ANTI-COLLISION DETECTION ON AERIAL WORK PLATFORMS

A Project Report

Submitted in partial fulfillment of the
requirements for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

SURAJ ADKE

Seat Number: _____

Under the esteemed guidance of

Mr. SABIR SHAIKH

Assistant Professor, Department of Information Technology

DEPARTMENT OF INFORMATION TECHNOLOGY

VIDYALANKAR SCHOOL OF INFORMATION TECHNOLOGY

(Affiliated to University of Mumbai)

MUMBAI, 400 037

MAHARASHTRA

2018 - 2019

VIDYALANKAR SCHOOL OF INFORMATION TECHNOLOGY

(Affiliated to University of Mumbai)

MUMBAI-MAHARASHTRA-400037

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the project entitled, **"ANTI-COLLISION DETECTION ON AERIAL WORK PLATFORMS"**, is bonafied work of **SURAJ ADKE** bearing Seat No: _____ submitted in partial fulfilment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.

Internal Guide **Coordinator**

Internal Examiner **External Examiner**

Date: **College Seal** **Principal**

INSERT COLOR XEROX OF SEMESTER V CERTIFICATE

ABSTRACT

Many a times we encounter life threatening accidents involving cranes (aerial work platform) at locations such as construction sites. Severe accidents occur due to collisions of the crane with obstacles like tress, buildings and while transporting objects from one place to another. Hence safety is a priority. This project proposes an intelligent collision avoidance system mounted on an arm. It is made up of a servo motor placed on the arm and it rotates the Ultrasonic sensor from 0 to 180 degrees. Ultrasonic sensor is used to measure the distance between an obstacle and crane. The overall system is controlled by ARDUINO. A Graphical representation of the data from the Ultrasonic Sensor is represented in a Radar type display with the help of Processing IDE which is placed in front of the operator. If the Ultrasonic Sensor detects any object within its range, the same will be displayed graphically on the screen, thus helping the operator to know the distance between the crane and the obstacle. If the distance is less than a fixed threshold, the crane movement stops completely to avoid collision. The project thus prevents collisions and ensures safety of personnel and equipment.

ACKNOWLEDGEMENT

The success and final outcome of our project “ANTI-COLLISION DETECTION ON AERIAL WORK PLATFORMS” required a lot of guidance and assistance from many people and we are extremely privileged to have got this all along the completion of our project. All that we have done is only due to such supervision and assistance and we would not forget to thank them.

Firstly we are very thankful to our guide Prof. SABIR SHAIKH, for guiding us to do the project work on time and giving us all support and guidance which made us complete the project duly. We are extremely thankful to him for providing such a nice support and guidance.

We are also thankful to and fortunate enough to get constant encouragement, support and guidance from the teachers of Information Technology who helped us in successfully completing our project work.

This list would be incomplete without mentioning all of the developers and education institutes around the world that share their knowledge, work, and wisdom over the Internet.

DECLARATION

I hereby declare that the project entitled, "**ANTI-COLLISION DETECTION ON AERIAL WORK PLAFORMS**" done at Vidyalankar School of Information Technology, has not been in any case duplicated to submit to any other universities for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as final semester project as part of our curriculum.

Name and Signature of the Student

TABLE OF CONTENTS

List of Figures

Chapter 1
Introduction

Ultrasonic sensors measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Ultrasonic sensors measure the distance to the target by measuring the time between the emission and the reception**.** Ultrasonic sound waves are vibrations at a frequency above the range of human hearing (>20kHz) that can travel through a wide variety of medium (air or fluid) to detect objects and measure its distance without making physical contact.

Ultrasonic sensors can handle collision avoidance for a robot and being moved often, as long as it isn't too fast. They can be reliably implemented in grain bin sensing applications, water level sensing, drone application and sensing cars at your local drive-thru restaurant or bank. Ultrasonic rangefinders are commonly used as devices to detect a collision.

In a reflective model ultrasonic sensor, a single oscillator emits and receives ultrasonic waves alternately. This enables miniaturization of the sensor head.

Distance Calculation:

The distance can be calculated with the following formula:

Distance L = 1/2 × T × C

where L is the distance, T is the time between the emission and reception, and C is the sonic speed. The value is multiplied by 1/2 because T is the time for go-and-return distance.

Features:

1. Accurate Detection: Accurately detect objects with a resolution of 1 mm - 1 cm at

a range of 10 cm - 10 m.

2. System Flexibility: Detect near and far objects of various shapes, sizes, colors and transparencies through air or liquid.

3. Integrated Solution: Simplify your design with highly integrated system on chip (SoC) ultrasonic transducer drivers and signal conditioners.

1.1 BACKGROUND

Drivable aerial work platforms are common on many construction sites. These platforms facilitate working in high locations and increase productivity. However, since severe accidents involving aerial work platforms may occur due to collisions, the safety aspect should not be ignored.

The construction industry is naturally one with many hazards because of having to work at tall heights, do heavy lifting, operate or work around heavy vehicles and working in an environment where there are many things being moved around such as wheelbarrows, timber and bricks which one could potentially trip over.

Construction workers work in varied environments, in all weather conditions. Each day, on average, two construction workers die of work-related injuries. In fact, one in five workplace fatalities are construction-related. The top causes of construction-related fatalities are falls, struck-by an object, electrocution and caught between objects.

Many of such incidents resulted in one or more worker fatalities, and most of them resulted in multi-million dollar property loss, lawsuits, or settlements.

Generally, an employee cannot sue his or her employer for on-the-job injuries other than using the workers' compensation system. However, at times the owner of the premises, the general contractor or other contractors could be liable in cases where employees have been injured.

1.2 OBJECTIVES

1. To safeguard the equipment, ultrasonic sensors are attached to the appropriate points on the aerial work platform.
2. To Continuous monitoring of aerial work platforms.
3. To Increased safety for personnel and machine.
4. Collisions between the machine and obstacles such as trees, buildings, or persons can be prevented.
5. Increases in productivity.

1.3: Purpose, Scope and Applicability:

1.3.1 PURPOSE

The main purpose of the project is to create a collision avoidance system for cranes used in construction sites which detects obstacles and prevents collision.

1.3.2 SCOPE

This prototype can be implemented for various domains like construction sites to continuously monitor the aerial work platforms and in vehicles to prevent the accidents. Ultrasonic sensor is used to measure the distance between an obstacle. If the distance is more than a fixed threshold, the machine movement stops completely.

