**COMSATS UNIVERSITY ISLAMABAD ATTOCK CAMPUS**



**LAB PROJECT REPORT**

**SIGNAL AND SYSTEMS**

**(CPE - 223)**

|  |  |
| --- | --- |
| **Student Names** |  |
| **Registration Numbers** |  |

**Lab Instructor : Engr. Ijaz Ahmed**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sr.no** | **CONTENTS** | **Page no** |
|  | Abstract | 3 |
| 1. | Project Tittle | 4 |
| 2. | Objectives | 4 |
| 3. | Introduction | 4 |
| 4. | Required Components | 5 |
| 5. | Components Details | 5-6 |
| 6. | Circuit diagram | 6 |
| 7. | Procedure | 6 |
| 8. | Working principle | 6 |
| 9. | Arduino Code | 7-8 |
| 10. | Processing IDE code | 9-11 |
| 11. | Applications | 12 |
| 12. | Conclusion | 12 |

**ABSTRACT**

Radar is an object detection system which uses radio waves to determine the range, altitude, direction, or speed of object. It can be used to detect aircraft, ships, spacecraft, guided missiles, motor vehicles, weather formations and terrains.

The radar dish or antenna transmits pulses of radio waves or microwaves which bounce off any object in their path. The object returns a tiny part of energy to a dish or antenna which is usually located at the same side of transmitter.

The modern uses of radar are highly diverse, including air traffic control, radar astronomy and air defense system.

High technique radar systems are associated with digital processing and can extract useful information from very high noise level.

The Arduino based project requires an ultrasonic sensor, the sensor released the waves which we want to measure the distance of the object.

The microcontrollers of Arduino board can be programmed using C and C++ languages. When code is written in Arduino UNO IDE software and connected to the board through a USB cable.

1. **PROJECT TITLE**

Radar System Using Ultrasonic Sensor and Arduino.

1. **OBJECTIVES**

* To achieve a radar system prototype based on an Arduino board that detects stationary and moving objects.
* To learn how Arduino works.
* To learn about ultrasonic sensor and servo motor.

1. **INTRODUCTION**

An Arduino is a microcontroller-based kit which can be either used directly by purchasing from the vendor or can be made at home by using some components. It is basically used in communications and in controlling or operating many devices.

The Arduino tool window consists of the toolbar with the buttons like verify, upload, new, open, save and serial monitor. It also consists of text editor to write code, a message area which displays feedback like showing the errors, the text console which displays the output and a series of menu like the File, Edit and Tool bars. Thus, the code is uploaded by the bootloader onto the microcontroller.

Everything produces sound wave just by existence and effect flow of air around them with their natural frequency. These frequencies are beyond hearing range of humans. Wave of frequency range of 20000hz and thereabouts are called ultra-sonic wave and these waves can be detected by an ultrasonic sensor which helps us to get various knowledge. They are used for measuring object position and orientation, collision avoidance system, surveillance system etc.

Basically, an ultrasonic sensor sends ultrasonic waves which travels in air and gets reflected after striking any object. By studying the property of reflected wave, we can get knowledge about objects distance, position, speed etc.

This project is to calculate the distance position 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 after striking some object. This wave is again sensed by the sensor and its characteristics is analyzed 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.

1. **REQUIRED COMPONENTS**

* Ultrasonic sensor
* Jumper wires
* Bread board
* Servo motor
* Arduino UNO
* Arduino software
* The processing foundation processor software

1. **COMPONENTS DETAILS**

Following are the list of the components used for our project:

* **Aurdino Uno R3**

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output pins that may be interfaced to various expansion boards and other circuits.

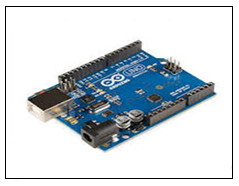


Figure 1

* **HC-SR04 Ultrasonic Module Distance Sensor**

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



Figure 2

* **Tower Pro SG90 RC Mini Servo Motor**

Micro Servo Motor SG90 is a tiny and lightweight server motor with high output power. Servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but smaller.

You can use any servo code, hardware or library to control these servos.



