The experiment purpose

1 Learn steering gear control principle;

2 Learn ultrasonic ranging methods;

3 Ultrasonic ranging module is used to realize the function of ultrasonic radar.

The component list

◆ Keywish Arduino Uno R3 mainboard

◆ Breadboard

◆ USB cable

◆ SG90 steering gear

◆ Ultrasonic module

◆ Steering gear bracket

◆ Jumper wires

Operating principle of steering gear

The steering gear control signal enters the signal modulation chip from the channel of the receiver to obtain the dc bias voltage. It has an internal reference circuit that generates a reference signal with a period of 20ms and a width of 1.5ms. The obtained dc offset voltage is compared with the voltage of the potentiometer to obtain the voltage difference output. Finally, the positive and negative output of the voltage difference to the motor drive chip determines the positive and negative rotation of the motor. When the motor speed is constant, the potentiometer is rotated by the cascade reduction gear, so that the voltage difference is 0 and the motor stops rotating. The steering gear has the maximum rotation Angle, and the middle position refers to the volume from that position to the minimum Angle, and the maximum Angle is exactly the same. The most important part, the maximum rotation Angle varies with the steering gear, but the bandwidth in the middle position is fixed, i.e. 1.5 ms.

Steering gear control

The control of the steering gear generally requires a time base pulse of about 20ms, and the high level part of the pulse is generally the Angle control pulse part within the range of 0.5ms~2.5ms. Take 180-degree servo as an example, then the corresponding control relationship is as follows:
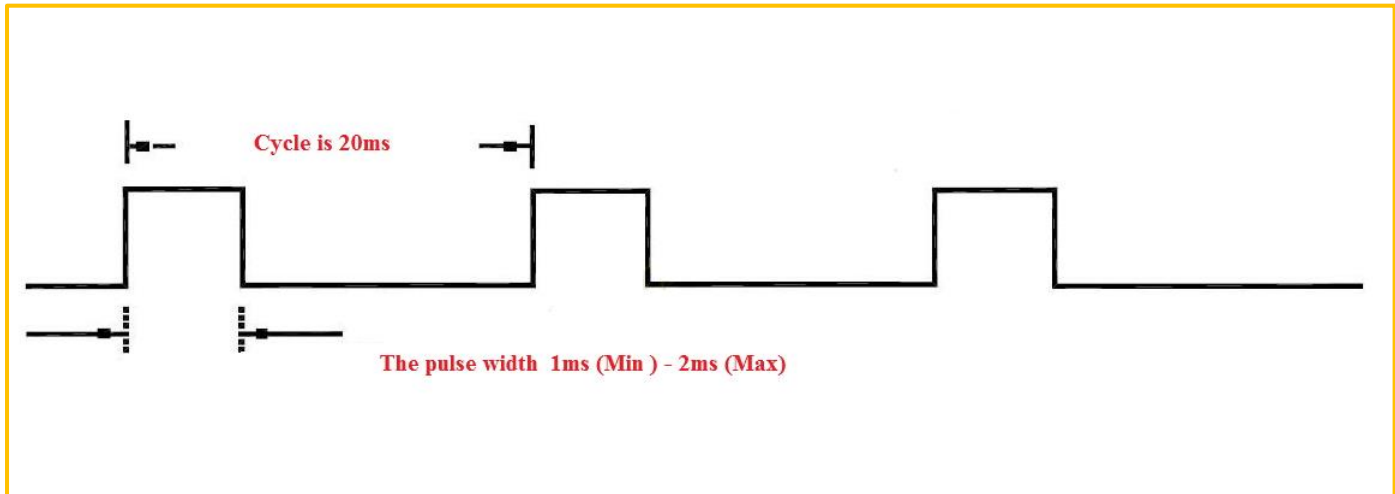
0.5ms--------------0degree；

1.0ms------------45degree；

1.5ms------------90degree；

2.0ms-----------135degree；

2.5ms-----------180degree；

Cycle is 20ms

The pulse width 1ms (Min) - 2ms (Max)

The rotation Angle is generated by a continuous pulse from the control line. This method is called pulse modulation. The length of the pulse determines the rotation Angle of the steering gear. For example, the steering gear rotates to a 1.5 millisecond pulse in the middle position (for a 180° steering gear, the middle position is 90°). When the control system issues a command to move the steering gear to a specific position and hold it at an Angle, the effect of external forces does not change the Angle. The Angle will not remain constant until the control system sends out pulses continuously to stabilize the steering Angle.

When the steering gear receives a pulse less than 1.5ms, the output shaft will be taken as the standard middle position and rotated anticlockwise at a certain Angle. When the received pulse is greater than 1.5ms, the output axis rotates clockwise. The maximum and minimum values may be different for different brands of steering gear, or even for different steering gear of the same brand.
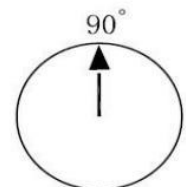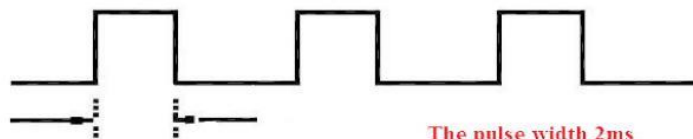
The Min pulse width
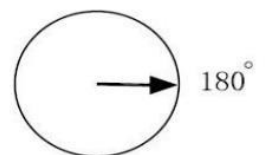
The pulse width 1ms

