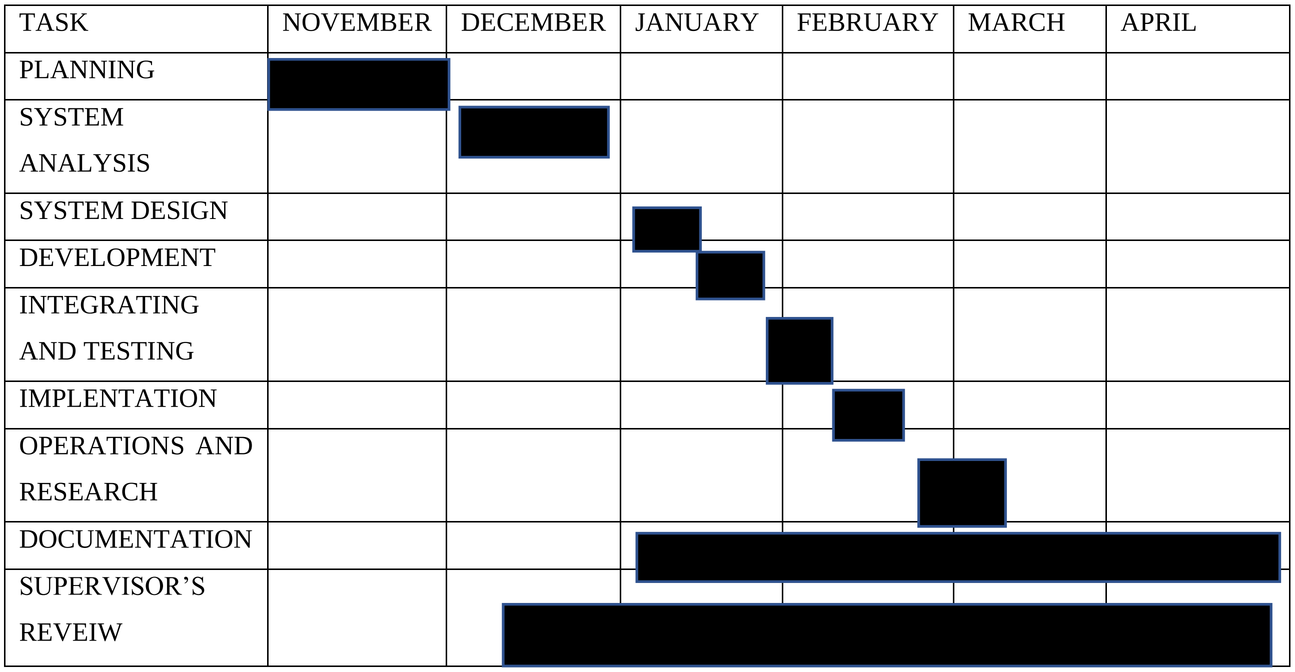


Figure 1.1 Gantt chart

1.12 Conclusion

This chapter discusses how ultrasonic RADAR works to detect objects and calculate distance and angles. It includes the recognized issues, the project's goals, the issues that will be encountered, the solutions, and the reasons. The system design, the details of the current measurement and how it will affect the organization will be analyzed in the next section.

CHAPTER 2: ANALYSIS PHASE

2.1 Introduction

The primary comparison in this chapter is between the current military radar system and the potential replacement. It identifies the main benefits and drawbacks and flags any prospective research gaps that need to be filled.

2.2 Business Value

It will defend the nation’s airspace and support national security by developing a precise ultrasonic RADAR measuring system that uses dependable and affordable sensors and actuators for object identification, distance calculation, and angle triangulation. The nation’s airspace will only be utilized by monitored aircraft, preventing it from being used to smuggle resources out of the country. Aside from that, there will be less risk of an airplane catastrophe, missed opportunities to spot damaged planes, delays, and growing airline operational costs.

2.3 Risk Analysis

There are numerous potential dangers that could have an impact on the project, including: Sensors are incredibly sensitive to changes in temperature. Understanding reflections from small, curved, and soft objects is difficult. These sensors have a minimum detecting range. When these sensors are interfaced with a microcontroller or other controller, a knowledgeable person or programmer is required. We can implement a temperature monitoring system that monitors and regulates the temperature of a specific environment, which would be a crucial component of the proposed system, to lower the chance that sensors are sensitive to temperature variations. You can simply monitor, manage, and adjust the sensors’ temperature in a particular environment with a temperature monitoring system.

2.4 Stakeholder Analysis

Stakeholders are individuals or groups that have an impact on the system and how it functions. Participants in the project include both the government and the Zimbabwe Defense Forces. The stakeholder’s interest or expectations that they would anticipate to be met by the system are to program the Arduino-controlled ultrasonic radar in the IDE and display the distance and servo angle measurements on the monitor of the Arduino Integrated Development Environment. Keep a record of the data you took from the prototype and run careful tests to make sure they’re accurate. Measurements from the ultrasonic sensor should be examined and recorded, then compared to actual scale readings obtained from various materials with various physical properties (from 0-40 cm, obstacle placed at every 10 cm). Examine and record the results once the barrier is positioned at intervals of 5 mm.

2.5 Information Gathering

It examines the methods used to access and collect data from system users and different stakeholders. We employed an interview and a questionnaire as our research tools or fact-finding approaches.

2.5.1 Interview

It involves face to face interaction between the interviewer and the subject, guarantees quick responses, and uses specific questions to gather information about a topic (George& Tegan, 2022). It is a technique for acquiring data in which the researcher questions the interviewee in order to learn more. For example, depending on the topic of the research, written or printed questions may be asked. However, the interviewee may also pose their own questions, such as how the research will benefit them or the organization, or how the questions were formulated, such as through consultations with the targeted area. When doing an interview when the interviewee asks some questions, tape recorders could be the best option. There are two types of interviews listed below.

- **Face to face** – made on the ground so that the whole face and mind can be looked at for a better assessment.
- **Phone calls -**
  Calls are made while moving from place to place, but you need to be careful with noise b
  ecause it can be affected by the noise of such devices, which can lead to inaccurate findin
  gs or interpretations.

Due to the interviewee’s busy schedule and the speed of telephone interviews, we employed them as the developer.

Phone call Interviews

When conducting research, phone call interviews are utilized when you need a quick answer or feedback from the interviewee. If the researcher and the respondent are in different locations, it can be used to find and contact the intended interviewee. It can also be used to conduct interviews that provide adequate or enough information and enable the interviewee to express their emotions or provide information about how they truly feel about the research project. Phone calls assist in gathering information that is useful for use in carrying out a research or project being studied. Instead of focusing on a single geographic region, it is beneficial to examine several regions. However, using this style of interview, the researcher or interviewer is unable to see how the response (gestures).

Why chose interviews?

- It is a quicker method of obtaining information from several locations. Without the interviewee's knowledge, information can be recorded or written down.

Disadvantages

- Prevents the researcher from observing the respondent's verbal and social clues to predict how they would answer. Examples include gestures and facial expressions, which are also effective ways to gather information.

Results of the interview

The researcher was able to understand the problems faced by the surveillance team after gathering data from the population sample during this interview, such as inability to spot malfunctioning aircraft in time, which resulted in delays and failure to detect low-flying objects and UFOs.

2.5.2 QUESTIONNAIRE

A questionnaire is a set of closed questions used to elicit information about the respondent's opinions, life experiences, or attitudes.

Advantages

- It gives respondents time to consider their answer before answering and enables them to comprehend the requirements of the inquiry.
- Why taking less time than interviews because there would be no need for the researcher to ask the participant to clarify whether they understand the question.
- In comparison to interviews, this strategy obtains more data and covers more ground

Demerits

- There is no assurance that all questions will be answered because the investigator observed as He gathered the papers demonstrating that the vast majority of responders did not address every inquiry.
- The papers could be harmed by weather conditions like precipitation and wind combined.
- The researcher found it challenging to express emotions and thoughts.
- Certain inquiries can be challenging to understand, and respondents might concealing information.

2.6 Weakness of the existing system

Zimbabwe's outdated and malfunctioning aviation radar control system compromises national security by making the country's airspace unsafe. These devices have a blind region of roughly 20–25 meters. Furthermore, these systems need to be positioned so that nothing can be in the transmission cone (usually 25° vertical). There are currently unreported aircraft operating in the nation's airspace, which has sparked concerns that some of them might be used to transport resources out of the country. Reports claim that sophisticated syndicates are to blame for leaks that cost the nation up to $100 million USD per month. The risks of an aircraft crash, failure to spot a troubled aircraft in time, delays, and increasing airline operating costs are also present.

2.7 Evaluate alternatives

There are alternatives to the current system, which has both advantages and disadvantages. The Zimbabwe Defense Forces must decide which alternative is best. The invention is fantastic because it will be a locally produced good that can be fixed inside the business through system calibration and tend to be inexpensive but expensive upon installation. A new system is usually less expensive than the same existing system, but often needs to be replaced because the old system is damaged and beyond repair. If the technology is developed, it will be used by the Zimbabwe Defense Force believes that the security of the airspace is in danger.

2.7.1 Outsource

According to Taplin (2020), outsourcing is the act of giving another company the responsibility of developing a system. In this process, the answer was found from a source other than the service providers. This procedure is not the most efficient technique to fix the system's present issues because of the company's restricted funding.

2.7.2 Improvement

The company needs to update the system that will employ an ultrasonic radar to identify objects, calculate distances, and triangulate angles. We faced situations where we had to watch out for flying objects like airplanes as well as keep an eye on restricted places like military bases to prevent trespassing and attacks. Now, using human labor for this job is expensive and also unreliable for constantly keeping an eye on a location. A vast region can be monitored by an ultrasonic radar system, which can also buzz in the authorities. We accomplish this by connecting a microcontroller circuit to a monitoring ultrasonic sensor mounted on a servo motor. To monitor the status of the detection, an LCD screen and buzzer are also connected and results are displayed on Processing IDE.

2.7.3 Development

In-house development puts more of an emphasis on creating a program or system from scratch with the aid of system developers currently employed and who possess the necessary abilities to accomplish so.

The institution keeps complete ownership and management of the system thanks to this method, which provides a number of advantages.

- A sense of ownership is felt when a system is developed using internal development alternatives.
- The ability to design a specific system that flawlessly achieves the goals of the system.

2.8 Requirements Analysis

To predict customer needs in the system development process, requirement analysis is a skill that must be used. Due to the clients' continually changing needs (Boogard, 2022). As a result, in order to keep them updated on the clients' continuously changing demands, potential system users should be considered by system developers while analyzing the requirements.

2.8.1 Functional Requirements

To meet the basic requirements of end users, the system must meet certain prerequisites (Chitra, 2020).This refers to the style and content of the most current structural proposal. The system must meet a wide range of functional criteria, including each system user's access control, which is accomplished by user login and registration, with restrictions placed on their accounts based on their commitments.

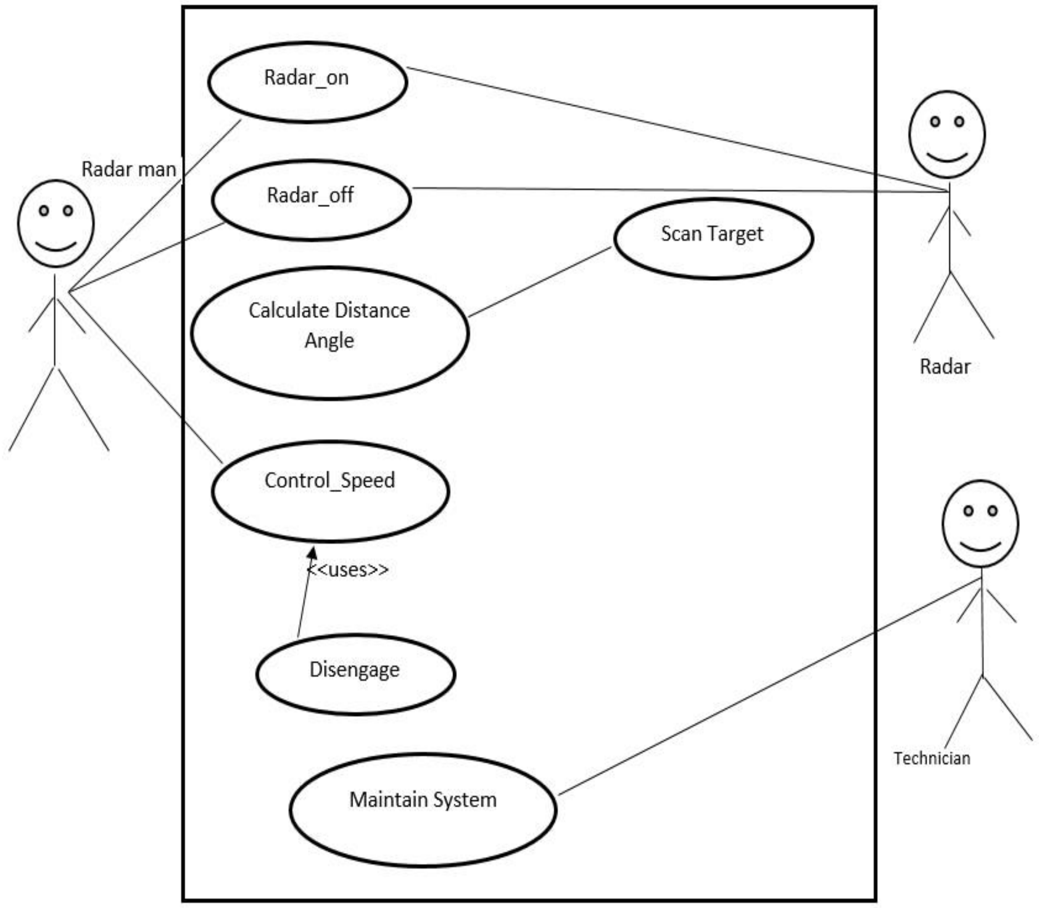


Figure 2.0 Use case diagram

According to Denis, case diagrams are accustomed to using diagrams to depict procedures where the system's primary attributes are shown. The system processes user input.

2.8.2 Non-functional Requirements

These requirements that have no bearing on how the system functions (Chitra, 2020). Non-functional requirements are focused on how the system will carry out a certain activity. Newly introduced non-functional requirements have no impact on the system's functioning, but they do have an impact on how effectively it works. (1921, Gorabenko) Examples of non-functional requirements include the following: Speed: The efficiency with which a system completes a given task.

The system's accessibility: For how long is it available? For instance, does it operate continuously? The degree of a system’s dependability is its reliability. Usability: How easy it is for a user to use the system

3.0 Conclusion

The functioning of the current system is described in the screen time section. In addition to giving developers a good understanding of how to fix and identify problems, this section also shows how manual processes do their job. The design phase will refer to the design of the process in the next phase.

CHAPTER 3: METHODOLOGY

3.1 Introduction

This chapter describes the hardware and software tools. That is, each and every piece of software or hardware that will be used needs to have its fundamental theoretical information stated additionally included in this section are the circuit design, algorithm design, and activity and flow diagrams.

3.2 Hardware and Software Component

On the other hand, software refers to the set of instructions that allow the hardware to perform certain tasks. Devices refer to the visible and visible part of the system, such as Arduino, potentiometers, and sensors. Both hardware and software are required for hardware to function properly and complete tasks. Both are interconnected but different from each other.

