Uart adjustment rudder angle experiment

Introduction to Servo Motor

I guess you must have seen robots or high-tech products in American science-fiction films, at least heard of the noise of some moving automatic mechanical arms and audiences' scream. The noise comes from therotation of the steering gear.

The steering gear is a kind of position (angle) servo driver, it can be rotated to any angle between 0 and 180 degrees, then precisely stop at your command, so it is suitable for those control systems which require angle changing and keeping . At present, it has been widely used in high-grade remote control toys, such as model aircraft, including the model plane, submarine model and remote control robot. Steering gear is an unprofessional name, in fact it is a kind of servo motor, a set of automatic control device which consists of DC motor, reduction gear group, sensor and control circuit. What is the automatic control? The so-called automatic control — continuously adjusting the output deviation by using a closed-loop feedback control circuit — makes the system output constant.

Experiment Principle

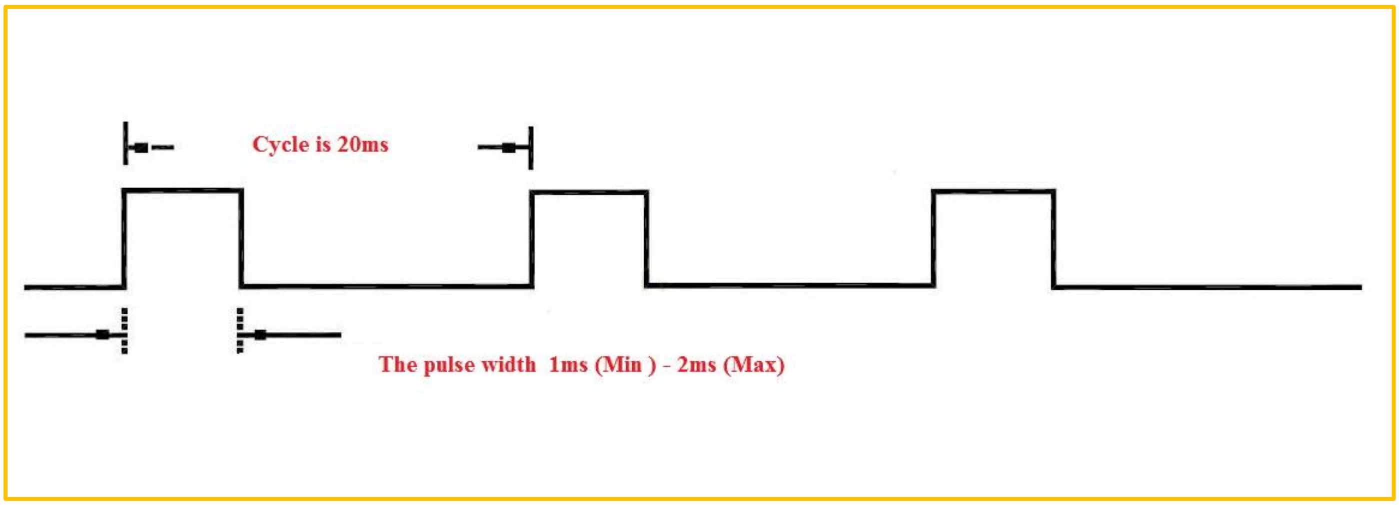
Use RaspberryPi motherboard to realize the experiment of adjusting the angle of the steering gear.

Component list

- RaspberryPi motherboard
- T-type expansion board
- Power cord
- SG90 steering gear
- Several jumpers