0°
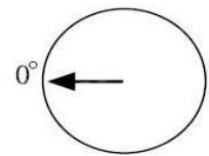
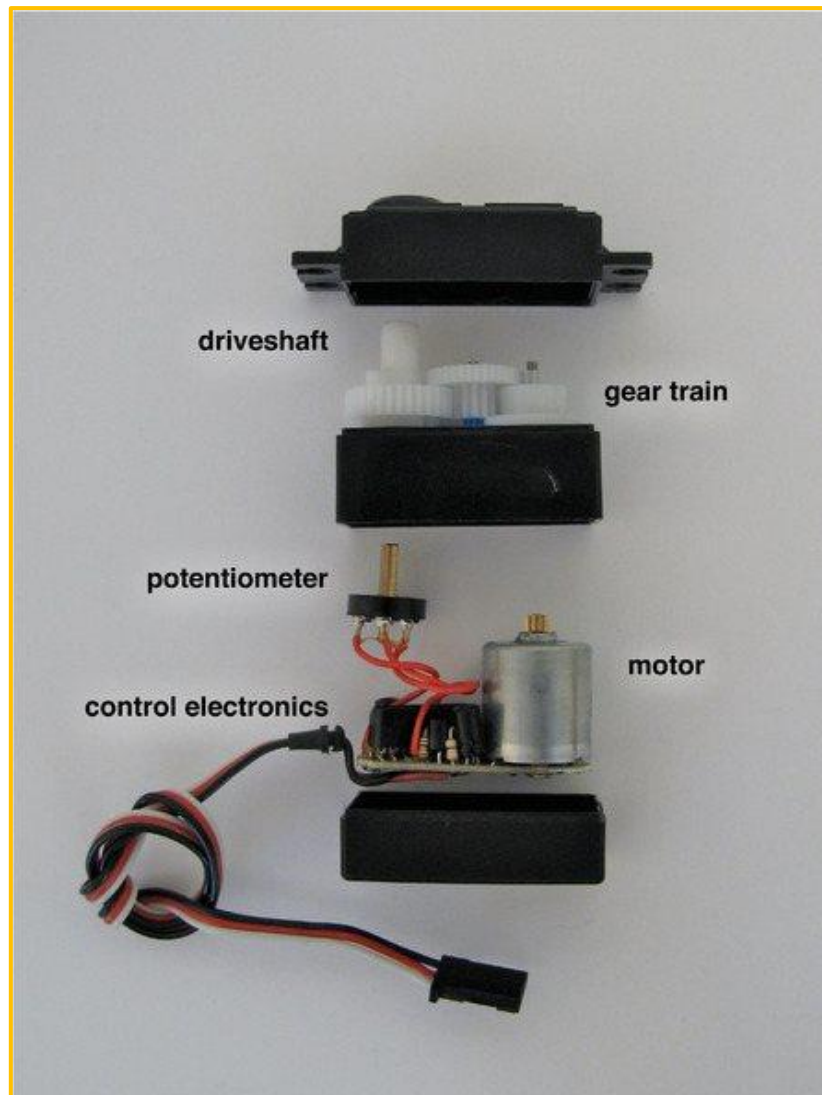The medium pulse width

The pulse width 1.5ms

90°

The Max pulse width

The pulse width 2ms

180°

Internal Structure of Steering Gear

Principle of ultrasonic ranging module

The ultrasonic transmitter fires the ultrasound in one direction or another, and we start the clock at the same time. When the ultrasound in the air hits an obstacle, it immediately returns, the ultrasonic receiver picks up the reflected wave, and we stop the clock. The velocity of sound waves in the air is 340m/s. According to the recorded time t, the distance s between the starting point and the obstacle can be calculated, that is, s = 340m/s * t / 2. Therefore, we can obtain the distance.

Ultrasonic ranging module has four pins, they are Vcc, Trig, Echo, GND, Trig is the distance measurement trigger pin, as long as the Trig pin at least 10 s high level, ultrasonic sending module will automatically send 8 40KHZ ultrasonic pulses, and automatically detect whether there is a return signal. This step will be done automatically by the internal module. If there is any return signal, the Echo pin will output a high level, and the duration of the high level is the time from the ultrasonic transmission to the

return. At this point, we can use the pulseIn () function to obtain the result of the distance measurement and calculate the actual distance.
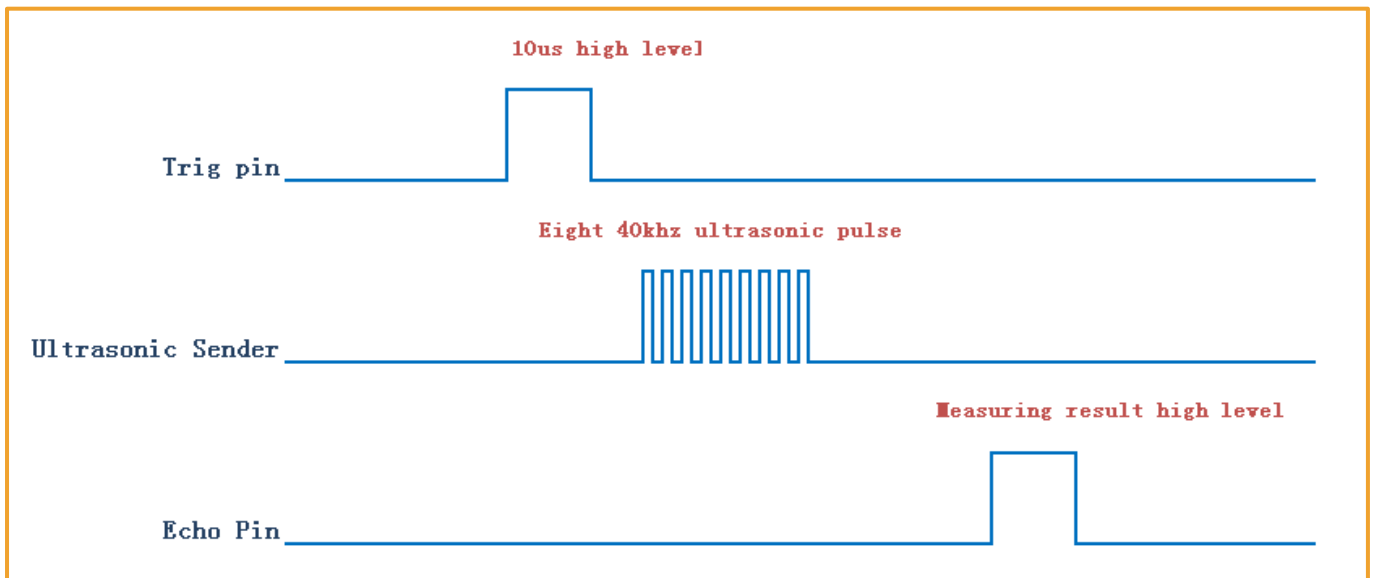
The experimental principle