Figure 3

1. **CIRCUIT DIAGRAM**

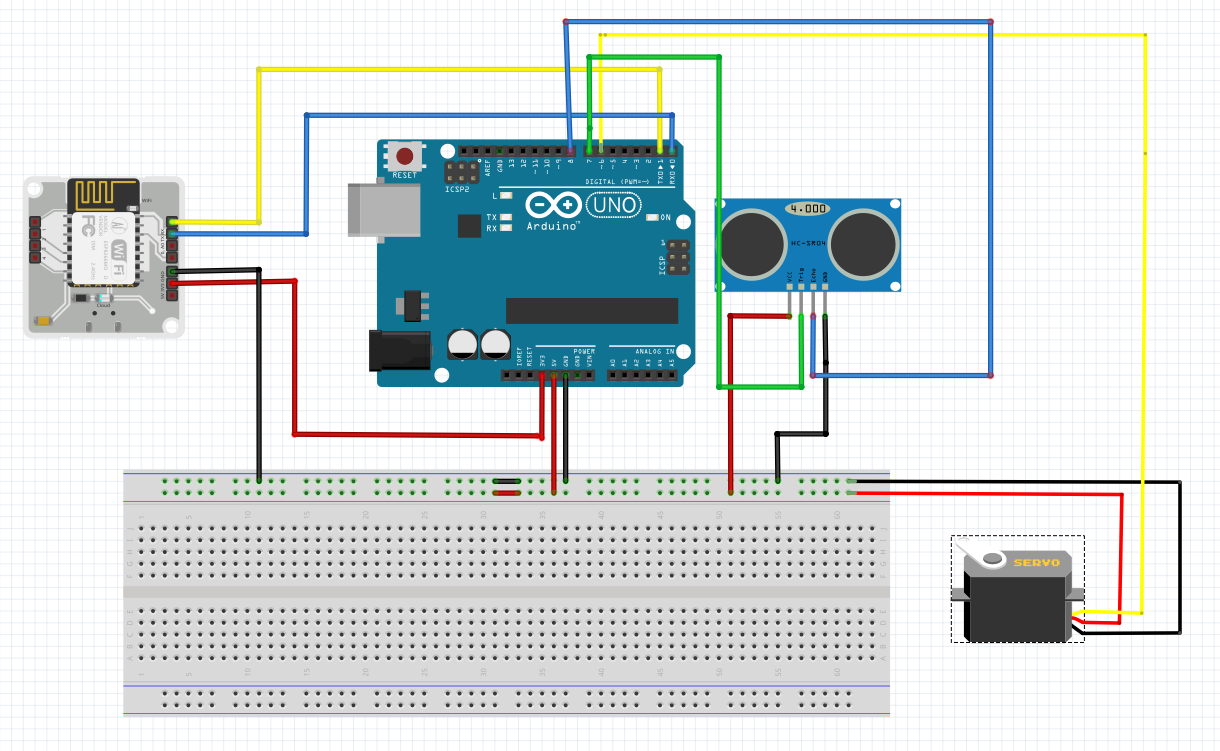


Figure 4: Complete Circuit for Radar System Using Ultrasonic Sensor and Arduino

1. **PROCEDURE**

* Make connections.
* Connect Arduino UNO
* Connect VCC of servomotor (red wire) and VCC of ultrasonic sensor to 5v of Arduino.
* Connect the GND of ultrasonic sensor and servo (black wire) to ground of the Arduino.
* Connect trig and echo pin of ultrasonic sensor to 10 and 11 of Arduino respectively.
* Connect signal pin of servo to pin 12 of Arduino.
* Write code in Arduino.
* Open processor software.
* Paste code in processing ide.
* Change the com3 on the code to your com port to which Arduino ide is connected otherwise code will give an error.
* At end, you observe results or output on processor software by taking an object near to ultrasonic sensor.

1. **WORKING PRINCIPLE**

Ultrasonic sensor sends the ultrasonic wave in different directions by rotating with help of servo motor. Sensor detects the distance of the closest object in front of it (from 3 cm up to 400 cm). It works by sending out a burst of ultrasound and listening for the echo when it bounces off an object. This wave travels in air and gets reflected after striking some object. This wave is again sensed by the sensor and its characteristics is analyzed 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 software to give final output in display screen.