1. **Arduino Uno**

For hardware and software developments for usage in both academic and professional settings, this electronic board was developed. This kind of board could be utilized in the respect that, after being employed for teaching process, it can be updated or formatted to be used once more. It is built around the programmable Atmega328P microprocessor chip. It is compatible with any programming language of choice and includes the open-source Arduino IDE coding environment. The voltage regulator on this sort of board, which allows it to use the suggested operating voltage supply of 5 volts even though the board needs an input voltage of at least 7 volts and a maximum of 12 volts, is interconnected with other elements in order to provide the facility's complete functionality.

Figure 3.0 Arduino Uno

2. **Servo Motor**

A fantastic tool which could develop into a predetermined position is a servo motor. Typically, Servo arm is present with a 180-degree range of motion. A servo may be directed to a certain location using the Arduino, and it will do so.

A servo motor's schematic is shown below.

Figure 3.1 Servo Motor

3. Breadboard

Breadboard can be used to construct electronics without the need for a soldering iron. Just after components are placed on the breadboard’s connectors, additional “jumper” wires are utilized to make connections.

A breadboard's diagram is shown below.

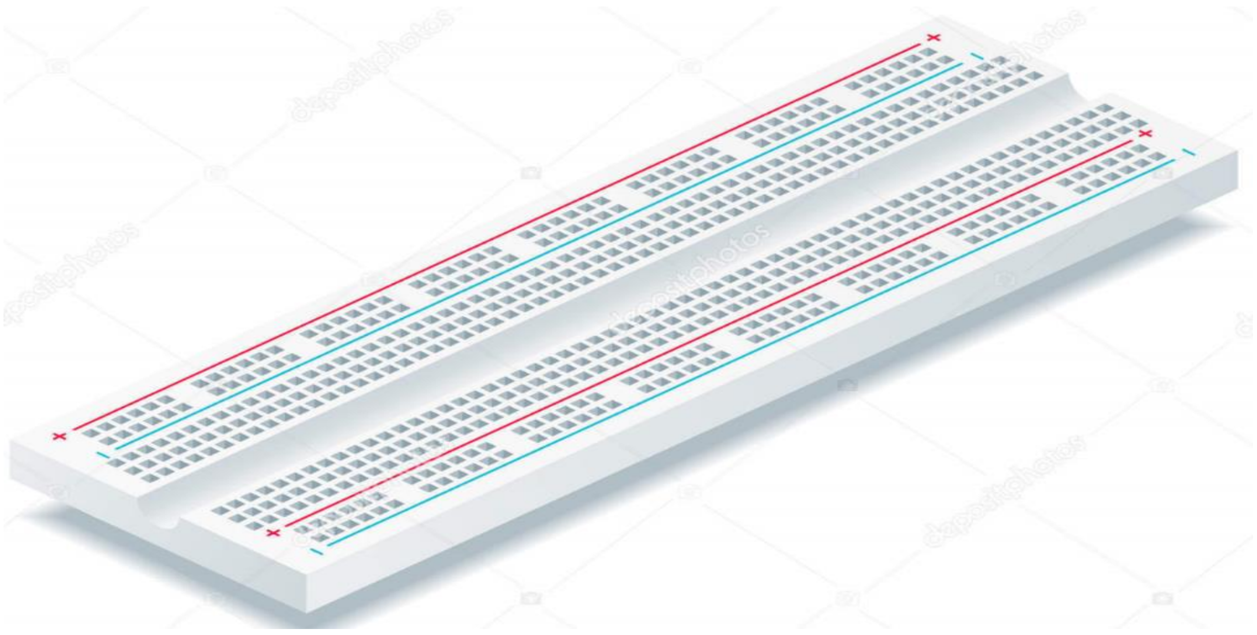


Figure 3.2 Breadboard

4. Connecting Wires

These are the numerous kinds of wires that are employed in the construction of projects or in the interface of various electronic device components to allow for the flow of current. Although these jumper wires come in a variety of colors and can be used for quick connections, their different hues have no bearing on their functionality. They differ in the ends where the connections are made.

Jumper wires are shown in the diagram below.

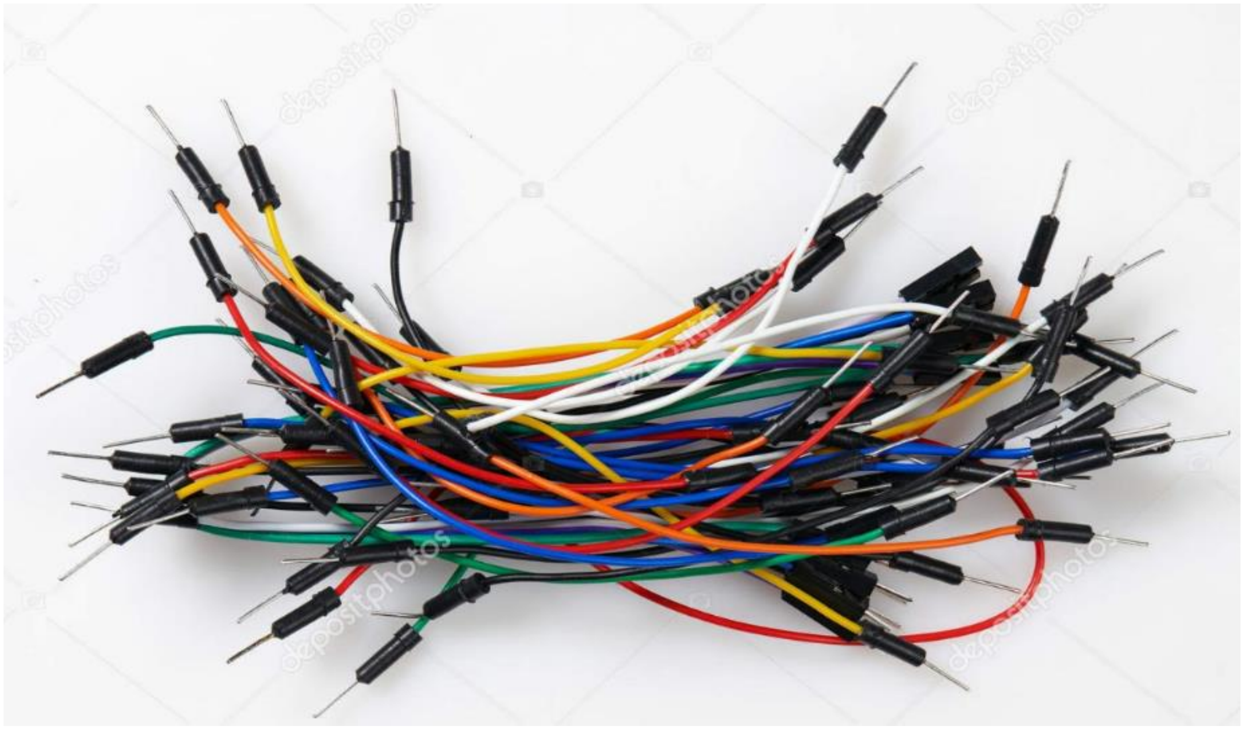


Figure 3.3 Jumper Wires

5. **Buzzer**

In fact, the ringtone is the bell. The Arduino buzzer beeps when a value is passed through it. The Arduino buzzer can be physically connected to the microcontroller and produce multiple sounds by sending electrical pulses to the buzzer at different frequencies.

Below is a diagram of a buzzer

Figure 3.4 Buzzer

6. **Sensor**

The ultrasonic sensor utilizes SONAR, just like bats use, to determine how far away an object is. With excellent precision and reliable readings, it provides detects objects from 2 cm to 400 cm.

Below is a diagram of an ultrasonic sensor

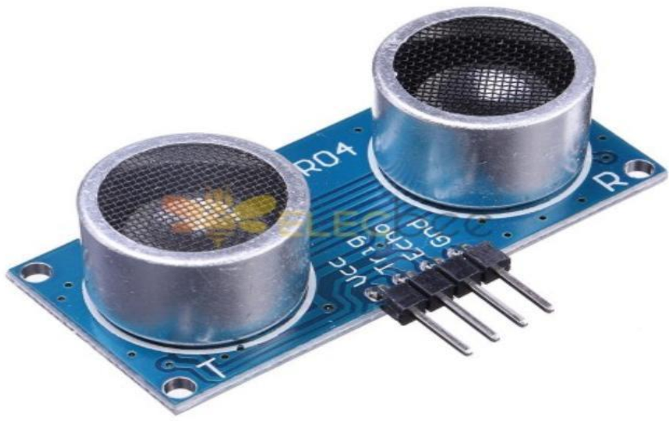
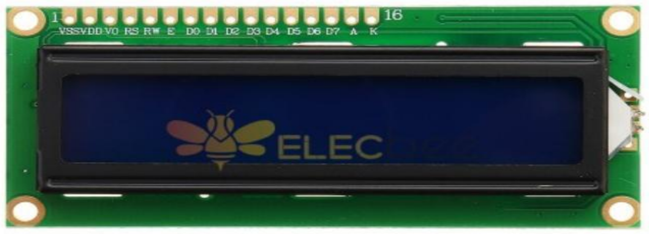


Figure 3.5 Ultrasonic Sensor

7. LCD Display

Liquid crystal is mainly used in the operation of liquid crystal soup type flat panel. LCDs have many uses for consumers and businesses as they are often used in phones, TVs, laptops and boards

Below is a schematic diagram of the LCD display



.

Figure 3.6 LCD Display

8. Tinkercad Software

For the objectives of circuit design and simulation, this program was created. It includes circuit design, PCB design, microprocessor design, and simulation models of component interactions. It aims to provide different electronic circuit designs. Software of this kind can be applied to the prototyping process. It offers the ability to mimic software for any analog or electronic electronics, including microcontrollers.

The purpose of this kind of program is to demonstrate system functionality without utilizing the actual resources or component.

3.3 Proposed System Architecture

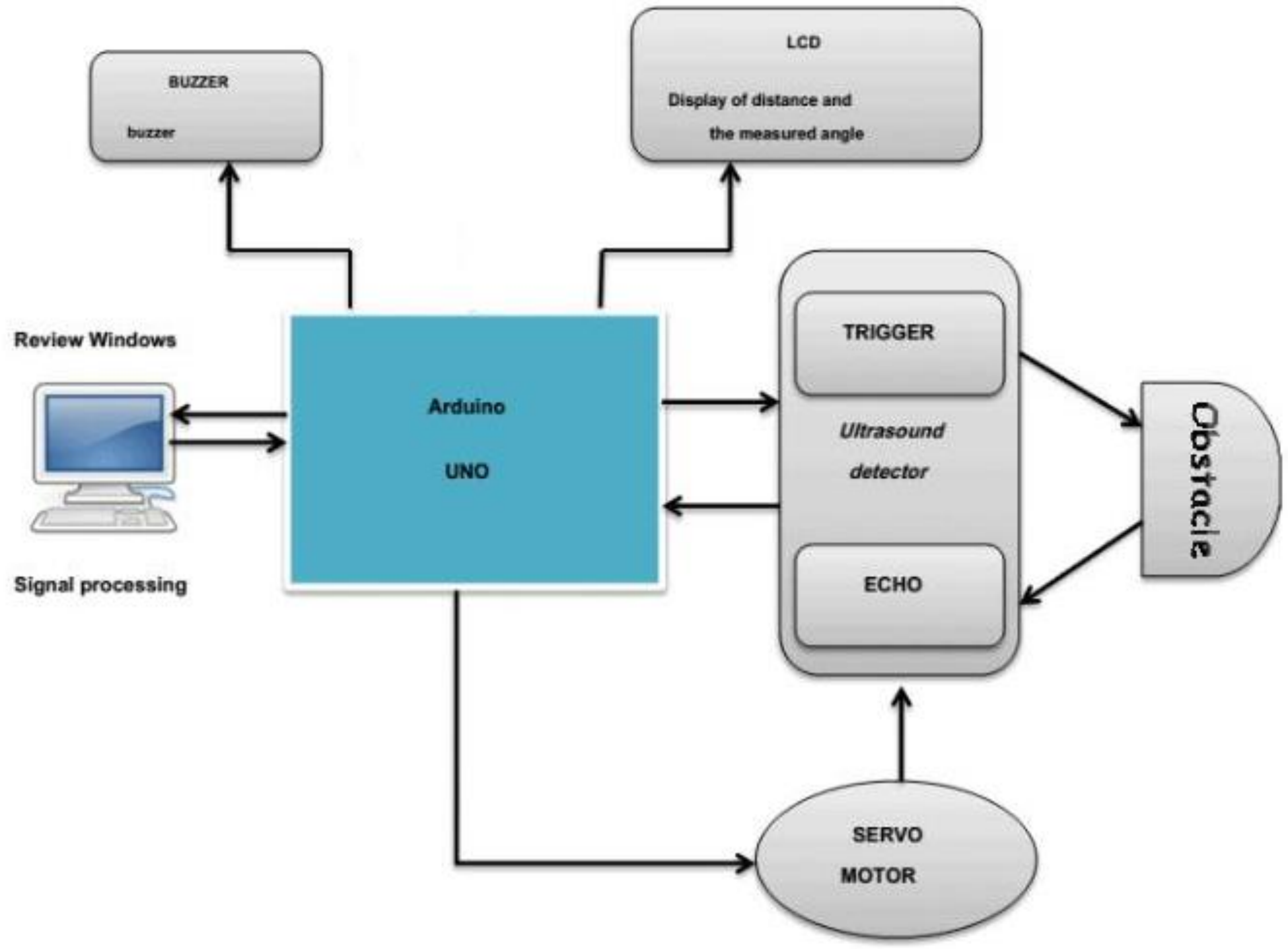


Figure 3.9 Proposed Architecture

3.4 Process Analysis

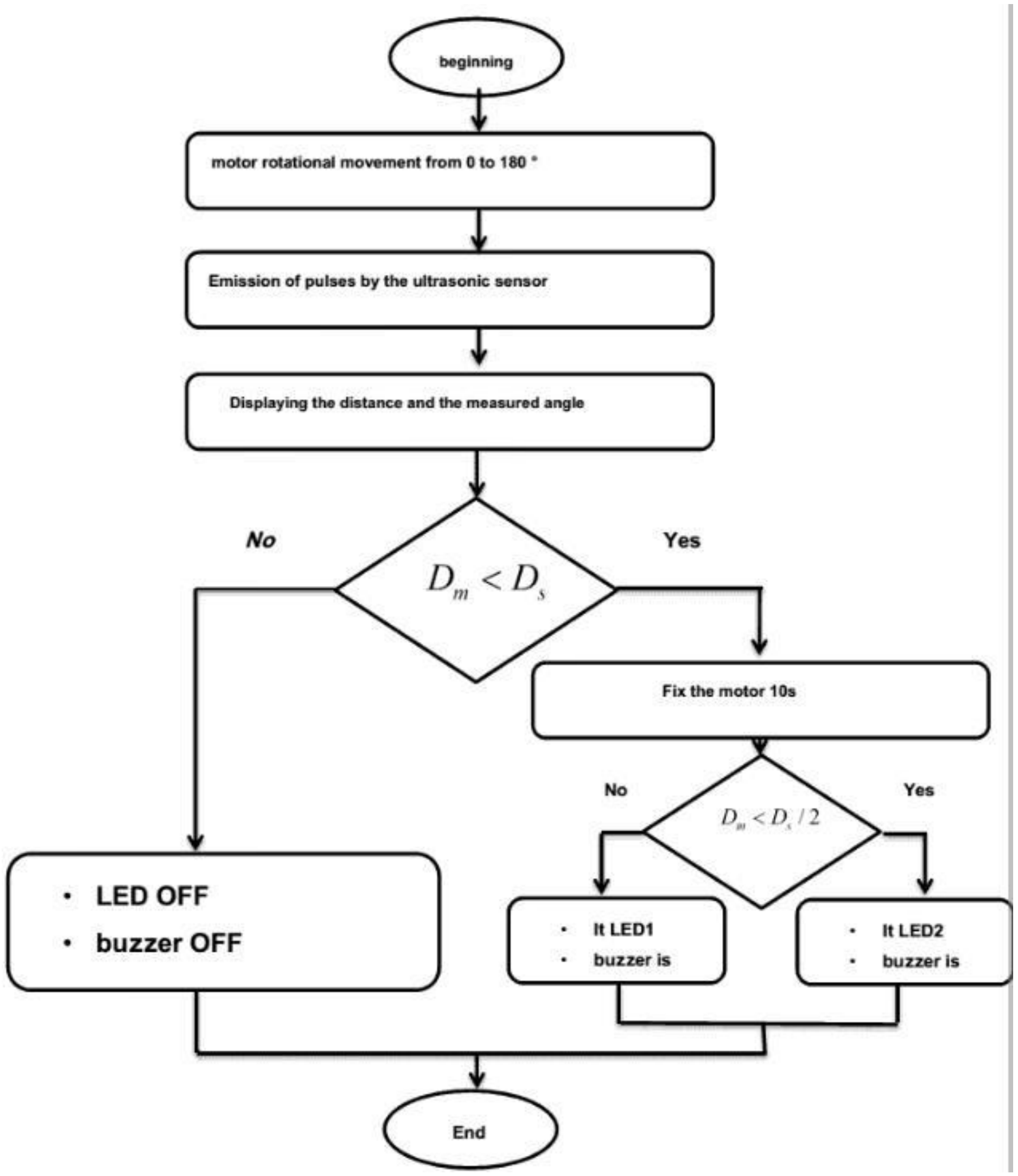

beginning
motor rotational movement from 0 to 180 °
Emission of pulses by the ultrasonic sensor
Displaying the distance and the measured angle
No
Yes
$D_m < D_s$
Fix the motor 10s
No
Yes
$D_m < D_s / 2$
• LED OFF
• buzzer OFF
• It LED1
• buzzer is
• It LED2
• buzzer is
End

