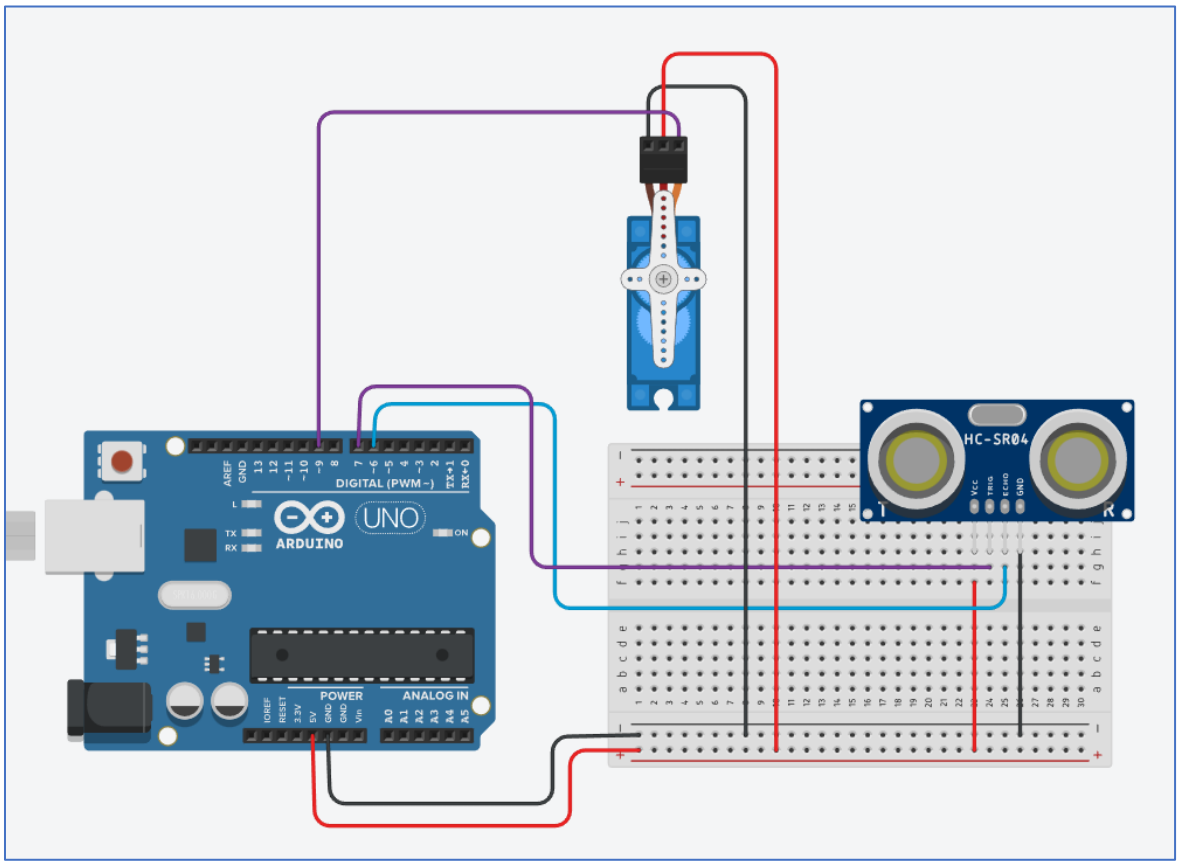
Radar uygulaması



Not: Bradboard' un motorun üzerine yerleştirilmesi gerekmektedir.