.1.3.3 APPLICABILITY

Applicability of the project anti-collision detection system is the construction industry. As many aerial work accidents happen around the world and many people

loose the lives in it, this system can certainly help to decrease the number of deaths that happen because of a aerial work accident.

Chapter 2
Survey of Technologies

A microcontroller (MCU for microcontroller unit, or UC for µ-controller) is a small computer on a single integrated circuit. In modern terminology, it is similar to, but less sophisticated than, a system on a chip (SoC); an SoC may include a microcontroller as one of its components. A microcontroller contains one or more CPUs (processor cores) along with memory and programmable input/output peripherals.

Figure 1: Microcontroller

LIMITATIONS OF MICROCONTROLLER:

- The microcontroller cannot interface high power devices directly.
- It has more complex structure as compared to microprocessor.
- It only performed limited number of executions simultaneously.
- It is generally used in micro equipment.
- Typically, micro-controller programs must fit in the available on-chip memory, since it would be costly to provide a system with external, expandable memory.

ARDUINO:

The Arduino Uno is a microcontroller board based on the ATmega328. It has

14 digital input/output pins (of which 6 can be used as PWM outputs),6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

Figure 2: Arduino

ADVANTAGES OF ARDUINO OVER MICROCONTROLLER:

- When you need more control and are actually thinking on converting your prototype into a real product, then yes, you need to get deep down into the microcontroller and get rid of all the excess fat, trim the circuit to just the bare bones, optimize the code, etc.
- For prototyping, the Arduino platform gives you a lot of pre-wiring and free code libraries that will let you concentrate on testing your idea instead of spending your time building supporting circuitry or writing tons of low level code.
- Nearly instantaneous start (plug in a USB cord and load an example program and you can see something work).
- A huge community of people working in the same environment.
- A large assortment of included libraries for interfacing to a wide range of

hardware.

- Ease of use. The Arduino Uno has built in pin outs for providing you with 5v, 3.3V, ground, analog input, digital output, SPI, I2C which comes in handy.

ULTRASONIC SENSOR:

The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. From 2cm to 400 cm or 1” to 13 feet.

Figure 3: Ultrasonic Sensor

ADVANTAGES OF ULTRASONIC SENSOR:

1. The ultrasonic sensor has high frequency, high sensitivity and high penetrating power therefore it can easily detect the external or deep objects.
2. These sensors easily interface with microcontroller or any type of controller
3. These sensors have greater accuracy then other methods for measuring the

thickness and depth of parallel surface.

4. These sensors could easily sense the nature, shape and orientation of that specific objects which is within the area of these sensors.
5. There sensors are easy to use, not dangerous during operation for nearby objects, person, equipment or material.

SERVO MOTOR:

A servomotor is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback.

Figure 4: Servo Motor

ADVANTAGES OF SERVO MOTOR:

- If a heavy load is placed on the motor, the driver will increase the current to the motor coil as it attempts to rotate the motor. Basically, there is no out-of-step condition.
- High-speed operation is possible.

Chapter 3
Requirements and Analysis

3.1 PROBLEM DEFINITION:

WORKING OF CURRENT SYSTEM:

- An aerial work platform (AWP), also known as an aerial device, elevating work platform (EWP), bucket truck or mobile elevating work platform (MEWP) is a mechanical device used to provide temporary access for people or equipment to inaccessible areas, usually at height. There are distinct types of mechanized access platforms and the individual types may also be known as a "cherry picker" or a "scissor lift".
- They are generally used for temporary, flexible access purposes such as maintenance and construction work or by firefighters for emergency access, which distinguishes them from permanent access equipment such as elevators. They are designed to lift limited weights — usually less than a ton, although some have a higher safe working load (SWL)— distinguishing them from most types of cranes. They are usually capable of being set up and operated by a single person.
- Regardless of the task they are used for, aerial work platforms may provide additional features beyond transport and access, including being equipped with electrical outlets or compressed air connectors for power tools. They may also be equipped with specialist equipment, such as carrying frames for window glass. Under bridge units are also available to lift operators down to a work area.

LIMITATION OF THE CURRENT SYSTEM:

- Equipments may get damage.
- Hazardous for human life.
- Machine may get damage.
- Collisions may occur between the machine and obstacles such as trees, buildings, or persons.

- Effects on productivity.
- Not properly monitoring of aerial work platforms.
- Large amount of accidents may arise because of mistake of driver who is monitoring the crane.

SOLUTIONS PROVIDED:

- To safeguard the equipment, ultrasonic sensors are attached to the appropriate points on the aerial work platform.
- Continuous monitoring of aerial work platforms.
- Increased safety for personnel and machine.
- Collisions between the machine and obstacles such as trees, buildings, or persons can be prevented.
- Increase the productivity.
- If the crane is to close to the obstacle then entire working of crane will automatically gets stop, in this way collision gets prevented.
- Driver of crane know how far our crane is from the obstacle by just seeing the screen which is fitted in front of him/her.

3.2 REQUIREMENTS SPECIFICATIONS

- Microcontroller: Simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. You can tell your board what to do by sending a set of instructions to the microcontroller on the board.
- Sensor: A sensor is a device, module, or subsystem whose purpose is to detect events or changes in its environment and send the information to other electronics, frequently a computer processor.
- Motor: A motor is a device that changes a form of energy into mechanical energy to produce motion.
- Car: Vehicle that is teleoperated by a means that does not restrict its motion with an origin external to the device.
- Laptop: A laptop, also called a notebook computer or simply a notebook, is a small, portable personal computer with a "clamshell" form factor, having, typically, a thin LCD or LED computer screen mounted on the inside of the upper lid of the "clamshell" and an alphanumeric keyboard on the inside of the lower lid.
- Wires: The purpose of the wires in a series circuit is to allow the electricity to flow from one device to the next. Wire is used to carry the flow of electrons.

3.3 PLANNING AND SCHEDULING

We are using Prototyping Model. The Prototyping Model is a systems development method (SDM) in which a prototype (an early approximation of a final system or product) is built, tested, and then reworked as necessary until an acceptable prototype is finally achieved from which the complete system or product can now be developed.

This model works best in scenarios where not all of the project requirements are known in detail ahead of time. It is an iterative, trial-and-error process that takes place between the developers and the users.