Figure 3.10 Process Analysis

3.5 Algorithm Design

Pseudo code is a technique for expressing in an easy-to-understand language how a system or program works, what it is under development or performs tasks for non-technical people to access. According to Dennis et al., pseudo code is a combination of programming languages, but is intended to be read by humans rather than machines.

Pseudo code For Arduino ultrasonic sensor radar system

```
Start:

Declare Variables

Send echo pulse to sensor
If distanceCm <= DistanceSec
If distanceCm<= DistanceSec/2
Send Sound signal to sensor

Else

Stop sound
Display Distance
Position Servomotor

End
```

3.6 Circuit Diagram

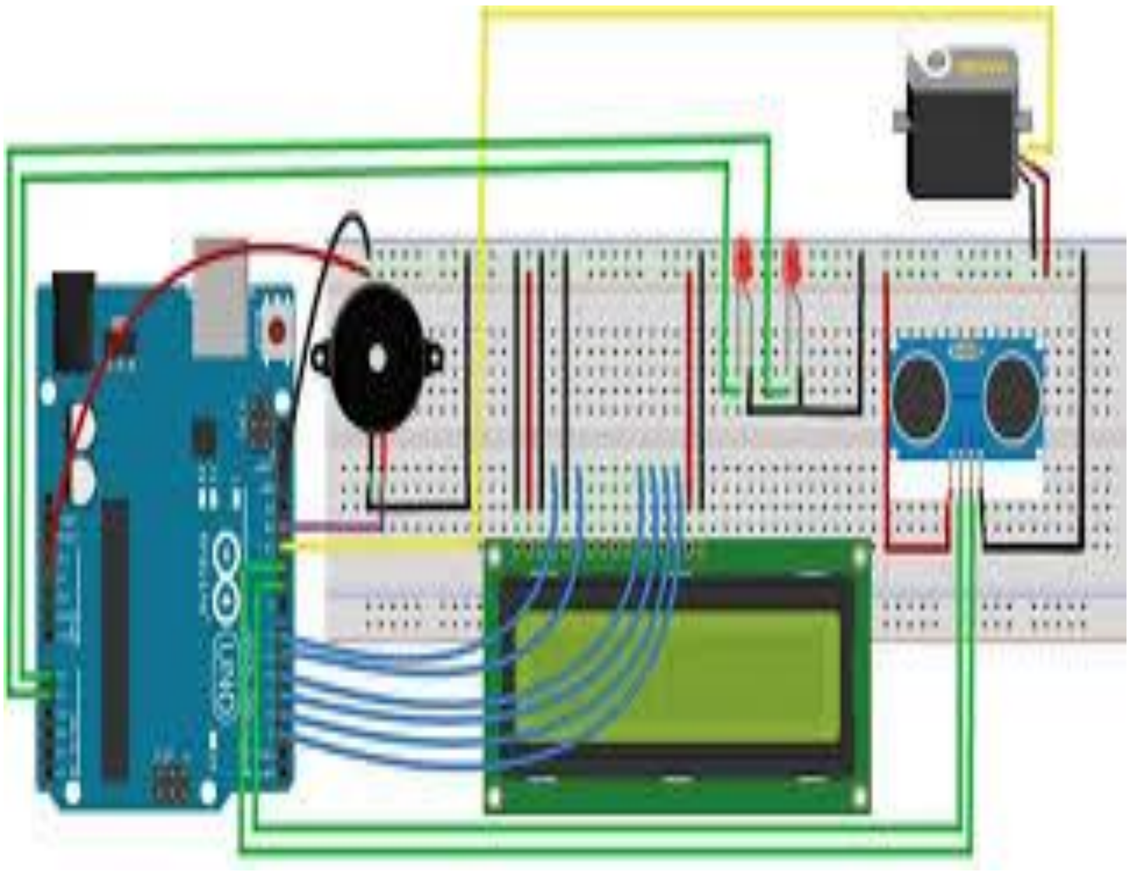


Figure 3.11 Circuit Diagram

An Arduino board's +5V signal is sent to the trig pin of the ultrasonic sensor to activate it. When an object is moving, it may be detected and located within 180 degrees thanks to rotational action provided by the servo motor that is mechanically attached to the ultrasonic Sensor HC-SR04.

The Arduino uses the sensor's TRIGGER pin to generate a series of ultrasonic waves that travel through the air until it encounters an obstacle and then return to the sensor pin in the opposite direction. To determine the distance, the meter measures the pulse width. The signal on the ECHO pin of the sensor stay high throughout the transmission and calculates the time it takes for the ultra sound to return, allowing you to measure the distance. The estimated distance and turning angle are displayed on the LCD. The buzzer is an additional component that when detected, emits a sound (Tone 1 and Tone 2) to determine the area (near or far) an object is in.

3.7 Conclusion

This chapter was successful in providing a thorough analysis of the setup and design of the Arduino ultrasonic radar system. It was successful in accurately describing the functions and intended uses of each component.

CHAPTER 4: RESULTS AND DISCUSSION

4.1 Introduction

The findings showing how the computations, programming, and tests were carried out in the results and debates. It clarifies and illustrates the steps that must be done to carry out the project plan, outlining all required measures, assessments of hardware devices, and simulations.

4.2 Implementation/ Experiments

The process of conducting a research or activity with the goal of attaining a certain outcome or for monitoring reasons is known as an investigation. In hopes of accomplishing a given goal, it also requires the recognition of results.

When using Tinkercad for simulation, the author had to make sure that the modules had been pre-installed as aspect of the simulation model. The libraries were built-in from the start.

Interface Of The Hardware Components

1. **Circuit Set Up**

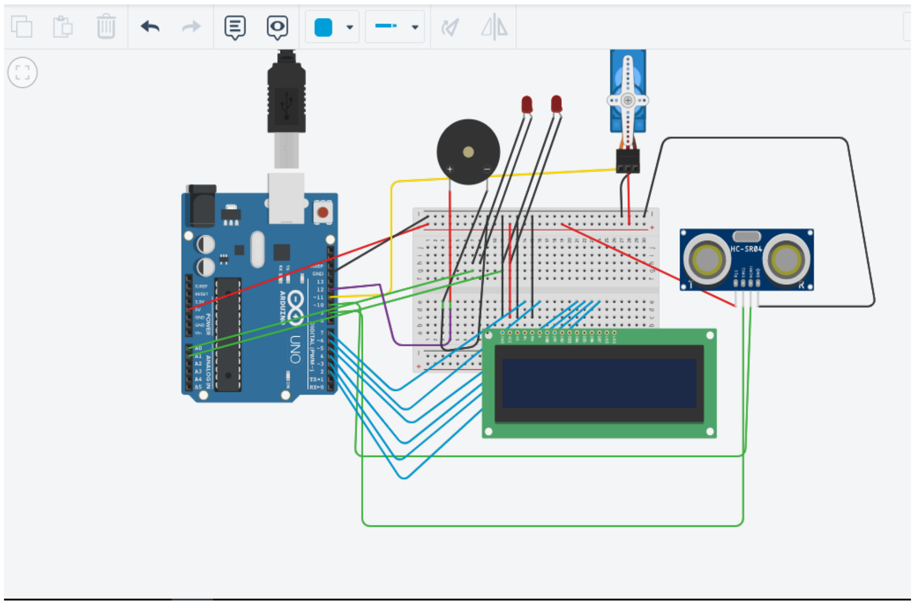


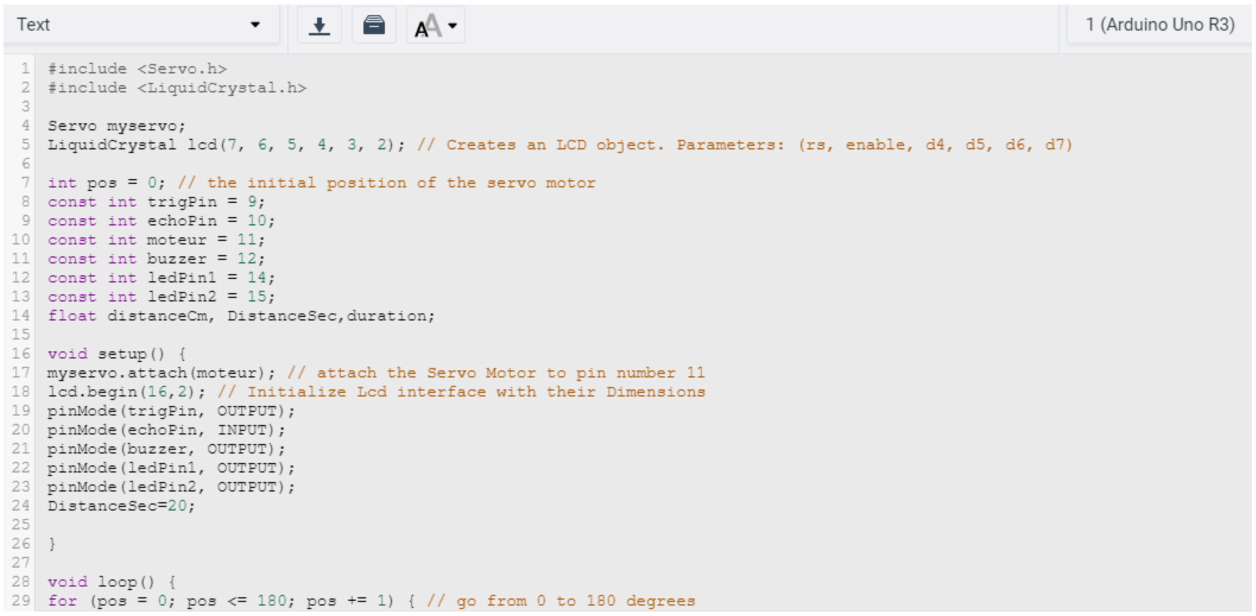
Figure 4.0 Circuit Set Up

In the image above, an Arduino military radar model employing an Arduino Uno microcontroller for distance detection and angle finding is shown. An Arduino board's 5V sends a signal to the trig pin of the (HC-SR04) ultrasonic sensor receives a signal to activate it. When an object is moving, it may be detected and located within 180 degrees thanks to rotational action provided by the servo motor that is mechanically attached to the ultrasonic Sensor HC-SR04.

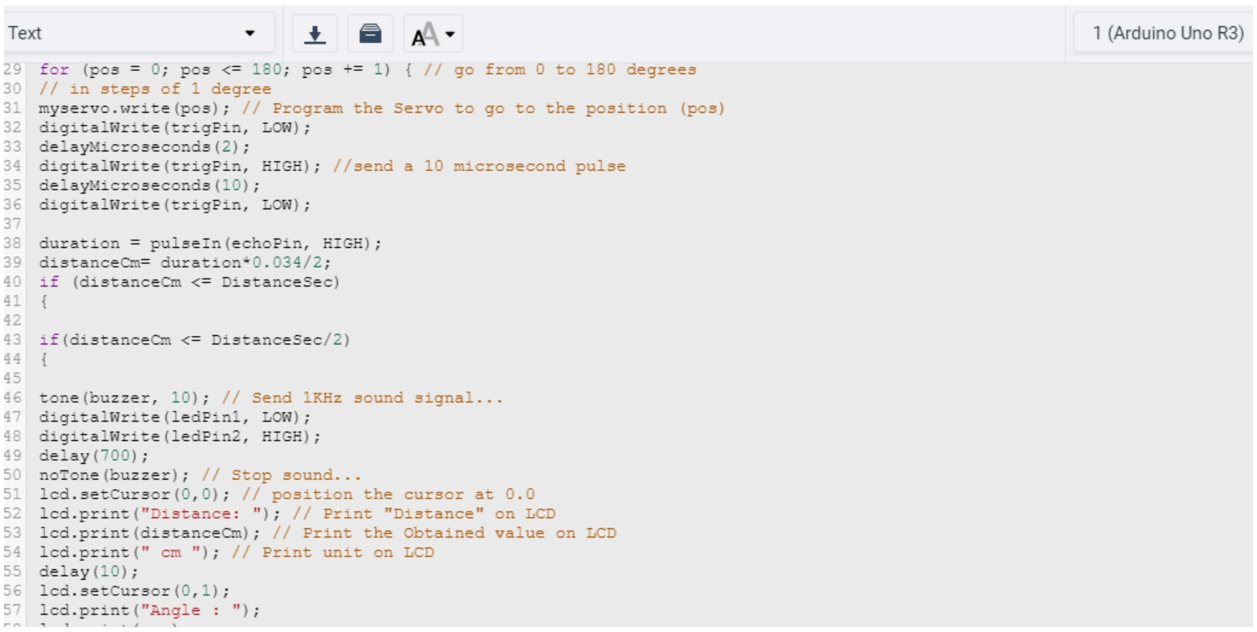
To create a system that will generate ultrasonic waves that travel through the air until it meets the problem, the Arduino sends a maximum 10 second pulse to the pin of the sensor. A sensor measures the pulse width for calculation.

During communication, the signal on the sensor pin ECHO is kept high, allowing you to measure the distance by calculating the time it takes for the ultrasound to return.Using an LCD display, the calculated distance and rotating degree are displayed. The buzzer is indeed an additional component that plays whenever a recognition is achieved along with LEDs. To identify the area that surrounds where the object is positioned, the buzzer and both Lights function in tandem.