1. **ARDUINO CODE**

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55 | // Includes the Servo library  #include <Servo.h>.  // Defines Tirg and Echo pins of the Ultrasonic Sensor  const int trigPin = 10;  const int echoPin = 11;  // Variables for the duration and the distance  long duration;  int distance;  Servo myServo; // Creates a servo object for controlling the servo motor  void setup() {  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output  pinMode(echoPin, INPUT); // Sets the echoPin as an Input  Serial.begin(9600);  myServo.attach(12); // Defines on which pin is the servo motor attached  }  void loop() {  // rotates the servo motor from 15 to 165 degrees  for(int i=15;i<=165;i++){  myServo.write(i);  delay(30);  distance = calculateDistance();// Calls a function for calculating the distance measured by the Ultrasonic sensor for each degree    Serial.print(i); // Sends the current degree into the Serial Port  Serial.print(","); // Sends addition character right next to the previous value needed later in the Processing IDE for indexing  Serial.print(distance); // Sends the distance value into the Serial Port  Serial.print("."); // Sends addition character right next to the previous value needed later in the Processing IDE for indexing  }  // Repeats the previous lines from 165 to 15 degrees  for(int i=165;i>15;i--){  myServo.write(i);  delay(30);  distance = calculateDistance();  Serial.print(i);  Serial.print(",");  Serial.print(distance);  Serial.print(".");  }  }  // Function for calculating the distance measured by the Ultrasonic sensor  int calculateDistance(){    digitalWrite(trigPin, LOW);  delayMicroseconds(2);  // Sets the trigPin on HIGH state for 10 micro seconds  digitalWrite(trigPin, HIGH);  delayMicroseconds(10);  digitalWrite(trigPin, LOW);  duration = pulseIn(echoPin, HIGH); // Reads the echoPin, returns the sound wave travel time in microseconds  distance= duration\*0.034/2;  return distance;  } | |