Arduino is a convenient, flexible and easy-to-use open source electronic prototype platform. The generated data is sent back to the computer through the serial port, and then the "radar" window is displayed through Processing. Processing originates from data visualization and has added a lot of multimedia Processing capabilities through evolution, so it can be expressed not only by graphics, but also by video Processing and sound Processing.

Wiring

Arduino UNO board	Sterring gear
GND	GND
5V	VCC
7	S
Arduino UNO board	**ultrasonic**
GNG	GND
3	Echo
2	Trig
5V	VCC
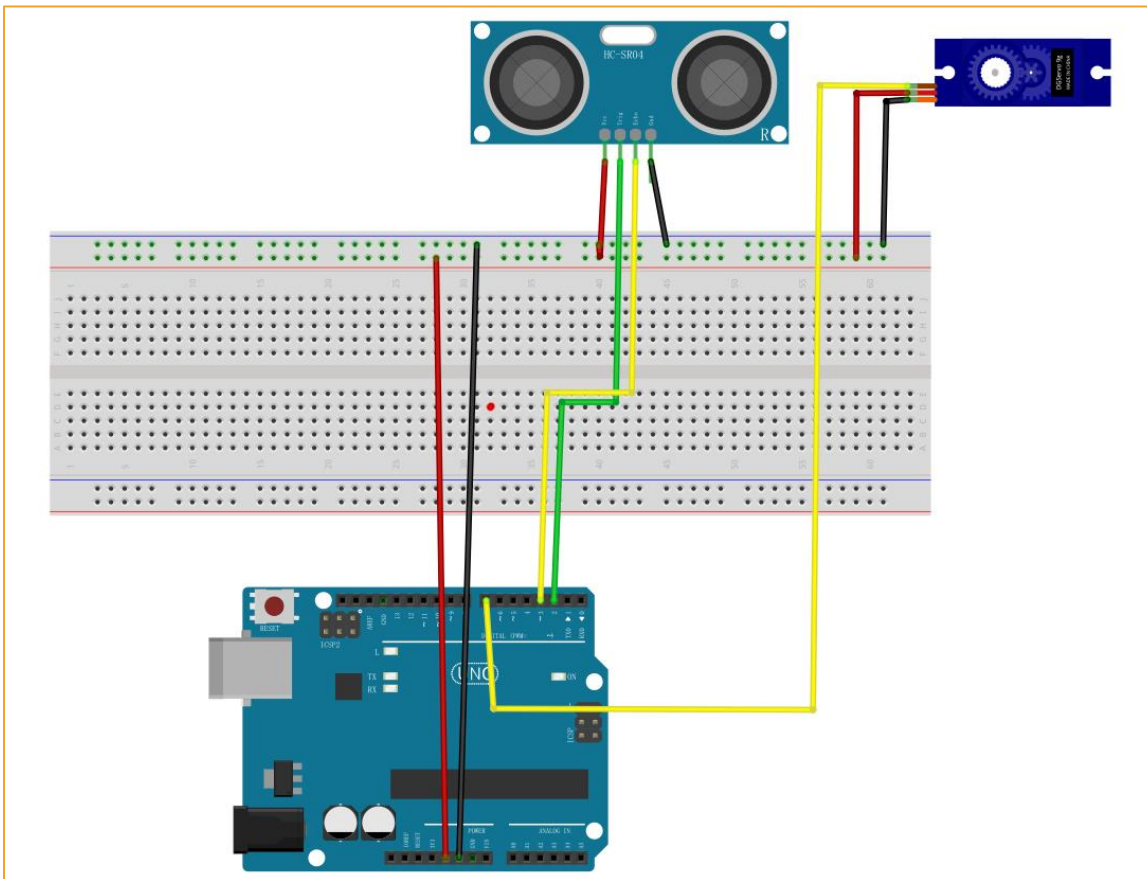
Arduino code

```
#include<Servo.h>
const int soundTriggerPin = 2;
const int soundEchoPin = 3;
const int motorSignalPin = 7;
const int startingAngle = 15;
const int minimumAngle = 15;
const int maximumAngle = 165;
const int rotationSpeed = 1;
Servo motor;
void setup(void)
{
    pinMode(soundTriggerPin, OUTPUT);
    pinMode(soundEchoPin, INPUT);
    motor.attach(motorSignalPin);
    Serial.begin(9600);
}
void loop(void)
{
    static int motorAngle = startingAngle;
```

```
    static int motorRotateAmount = rotationSpeed;
    motor.write(motorAngle);
    delay(30);
    SerialOutput(motorAngle, CalculateDistance());
    motorAngle += motorRotateAmount;
    if(motorAngle <= minimumAngle || motorAngle >= maximumAngle) {
        motorRotateAmount = -motorRotateAmount;
    }}
int CalculateDistance(void)
{
    //digitalWrite(soundTriggerPin, LOW);
    //delayMicroseconds(2);
    digitalWrite(soundTriggerPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(soundTriggerPin, LOW);
      long duration = pulseIn(soundEchoPin, HIGH);
    float distance = duration * 0.017F;
    return int(distance);
}
void SerialOutput(const int angle, const int distance)
{
    String angleString = String(angle);
    String distanceString = String(distance);
    Serial.println(angleString + "," + distanceString);
}
```