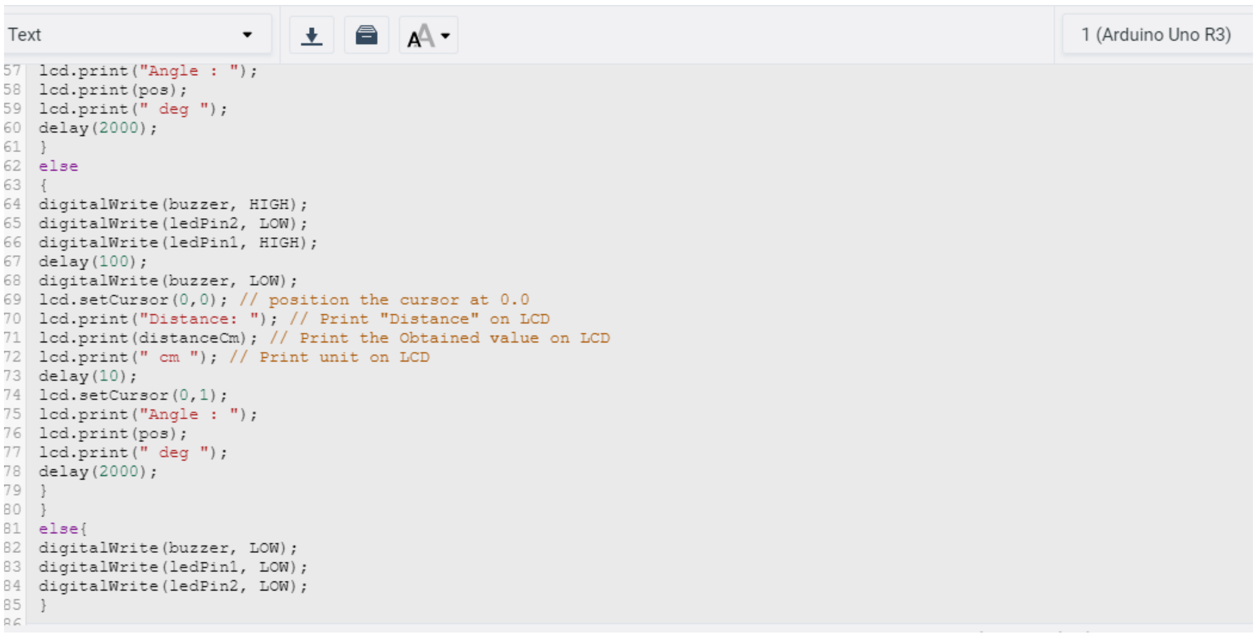
Code Snippets



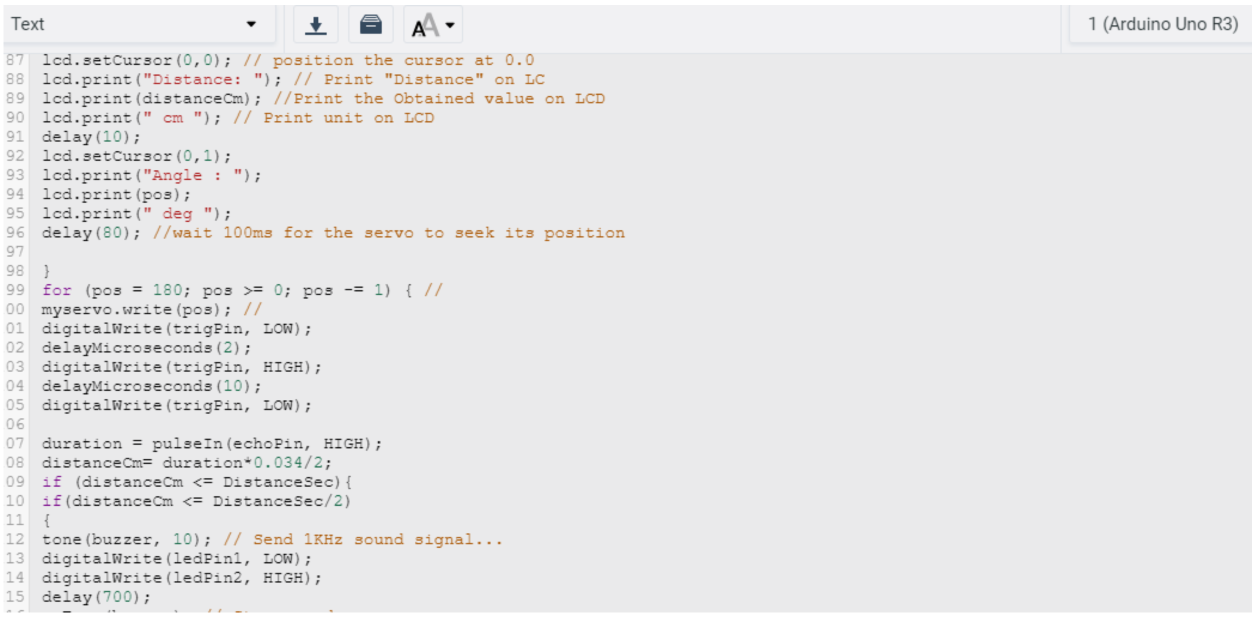
```
#include <Servo.h>
#include <LiquidCrystal.h>

Servo myservo;
LiquidCrystal lcd(7, 6, 5, 4, 3, 2); // Creates an LCD object. Parameters: (rs, enable, d4, d5, d6, d7)

int pos = 0; // the initial position of the servo motor
const int trigPin = 9;
const int echoPin = 10;
const int moteur = 11;
const int buzzer = 12;
const int ledPin1 = 14;
const int ledPin2 = 15;
float distanceCm, DistanceSec,duration;

void setup() {
myservo.attach(moteur); // attach the Servo Motor to pin number 11
lcd.begin(16,2); // Initialize Lcd interface with their Dimensions
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(buzzer, OUTPUT);
pinMode(ledPin1, OUTPUT);
pinMode(ledPin2, OUTPUT);
DistanceSec=20;

}

void loop() {
for (pos = 0; pos <= 180; pos += 1) { // go from 0 to 180 degrees
```



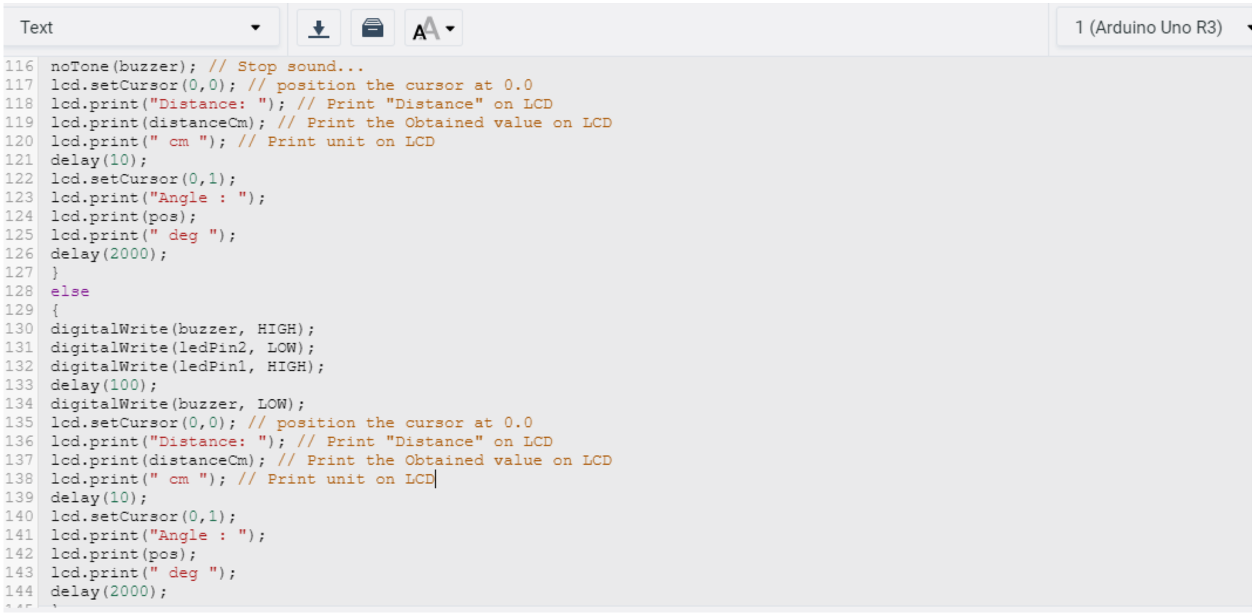
```
for (pos = 0; pos <= 180; pos += 1) { // go from 0 to 180 degrees
// in steps of 1 degree
myservo.write(pos); // Program the Servo to go to the position (pos)
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH); //send a 10 microsecond pulse
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);
distanceCm= duration*0.034/2;
if (distanceCm <= DistanceSec)
{

if(distanceCm <= DistanceSec/2)
{

tone(buzzer, 10); // Send 1KHz sound signal...
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, HIGH);
delay(700);
noTone(buzzer); // Stop sound...
lcd.setCursor(0,0); // position the cursor at 0.0
lcd.print("Distance: "); // Print "Distance" on LCD
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
```



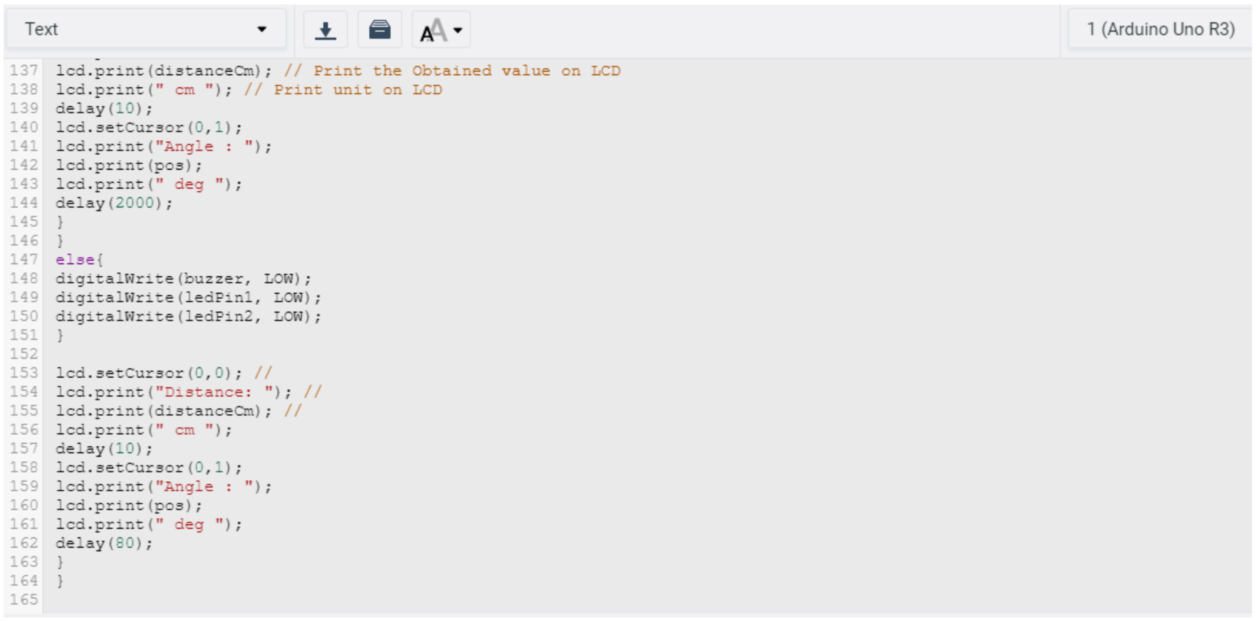
```
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
}
else
{
digitalWrite(buzzer, HIGH);
digitalWrite(ledPin2, LOW);
digitalWrite(ledPin1, HIGH);
delay(100);
digitalWrite(buzzer, LOW);
lcd.setCursor(0,0); // position the cursor at 0.0
lcd.print("Distance: "); // Print "Distance" on LCD
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
}
}
else{
digitalWrite(buzzer, LOW);
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, LOW);
}
```


Text
1 (Arduino Uno R3)

```
lcd.setCursor(0,0); // position the cursor at 0.0
lcd.print("Distance: "); // Print "Distance" on LC
lcd.print(distanceCm); //Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(80); //wait 100ms for the servo to seek its position

}
for (pos = 180; pos >= 0; pos -= 1) { //
myservo.write(pos); //
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);
distanceCm= duration*0.034/2;
if (distanceCm <= DistanceSec){
if(distanceCm <= DistanceSec/2)
{
tone(buzzer, 10); // Send 1KHz sound signal...
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, HIGH);
delay(700);
```


Text
1 (Arduino Uno R3)

```
noTone(buzzer); // Stop sound...
lcd.setCursor(0,0); // position the cursor at 0.0
lcd.print("Distance: "); // Print "Distance" on LCD
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
}
else
{
digitalWrite(buzzer, HIGH);
digitalWrite(ledPin2, LOW);
digitalWrite(ledPin1, HIGH);
delay(100);
digitalWrite(buzzer, LOW);
lcd.setCursor(0,0); // position the cursor at 0.0
lcd.print("Distance: "); // Print "Distance" on LCD
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
```



```
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
}
}
else{
digitalWrite(buzzer, LOW);
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, LOW);
}

lcd.setCursor(0,0); //
lcd.print("Distance: "); //
lcd.print(distanceCm); //
lcd.print(" cm ");
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(80);
}
}
```

Figure 4.1 Code Snippets

The snippets above demonstrate simulation processes for real circuitry setup, code snippets, and modeling procedures used when the entire program was evaluated using Tinkercad to see if the installed element library is operating properly using sample codes that come preloaded with the application.

4.3 Testing and Results

Evaluation is a method of verifying or evaluating requirements to determine whether the system design meets stated objectives and to identify defects in implementation.

This helps determine whether the results are reliable and helps reduce errors, allow and ensure that the system's goals are met. The output of the system will be displayed on the LCD ,but if there is an error in the number or a component is not working properly, another tab will be displayed showing the error in the code, this will help to show the generated problem, for example as a probability check and code statement.

When testing the system, we evaluate

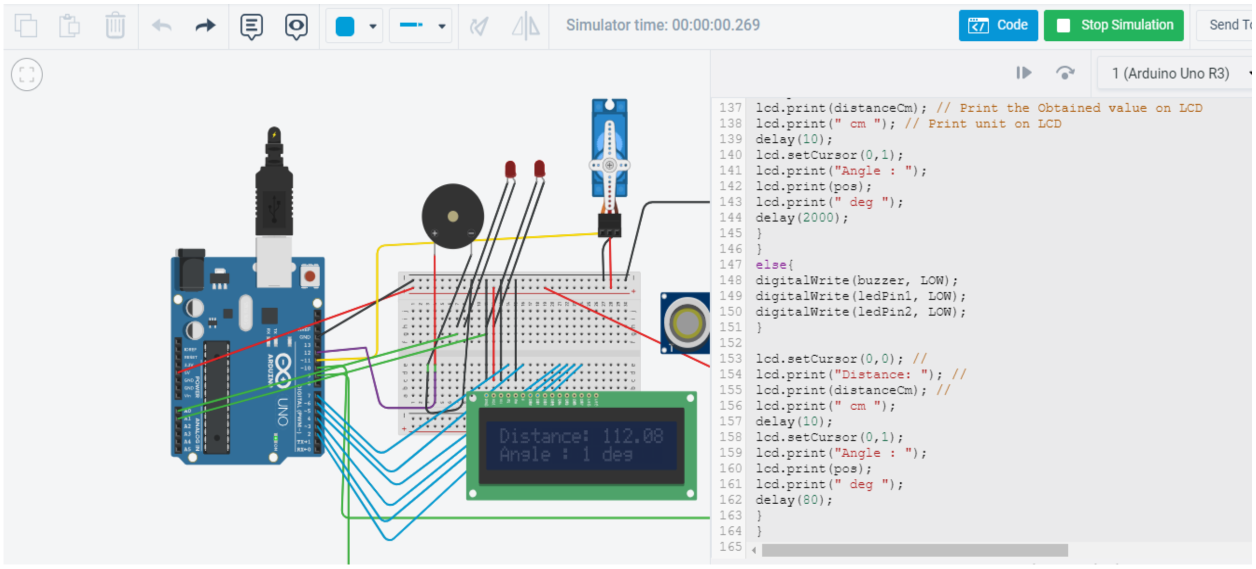
- Output and accuracy of the system
- The system's level of reliability

Unit Testing

At this point, each hardware device is tested individually using the code created for it in the Tinkercad simulation. This aids in authenticating the system's single planned unit.