1. **PROCESSING IDE CODE**

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  15  157  158 | **import** **processing.serial.**\*; // imports library **for** serial communication  **import** **java.awt.event.KeyEvent**; // imports library **for** reading the data **from** **the** **serial** **port**  **import** **java.io.IOException**;  Serial myPort; // defines Object Serial  // defubes variables  String angle="";  String distance="";  String data="";  String noObject;  float pixsDistance;  int iAngle, iDistance;  int index1=**0**;  int index2=**0**;  PFont orcFont;  void setup() {    size (**1200**, **700**); // \*\*\*CHANGE THIS TO YOUR SCREEN RESOLUTION\*\*\*  smooth();  myPort = new Serial(this,"COM3", **9600**); // starts the serial communication  myPort.bufferUntil('.'); // reads the data **from** **the** **serial** **port** **up** **to** **the** **character** '.'. So actually it reads this: angle,distance.  }  void draw() {    fill(**98**,**245**,**31**);  // simulating motion blur **and** slow fade of the moving line  noStroke();  fill(**0**,**4**);  rect(**0**, **0**, width, height-height\***0.065**);    fill(**98**,**245**,**31**); // green color  // calls the functions **for** drawing the radar  drawRadar();  drawLine();  drawObject();  drawText();  }  void serialEvent (Serial myPort) { // starts reading data **from** **the** **Serial** **Port**  // reads the data **from** **the** **Serial** **Port** **up** **to** **the** **character** '.' **and** puts it into the String variable "data".  data = myPort.readStringUntil('.');  data = data.substring(**0**,data.length()-**1**);    index1 = data.indexOf(","); // find the character ',' **and** puts it into the variable "index1"  angle= data.substring(**0**, index1); // read the data **from** **position** "0" to position of the variable index1 **or** thats the value of the angle the Arduino Board sent into the Serial Port  distance= data.substring(index1+**1**, data.length()); // read the data **from** **position** "index1" to the end of the data pr thats the value of the distance    // converts the String variables into Integer  iAngle = int(angle);  iDistance = int(distance);  }  void drawRadar() {  pushMatrix();  translate(width/**2**,height-height\***0.074**); // moves the starting coordinats to new location  noFill();  strokeWeight(**2**);  stroke(**98**,**245**,**31**);  // draws the arc lines  arc(**0**,**0**,(width-width\***0.0625**),(width-width\***0.0625**),PI,TWO\_PI);  arc(**0**,**0**,(width-width\***0.27**),(width-width\***0.27**),PI,TWO\_PI);  arc(**0**,**0**,(width-width\***0.479**),(width-width\***0.479**),PI,TWO\_PI);  arc(**0**,**0**,(width-width\***0.687**),(width-width\***0.687**),PI,TWO\_PI);  // draws the angle lines  line(-width/**2**,**0**,width/**2**,**0**);  line(**0**,**0**,(-width/**2**)\*cos(radians(**30**)),(-width/**2**)\*sin(radians(**30**)));  line(**0**,**0**,(-width/**2**)\*cos(radians(**60**)),(-width/**2**)\*sin(radians(**60**)));  line(**0**,**0**,(-width/**2**)\*cos(radians(**90**)),(-width/**2**)\*sin(radians(**90**)));  line(**0**,**0**,(-width/**2**)\*cos(radians(**120**)),(-width/**2**)\*sin(radians(**120**)));  line(**0**,**0**,(-width/**2**)\*cos(radians(**150**)),(-width/**2**)\*sin(radians(**150**)));  line((-width/**2**)\*cos(radians(**30**)),**0**,width/**2**,**0**);  popMatrix();  }  void drawObject() {  pushMatrix();  translate(width/**2**,height-height\***0.074**); // moves the starting coordinats to new location  strokeWeight(**9**);  stroke(**255**,**10**,**10**); // red color  pixsDistance = iDistance\*((height-height\***0.1666**)\***0.025**); // covers the distance **from** **the** **sensor** **from** **cm** **to** **pixels**  // limiting the range to **40** cms  **if**(iDistance<**40**){  // draws the object according to the angle **and** the distance  line(pixsDistance\*cos(radians(iAngle)),-pixsDistance\*sin(radians(iAngle)),(width-width\***0.505**)\*cos(radians(iAngle)),-(width-width\***0.505**)\*sin(radians(iAngle)));  }  popMatrix();  }  void drawLine() {  pushMatrix();  strokeWeight(**9**);  stroke(**30**,**250**,**60**);  translate(width/**2**,height-height\***0.074**); // moves the starting coordinats to new location  line(**0**,**0**,(height-height\***0.12**)\*cos(radians(iAngle)),-(height-height\***0.12**)\*sin(radians(iAngle))); // draws the line according to the angle  popMatrix();  }  void drawText() { // draws the texts on the screen    pushMatrix();  **if**(iDistance>**40**) {  noObject = "Out of Range";  }  **else** {  noObject = "In Range";  }  fill(**0**,**0**,**0**);  noStroke();  rect(**0**, height-height\***0.0648**, width, height);  fill(**98**,**245**,**31**);  textSize(**25**);    text("10cm",width-width\***0.3854**,height-height\***0.0833**);  text("20cm",width-width\***0.281**,height-height\***0.0833**);  text("30cm",width-width\***0.177**,height-height\***0.0833**);  text("40cm",width-width\***0.0729**,height-height\***0.0833**);  textSize(**40**);  text("Behind The Clutter ", 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();  } | |

1. **APPLICATIONS**

* Air-traffic control at airports
* Ocean surveillance systems
* Outer space surveillance systems
* Marine
* Guided Missiles
* Motor vehicles
* Weather formations
* Radar astronomy
* Ground penetrating radar for geological observations
* Meteorological precipitation monitoring

1. **CONCLUSION**

Besides proper distance and velocity, a large set of data was taken to find the accuracy of the radar for objects of different shapes. For a cylindrical object, the radar provided 100% efficiency in a constant environment when the object was 5 cm away. The accuracy decreased to 30% when the distance was 17 cm away. The limitation of this system is that it was unable to detect small object or if the object was very close (1 cm).

In this project a 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 system can be used in robotics for object detection and avoidance system or can also be used for intrusion detection for location sizes. Range of the system depends upon type of ultra-sonic sensor used.

**------------------------------------------------------------------------------------**