Processing

```
import processing.serial.*;
import java.awt.event.KeyEvent;
import java.io.IOException;

Serial  myPort;
PFont orcFont;
int iAngle;
int iDistance;
void setup() {
size(1350, 760);
        smooth();

        myPort  = new Serial(this, "COM31", 9600);
        myPort.clear();
```

```
        myPort.bufferUntil('\n');
            orcFont  =  loadFont("OCRAExtended-48.vlw");
}
void draw()
{
        fill(98, 245, 31);
        textFont(orcFont);
        noStroke();
        fill(0, 4);
        rect(0, 0, width, 0.935 * height);
        fill(98, 245, 31);

        DrawRadar();
        DrawLine();
        DrawObject();
        DrawText();
}
void serialEvent (Serial myPort)
{
        try {
                String data = myPort.readStringUntil('\n');
                        if (data == null) {
                        return;
                }
                  int commaIndex = data.indexOf(",");
                String angle = data.substring(0, commaIndex);
                String distance = data.substring(commaIndex+1, data.length()-1);
        iAngle  =  StringToInt(angle);
                iDistance  =  StringToInt(distance);
        } catch(RuntimeException e) {}
}
void DrawRadar()
{
        pushMatrix();
        translate(width/2, 0.926 * height);
        noFill();
        strokeWeight(2);
        stroke(98, 245, 31);

        // draws the arc lines
        DrawRadarArcLine(0.9375);
        DrawRadarArcLine(0.7300);
```

```
        DrawRadarArcLine(0.5210);
        DrawRadarArcLine(0.3130);

        // draws the angle lines
        final int halfWidth = width/2;
        line(-halfWidth, 0, halfWidth, 0);
        for(int angle = 30; angle <= 150; angle+=30) {
                DrawRadarAngledLine(angle);
        }
        line(-halfWidth * cos(radians(30)), 0, halfWidth, 0);
                popMatrix();
}
void DrawRadarArcLine(final float coefficient)
{
        arc(0, 0, coefficient * width, coefficient * width, PI, TWO_PI);
}
void DrawRadarAngledLine(final int angle){
        line(0, 0, (-width/2) * cos(radians(angle)), (-width/2) * sin(radians(angle)));
}
void DrawObject()
{
        pushMatrix();
        translate(width/2, 0.926 * height);
        strokeWeight(9);
        stroke(255,10,10);
        int pixsDistance = int(iDistance * 0.020835 * height);
        if(iDistance < 40 && iDistance != 0) {
                float cos = cos(radians(iAngle));
                float sin = sin(radians(iAngle));
                int x1 = +int(pixsDistance * cos);
                int y1 = -int(pixsDistance * sin);
                int x2 = +int(0.495 * width * cos);
                int y2 = -int(0.495 * width * sin);

                line(x1, y1, x2, y2);
        }
            popMatrix();
}
void DrawLine()
{
        pushMatrix();
        strokeWeight(9);
```

```
                stroke(30, 250, 60);
                translate(width/2, 0.926 * height);

                float angle = radians(iAngle);
                int x = int(+0.88 * height * cos(angle));
                int y = int(-0.88 * height * sin(angle));
                line(0, 0, x, y);
                popMatrix();
}
void DrawText()
{
                pushMatrix();
                fill(0, 0, 0);
                noStroke();
                rect(0, 0.9352 * height, width, height);
                fill(98, 245, 31);
                textSize(25);
                text("10cm", 0.6146 * width, 0.9167 * height);
                text("20cm", 0.7190 * width, 0.9167 * height);
                text("30cm", 0.8230 * width, 0.9167 * height);
                text("40cm", 0.9271 * width, 0.9167 * height);

                textSize(40);
                text("Object: " + (iDistance > 40 ? "Out of Range" : "In Range"), 0.125 * width,
0.9723 * height);
                text("Angle: " + iAngle + " ° ", 0.52 * width, 0.9723 * height);
                text("Distance: ", 0.74 * width, 0.9723 * height);
                if(iDistance < 40) {
                        text("         " + iDistance +" cm", 0.775 * width, 0.9723 * height);
                }
                  textSize(25);
                fill(98, 245, 60);
                translate(0.5006 * width + width/2 * cos(radians(30)), 0.9093 * height - width/2
* sin(radians(30)));
                rotate(-radians(-60));
                text("30° ",0,0);

                resetMatrix();

                translate(0.497 * width + width/2 * cos(radians(60)), 0.9112 * height - width/2 *
sin(radians(60)));
                rotate(-radians(-30));
```

```
        text("60°",0,0);
          resetMatrix();
          translate(0.493 * width + width/2 * cos(radians(90)), 0.9167 * height - width/2
* sin(radians(90)));
        rotate(radians(0));
        text("90°",0,0);
        resetMatrix();

        translate(0.487 * width + width/2 * cos(radians(120)), 0.92871 * height - width/2
* sin(radians(120)));
        rotate(radians(-30));
        text("120°",0,0);
        resetMatrix();

        translate(0.4896 * width + width/2 * cos(radians(150)), 0.9426 * height - width/2
* sin(radians(150)));
        rotate(radians(-60));
        text("150°",0,0);
        popMatrix();
}

int StringToInt(String string)
{
        int value = 0;
        for(int i = 0; i < string.length(); ++i) {
                if(string.charAt(i) >= '0' && string.charAt(i) <= '9')
{
                        value *= 10;
                        value += (string.charAt(i) - '0');
                }}
          return value;}
```