System Testing

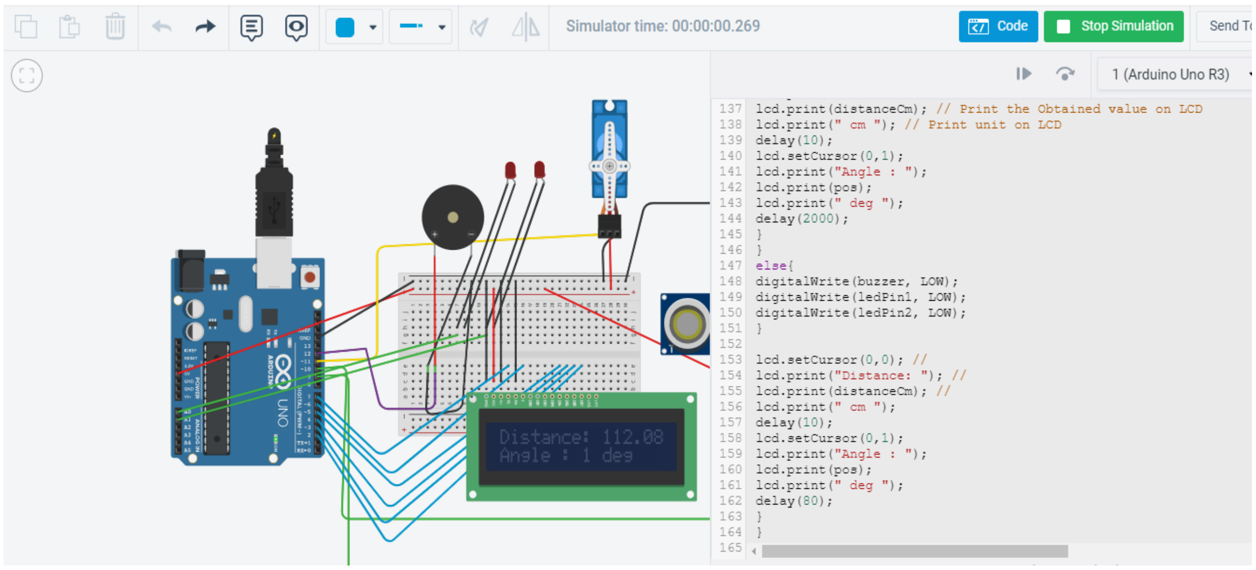
The primary objective of this was to verify the system's operation and efficiency on a full system display where all the parts were interconnected.



```
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
}
}
else{
digitalWrite(buzzer, LOW);
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, LOW);
}

lcd.setCursor(0,0); //
lcd.print("Distance: "); //
lcd.print(distanceCm); //
lcd.print(" cm ");
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(80);
}
}
```

Figure 4.2 Code Compilation

Screenshot For Results



```
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
}
}
else{
digitalWrite(buzzer, LOW);
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, LOW);
}

lcd.setCursor(0,0); //
lcd.print("Distance: "); //
lcd.print(distanceCm); //
lcd.print(" cm ");
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(80);
}
}
```

Figure 4.3 Screenshot Results

4.4 Discussion Of Findings

The LCD, buzzer, and ultrasonic sensor were all controlled by the Arduino, which served as the system's main board or microcontroller and served as their central processing unit. The calculated distance and rotational angle were shown on the LCD.

Throughout transmission, the sensor's pin ECHO signal remained in the HIGH condition allowing us to time the ultrasound's round journey and calculate the distance.

4.5 Conclusion

According to this research, the creation of an Arduino military radar model that uses an ultrasonic sensor for distance detection and angle finding using an Arduino Uno is successful and will result in the building of a system. The technology can keep an eye on a wide area and alarm the officials who use a buzzer. We accomplish this by connecting a microprocessor circuitry to a surveillance ultrasonic sensor installed on a servo motor.

To monitor the recognizing status, an LCD screen and buzzer are also connected. The radar constantly monitor the ultrasonic sensor echo as it examines the area.

CHAPTER 5: SUMMARY AND CONCLUSION

5.1 Introduction

The next section concludes the project. Studies are summarized, findings are discussed and inter preted. The results of this research were investigated. Recommendations for further research are at the end of this section.

5.2 Summary of Chapters

Chapter 5 of the Arduino Uno Ultrasonic Radar project report presents the results and analysis of the project. The chapter starts with a brief overview of the project goals and objectives, followed by a presentation of the final design of the system. The chapter then presents the results of the testing and analysis conducted on the system, which showed that the system was accurate and reliable in detecting objects and displaying their distance and position on the screen.

The chapter then concludes with a discussion of the project’s successes and limitations, along with recommendations for future improvements. The project was successful in achieving its goals and objectives, with the final design being functional and accurate. However, the system had limitations, such as the limited range of the ultrasonic sensor and the slow scanning speed of the servo motor. The chapter suggests several recommendations for future improvements, such as using a higher-range ultrasonic sensor and a faster servo motor. Overall, Chapter 5 provides a comprehensive summary of the project's results and analysis, highlighting its successes and limitations while presenting recommendations for future work.

5.3 Significance of Project

The Arduino Uno Ultrasonic Radar project is a significant development in the field of electronics and programming. It demonstrates the use of the Arduino Uno microcontroller and its capabilities in building a functional radar system. The project has several important implications and contributions:

1. **Educational Value**: The project provides a valuable resource for students, hobbyists, and professionals who are interested in learning about microcontroller programming and its applications in electronics. It can serve as a practical example of how to design and build

a functional radar system using commonly available components and open-source software.

2. **Low-Cost Alternative**: The project offers a low-cost alternative to commercial radar systems, which can be prohibitively expensive for many individuals and organizations. By using readily available components and open-source software, the Arduino Uno Ultrasonic Radar project can be replicated and customized according to specific needs and requirements.
3. **Practical Applications**: The project has practical applications in various fields, such as robotics, automation, and security. For example, the system can be used in robotics to detect obstacles and avoid collisions, or in security systems to detect intruders and trigger alarms.
4. **Innovation and Creativity**: The project encourages innovation and creativity by allowing individuals to customize and improve upon the design to suit their specific needs. It also provides a platform for experimenting with different components and programming techniques, which can lead to new discoveries and breakthroughs.

Overall, the Arduino Uno Ultrasonic Radar project has significant contributions to the fields of electronics, programming, and engineering. It provides a practical example of how to design and build a functional radar system, while also encouraging innovation and creativity in the process. It has potential applications in various industries and can serve as a valuable resource for individuals and organizations seeking low-cost alternatives to commercial radar systems.

5.4 Recommendations and Future Work

The Arduino Uno Ultrasonic Radar project has demonstrated the potential of using an Arduino microcontroller to create a functional radar system. However, there are still areas for improvement that can be explored in future work. Here are some recommendations for future work and improvements:

1. **Improving Sensitivity and Accuracy**: One area that can be improved in the radar system is its sensitivity and accuracy. This can be achieved by using a higher-quality

ultrasonic sensor or combining it with other sensors such as infrared or laser sensors for better range and accuracy.

2. **Increasing Range**: The range of the radar system can be increased by using a higher-frequency ultrasonic sensor that can detect objects at a greater distance. Alternatively, other sensors such as Lidar or Radar can be used to increase the range of the system.
3. **Enhancing Scanning Capabilities**: The servo motor used in the radar system can be replaced with a stepper motor or other types of motors that can provide faster and smoother scanning capabilities.
4. **Adding Data Logging Capabilities**: The system can be further enhanced by adding data logging capabilities that can collect and store data from the radar system for future analysis.
5. **Integrating Wireless Communication**: The system can be improved by integrating wireless communication capabilities such as Bluetooth or Wi-Fi that can allow remote access and control of the radar system.
6. **Developing User Interface**: A user-friendly interface can be developed for the system to make it easier to operate and understand. This can include developing a custom app or software for controlling and displaying the results from the radar system.

In conclusion, the Arduino Uno Ultrasonic Radar project has shown the potential of using an Arduino microcontroller to create a functional radar system. By making improvements in sensitivity, accuracy, range, scanning capabilities, data logging, and user interface, the radar system can be further enhanced and customized for a wide range of applications.

X

5.5 Conclusion

In conclusion, the Arduino Uno Ultrasonic Radar project has demonstrated the capabilities of a versatile microcontroller in creating a functional radar system. The project has shown that a low-cost alternative to commercial radar systems can be achieved by using commonly available components and open-source software. The project has also highlighted the potential applications of the system in various fields, including robotics, automation, and security.

The testing and analysis of the project showed that the system was accurate and reliable in detecting objects and displaying their distance and position on the screen. However, the project also revealed some limitations of the system, such as the limited range of the ultrasonic sensor and the slow scanning speed of the servo motor.

To further enhance the capabilities of the system, future work and improvements can be made in areas such as sensitivity, accuracy, range, scanning capabilities, data logging, and user interface. These improvements can make the system more versatile and customized for a wide range of applications.

Overall, the Arduino Uno Ultrasonic Radar project has significant contributions to the fields of electronics, programming, and engineering. It provides a valuable resource for students, hobbyists, and professionals who are interested in learning about microcontroller programming and its applications in electronics. The project also encourages innovation and creativity by allowing individuals to customize and improve upon the design to suit their specific needs.

References

1. Arduino. (2021). Arduino Uno. [online] Available at: https://www.arduino.cc/en/Guide/ArduinoUno [Accessed 18 Dec. 2022].

2. HC-SR04 Ultrasonic Sensor. (2021). HC-SR04 Ultrasonic Sensor. [online] Available at: https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/ [Accessed 23 Jan. 2023].

3. Servo Motor. (2021). Servo Motor. [online] Available at: https://www.arduino.cc/en/reference/servo [Accessed 14 Feb. 2023].

4. Arduino Radar Project. (2021). Arduino Radar Project. [online] Available at: https://circuitdigest.com/microcontroller-projects/arduino-radar-project [Accessed 23 Apr. 2023].

5. Parallax. (2021). Parallax Ping Ultrasonic Sensor. [online] Available at: https://www.parallax.com/product/28015 [Accessed 10 May. 2023].

6. Arduino Project Hub. (2021). Arduino Radar System Using Ultrasonic Sensor. [online] Available at: https://create.arduino.cc/projecthub/Abdullah_Saad_Farooq/arduino-radar-system-using-ultrasonic-sensor-fd5c8e [Accessed 23 Jan. 2023].

7. HowToElectronics. (2021). Arduino Ultrasonic Radar System. [online] Available at: https://how2electronics.com/arduino-ultrasonic-radar-system/ [Accessed 14 Apr. 2023].

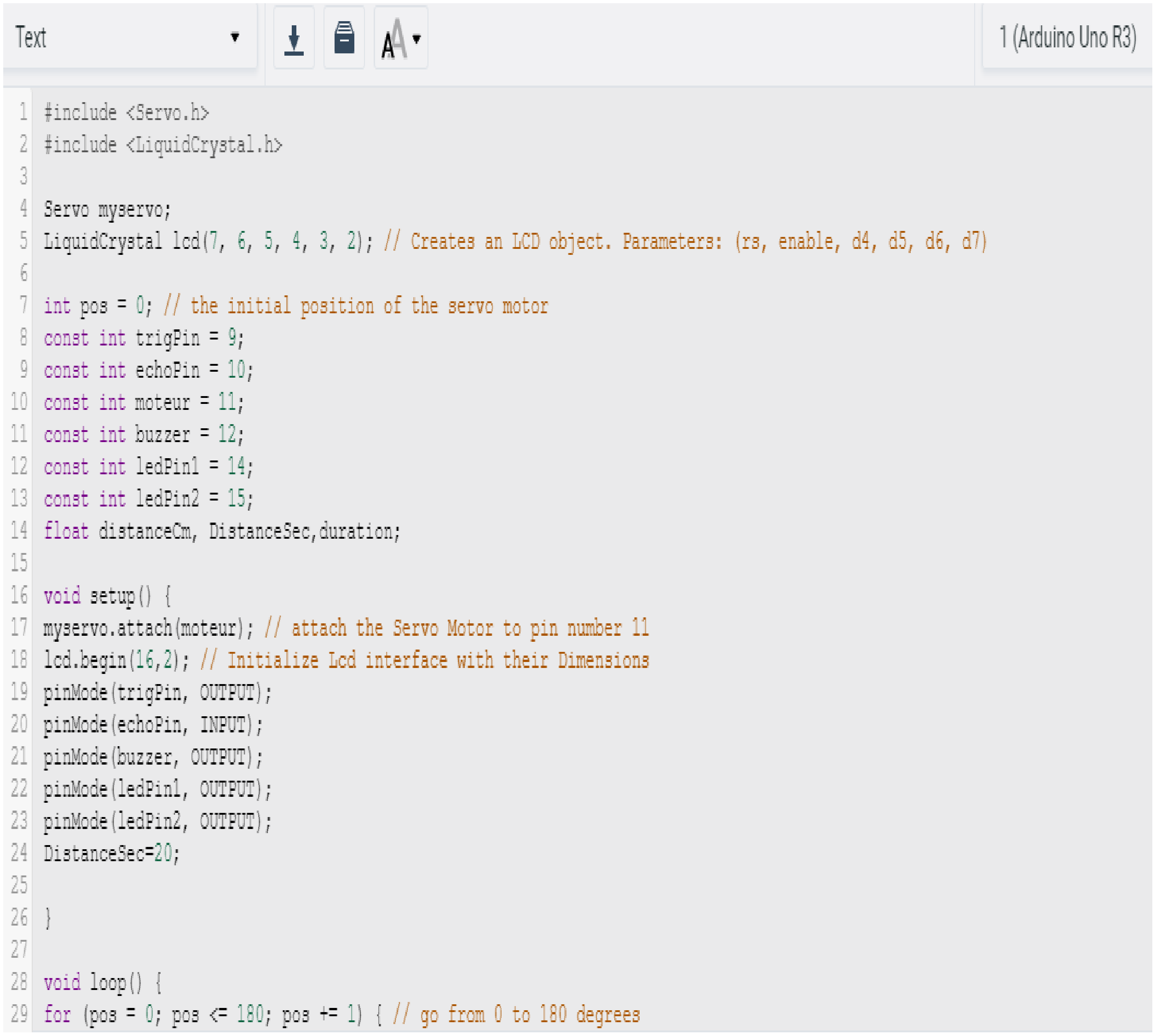
8. HC-SR04 Ultrasonic Sensor. (2021). HC-SR04 Ultrasonic Sensor. [online] Available at: https://how2electronics.com/hc-sr04-ultrasonic-sensor-working-pinout-specifications/ [Accessed 16 Feb. 2023].