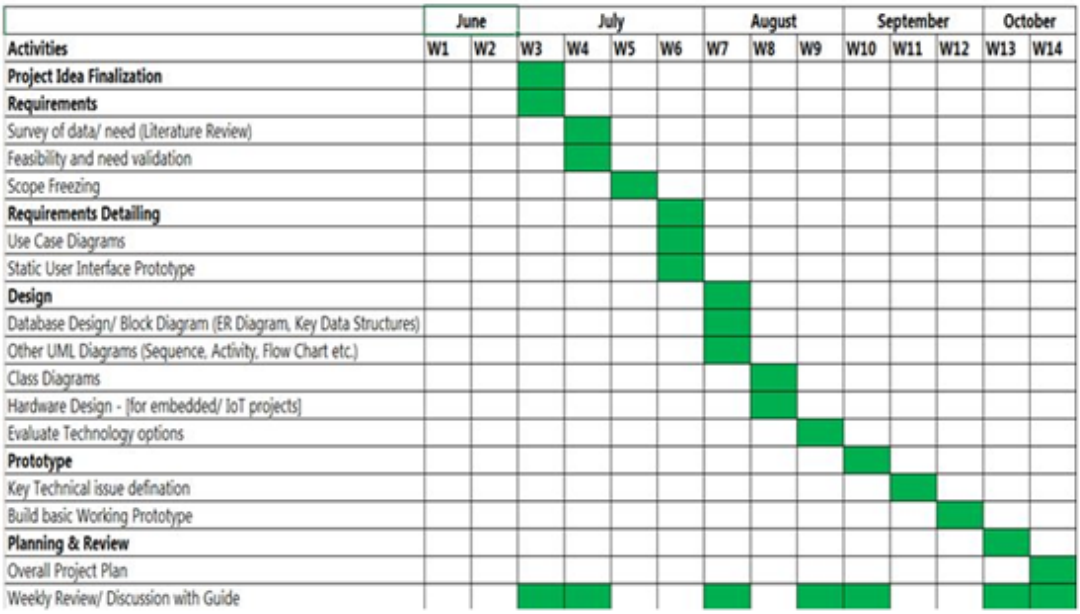


Figure 5: Gantt Chart

3.4 SOFTWARE AND HARDWARE REQUIREMENTS

HARDWARE SPECIFICATION:

Arduino UNO

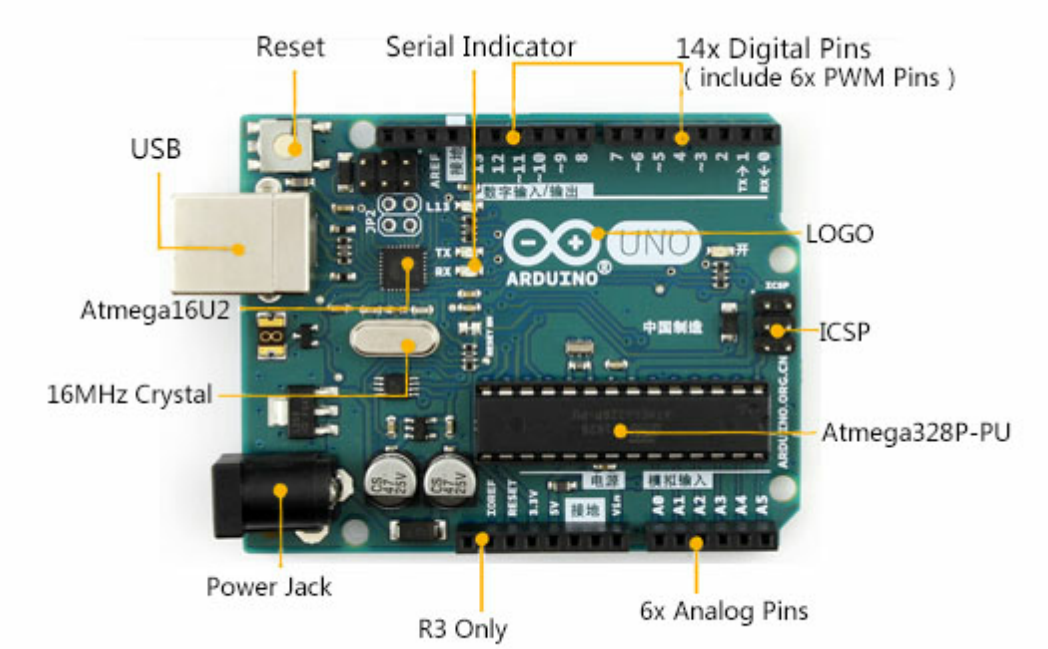


Figure 6: Arduino

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC

adapter or battery to get started.

You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

The Arduino project provides the Arduino integrated development environment (IDE), which is a cross – platform application written in the programming language Java. It originated from the IDE for the languages Processing and Wiring . It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and provides simple one – click mechanism to compile and load programs to an Arduino board. A program written with the IDE for Arduino is called a "sketch".

Ultrasonic Sensor HC-SR04

Vcc- Connects to 5V of positive voltage for power
Trig- A pulse is sent here for the sensor to go into ranging mode for object detection
Echo- The echo sends a signal back if an object has been detected or not. If a signal is returned, an object has been detected. If not, no object has been detected.
GND- Completes electrical pathway of the power.

Figure 7: Ultrasonic Sensor HC-SR04

The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. From 2cm to 400 cm or 1” to 13 feet. Its operation is not affected by sunlight or black material like sharp rangefinders are (although acoustically soft materials like cloth can be difficult to detect). It comes complete with ultrasonic transmitter and receiver module.

USB Cable

Figure 8: USB Cable

USB cable is used to insert the code written in Arduino IDE in the Arduino board.

Male to Female wires, Wires, Male to Male wires

Male to female jumper wires

Jumper wire male to female, used in connecting female header pin of any development board (like Arduino) to other development board having male connector. In this one end of the wire is male connector and other one is female. The length of the wire is 20 cm approx.

Figure 9: Male to Female Jumper Wires

Male to male jumper wires

In male to male both the ends have male connectors. They are multipurpose use and very handy. Generally used for connecting FRC pins, Header pins, Berg pins etc. Male connector at both the ends. It's Length id about 18cm (approx.)

Figure 10: Male to Male Jumper Wires

Servo Motor SG-90

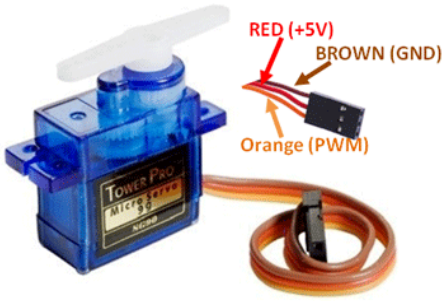


Figure 11: Servo Motor SG-90

A servomotor is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servo motors. A servomotor is a closed-loop servomechanism that uses position feedback to control its motion and final position. The input to its control is a signal (either analogue or digital) representing the position commanded for the output shaft. Servomotors are used in applications such as robotics, CNC machinery or automated manufacturing.