Software operation procedure

In this experiment, we need to use Processing software, Processing software download address

https://processing.org/download/

We can download the corresponding version according to the model of our computer system

Download Processing. Processing is available for Linux, Mac OS X, and Windows. Select your choice to download the software below.

3.5.3 (3 February 2019)

Windows 64-bit   Linux 64-bit   Mac OS X

Windows 32-bit   Linux 32-bit

                        Linux ARM

                        (running on Pi?)

» Github   Read about the changes in 3.0. The list of revisions covers the differences
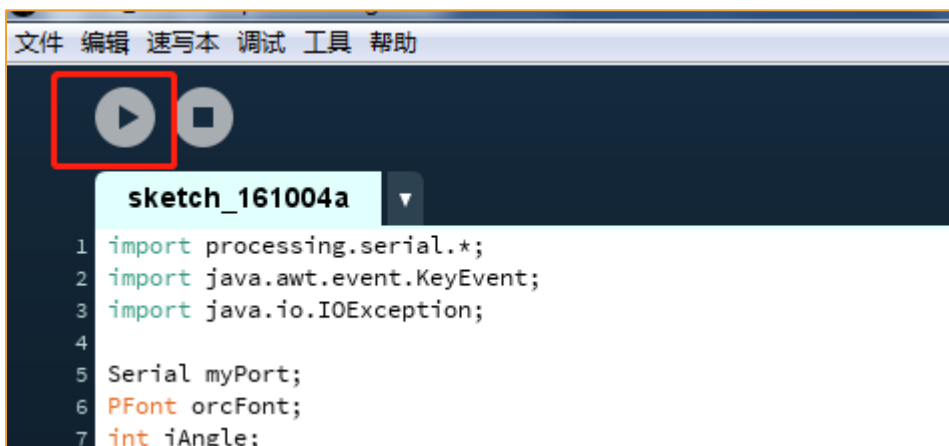» Report Bugs   between releases in detail.
» Wiki
» Supported Platforms

1) install the Processing software and open the ultrasonic radar experiment \Processing\sketch_161004a\ sketch_161004a.pde program

2) use Arduino IDE 1.6.5 software to open the program of ultrasonic radar experiment \Arduino_Radar\ arduino_radar.ino.

3) burn the arduino_radar. ino program to the Arduino UNO board

4) open the serial port monitor and observe the measurement data of ultrasonic ranging module printed by the serial port monitor

5) check the port number of Arduino UNO, if it is COM31; Set myPort = new Serial(this, "COM31", 9600) in the processing program; The port parameter of this function is changed to COM31, consistent with the Arduino port, otherwise an error will be reported.
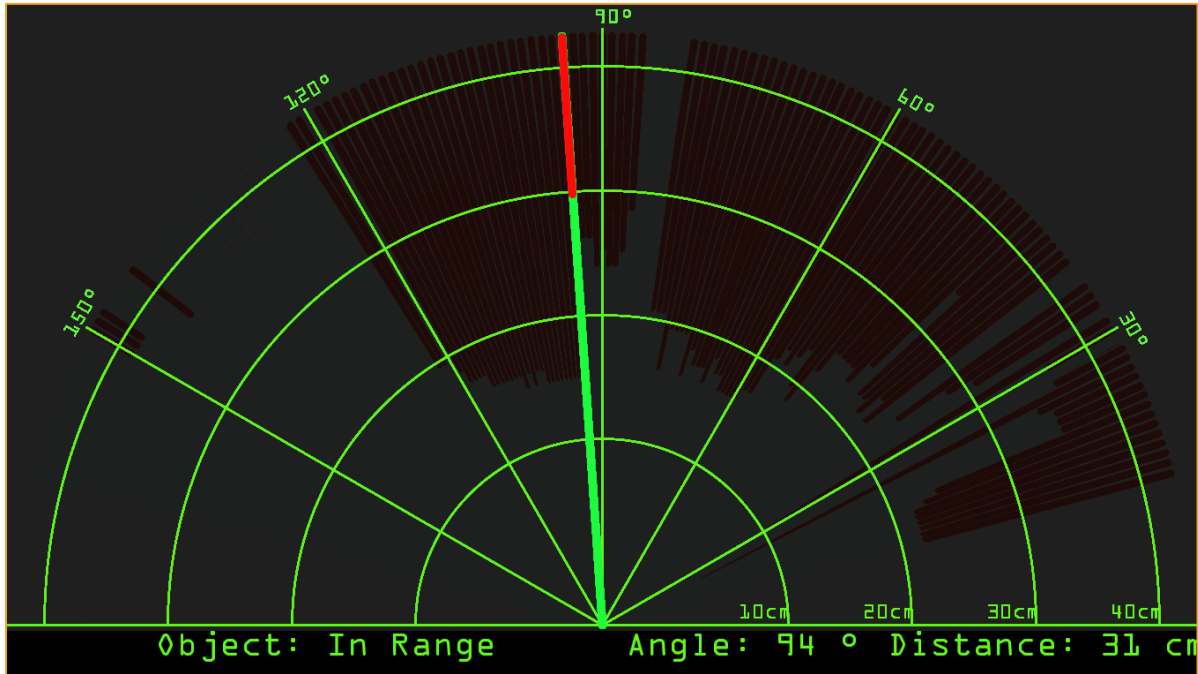
6) click the processing run button, as shown in the figure below;