9. Servo Motor. (2021). Servo Motor. [online] Available at: https://how2electronics.com/servo-motor-working-pinout-datasheet/ [Accessed 2 Mar. 2023].

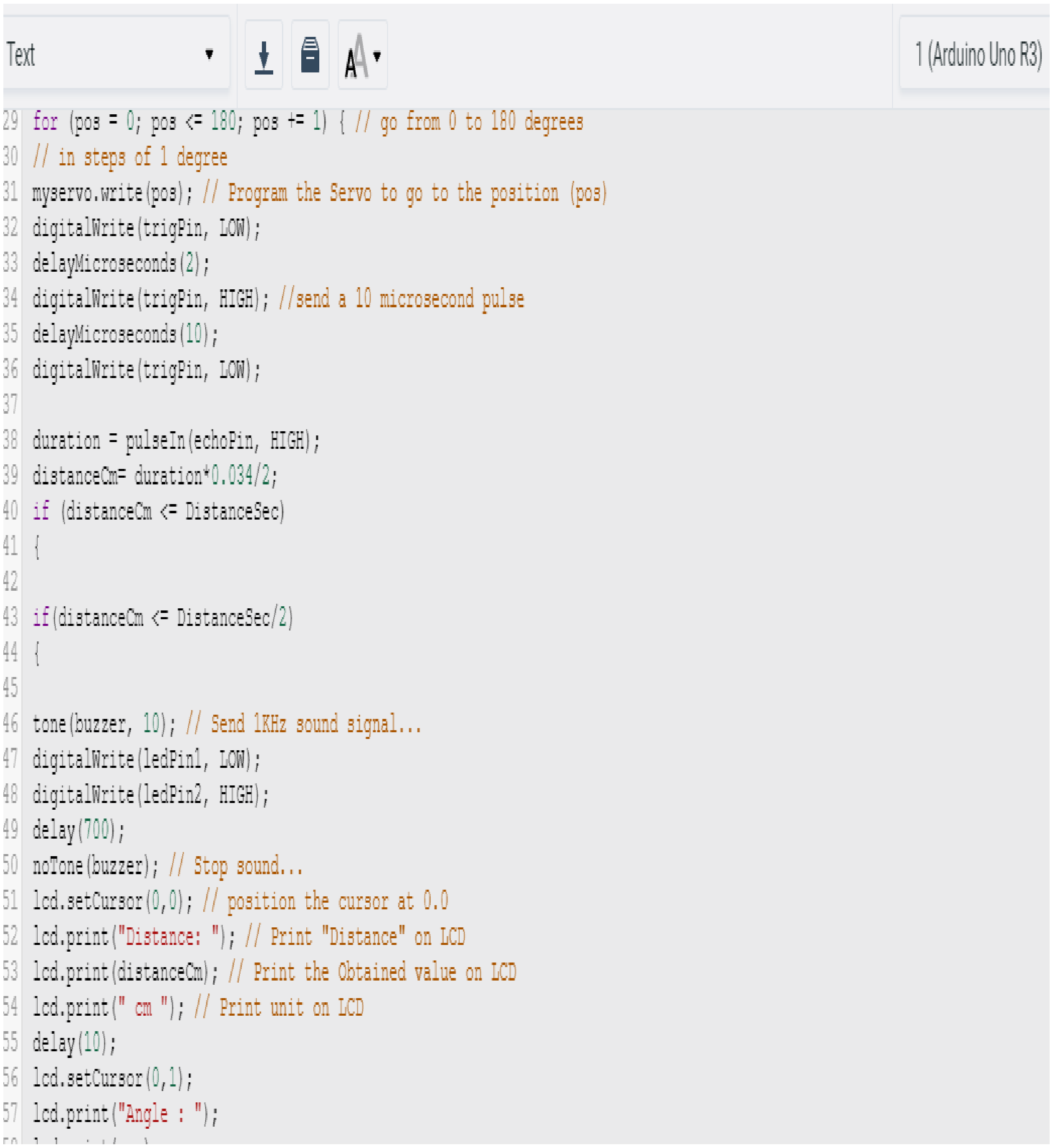
10. Arduino. (2021). Arduino Uno. [online] Available at: https://how2electronics.com/arduino-uno-board-pinout-specifications/ [Accessed 21 Jan. 2023].

11. HowToMechatronics. (2021). Arduino Radar System Using Ultrasonic Sensor. [online] Available at: https://howtomechatronics.com/projects/arduino-radar-system-using-ultrasonic-sensor/ [Accessed 17 Feb. 2023].

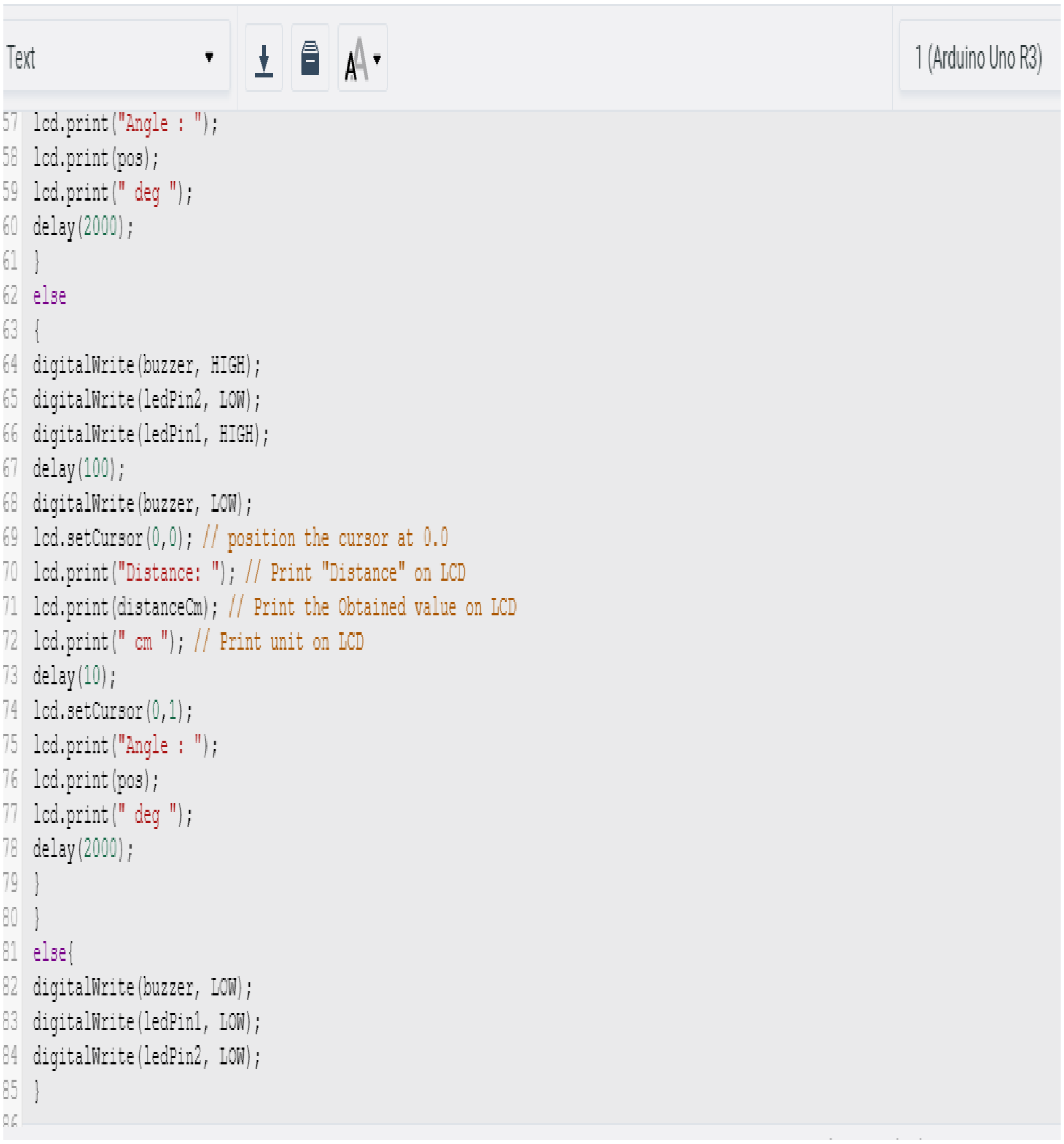
APPENDIX A: CODE SNIPPETS



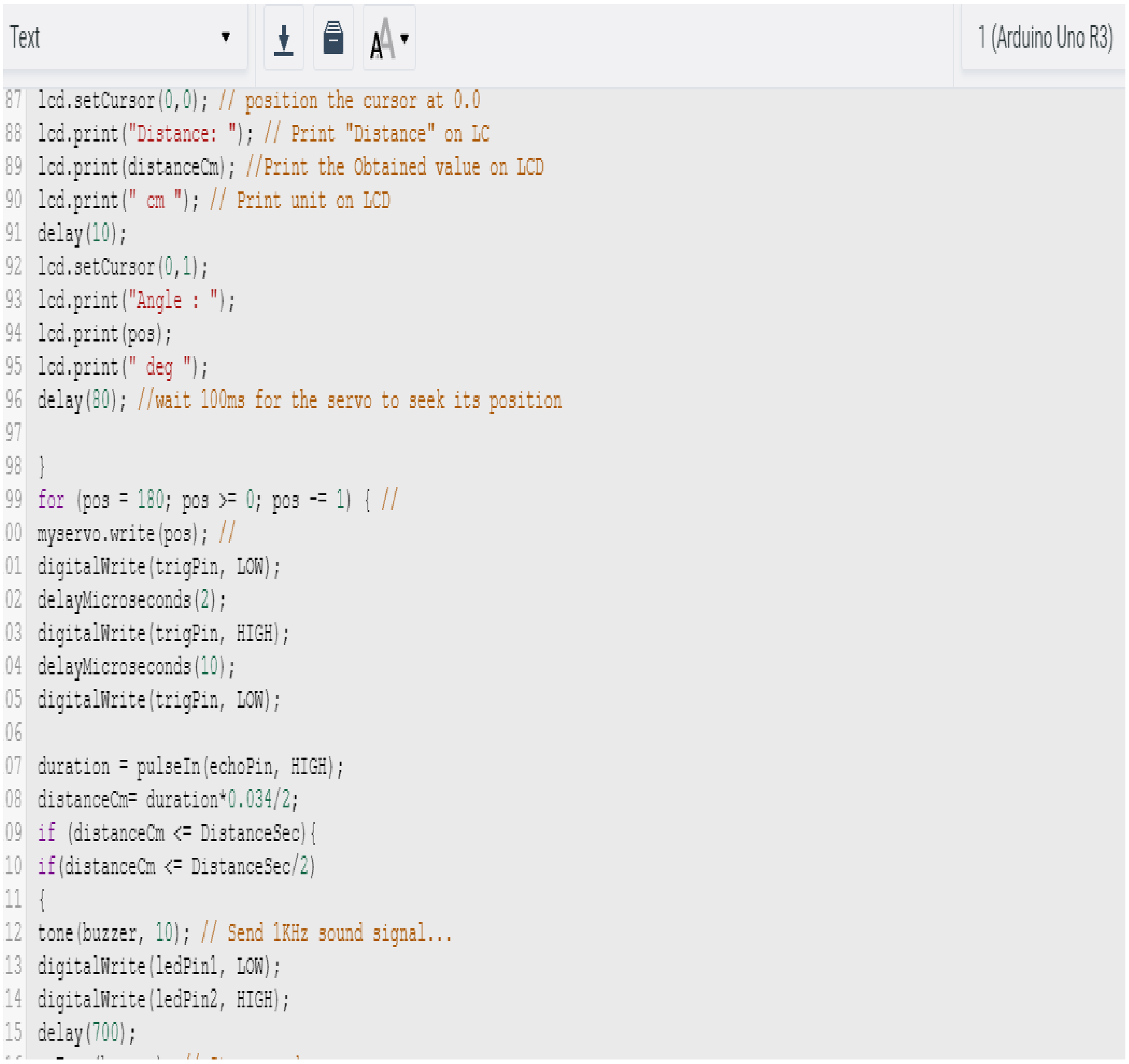
```
#include <Servo.h>
#include <LiquidCrystal.h>

Servo myservo;
LiquidCrystal lcd(7, 6, 5, 4, 3, 2); // Creates an LCD object. Parameters: (rs, enable, d4, d5, d6, d7)

int pos = 0; // the initial position of the servo motor
const int trigPin = 9;
const int echoPin = 10;
const int moteur = 11;
const int buzzer = 12;
const int ledPin1 = 14;
const int ledPin2 = 15;
float distanceCm, DistanceSec,duration;

void setup() {
myservo.attach(moteur); // attach the Servo Motor to pin number 11
lcd.begin(16,2); // Initialize Lcd interface with their Dimensions
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(buzzer, OUTPUT);
pinMode(ledPin1, OUTPUT);
pinMode(ledPin2, OUTPUT);
DistanceSec=20;

}

void loop() {
for (pos = 0; pos <= 180; pos += 1) { // go from 0 to 180 degrees
```