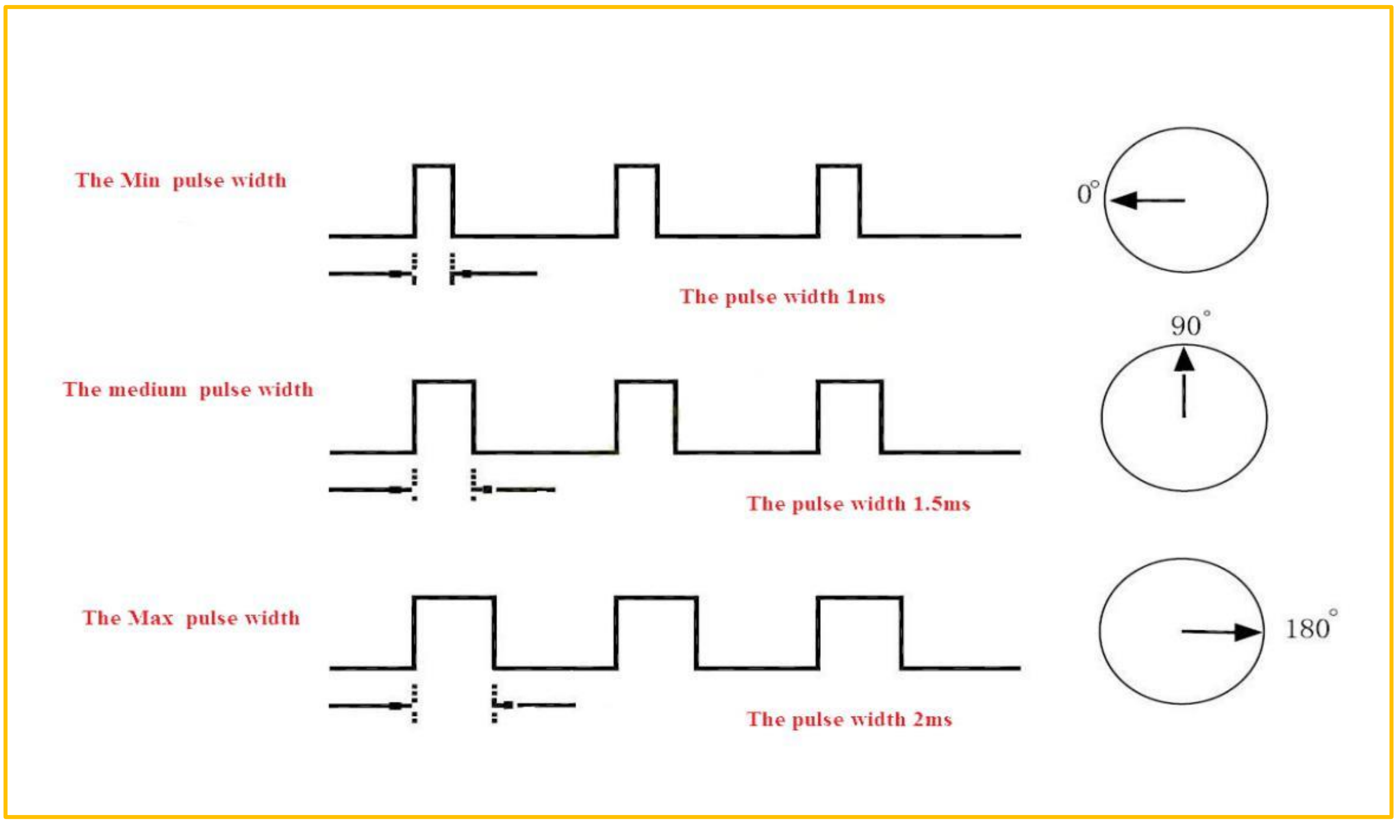
working principle

The steering gear servo system can be controlled by variable bandwidth pulse, the control line is used to transmit pulse. The parameters of the pulse consist minimum value, maximum value and frequency. In general, the cycle of the reference signal of the steering gear is 20ms, the bandwidth is 1.5ms. The reference signal is from the middle position. The steering gear has the maximum rotation angle, the middle position refers to the volumes from this position to the minimum angle and the maximum angle are exactly identical. The most important part, the maximum rotation angle varies with different steering gears, but of which the bandwidth of the middle position is certain, that is 1.5 ms.