7) after normal communication, the processing software will appear the following interface, and the ultrasonic radar will start to work;

Object: In Range   Angle: 94 ° Distance: 31 cm

The experimental results

Experiment purpose

◆ Learn the control principle of steering gear;

◆ Learning ultrasonic ranging method;

◆ Ultrasonic ranging module is used to realize the function of ultrasonic radar.

The component list

◆ Arduinos Uno motherboard
◆ Bread plate
◆ USB cable
◆ Servo Motor*1
◆ Jumpers
◆ Ultrasonic module
◆ Steering gear bracket

Wiring

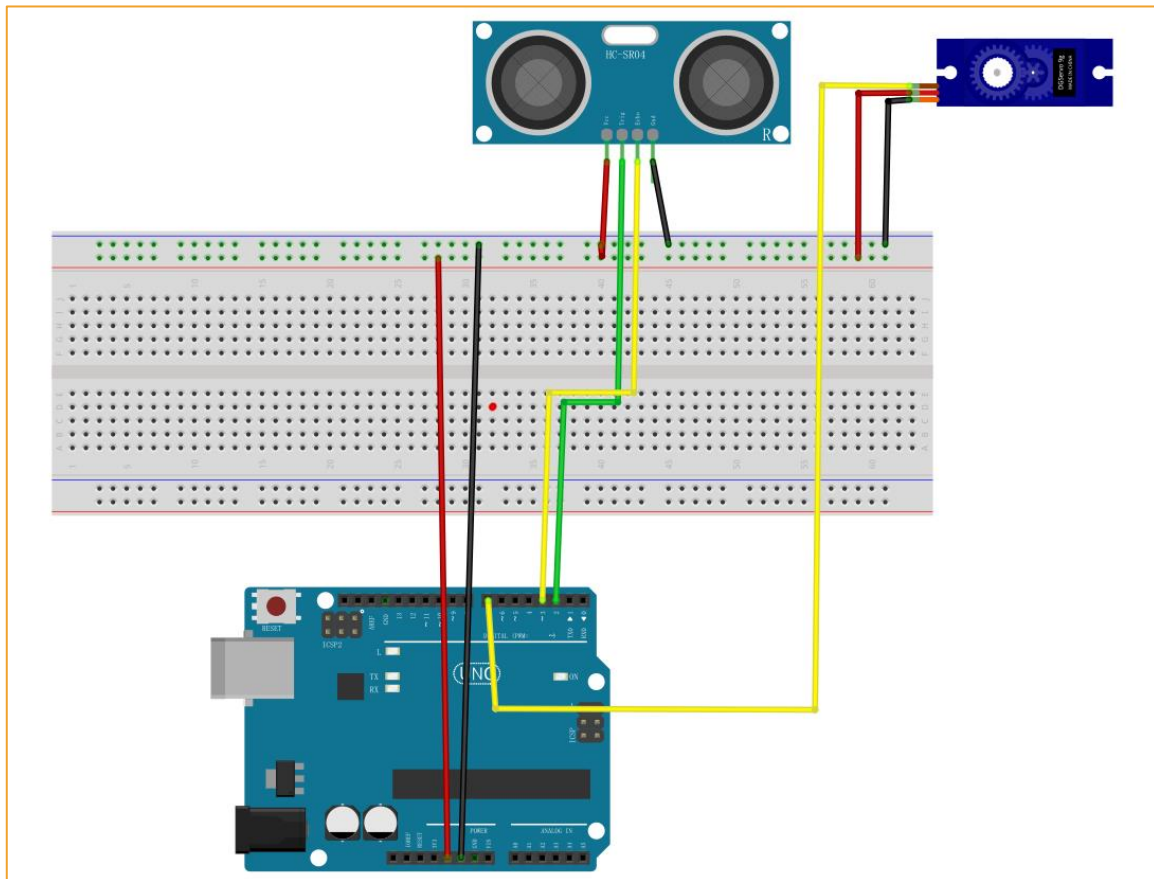| Arduino UNO	Servo motor
GND	GND
5V	VCC
7	S
Arduino UNO	Ultrasonic module
GNG	GND
3	Echo
2	Trig
5V	VCC

Code

```cpp
#include<Servo.h>
const int soundTriggerPin = 2;
const int soundEchoPin = 3;
const int motorSignalPin = 7;
const int startingAngle = 15;
const int minimumAngle = 15;
const int maximumAngle = 165;
const int rotationSpeed = 1;
Servo motor;
void setup(void)
{
    pinMode(soundTriggerPin, OUTPUT);
    pinMode(soundEchoPin, INPUT);
    motor.attach(motorSignalPin);
    Serial.begin(9600);
}
void loop(void)
{
    static int motorAngle = startingAngle;
```

```
   static int motorRotateAmount = rotationSpeed;
   motor.write(motorAngle);
   delay(30);
   SerialOutput(motorAngle, CalculateDistance());
   motorAngle += motorRotateAmount;
   if(motorAngle <= minimumAngle || motorAngle >= maximumAngle) {
       motorRotateAmount = -motorRotateAmount;
   }}
int CalculateDistance(void)
{
   //digitalWrite(soundTriggerPin, LOW);
   //delayMicroseconds(2);
   digitalWrite(soundTriggerPin, HIGH);
   delayMicroseconds(10);
   digitalWrite(soundTriggerPin, LOW);
     long duration = pulseIn(soundEchoPin, HIGH);
   float distance = duration * 0.017F;
   return int(distance);
}
void SerialOutput(const int angle, const int distance)
{

 String angleString = String(angle);
   String distanceString = String(distance);
   Serial.println(angleString + "," + distanceString);
}
```

