|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Akademik Yılı** | **2023 / 2024** | **Dönem** | **Bahar** | **Bölüm/Program** | **Bilgisayar Teknolojileri /Bilgisayar Programcılığı** | |
| **Dersin Adı** | **Bilgisayarlı Kontrol** | | | **Sınav Türü** | **Vize** | |
| **Proje Adı** | **ARDUINO İLE RADAR GÖRÜNTÜLEME SİSTEMİ** | | | | | |
| **Öğrenci Adı-Soyadı** | **Duran ERENSOY** | | **Öğrenci No** | **223010720059** | **Proje Notu** |  |
| **Öğrenci İmza** |  | | **Tarih** | **24/04/2024** |
| **Proje Grup Üyeleri:**  **1- Duran ERENSOY**  **2- Fevzi YÜKSEL**  **3- Abdülkadir BAYRAK** | | | **Öğretim Elemanının Adı-Soyadı:**  **Öğr. Gör. Fatma Nur KILIÇKAYA** | | | |

PROJE İÇİN GEREKLİ OLAN ARDUINO PARÇALARI

1. 1 ADET ARDUNIO UNO
2. 1 ADET BREADBOARD
3. 1 ADET SERVO MOTOR
4. 1 ADET ULTRASONİK SENSÖR
5. 1 ADET PROGRAMLAMA KABLOSU
6. YETERİNCE JUMPER KABLO

ARDUINO KODLARI

// Mini radar projesi

// Duran ERENSOY

#include <Servo.h>

// Ultrasonik Sinyal pinleri

const int trigPin = 10;

const int echoPin = 11;

long duration;

int distance;

Servo myServo;

void setup() {

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

Serial.begin(9600);

myServo.attach(12); // Servo motor sinyal pini

}

void loop() {

// 15 derece ile 165 derece arasında dön

for(int i=10;i<=180;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;

}

PROCESSING (GÖRÜNTÜLEME) KODLARI

// Mini radar projesi

// Duran ERENSOY

import processing.serial.\*; // kütüphane entegresi

import java.awt.event.KeyEvent;

import java.io.IOException;

Serial myPort;

String angle="";

String distance="";

String data="";

String noObject;

float pixsDistance;

int iAngle, iDistance;

int index1=0;

int index2=0;

PFont orcFont;

void setup() {

size (1366, 700);

smooth();

myPort = new Serial(this,"COM3", 9600); // Com portunu seçin

myPort.bufferUntil('.');

}

void draw() {

fill(98,245,31);

noStroke();

fill(0,4);

rect(0, 0, width, 1010);

fill(98,245,31); // yeşil renk

drawRadar();

drawLine();

drawObject();

drawText();

}

void serialEvent (Serial myPort) {

data = myPort.readStringUntil('.');

data = data.substring(0,data.length()-1);

index1 = data.indexOf(",");

angle= data.substring(0, index1);

distance= data.substring(index1+1, data.length());

iAngle = int(angle);

iDistance = int(distance);

}

void drawRadar() {

pushMatrix();

translate(683,700);

noFill();

strokeWeight(2);

stroke(98,245,31);

// draws the arc lines

arc(0,0,1300,1300,PI,TWO\_PI);

arc(0,0,1000,1000,PI,TWO\_PI);

arc(0,0,700,700,PI,TWO\_PI);

arc(0,0,400,400,PI,TWO\_PI);

// draws the angle lines

line(-700,0,700,0);

line(0,0,-700\*cos(radians(30)),-700\*sin(radians(30)));

line(0,0,-700\*cos(radians(60)),-700\*sin(radians(60)));

line(0,0,-700\*cos(radians(90)),-700\*sin(radians(90)));

line(0,0,-700\*cos(radians(120)),-700\*sin(radians(120)));

line(0,0,-700\*cos(radians(150)),-700\*sin(radians(150)));

line(-700\*cos(radians(30)),0,700,0);

popMatrix();

}

void drawObject() {

pushMatrix();

translate(683,700);

strokeWeight(9);

stroke(255,10,10); // kırmızı renk

pixsDistance = iDistance\*22.5;

// 40 cm ye kadar ölçer

if(iDistance<40){ line(pixsDistance\*cos(radians(iAngle)),-pixsDistance\*sin(radians(iAngle)),700\*cos(radians(iAngle)),-700\*sin(radians(iAngle))); } popMatrix(); } void drawLine() { pushMatrix(); strokeWeight(9); stroke(30,250,60); translate(683,700); line(0,0,700\*cos(radians(iAngle)),-700\*sin(radians(iAngle))); popMatrix(); } void drawText() { pushMatrix(); if(iDistance>40) {

noObject = "Out of Range";

}

else {

noObject = "In Range";

}

fill(0,0,0);

noStroke();

rect(0, 1010, width, 1080);

fill(98,245,31);

textSize(25);

text("10cm",800,690);

text("20cm",950,690);

text("30cm",1100,690);

text("40cm",1250,690);

textSize(40);

text("Object: " + noObject, 240, 1050);

text("Angle: " + iAngle +" °", 1050, 1050);

text("Distance: ", 1380, 1050);

if(iDistance<40) {

text(" " + iDistance +" cm", 1400, 1050);

}

textSize(25);

fill(98,245,60);

translate(390+960\*cos(radians(30)),780-960\*sin(radians(30)));

rotate(-radians(-60));

text("30°",0,0);

resetMatrix();

translate(490+960\*cos(radians(60)),920-960\*sin(radians(60)));

rotate(-radians(-30));

text("60°",0,0);

resetMatrix();

translate(630+960\*cos(radians(90)),990-960\*sin(radians(90)));

rotate(radians(0));

text("90°",0,0);

resetMatrix();

translate(760+960\*cos(radians(120)),1000-960\*sin(radians(120)));

rotate(radians(-38));

text("120°",0,0);

resetMatrix();

translate(840+900\*cos(radians(150)),920-960\*sin(radians(150)));

rotate(radians(-60));

text("150°",0,0);

popMatrix();

}