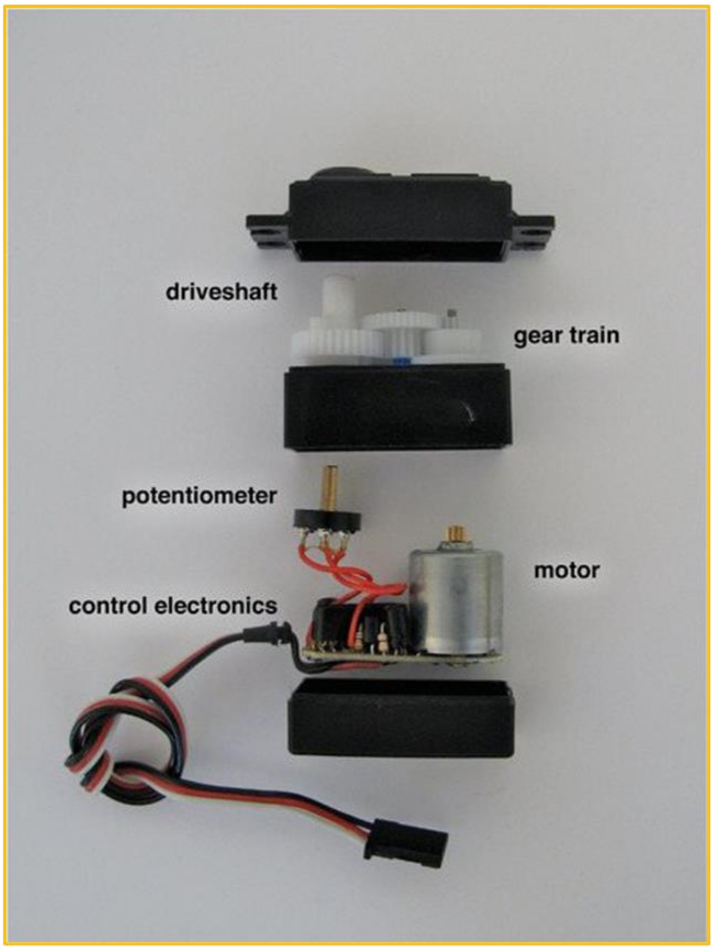


The rotation angle is produced by the continuous pulse from control line. This method is called pulse modulation. The length of pulse decides the rotation angle of steering gear. For example: the steering gear rotates to the middle position by 1.5 millisecond pulse(for 180° steering gear, the middle position is 90°). When the control system issues commands to move the steering gear to a particular position and make it keep a certain angle, then the influence of the external force won't change the angle, but the ceiling is its biggest torsion. Unless the control system continuously issues pulse to stable the steering angle, the angle will not always stay the same.

When the steering gear receives a pulse less than 1.5ms , the output shaft will take the middle position as standard and rotate a certain angle counterclockwise; when the received pulse is greater than 1.5ms, then the output shaft rotates clockwise. Different brands of steering gears, and even the same brand of different steering gears, the maximum and minimum value could be different.