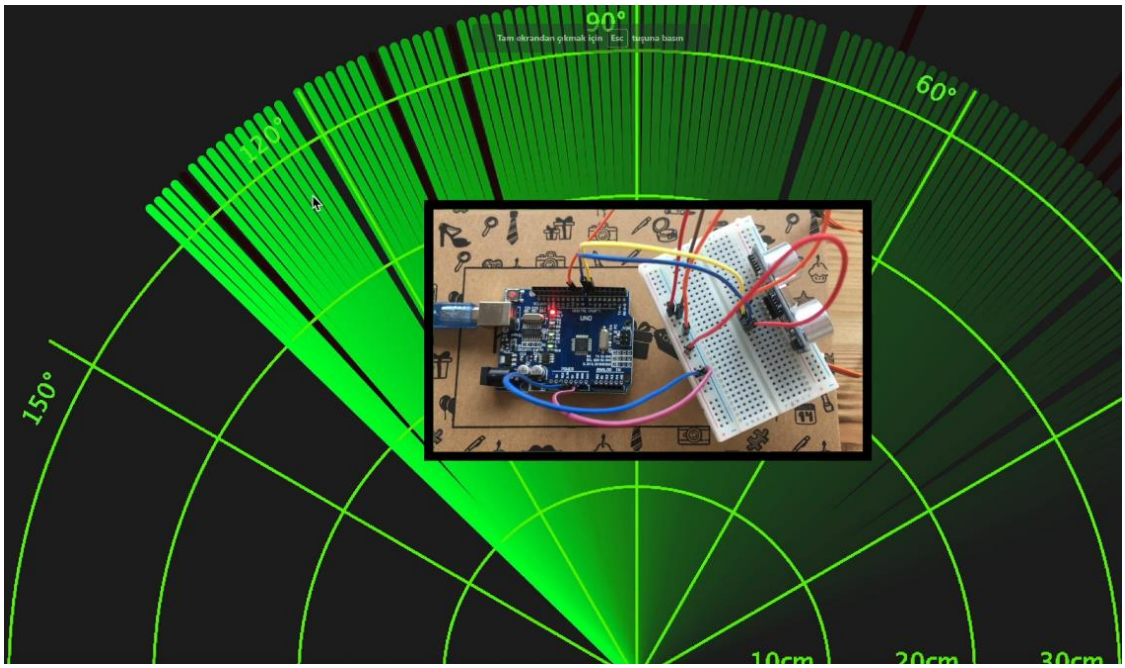
```
# include <Servo.h>
# define echo 6
# define trig 7

long zaman;
int cm;

Servo motor;

void setup()
{
  pinMode(trig,OUTPUT);
  pinMode(echo, INPUT);
  motor.attach(9);
  Serial.begin(9600);
}

void loop()
{
  for(int i = 1; i <= 150; i++){
  motor.write(i);
  delay(20);
```

```
   cm = mesafebulma ();

   Serial.print(i);
   Serial.print(",");
   Serial.print(cm);
   Serial.print("//");
  }

  for(int i = 150; i <= 15; i--){
  motor.write(i);
  delay(20);

   cm = mesafebulma ();

   Serial.print(i);
   Serial.print(",");
   Serial.print(cm);
   Serial.print("//");
  }
}

int mesafebulma()

{
    digitalWrite(trig, LOW);
    delayMicroseconds(2);
    digitalWrite(trig, HIGH);
    delayMicroseconds(10);
    digitalWrite(trig, LOW);

  zaman = pulseIn(echo, HIGH);
  float cm = zaman / 58.2;
  return cm;
  }
```

Görüntü alabilmek için processing programı kullanılır. Bu amaçla processing.org sitesine girerek Download bölümüne gelerek işletim sisteminin uygun versiyonuna göre uygun program indirilerek kurulmalıdır.

Bu uygulama yüklendiğinde görünüm aşağıdaki gibi olacaktır. Processing programın kodu aşağıda, şeklin altında verilmiştir.