Wires

A wire is a single, usually cylindrical, flexible strand or rod of metal. Wires are used to bear mechanical loads or electricity and telecommunications signals. Wire is commonly formed by drawing the metal through a hole in a die or drawplate.

Figure 12: Wires

Chassis

A chassis is the internal framework of an artificial object, which supports the object in its construction and use. An example of a chassis is a vehicle frame the under part of a motor vehicle, on which the body is mounted

Figure 13: Chassis

Wheels

Wheels are used to move the prototype.

Figure 14: Wheels

SOFTWARE SPECIFICATION:

ARDUINO IDE:

A program for Arduino may be written in any programming language with compilers that produce binary machine code for the target processor. Atmel provides a development environment for their microcontrollers, AVR Studio and the newer Atmel Studio. The Arduino project provides the Arduino integrated development environment(IDE), which is a cross-platform application written in the programming language Java. A program written with the IDE for Arduino is called a sketch.

Sketches are saved on the development.

Computer as text files with the file extension .ino. Arduino Software (IDE) pre-1.0 saved sketches with the extension .pde.

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU tool chain, also included with the IDE distribution.

Program structure

A minimal Arduino C/C++ program consist of only two functions:

- setup(): This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
- loop(): After setup() has been called, function *loop()* is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.

Processing IDE:

Processing is an open-source graphical library and integrated development environment (IDE) / playground built for the electronic arts, new media art, and visual design communities with the purpose of teaching non-programmers the fundamentals of computer programming in a visual context. Processing uses the Java Language, with additional simplifications such as additional classes and aliased mathematical functions and operations. As well as this, it also has a

graphical user interface for simplifying the compilation and execution stage.

The Processing language and IDE were the precursor to numerous other projects, notably Arduino, Wiring and P5.js.Processing includes a sketchbook, a minimal alternative to an integrated development environment (IDE) for organizing projects. Every Processing sketch is actually a subclass of the P applet Java class (formerly a subclass of Java's built-in Applet) which implements most of the Processing language's features. When programming in Processing, all additional classes defined will be treated as inner classes when the code is translated into pure Java before compiling. This means that the use of static variables and methods in classes is prohibited unless Processing is explicitly told to code in pure Java mode. Processing also allows for users to create their own classes within the P Applet sketch. This allows for complex data types that can include any number of arguments and avoids the limitations of solely using standard data types such as:int(integer),char(character),float(real number) and color(RGB, hex).

3.5 Preliminary Product Description:

Functional Requirements:

How the system should react to particular inputs and how the system should behave in particular situations:

- If ultrasonic sensor detects any obstacle which is close to the crane then it will show graph on to the screen.

- If the obstacle is too close to the crane & driver does not take any actions then the entire system will stop automatically.

What the system should not do:

- System should not work improperly.
- System must not exceed a specified time to generate a response.

Non-functional Requirements:

Usability:

Frequently used functions should be tested for usability.

Reliability:

Users have to trust the system.

Performance:

Think of stress periods.

Supportability:

The system needs to be cost-effective to maintain.

Functions and operation of the system:

The prototype consists of a car on which an arm is fitted. The servo motor which is placed on the arm rotates the sensor from 0 degree to 180 degrees. Ultrasonic sensor is used to measure the distance between an obstacle. If the distance is more than a fixed threshold, the machine movement stops completely.

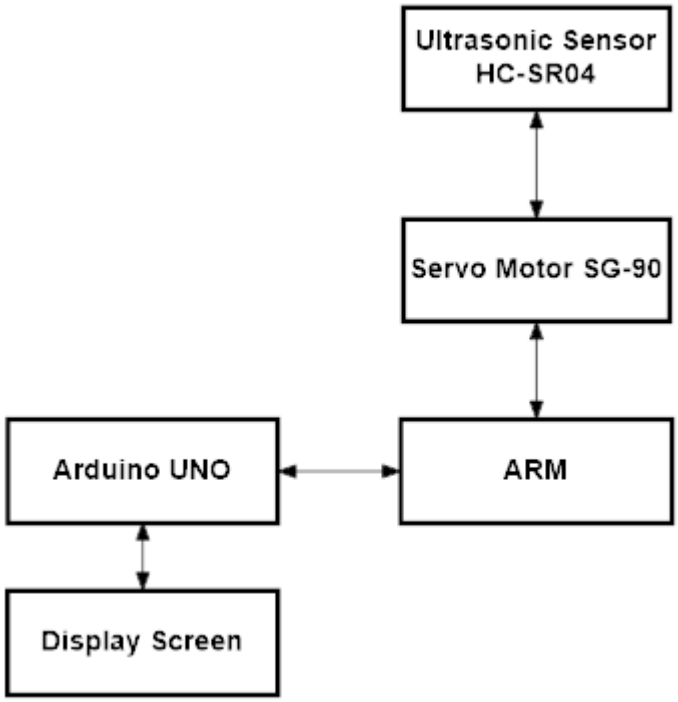
3.6 CONCEPTUAL MODELS:

The project Anti-Collision Detection on Aerial Work Platforms proposes an intelligent collision avoidance system as a prototype which avoids accidents in aerial work platforms like construction sites. There are different types of cranes used in construction sites like Telescopic Crane, Tower Crane, Mobile Crane, Rough Terrain Crane, Loader Crane. Many several technology and innovations are available