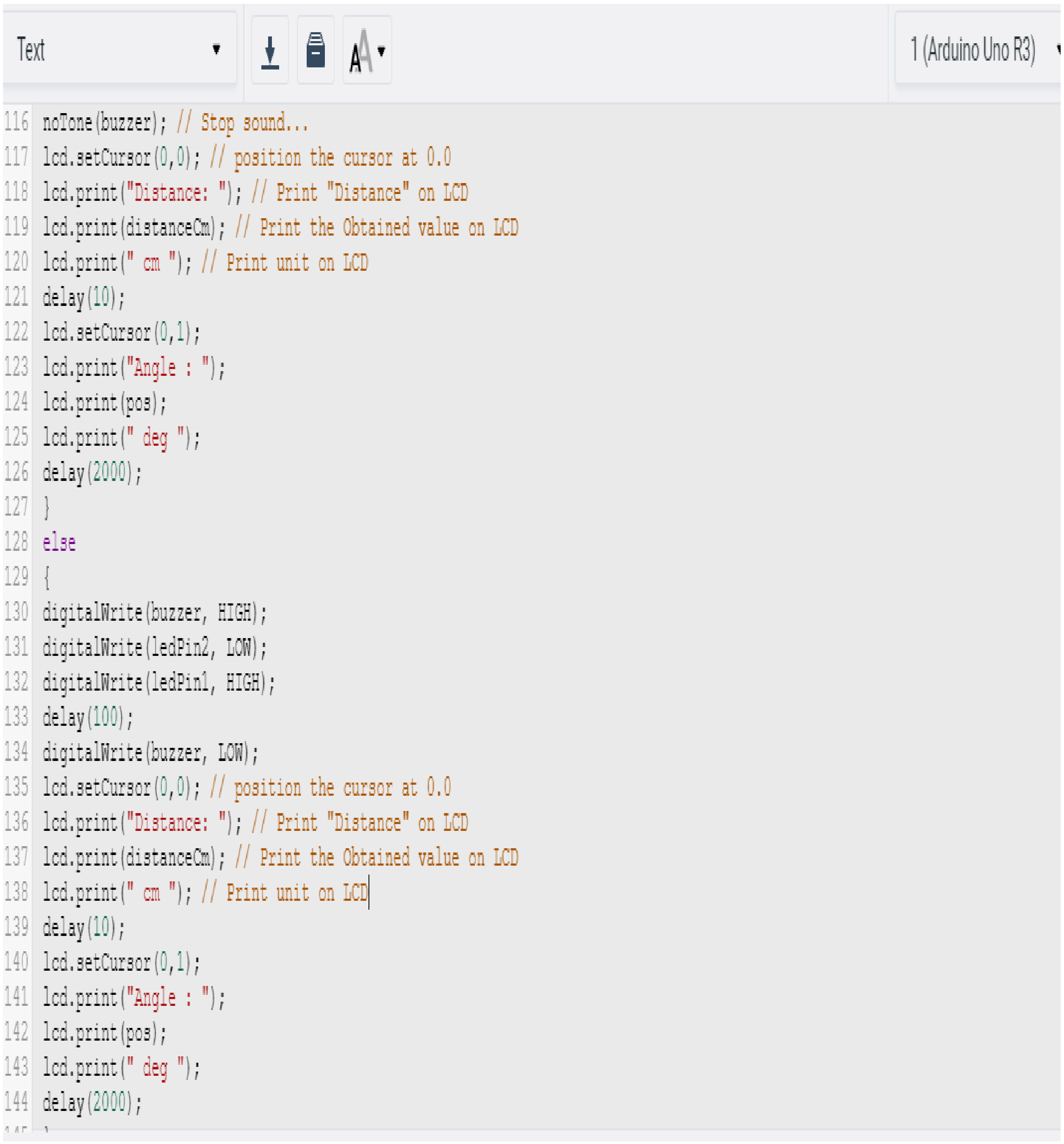
```
for (pos = 0; pos <= 180; pos += 1) { // go from 0 to 180 degrees
// in steps of 1 degree
myservo.write(pos); // Program the Servo to go to the position (pos)
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH); //send a 10 microsecond pulse
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);
distanceCm= duration*0.034/2;
if (distanceCm <= DistanceSec)
{

if(distanceCm <= DistanceSec/2)
{

tone(buzzer, 10); // Send 1KHz sound signal...
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, HIGH);
delay(700);
noTone(buzzer); // Stop sound...
lcd.setCursor(0,0); // position the cursor at 0.0
lcd.print("Distance: "); // Print "Distance" on LCD
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
```



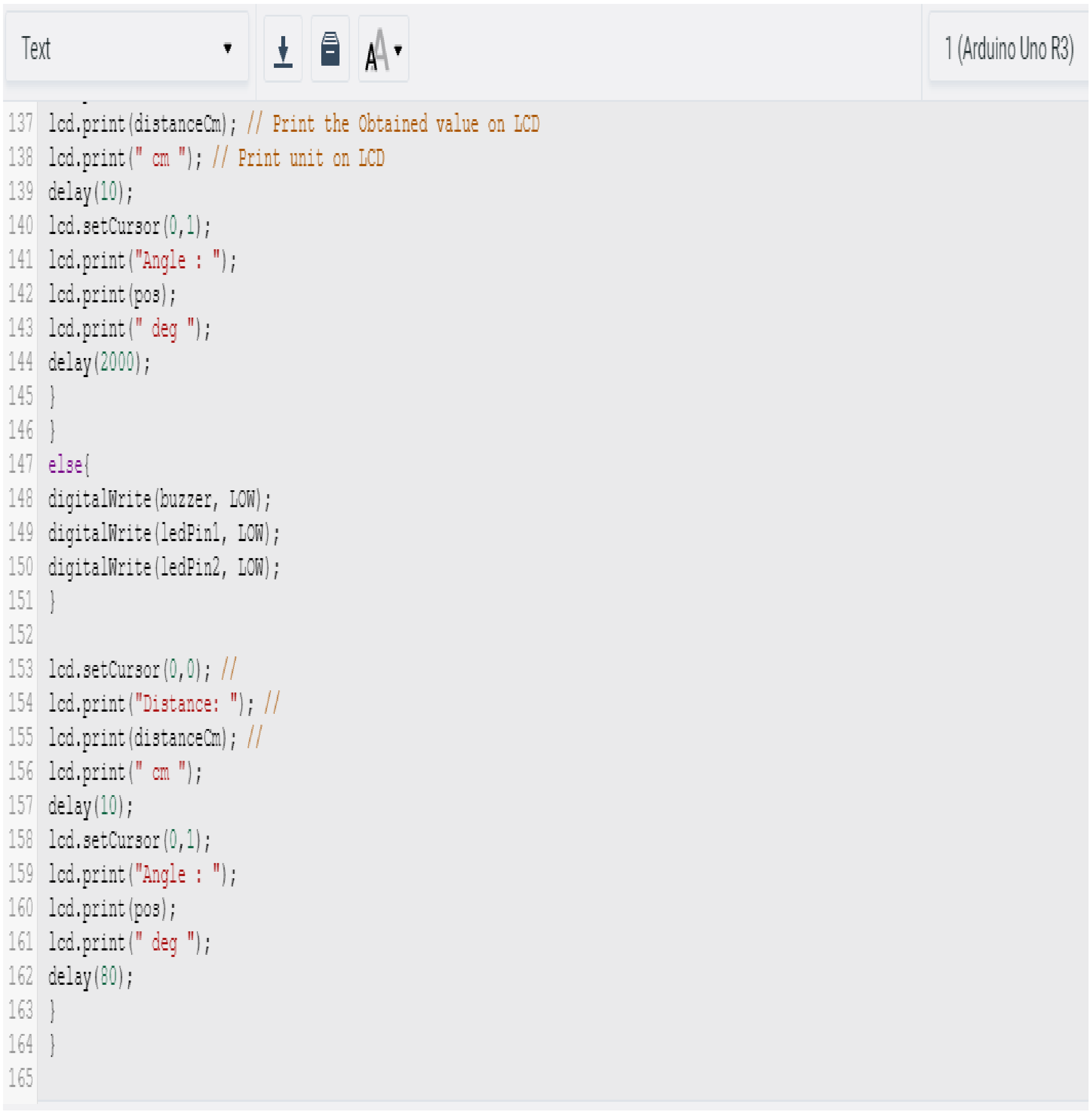
```
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
}
else
{
digitalWrite(buzzer, HIGH);
digitalWrite(ledPin2, LOW);
digitalWrite(ledPin1, HIGH);
delay(100);
digitalWrite(buzzer, LOW);
lcd.setCursor(0,0); // position the cursor at 0.0
lcd.print("Distance: "); // Print "Distance" on LCD
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
}
}
else{
digitalWrite(buzzer, LOW);
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, LOW);
}
```



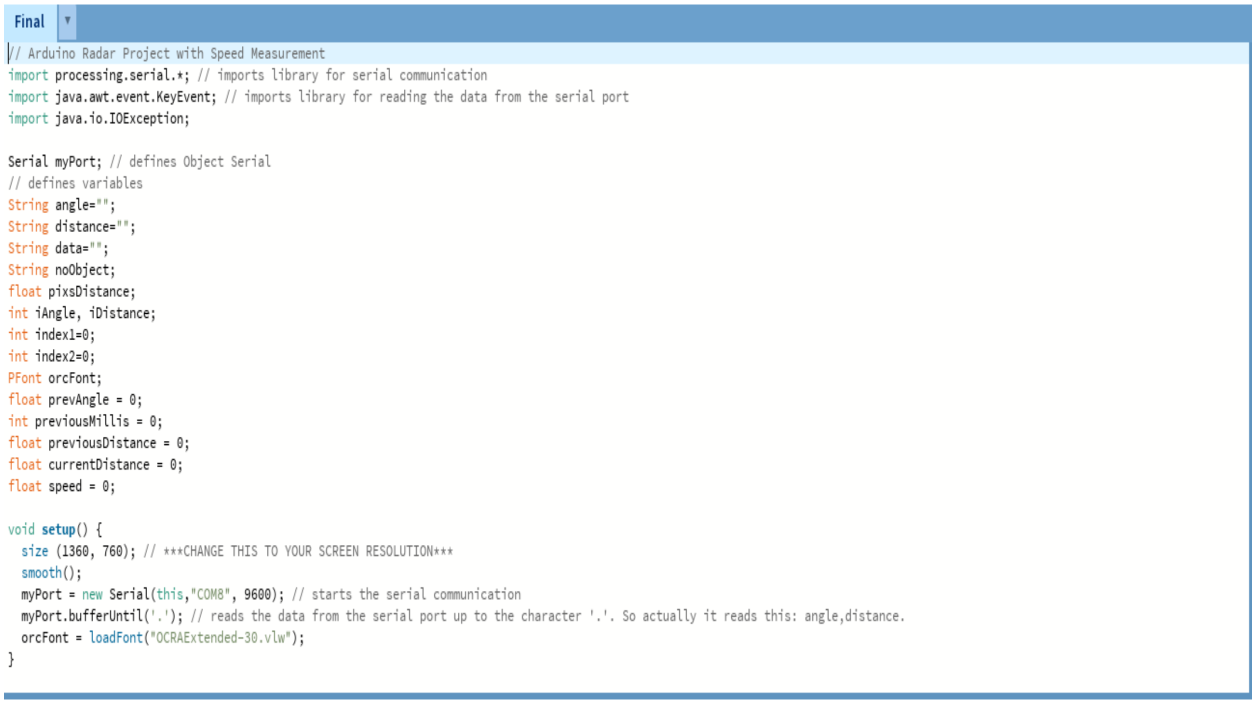
```
lcd.setCursor(0,0); // position the cursor at 0.0
lcd.print("Distance: "); // Print "Distance" on LC
lcd.print(distanceCm); //Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(80); //wait 100ms for the servo to seek its position

}
for (pos = 180; pos >= 0; pos -= 1) { //
myservo.write(pos); //
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);
distanceCm= duration*0.034/2;
if (distanceCm <= DistanceSec){
if(distanceCm <= DistanceSec/2)
{
tone(buzzer, 10); // Send 1KHz sound signal...
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, HIGH);
delay(700);
```



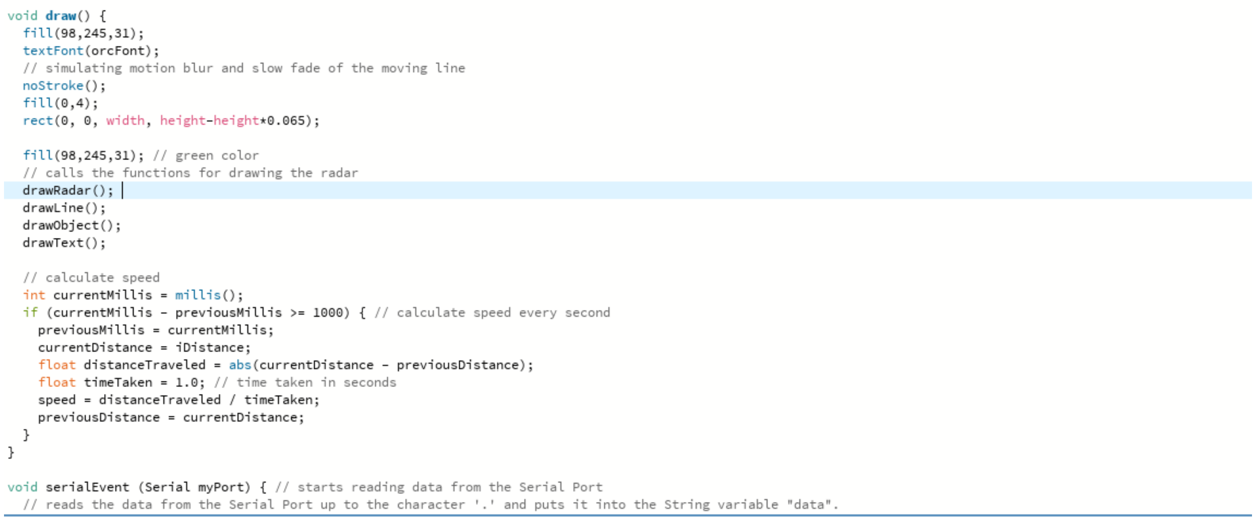
```
noTone(buzzer); // Stop sound...
lcd.setCursor(0,0); // position the cursor at 0.0
lcd.print("Distance: "); // Print "Distance" on LCD
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
}
else
{
digitalWrite(buzzer, HIGH);
digitalWrite(ledPin2, LOW);
digitalWrite(ledPin1, HIGH);
delay(100);
digitalWrite(buzzer, LOW);
lcd.setCursor(0,0); // position the cursor at 0.0
lcd.print("Distance: "); // Print "Distance" on LCD
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
```



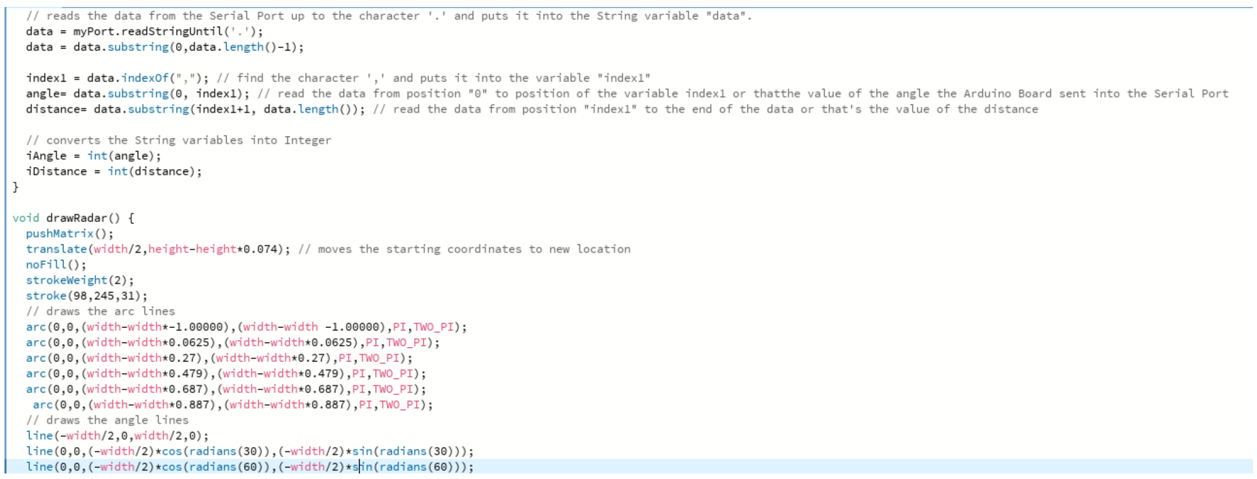
```
lcd.print(distanceCm); // Print the Obtained value on LCD
lcd.print(" cm "); // Print unit on LCD
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(2000);
}
}
else{
digitalWrite(buzzer, LOW);
digitalWrite(ledPin1, LOW);
digitalWrite(ledPin2, LOW);
}

lcd.setCursor(0,0); //
lcd.print("Distance: "); //
lcd.print(distanceCm); //
lcd.print(" cm ");
delay(10);
lcd.setCursor(0,1);
lcd.print("Angle : ");
lcd.print(pos);
lcd.print(" deg ");
delay(80);
}
}
```