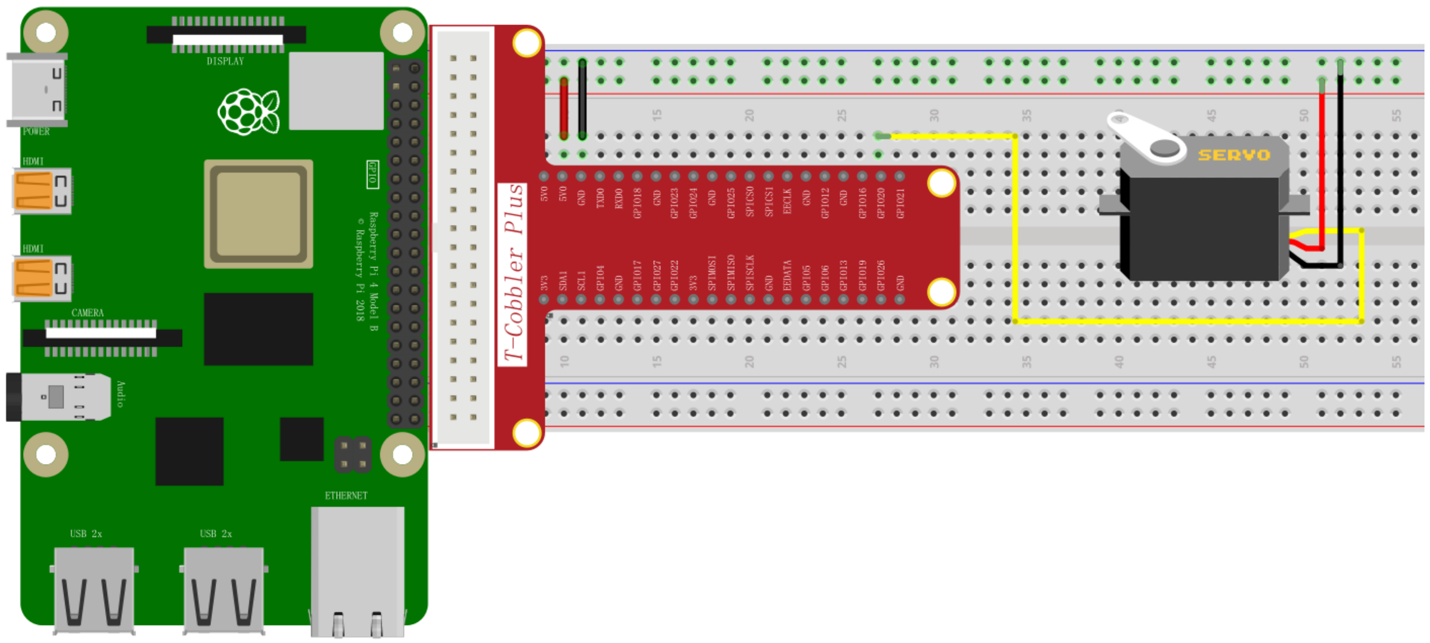
Internal Structure of Steering Gear

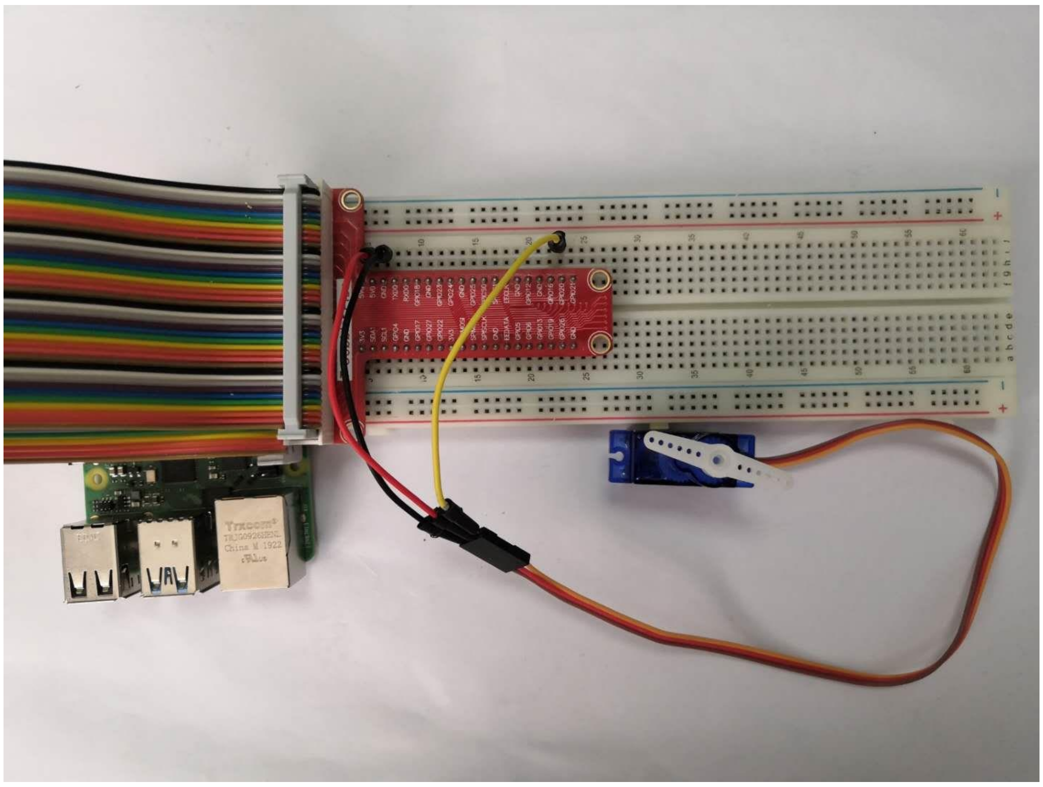


Experimental principle

The servo signal line is connected to the RaspberryPi motherboard. After running the program, enter the angle of the servo to be adjusted in the serial monitor to turn the servo to the corresponding angle.接线