Processing Code

```
import processing.serial.*;
import java.awt.event.KeyEvent;
import java.io.IOException;

Serial  myPort;
PFont orcFont;
int iAngle;
int iDistance;
void setup() {
size(1350, 760);
        smooth();

        myPort  =  new Serial(this, "COM31", 9600);
```

```
        myPort.clear();
        myPort.bufferUntil('\n');
            orcFont  =  loadFont("OCRAExtended-48.vlw");
}
void draw()
{

        fill(98, 245, 31);
        textFont(orcFont);
        noStroke();
        fill(0, 4);
        rect(0, 0, width, 0.935 * height);
        fill(98, 245, 31);


        DrawRadar();
        DrawLine();
        DrawObject();
        DrawText();
}
void serialEvent (Serial myPort)
{

        try {
                String data = myPort.readStringUntil('\n');
                    if (data == null) {
                        return;
                }
                  int commaIndex = data.indexOf(",");
                String angle = data.substring(0, commaIndex);
                String distance = data.substring(commaIndex+1, data.length()-1);
        iAngle  =  StringToInt(angle);
                iDistance  =  StringToInt(distance);
        }  catch(RuntimeException e)  {}
}
void DrawRadar()
{

        pushMatrix();
        translate(width/2, 0.926 * height);
        noFill();
        strokeWeight(2);
        stroke(98, 245, 31);


        // draws the arc lines
        DrawRadarArcLine(0.9375);
```

```
        DrawRadarArcLine(0.7300);
        DrawRadarArcLine(0.5210);
        DrawRadarArcLine(0.3130);

        // draws the angle lines
        final int halfWidth = width/2;
        line(-halfWidth, 0, halfWidth, 0);
        for(int angle = 30; angle <= 150; angle+=30) {
                DrawRadarAngledLine(angle);
        }
        line(-halfWidth * cos(radians(30)), 0, halfWidth, 0);
                popMatrix();
}
void DrawRadarArcLine(final float coefficient)
{
        arc(0, 0, coefficient * width, coefficient * width, PI, TWO_PI);
}
void DrawRadarAngledLine(final int angle){
        line(0, 0, (-width/2) * cos(radians(angle)), (-width/2) * sin(radians(angle)));
}
void DrawObject()
{

        pushMatrix();
        translate(width/2, 0.926 * height);
        strokeWeight(9);
        stroke(255,10,10);
        int pixsDistance = int(iDistance * 0.020835 * height);
        if(iDistance < 40 && iDistance != 0) {
                float cos = cos(radians(iAngle));
                float sin = sin(radians(iAngle));
                int x1 = +int(pixsDistance * cos);
                int y1 = -int(pixsDistance * sin);
                int x2 = +int(0.495 * width * cos);
                int y2 = -int(0.495 * width * sin);

                line(x1, y1, x2, y2);
        }
            popMatrix();
}
void DrawLine()
{
        pushMatrix();
```

```
        strokeWeight(9);
        stroke(30, 250, 60);
        translate(width/2, 0.926 * height);

        float angle = radians(iAngle);
        int x = int(+0.88 * height * cos(angle));
        int y = int(-0.88 * height * sin(angle));
        line(0, 0, x, y);
        popMatrix();
}
void DrawText()
{
        pushMatrix();
        fill(0, 0, 0);
        noStroke();
        rect(0, 0.9352 * height, width, height);
        fill(98, 245, 31);
        textSize(25);
        text("10cm", 0.6146 * width, 0.9167 * height);
        text("20cm", 0.7190 * width, 0.9167 * height);
        text("30cm", 0.8230 * width, 0.9167 * height);
        text("40cm", 0.9271 * width, 0.9167 * height);

        textSize(40);
        text("Object: " + (iDistance > 40 ? "Out of Range" : "In Range"), 0.125 * width,
0.9723 * height);
        text("Angle: " + iAngle + " ° ", 0.52 * width, 0.9723 * height);
        text("Distance: ", 0.74 * width, 0.9723 * height);
        if(iDistance < 40) {
                text("        " + iDistance +" cm", 0.775 * width, 0.9723 * height);
        }
          textSize(25);
        fill(98, 245, 60);
        translate(0.5006 * width + width/2 * cos(radians(30)), 0.9093 * height - width/2
* sin(radians(30)));
        rotate(-radians(-60));
        text("30° ",0,0);

        resetMatrix();

        translate(0.497 * width + width/2 * cos(radians(60)), 0.9112 * height - width/2 *
sin(radians(60)));
```

```
        rotate(-radians(-30));
        text("60° ",0,0);
          resetMatrix();
          translate(0.493 * width + width/2 * cos(radians(90)), 0.9167 * height - width/2
* sin(radians(90)));
        rotate(radians(0));
        text("90° ",0,0);
        resetMatrix();

        translate(0.487 * width + width/2 * cos(radians(120)), 0.92871 * height - width/2
* sin(radians(120)));
        rotate(radians(-30));
        text("120° ",0,0);
        resetMatrix();

        translate(0.4896 * width + width/2 * cos(radians(150)), 0.9426 * height - width/2
* sin(radians(150)));
        rotate(radians(-60));
        text("150° ",0,0);
        popMatrix();
}

int StringToInt(String string)
{
        int value = 0;
        for(int i = 0; i < string.length(); ++i) {
                if(string.charAt(i) >= '0' && string.charAt(i) <= '9')
{
                        value *= 10;
                        value += (string.charAt(i) - '0');
                }}
        return value;}
```