Final

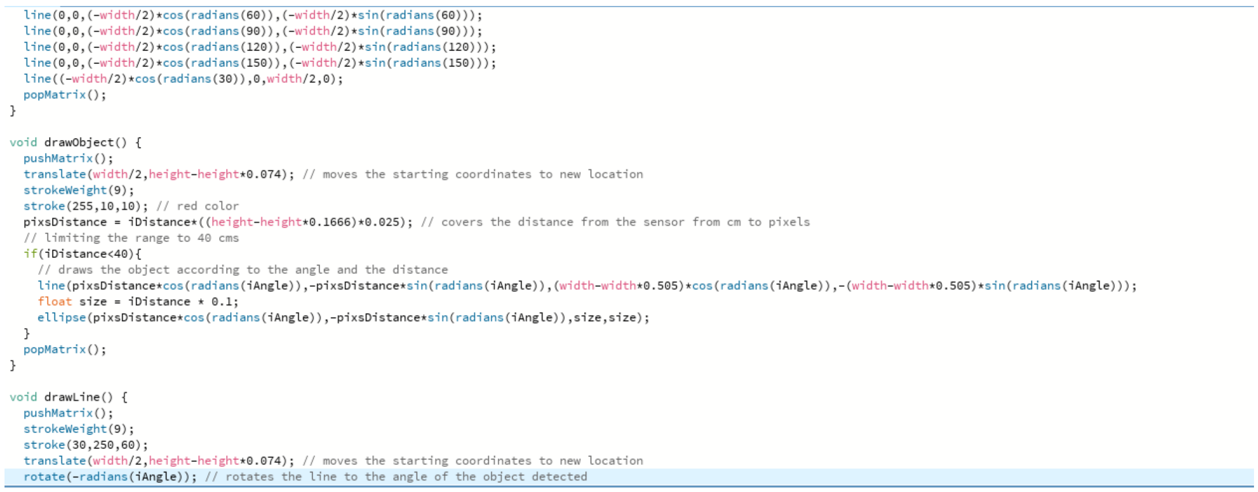
```
// Arduino Radar Project with Speed Measurement
import processing.serial.*; // imports library for serial communication
import java.awt.event.KeyEvent; // imports library for reading the data from the serial port
import java.io.IOException;

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

void setup() {
  size (1360, 760); // ***CHANGE THIS TO YOUR SCREEN RESOLUTION***
  smooth();
  myPort = new Serial(this,"COM8", 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.
  orcFont = loadFont("OCRAExtended-30.vlw");
}
```



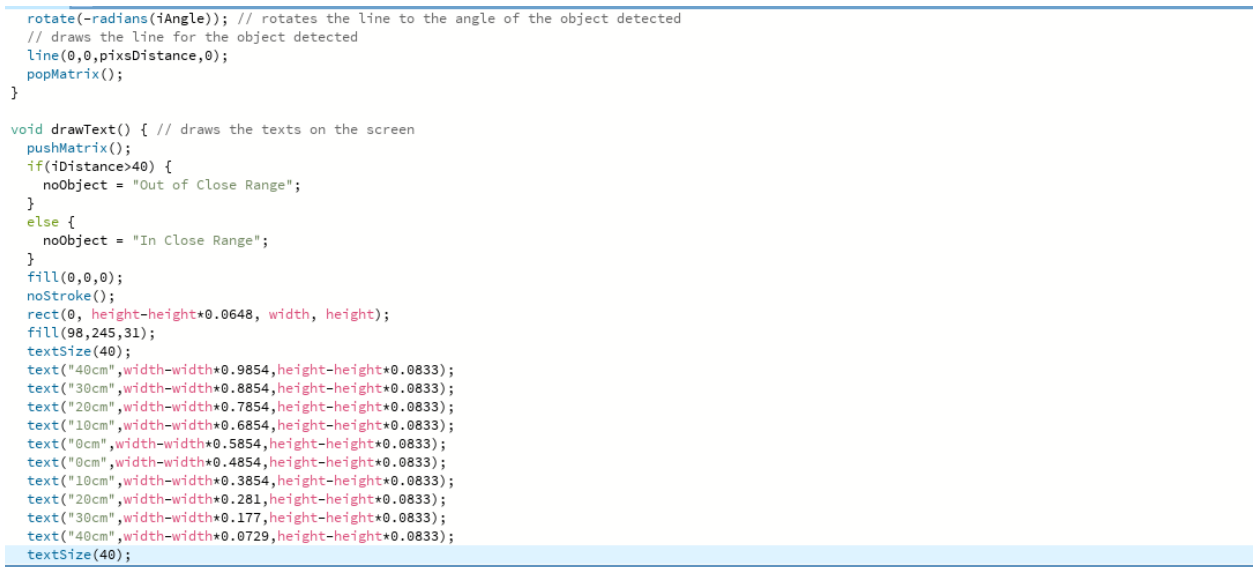
```
void draw() {
  fill(98,245,31);
  textFont(orcFont);
  // simulating motion blur and slow fade of the moving line
  noStroke();
  fill(0,4);
  rect(0, 0, width, height-height*0.065);

  fill(98,245,31); // green color
  // calls the functions for drawing the radar
  drawRadar();
  drawLine();
  drawObject();
  drawText();

  // calculate speed
  int currentMillis = millis();
  if (currentMillis - previousMillis >= 1000) { // calculate speed every second
    previousMillis = currentMillis;
    currentDistance = iDistance;
    float distanceTraveled = abs(currentDistance - previousDistance);
    float timeTaken = 1.0; // time taken in seconds
    speed = distanceTraveled / timeTaken;
    previousDistance = currentDistance;
  }
}

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".
```



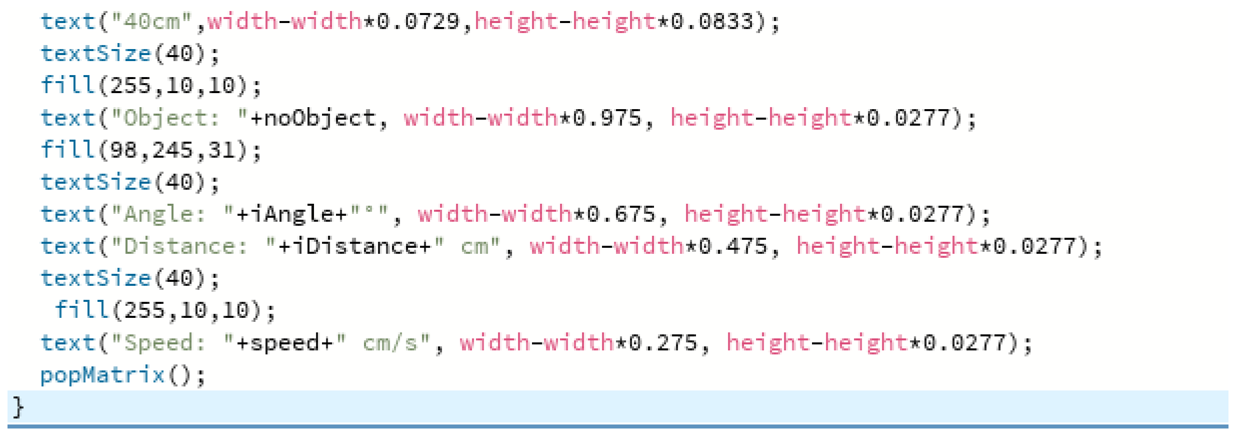
```
  // reads the data from the Serial Port up to the character '.' and puts it into the String variable "data".
  data = myPort.readStringUntil('.');
  data = data.substring(0,data.length()-1);

  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 thatthe 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 or that's 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 coordinates to new location
  noFill();
  strokeWeight(2);
  stroke(98,245,31);
  // draws the arc lines
  arc(0,0,(width-width*-1.00000),(width-width -1.00000),PI,TWO_PI);
  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);
   arc(0,0,(width-width*0.887),(width-width*0.887),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(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 coordinates 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)));
    float size = iDistance * 0.1;
    ellipse(pixsDistance*cos(radians(iAngle)),-pixsDistance*sin(radians(iAngle)),size,size);
  }
  popMatrix();
}

void drawLine() {
  pushMatrix();
  strokeWeight(9);
  stroke(30,250,60);
  translate(width/2,height-height*0.074); // moves the starting coordinates to new location
  rotate(-radians(iAngle)); // rotates the line to the angle of the object detected
```



```
  rotate(-radians(iAngle)); // rotates the line to the angle of the object detected
  // draws the line for the object detected
  line(0,0,pixsDistance,0);
  popMatrix();
}

void drawText() { // draws the texts on the screen
  pushMatrix();
  if(iDistance>40) {
    noObject = "Out of Close Range";
  }
  else {
    noObject = "In Close Range";
  }
  fill(0,0,0);
  noStroke();
  rect(0, height-height*0.0648, width, height);
  fill(98,245,31);
  textSize(40);
  text("40cm",width-width*0.9854,height-height*0.0833);
  text("30cm",width-width*0.8854,height-height*0.0833);
  text("20cm",width-width*0.7854,height-height*0.0833);
  text("10cm",width-width*0.6854,height-height*0.0833);
  text("0cm",width-width*0.5854,height-height*0.0833);
  text("0cm",width-width*0.4854,height-height*0.0833);
  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("40cm",width-width*0.0729,height-height*0.0833);
  textSize(40);
  fill(255,10,10);
  text("Object: "+noObject, width-width*0.975, height-height*0.0277);
  fill(98,245,31);
  textSize(40);
  text("Angle: "+iAngle+"°", width-width*0.675, height-height*0.0277);
  text("Distance: "+iDistance+" cm", width-width*0.475, height-height*0.0277);
  textSize(40);
   fill(255,10,10);
  text("Speed: "+speed+" cm/s", width-width*0.275, height-height*0.0277);
  popMatrix();
}
```

APPENDIX B: USER MANUAL

6.1 Introduction

Thank you for choosing the Arduino Military Radar Model using Ultrasonic Sensor for Distance, Detection and Angle Finding for Zimbabwe Defence Forces. This user manual is designed to guide you through the setup, operation, and maintenance of the system.

6.2 System Overview

The Arduino Military Radar Model using Ultrasonic Sensor for Distance, Detection and Angle Finding is a sophisticated system designed for use by the Zimbabwe Defence Forces. The system is built on the Arduino platform and uses ultrasonic sensors for distance measurement and angle finding. The system is capable of detecting objects within a range of up to 4 meters and can provide angle information for up to 180 degrees.

6.3 System Setup

The following steps should be followed to set up the system:

Step 1: Unpack the system and ensure that all components are present. The system should include an Arduino board, ultrasonic sensors, a breadboard, jumper wires, LCD and servo motor.

Step 2: Connect the ultrasonic sensors to the Arduino board using the jumper wires. The sensors should be connected to pins 8 and 9 on the board.

Step 3: Connect servo motor to pin 12 on the Arduino board.

Step 4: Connect the power source to the Arduino board.

Step 5: Upload the code to the Arduino board using the Arduino IDE.

Step 6: Upload the code to the Processing IDE.

6.4 System Operation

The following steps should be followed to operate the system:

Step 1: Power on the system by connecting the power source to the Arduino board.

Step 2: The system will start scanning for objects within its range. When an object is detected, the buzzer will light up.

Step 3: The system will also display the angle of the detected object on the Processing IDE.

Step 4: The system can be recalibrated by adjusting the position of the sensors.

6.5 System Maintenance

The following steps should be followed to maintain the system:

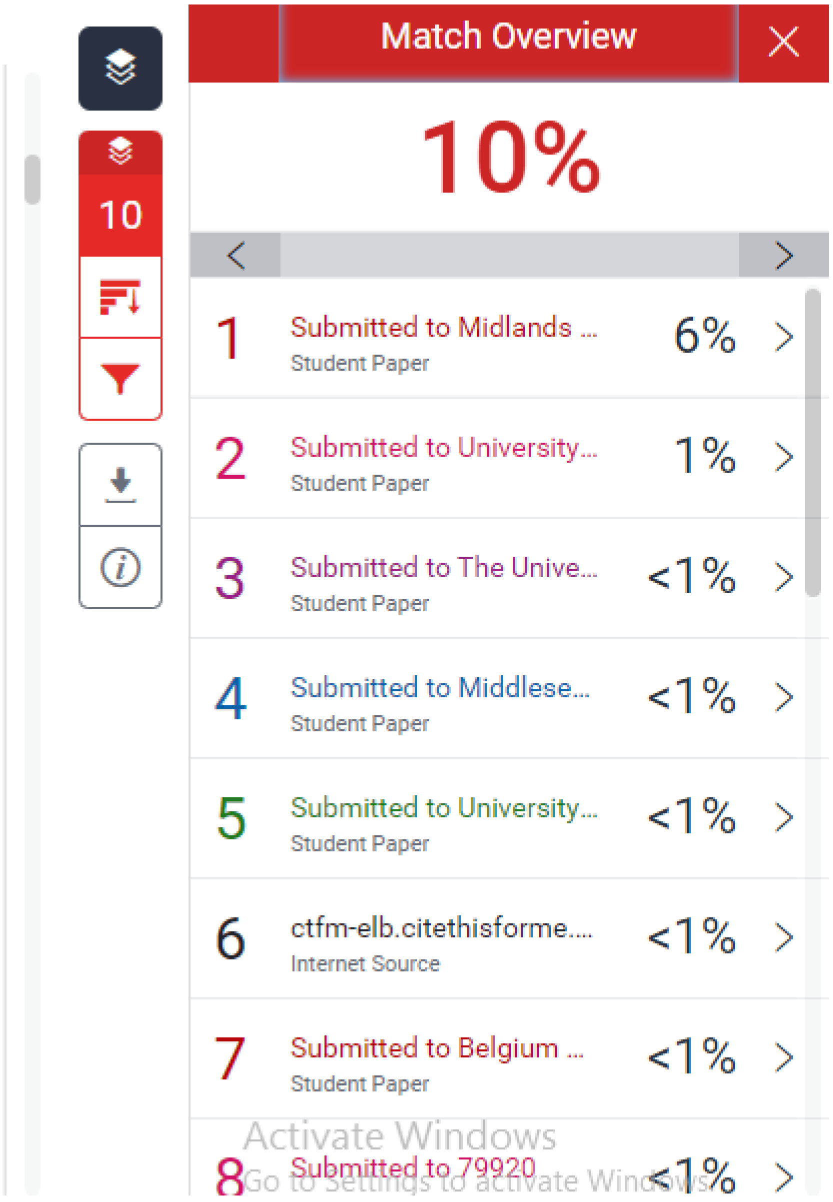
Step 1: Regularly check the connections between the sensors and the Arduino board to ensure they are secure.

Step 2: Check the power source and ensure that it is providing the correct voltage.

Step 3: Test the system regularly to ensure that it is functioning properly.

6.6 Conclusion

The Arduino Military Radar Model using Ultrasonic Sensor for Distance, Detection and Angle Finding for Zimbabwe Defence Forces is a powerful system that can provide accurate distance and angle information. By following the setup, operation, and maintenance guidelines outlined in this user manual, you can ensure that the system is functioning properly and providing accurate information.


Match Overview

10%

1	Submitted to Midlands ... Student Paper	6%	>
2	Submitted to University... Student Paper	1%	>
3	Submitted to The Unive... Student Paper	<1%	>
4	Submitted to Middlese... Student Paper	<1%	>
5	Submitted to University... Student Paper	<1%	>
6	ctfm-elb.citethisforme.... Internet Source	<1%	>
7	Submitted to Belgium ... Student Paper	<1%	>
8	Submitted to 79920	<1%	>