RaspberryPi	Steering gear
GND	Brown (black line)
VCC	Red (red line)
28(wiringPi)/20(BCM)	Orange (yellow line)





C++ program

```
#include <stdio.h>
#include <string.h>
#include <wiringPi.h>
#include <wiringSerial.h>
#include <iostream>

using namespace std;
int cyc =20000;
int pwmPin = 28;
int value;

void pwm_fun(int temp)
{
    digitalWrite(pwmPin, HIGH);
    delayMicroseconds(500+temp*500/45);
    digitalWrite(pwmPin,0);
    delayMicroseconds((cyc-(500+temp*500/45)));
}

int main()
{
    wiringPiSetup();
    pinMode(pwmPin, OUTPUT);
    while(1)
    {
        std::cout<<"input please:"<<endl;
        std::cin>>value;
        pwm_fun(value);
        delay(500);
    }
}
```

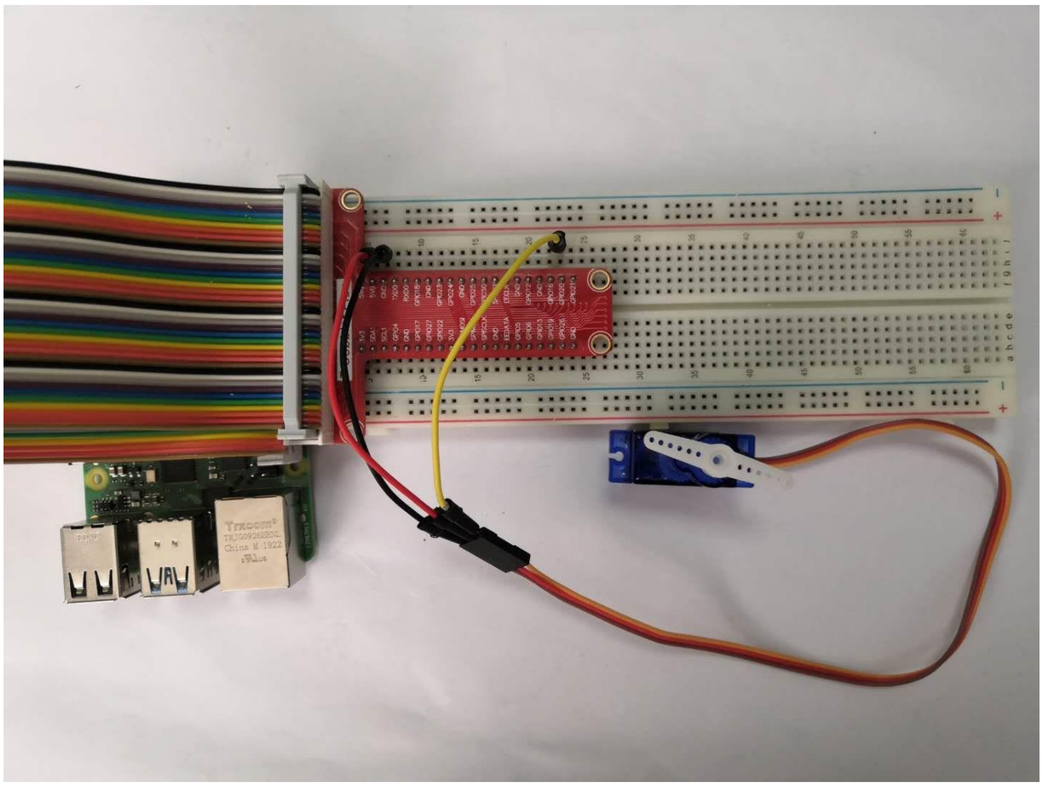
Python program

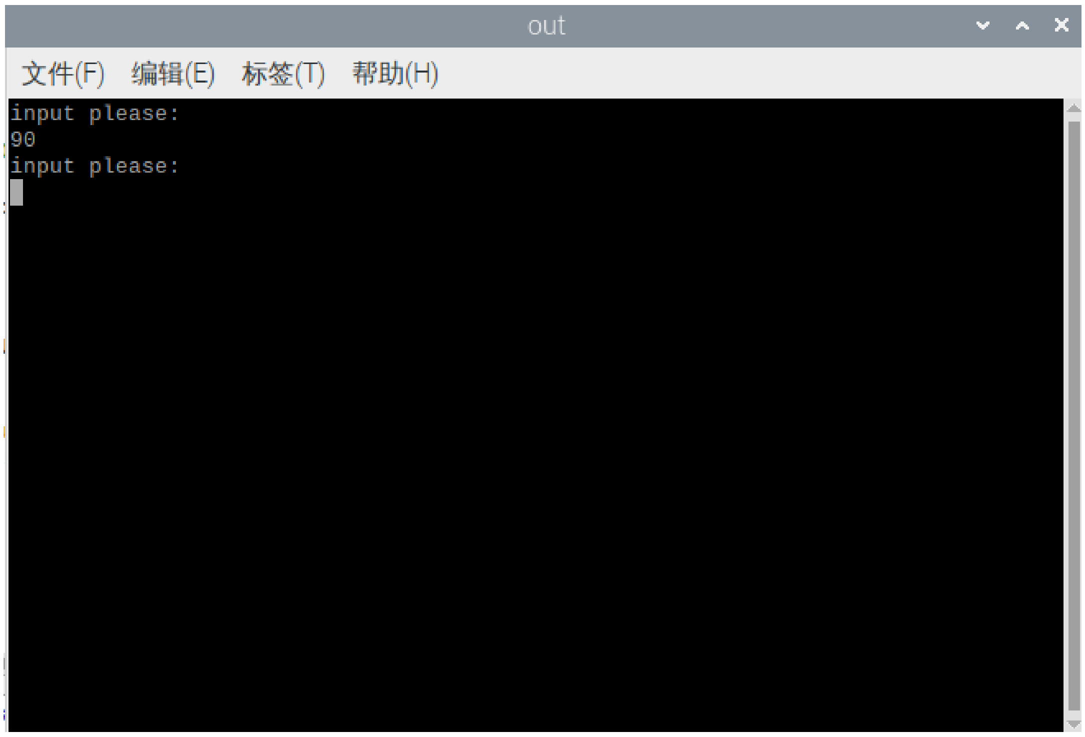
```
# -*- coding: utf-8 -*-
#!/usr/bin/env python

import RPi.GPIO as GPIO
import time
import signal
import atexit

atexit.register(GPIO.cleanup)

servopin = 20
GPIO.setmode(GPIO.BCM)
GPIO.setup(servopin, GPIO.OUT)
cyc = 0.2

def pwm_change(temp):
    GPIO.output(servopin ,True)
    time.sleep(0.0005+float(temp)*0.0005/45)
    print(0.0005+float(temp)*0.0005/45)
    GPIO.output(servopin , False)
    time.sleep(0.02-(0.0005+float(temp)*0.0005/45))

while True:
    val=input("input num:")
    pwm_change(val)
GPIO.cleanup()
```