Software operation steps

In this experiment, we need to use Processing software, Processing software download address:

https://processing.org/download/

We can download the corresponding version according to our computer system model

Download Processing. Processing is available for Linux, Mac OS X, and Windows. Select your choice to download the software below.

3.5.3 (3 February 2019)

Windows 64-bit   Linux 64-bit   Mac OS X
Windows 32-bit   Linux 32-bit
                  Linux ARM
                  (running on Pi?)

» Github
» Report Bugs
» Wiki
» Supported Platforms

Read about the changes in 3.0. The list of revisions covers the differences between releases in detail.

1) install the Processing software and turn on the program of ultrasonic radar experiment \Processing\sketch_161004a\ sketch_161004a.pde

2) turn on the ultrasonic radar experiment \Arduino_Radar\Arduino_Radar. Ino program with Arduino IDE 1.8.9

3) burn the arduino_radar. ino program to the Arduino UNO board

4) open the serial port monitor to observe the measurement data of ultrasonic ranging module printed by the serial port monitor

Check the port number of Arduino UNO, say COM31;MyPort = new Serial(this, "COM31", 9600);The port parameter of this function is changed to COM31, which is consistent with the Arduino port, otherwise an error will be reported
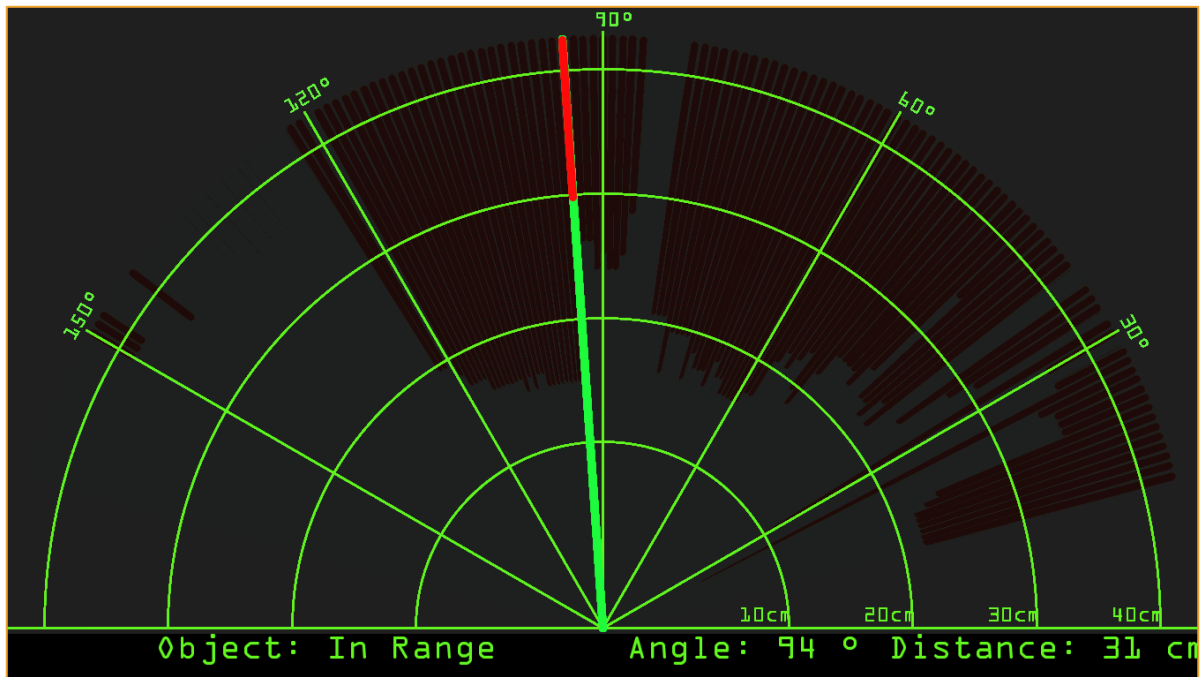
```
sketch_161004a    ▼
1  import processing.serial.*;
2  import java.awt.event.KeyEvent;
3  import java.io.IOException;
4
5  Serial myPort;
6  PFont orcFont;
7  int iAngle;
8  int iDistance;
9  void setup() {
10 size(1350, 760);
11     smooth();
12
13     myPort = new Serial(this, "COM31", 9600);
14     myPort.clear();
15     myPort.bufferUntil('\n');
16       orcFont = loadFont("OCRAExtended-48.vlw");
17 }
18 void draw()
```

6) click the run button of processing, as shown in the figure below;


```
文件 编辑 速写本 调试 工具 帮助

sketch_161004a    ▼
1  import processing.serial.*;
2  import java.awt.event.KeyEvent;
3  import java.io.IOException;
4
5  Serial myPort;
6  PFont orcFont;
7  int iAngle;
```

7) after normal communication, processing software will show the interface shown in the figure below, and then the ultrasonic radar will start to work;

Object: In Range   Angle: 94 ° Distance: 31 cm

Experiment Result