INSTITUTO TECNOLÓGICO DE REYNOSA

Ingeniería Mecatrónica

“Reporte Práctica Arduino, Radar”

Alumnos

Aarón Javier Ávila López 24580067

Gregorio Valdez Vez 24580119

Víctor Manuel Guerrero Huerta 24580088

Daniel Alejandro Saucedo Gutiérrez 24580123

Asignatura

Programación Básica

Maestro

Ing. Miriam Puente Jiménez

Fecha de entrega

12-05-2025

TECNOLÓGICO NACIONAL DE MÉXICO

INDICE

1. Resumen

La presente práctica tuvo como objetivo desarrollar un sistema de radar utilizando Arduino Uno y un sensor ultrasónico, permitiendo la detección de objetos a diferentes distancias mediante una interfaz gráfica.

El sistema fue montado sobre un servomotor para simular el barrido de un radar, fijando el sensor con silicón para mayor estabilidad. La programación se realizó en el entorno de Arduino IDE y la visualización se logró con el software Processing, el cual generó una interfaz que simula el radar en funcionamiento.

Se logró detectar correctamente la distancia de objetos en el rango de acción del sensor, visualizándolo en pantalla. Esta experiencia permitió fortalecer los conocimientos de electrónica básica, programación y comunicación entre hardware y software.

2. Introducción

En el ámbito de la electrónica y la programación, el desarrollo de proyectos que integran hardware con software visual representa una valiosa herramienta de aprendizaje. En esta práctica, se diseñó un radar funcional con Arduino Uno, utilizando componentes electrónicos básicos y programación en el entorno Arduino IDE y Processing.

Arduino es una plataforma de desarrollo abierta que facilita la creación de proyectos interactivos, especialmente en entornos educativos. Gracias a su versatilidad y bajo costo, es ampliamente utilizada para enseñar conceptos de programación, electrónica y automatización.

El objetivo general de esta práctica fue diseñar e implementar un radar funcional capaz de detectar objetos cercanos y representar gráficamente la distancia mediante una interfaz desarrollada en Processing.

"Dame unas semanas más, y tendré un sistema que podrá ver en el patio trasero del enemigo.” (Sir Robert Watson-Watt, 1935).

3. Materiales y Métodos

Materiales utilizados:

- 1 placa Arduino Uno
- 1 Protoboard
- Cables tipo jumper macho-macho
- Cables tipo jumper hembra-macho
- 1 sensor ultrasónico HC-SR04
- 1 servomotor SG90
- Silicón líquido o caliente (para fijar el sensor al servomotor)
- Cable USB para conexión a la PC
- Computadora con Arduino IDE y Processing instalados.

Métodos:

El diseño del circuito consistió en montar el sensor ultrasónico sobre el servomotor, permitiendo que este gire horizontalmente para escanear el área. El sensor se fijó con silicón al eje del servomotor para asegurar su movimiento conjunto. Se conectó el sensor y el servomotor a la placa Arduino Uno mediante cables jumper, usando una protoboard para facilitar las conexiones.

La programación de la placa se realizó utilizando el entorno Arduino IDE. El Arduino se encargó de controlar el movimiento del servomotor y de medir la distancia utilizando el sensor ultrasónico. Los datos obtenidos fueron enviados por el puerto serial a la computadora.

En Processing se desarrolló una interfaz gráfica tipo radar, que recibía los datos del Arduino y los visualizaba en forma de barrido, representando en pantalla la distancia de los objetos detectados.

4. Resultados

El sistema radar logró realizar un barrido efectivo del área frente al sensor, detectando la presencia de objetos y reflejándolo en la interfaz de Processing en tiempo real. La visualización simuló correctamente el funcionamiento de un radar, mostrando la distancia de los objetos mediante líneas y puntos en la interfaz gráfica.

Se comprobó que el sensor ultrasónico respondía adecuadamente dentro de su rango de medición (aproximadamente entre 2 cm y 40 cm). A pesar de variaciones menores por ángulos o superficies reflectantes, el sistema se comportó de manera estable.

5. Discusión

Durante esta práctica se comprendió cómo integrar sensores, actuadores y software para construir un sistema funcional y visualmente interactivo. Una de las principales dificultades fue la alineación precisa del sensor ultrasónico sobre el servomotor, que afectaba la lectura si no estaba firmemente fijado, se probaron varias cosas para unir el sensor al servomotor, al final; esto se resolvió con el uso de silicón.

Otra dificultad fue la sincronización entre Arduino y Processing, la cual requirió asegurarse de que ambos programas usaran el mismo puerto serial y velocidad de transmisión. Esto permitió lograr una comunicación fluida entre ambas plataformas.

El proyecto permitió aplicar conocimientos de electrónica básica, estructuras de control en programación, y el uso de herramientas gráficas para visualización de datos, reforzando la comprensión del trabajo con sensores y motores.

6. Conclusiones

Esta práctica permitió a los estudiantes comprender el funcionamiento de un radar básico mediante el uso de Arduino y Processing. Se lograron integrar componentes electrónicos, programación y visualización gráfica para simular un sistema de detección real.

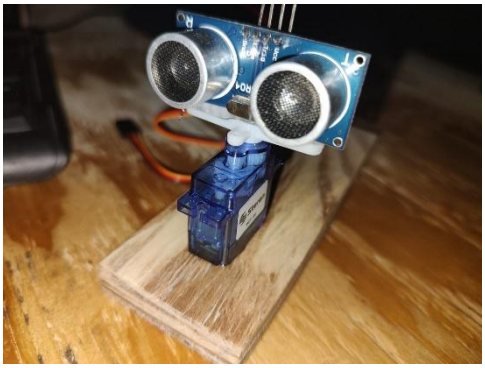
El ejercicio fomentó el desarrollo de habilidades prácticas como el armado de circuitos, depuración de código y visualización de datos. Además, se resaltó la importancia de la comunicación entre diferentes plataformas de software para lograr un sistema funcional e interactivo.

7. Referencias

Fontelles, R. (2024, June 9). Radar con Arduino usando sensor de ultrasonidos HC-SR04. RobotUNO - Aprende sobre electrónica y Arduino. https://robotuno.com/radar-arduino-sensor-ultrasonidos-hc-sr04/

Instructables. (2017, September 20). Radar sencillo con processing y arduino. Instructables. https://www.instructables.com/Radar-Sencillo-Con-Processing-Y-Arduino/

8. Anexos (Opcional)



VIDEO YOUTUBE: https://youtu.be/Siw_jXKncC8

CODIGO ARDUINO IDE:

```
#include <Servo.h>
const int trigPin = 2;
const int echoPin = 3;
long duration;
int distance;
Servo myServo;
void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
  myServo.attach(5);
}
void loop() {
  for(int i=15;i<=165;i++){
  myServo.write(i);
  delay(30);
  distance = calculateDistance();
  Serial.print(i);
  Serial.print(",");
  Serial.print(distance);
  Serial.print(".");
  }
  for(int i=165;i>15;i--){
  myServo.write(i);
  delay(30);
  distance = calculateDistance();
  Serial.print(i);
  Serial.print(",");
  Serial.print(distance);
  Serial.print(".");
  }
}
int calculateDistance(){
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance= duration*0.034/2;
  return distance;
}
```

CODIGO PROCESSING:

```
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("AA", width-width*0.875, height-
height*0.0277);
  text("Ángulo: " + iAngle +" °", width-width*0.48,
height-height*0.0277);
  text("Dist:", 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();
}
```