Java program

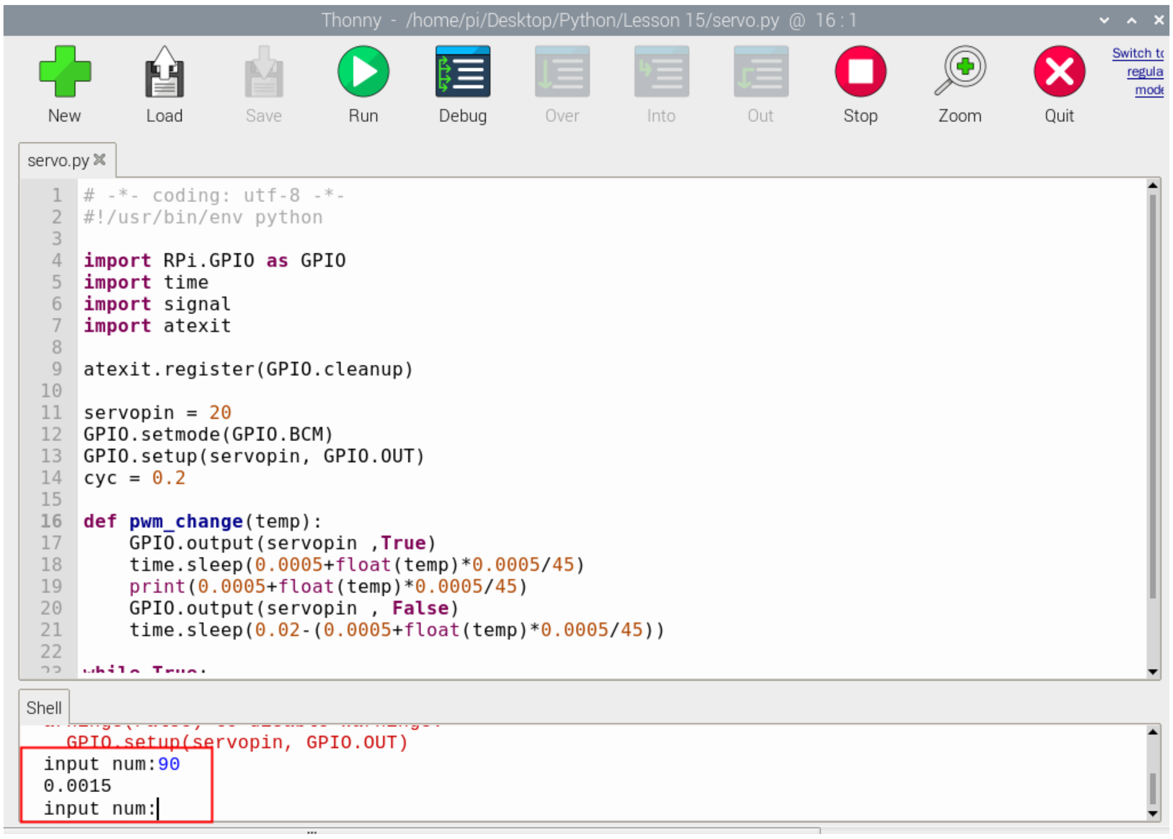
```
import com.pi4j.wiringpi.Gpio;
import com.pi4j.io.gpio.impl.PinImpl;
import com.pi4j.wiringpi.GpioInterrupt;
import com.pi4j.wiringpi.GpioInterruptListener;
import com.pi4j.wiringpi.GpioInterruptEvent;
import java.util.Scanner;


public class Adjustment_servo {
    static int LEDPIN  = 21;
    static int value;
    static {
        // setup wiring pi
        if (Gpio.wiringPiSetup() == -1) {
            System.out.println(" ==>> GPIO SETUP FAILED");
        }

        Gpio.pinMode(LEDPIN, Gpio.OUTPUT);
    }

    static void led_pwm(int val){
        Gpio.digitalWrite(LEDPIN, Gpio.HIGH);
        Gpio.delayMicroseconds(500 + val*500 / 45);
        Gpio.digitalWrite(LEDPIN,Gpio.LOW);
        Gpio.delayMicroseconds((20000 - (500 + val*500 / 45)));
    }

    public static void main(String args[]) throws InterruptedException{

        for ( ; ; ) {
            System.out.println("输入的数据范围为0~180");
            Scanner scan = new Scanner(System.in);
            // 判断是否还有输入
            if (scan.hasNext()) {
                String str1 = scan.next();
                value = Integer.parseInt(str1);
                System.out.println("输入的数据为: " + value);
            }

            for (int i = 0; i < 10; i++) {
                Adjustment_servo.led_pwm(value);
            }
        }
    }
}
```

Experimental results





```
input please:
90
input please:
```



```
# -*- coding: utf-8 -*-
#!/usr/bin/env python

import RPi.GPIO as GPIO
import time
import signal
import atexit

atexit.register(GPIO.cleanup)

servopin = 20
GPIO.setmode(GPIO.BCM)
GPIO.setup(servopin, GPIO.OUT)
cyc = 0.2

def pwm_change(temp):
    GPIO.output(servopin ,True)
    time.sleep(0.0005+float(temp)*0.0005/45)
    print(0.0005+float(temp)*0.0005/45)
    GPIO.output(servopin , False)
    time.sleep(0.02-(0.0005+float(temp)*0.0005/45))

while True:
```

```
  GPIO.setup(servopin, GPIO.OUT)
input num:90
0.0015
input num:
```

Control the rotation angle of the servo through the Raspberry Pi, and input the angle value in the serial port to control the rotation of the servo at any angle.