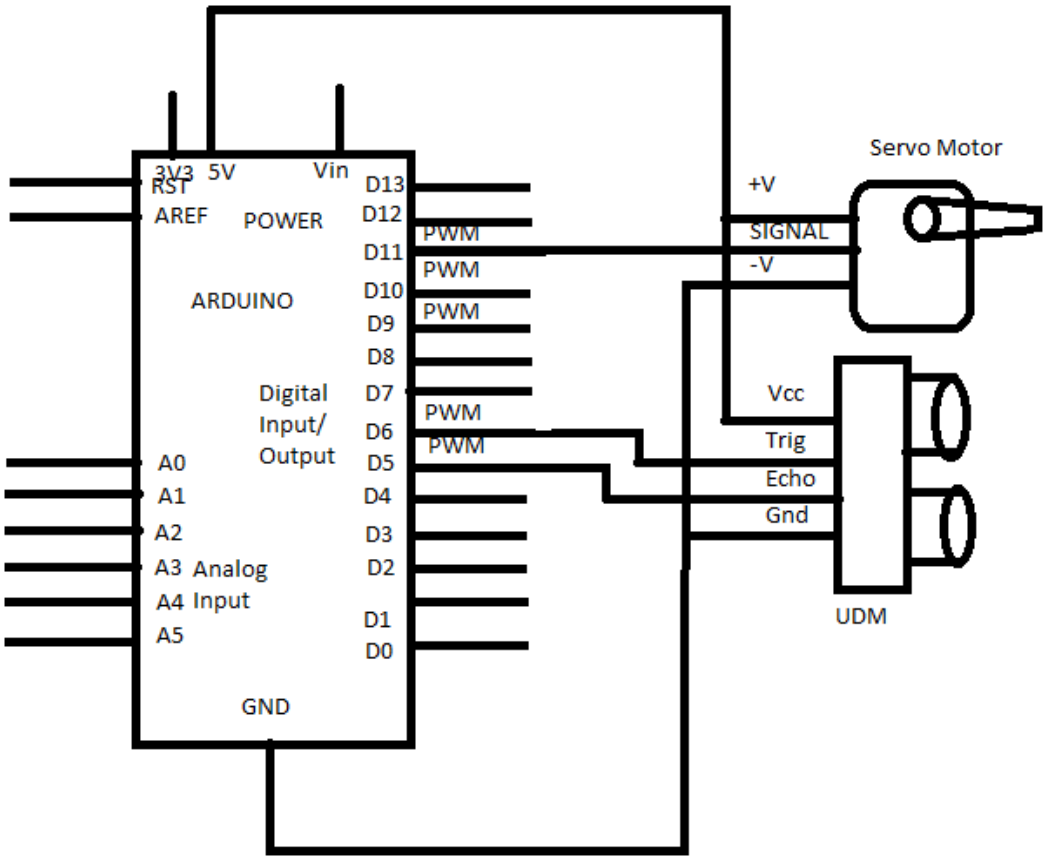
for crane safety. The prototype consists of a car on which an arm is fitted. The servo motor which is placed on the arm rotates the sensor from 0 degree to 180 degrees. Ultrasonic sensor is used to measure the distance between an obstacle. If the distance is more than a fixed threshold, the machine movement stops completely. Driver of crane know how far our crane is from the obstacle by just seeing the screen which is fitted in front of him/her. In this way collision will be prevented in construction sites. It helps to increase the productivity and safety of human life as well as machinery. So we can continuously monitor the aerial work platforms.

Chapter 4
System Designs

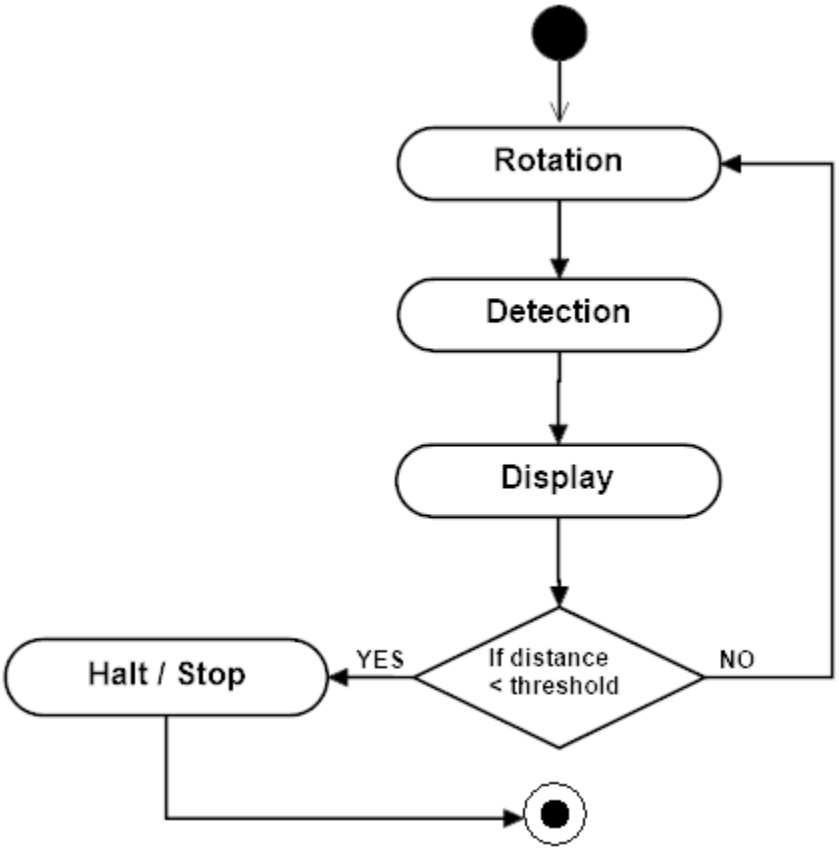
4.1 BLOCK DIAGRAM:



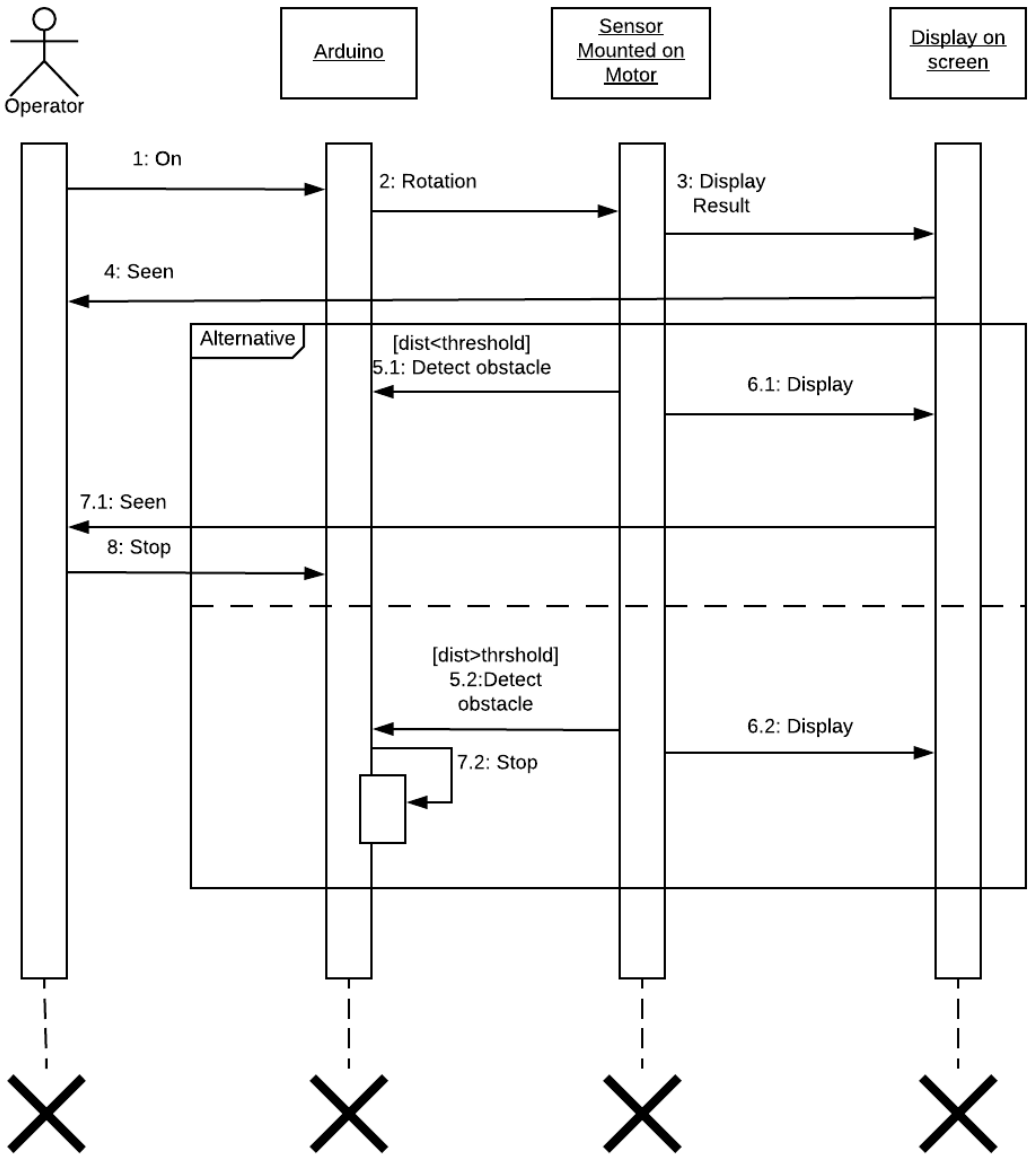
4.2 CIRCUIT DIAGRAM:



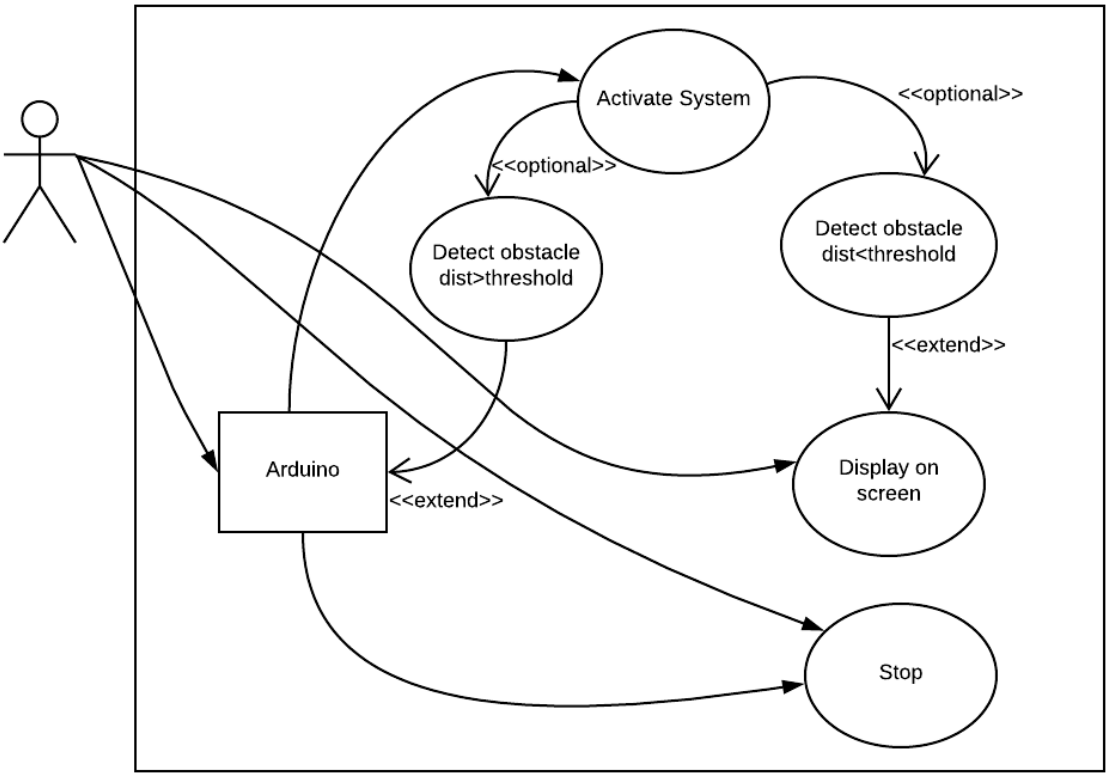
4.3 ACTIVITY DIAGRAM:



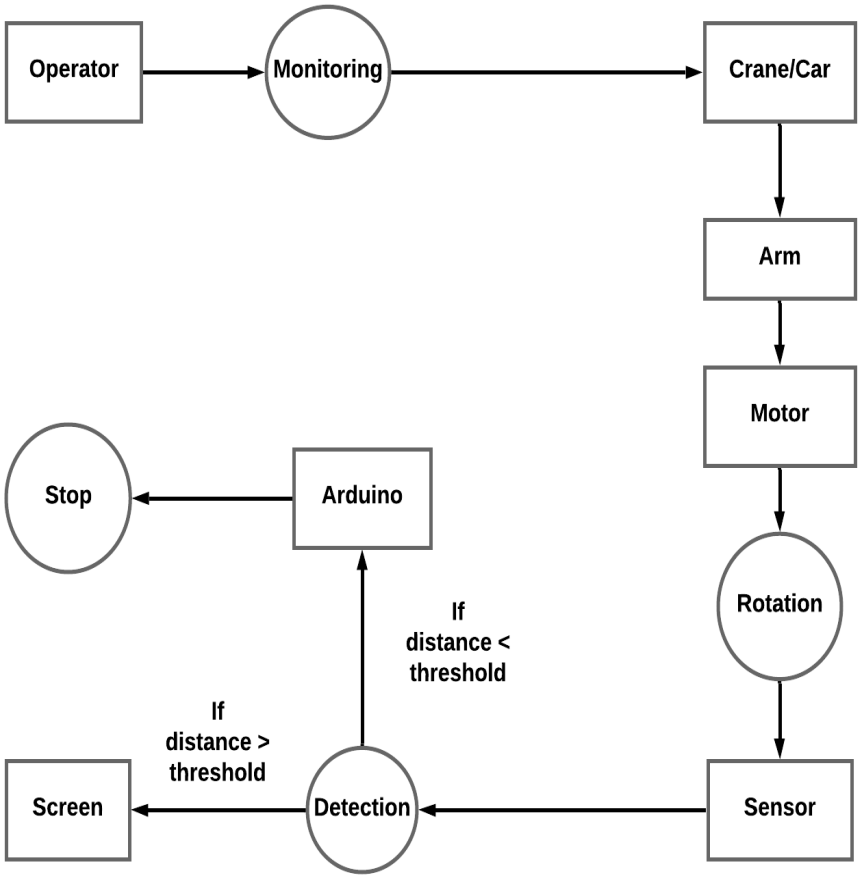
4.4 SEQUENCE DIAGRAM



4.5 USE CASE DIAGRAM



4.6 DATA FLOW DIAGRAM:



Chapter 5
Implementation and Testing

5.1 Implementation Approaches

Sr. No.	Implementation Plan	Action
1.	Module	1.Ultrasonic Sensor 2.ServoMotor
2.	Percentage Completed	1.Ultrasonic Sensor:100% 2.ServoMotor:100%
3.	Status	Completed
4.	Day Started	5 August 2018
5.	Day to be Completed	19 October 2018
6.	Actual Completion Date	1 December 2018
7.	Module Assignment	• Servo motor Module:Sai Warekar • Ultrasonic sensor Module:Suraj Adke
8.	Importance of Module	Servo motor:High Ultrasonic sensor:High

5.2 Coding Details and Code Efficiency:

Arduino Code

```
#include <Servo.h>
// Defines Trig 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(".");
if(distance < 5)
  {
myServo.detach();
    }
  }
}
// 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 IDE 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;
PFontorcFont;
void setup() {

 size (1920, 1080); // ***CHANGE THIS TO YOUR SCREEN RESOLUTION***
smooth();
```

```
myPort = new Serial(this,"COM1", 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();
}
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 prthats 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
coordinates 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("Object: " + noObject, 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();
}
```

• Code Efficiency:

- Our code doesn’t need enough memory to load and work as it is coded in C language with less memory utilization.
- Less global variables were used compare to local variables to minimize memory usage.
- Basic datatypes and functions were used to enhanced the code.
- Arduino IDE was used to code and test the code and upload to the Arduino board without any conversion in the code.
- Tested the code several times which work without any fail.
- Different libraries were used to increase the code functionality.
- The code is reliable for the system for which it was coded.
- Our code uses simple basic data types instead of derived data types.
- Comments were used to understand the functionality of the statement.
- Different loops are used for repeatation of code which minimizes the memory usage.
- In our code we avoid using some keywords which waste some processing cycles due to their presence in the code.

5.3Testing approaches

- Functional Testing is a testing technique which is used to test the features and the functionality of the system and the software it covers all the scenarios including failure paths and boundary cases.

5.3.1Unit Testing

- Unit testing deals with testing a unit or module as a whole. It is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine if they are fit for use.

- Its implementation can vary. Unit testing first focuses on the individual modules, independently one after another, to locate errors. This enables the tester to detect errors in coding and logical errors that is contained within that module alone. The resulting from the interaction between modules are initially avoided.

- So, while doing unit testing, we inserted the bug free code into our Arduino board to check its working properly and give us the expected output. We had tested all the components individually one after another and found that all the individual component works properly and give us the expected output.

- Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended, but in our case, we are only the developer so the unit testing is performed by us.

5.3.2 Integration Testing

- Integration testing is a systematic technique for constructing the program structure while at the same time to uncover the errors associated with interfacing.

- It brings all the modules together into a special testing environment, then it checks for errors, bugs and interoperability. It deals with tests for the entire application.

- The objective is to take unit-tested module and build a program structure. As

during integration several tests were conducted at each integrated module before and after integration.

- On completion of integrating all the components, then it is finally tested once again for any other error occurring after integrating and it also checks that each module is working properly.

- **Testing of the system:**

A) Object 1 is placed 30.5 far from the radar, radar gives the distance 32 cm, so:

- error =((32-30.5)/30.5)*100= 4.918%
- efficiency 1 = 100-error =95.08%

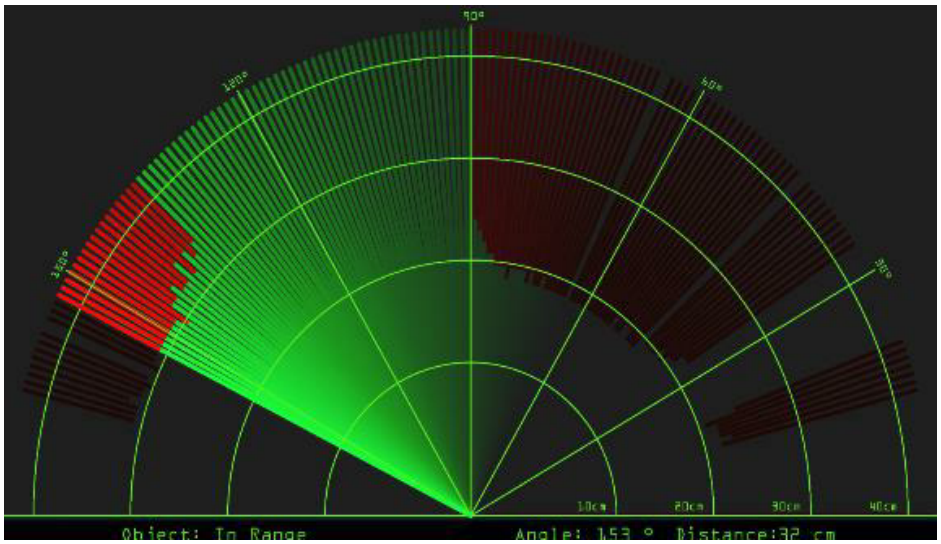


Figure 15: Radar Screen Shot 1

B) Object 2 placed at a distance of 20.3 cm, radar gives the distance 21 cm so:

- error = ((21-20.3)/20.3)*100 = 3.44%
- efficiency 2 =100-error= 96.55%

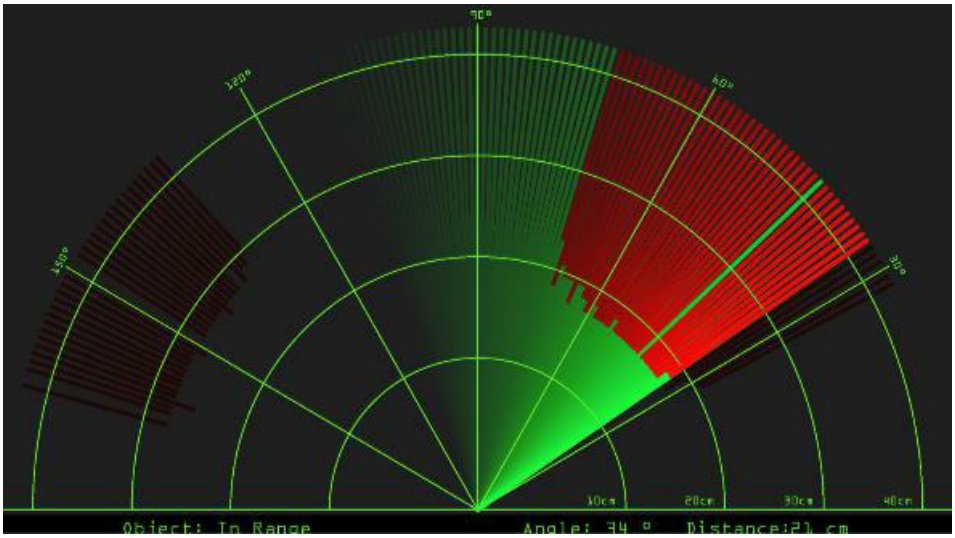


Figure 16: Radar Screen Shot 2

- After the observations and calculations we can conclude that this system is 95.815% efficient.

5.4 Modification and Improvements

- Initially our implemented model only asks for displaying a radar graph on the screen which is fitted in front of the operator whereas it was not very much convenient as it was only detecting obstacle and display it. So, to improve this, we set a threshold and if the distance the between crane and obstacle is less than the threshold then machine movement stops completely.

- By applying the threshold collision will be avoided completely.

Chapter 6
Results and Discussion

6.1Test Reports

Input	Expected Output	Error
Ultrasonic Sensor	Obstacle Detection	When the sensor doesn’t have obstacles in front or when the obstacles are soft, it returns 0 or erroneous results.
Servo Motor	To rotate sensor in the defined position	Servo Motor Chatter can be either intermittent or constant but is characterized by the motor standing still and making a chattering or buzzing

		noise.

6.2User Documentation

The aim of this project is to calculate the distance position and speed of the object placed at some distance from the sensor. Ultrasonic sensor sends the ultrasonic wave in different directions by rotating with help of servo motor. This wave travels in air and gets reflected back after striking some object. This wave is again sensed by the sensor and its characteristics is analysed and output is displayed in screen showing parameters such as distance and position of object. Arduino IDE is used to write code and upload coding in Arduino and helps us to sense position of servo motor and posting it to the serial port along with the distance of the nearest object in its path. The output of sensor is displayed with the help of Processing IDE to give final output in display screen. How the system should react to particular inputs and how the system should behave in particular situations:

- If ultrasonic sensor detects any obstacle which is close to the crane then it will show graph on to the screen.
- If the obstacle is too close to the crane & driver does not take any actions then the entire system will stop automatically.

Chapter 7
Conclusion

7.1 Conclusion

By considering the detection of the obstacle, the collision warning system focuses on the study of avoidance control. Radar system was designed with the help of Arduino, servomotor and ultrasonic sensor which can detect the position, distance of obstacle which comes in its way and converts it into visually representable form. This ultrasonic collision detection system can be used on stationary and mobile robots, automatic-guided vehicles, and other manufacturing applications.

7.2 Limitation of the system

1. Entire system is controlled by arduino. If arduino isn't working properly then overall system behave improperly.

2. If the operator of the crane not operating the system properly then the entire system stops again & again. It may result in delay in productivity.

3. Defect in Ultrasonic Sensor result in displaying improper radar display.

4. Only Collision is detected & prevented by this prototype. Other working is similar like normal cranes that are used in construction sites.

7.3 Future

7.3 Scope of the Project

- Using a Ethernet shield we can control the arduino means system from anywhere & it helps to easily operate the system and increases a safety.

- We can also control the movement of the crane if obstacle fall down. Suppose the obstacle fall down then machine moves back automatically.

I. References

1. T. Wilson, et al., “Forward-Looking Collision Warning System Performance Guidelines,” SAE No. 970456.
2. Nithya.V, Robin Alias2, Sreepriya N. S, Vivek Joseph,Yathukrishna.K.T-“Ultrasonic RADAR”,Vol.4,No.3,March 2016,E-ISSN:2321-9637.

II. Bibliography

III. Website Used

IV. Glossary

If you the students any acronyms, abbreviations, symbols, or uncommon terms in the project report then their meaning should be explained where they first occur. If they go on to use any of them extensively then it is helpful to list them in this section and define the meaning.

V. Appendices

These may be provided to include further details of results, mathematical derivations, certain illustrative parts of the program code (e.g., class interfaces), user documentation etc.

In particular, if there are technical details of the work done that might be useful to others who wish to build on this work, but that are not sufficiently important to the project as a whole to justify being discussed in the main body of the project, then they should be included as appendices.

VI. Summary

Project development usually involves an engineering approach to the design and development of a software system that fulfils a practical need. Projects also often form an important focus for discussion at interviews with future employers as they provide a detailed example of what the students are capable of achieving.

VII. Further Reading

1. Modern Systems Analysis and Design; Jeffrey A. Hoffer, Joey F. George, Joseph,S.Valacich; Pearson Education; Third Edition; 2002.

2. ISO/IEC 12207: Software Life Cycle Process

(http://www.software.org/quagmire/descriptions/iso-iec12207.asp).