Not: Programda dikkat edilecek önemli bir husus, programın 10. Satırında yazan COM4' ün sizin arduinonuzun bağlı olduğu portun isminin yazılmasıdır.




```
import processing.serial.*; // imports library for serial communication

Serial myPort;  // defines Object for Serial

String ang="";
String distance="";
String data="";

int angle, dist;

void setup() {

   size (1200, 700);

   myPort = new Serial(this,"COM4", 9600); // starts the serial communication
   myPort.bufferUntil('.');    // reads the data from the serial port up to the character
'.' before calling serialEvent

  background(0);
}

void draw() {
                              //for the blur effect
      fill(0,5);              //colour,opacity
      noStroke();
      rect(0, 0, width, height*0.93);

      noStroke();
      fill(0,255);
      rect(0,height*0.93,width,height); // so that the text having angle and distance
doesnt blur out

      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);

      int index1 = data.indexOf(",");
      ang= data.substring(0, index1);
      distance= data.substring(index1+1, data.length());
```

```
      angle = int(ang);
      dist = int(distance);
      System.out.println(angle);
}

void drawRadar()
{
    pushMatrix();
    noFill();
    stroke(10,255,10);        //green
    strokeWeight(3);

    translate(width/2,height-height*0.06);

    line(-width/2,0,width/2,0);

    arc(0,0,(width*0.5),(width*0.5),PI,TWO_PI);
    arc(0,0,(width*0.25),(width*0.25),PI,TWO_PI);
    arc(0,0,(width*0.75),(width*0.75),PI,TWO_PI);
    arc(0,0,(width*0.95),(width*0.95),PI,TWO_PI);

    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)));

    stroke(175,255,175);
    strokeWeight(1);
    line(0,0,(-width/2)*cos(radians(15)),(-width/2)*sin(radians(15)));
    line(0,0,(-width/2)*cos(radians(45)),(-width/2)*sin(radians(45)));
    line(0,0,(-width/2)*cos(radians(75)),(-width/2)*sin(radians(75)));
    line(0,0,(-width/2)*cos(radians(105)),(-width/2)*sin(radians(105)));
    line(0,0,(-width/2)*cos(radians(135)),(-width/2)*sin(radians(135)));
    line(0,0,(-width/2)*cos(radians(165)),(-width/2)*sin(radians(165)));

    popMatrix();
}

void drawLine() {

    pushMatrix();

    strokeWeight(9);
    stroke(0,255,0);
    translate(width/2,height-height*0.06);

   line(0,0,(width/2)*cos(radians(angle)),(-width/2)*sin(radians(angle)));

    popMatrix();

}

void drawObject()
{
    pushMatrix();
    strokeWeight(9);
    stroke(255,0,0);
    translate(width/2,height-height*0.06);

    float pixleDist = (dist/40.0)*(width/2.0);    // covers the distance from the sensor
from cm to pixels
    float pd=(width/2)-pixleDist;
    float x=-pixleDist*cos(radians(angle));
    float y=-pixleDist*sin(radians(angle));

    if(dist<=40)                                          // limiting the range
to 40 cms
    {
       //line(0,0,pixleDist,0);
       line(-x,y,-x+(pd*cos(radians(angle))),y-(pd*sin(radians(angle))));
    }
    popMatrix();
}

void drawText()
{
    pushMatrix();
```

```
    fill(100,200,255);
    textSize(25);

    text("10cm",(width/2)+(width*0.115),height*0.93);
    text("20cm",(width/2)+(width*0.24),height*0.93);
    text("30cm",(width/2)+(width*0.365),height*0.93);
    text("40cm",(width/2)+(width*0.45),height*0.93);

    textSize(40);
    text("YAinnoware",width*0.08,height*0.99);
    text("Angle :"+angle,width*0.45,height*0.99);

    if(dist<=40) {
      text("Distance :"+dist,width*0.7,height*0.99);
    }

   translate(width/2,height-height*0.06);
   textSize(25);

   text(" 30°",(width/2)*cos(radians(30)),(-width/2)*sin(radians(30)));
   text(" 60°",(width/2)*cos(radians(60)),(-width/2)*sin(radians(60)));
   text("90°",(width/2)*cos(radians(91)),(-width/2)*sin(radians(90)));
   text("120°",(width/2)*cos(radians(123)),(-width/2)*sin(radians(118)));
   text("150°",(width/2)*cos(radians(160)),(-width/2)*sin(radians(150)));

    popMatrix();
}
```