Design Experiment of Smart Lamp Based on Python Coding

Han Zheng

302, No.16, 1515 Lane Lujiazui International Huacheng, Zhangyang Road, Pudong District, Shanghai, 200135, China
Corresponding Author: Han Zheng, Email: zhenghan_study@163.com

Abstract

Energy furniture is emerging as an alternative to traditional furniture in a rapidly developing society. Smart lighting is an essential part of the smart furniture ecosystem. This experiment is to design an experiment of Smart Lamp Based on Python Coding. The main use is to control the switching of luminaires using sensors, thus replacing conventional switches. The Raspberry Pi is a mini-computer on the ARM Linux architecture designed for learning computer programming education, and many microcontroller modules are also available on the Raspberry Pi, so the sensors can be connected by calling python code in the Raspberry Pi. The experiment resulted in two consecutive waves of the hand in front of the ultrasonic rangefinder to control the switching on and off of the luminaire, but there would be an isolated number of failures, with a measurement accuracy of up to 80%. The most accurate operation was waving the hand 13cm from the sensor with a waving frequency of 10 to 12 seconds.

Keywords

Python coding; Raspberry Pi; Smart lamp; Relay

Introduction

With the rapid development of technology, people are looking for an efficient, convenient, and intelligent life. Smart homes stand out at this time. Compared to traditional furniture, the advantages over traditional furniture are that some furniture retains its value, is safe and secure, and has certain artistic and aesthetic features(Lv, 2014). However, traditional furniture is also heavy with materials, difficult to handle, and expensive, while quality varies and furniture is difficult to repair if damaged. Smart homes have obvious advantages, improving the comfort of life, ease of use, environmental protection, and energy-saving while also allowing for many personalized designs that cannot be achieved by traditional furniture. Any emerging product is like a double-edged sword. Smart home products also have a variety of problems in front of them—the use of complex products, poor compatibility of various products, rapid loss of electronic components, quick replacement, high prices, slow response time, and little consumer awareness of smart homes. Many companies in China such as Baidu, Haier, and Gree are now embedding artificial intelligence into their smart

Citation: Han Zheng. (2022) Design Experiment of Smart Lamp Based on Python Coding The Journal of Young Researchers 4(20): e20220706

Received on June 7, 2022; Accepted on June 17, 2022; Published on July 6, 2022

home products, products and Xiaomi has a dedicated smart home industry chain and smart home ecosystem. Foreign companies such as Amazon Echo, Google Home, and Apple Homekit of which Apple Homekit is the leader in voice control for smart homes in the US, London Oxford launched an integrated experiential showroom for smart homes in April 2016, and in 2017 Panasonic Japan launched "Panasonic Home + Whole House Intelligence" strategy (Rong & Lian, 2017). So it seems that the smart home is gradually emerging in the market and is also maturing.

This research is about smart lamps. Smart lamps belong to the popular field of the smart home. Compared with the traditional electric key switches, there is a wide range of smart lamps, remote control switches, object distance induction switches, infrared induction switches, voice-controlled switches, intelligent voice switches, etc. The technologies commonly involved in smart lamps are radio signal control, capacitive sensing, sensor distance measurement, voice recognition, as well as real-time updates on the status of the lamps (Zhou, 2017). Smart lamps avoid the limitation of the number of traditional lamp switches, as well as the limitation of space, and reduce the waste of electricity to a certain extent. At present, most of the intelligent lighting market in China is mainly based on dimming and colour mixing, while other countries are based on dimming and colour mixing in addition to the Internet of Things (An, 2018).

The purpose of this research is to initially build a smart lamp environment, mainly using measurement applications to achieve the switching of lamps. This work is to build a smart lamp to investigate how these work and to hypothesize about future developments. This will be done by using python programming to connect a sensor to measure the height of the waving hand to control the switch of a small light. The experiment aims to simulate the simple smart home working, at the same time design principles to achieve an efficient and convenient smart lamp life, explore the future development and technical improvement of the smart home market, let people understand the principle of smart home, and let businessmen understand the needs of consumers and where technical improvements can be made.

Literature Review

Control Methods

Based on the current academic design and control of smart homes there are two main ways of implementation, the first is to design an App through assembly language while using a variety of wireless protocols to achieve remote control of lights. The second is to use a single chip microcomputer (e.g. AVR series, STM32 series, 51 series) or Raspberry Pi to build a cloud server to detect the status of smart home devices in real-time.

The advantage of designing an app to control each device is that there are a large number of smart homes in the home, forming a certain smart home ecology. The interactive interface on the app, therefore, allows the status of each device to be displayed and monitored, making it easy to control all the devices from the mobile phone as the master console and also to adjust them remotely.

After observation, the disadvantage is that the app is aimed at a limited age group and requires a complete smart home environment. App itself is software on a smartphone and it will take some time for the elderly to get started. Building an App also demands occasional maintenance and is likely to involve some privacy and security issues. If there is only a single smart light fixture in the home that does not require an app to control and monitor, a traditional physical switch or infrared remote control may be less time-consuming and less expensive than app development.

This project controls smart lamps by using a Raspberry Pi connected to a modular sensor and using the python programming language to control the work of the sensor. The single-chip microcomputer is an integrated circuit chip with a high processing power for data while running at low power consumption. The Raspberry Pi is an ARM architecture with a Linux system and a based microcomputer designed for learning computer programming education. The reason for using the Raspberry Pi in this experiment is that it does not require a chip with a high level of data processing power, but rather a console that is easy to use and comes with a Python or C++ programming environment, as well as a wide range of sensors to build the device. Therefore, this experiment is different from the principle of making an APP and building a cloud server to implement a smart home. People only need to connect to the Raspberry Pi or remotely to the Raspberry Pi and call the code program to achieve the effect of smart lamps.

Sensors Used in Smart Home

Many academics are using infrared sensors or light sensors as smart home sensors. Infrared sensors are more commonly used in home security environments and no contact temperature measurements. For example, infrared sensors detecting the proximity of people can be combined with a buzzer alarm. As well as infrared sensors can also be used to measure people's temperature to estimate if family members might get ill. Light sensors are mostly used to dim the light and colour of intelligent lamps. The HC-SR04 ultrasonic ranging module is used in this experiment. The sensor calculates the distance from the time the ultrasound hits an obstacle and reflects it, and HC-SR04 is also widely used by many academics. It is highly accurate, measuring between 2cm and 400cm and needs to be non-contact. HC-SR04 has a favourable price, making it a very cost-effective sensor (Wang, 2014).

Lamps for Experimental Testing

At present, there are mostly energy-saving light bulbs and LED bulbs on the market. Energy-saving energy lamps are mainly filament-end emitted electrons, electrons collide with argon atoms in the lamp, and argon atoms then collide with mercury atoms, thus generating ionization. The lampion state is formed in the lamp. The voltage passes through and then emits ultraviolet light. The ultraviolet light excites the phosphor to emit light. Energy-saving lamps have high luminous efficacy, small size, and long life but can cause mercury volatilization if the lamp is accidentally broken, which may cause harm to humans. On top of the advantages of energy-saving lamps, LED lamps also have the advantages of easy light-gathering, abundant colours, and strong monochromaticity. At the same time, the disadvantages are high cost, high price, susceptibility to high temperatures, the need for good heat dissipation, etc. Both bulbs are planned to be used in this study.

Programming Languages

Depending on the development phase, developers choose different programming languages such as C, C++, Java, JavaScript, Python, and PHP. C is suitable for writing hardware layer code that does not require powerful processing power and can use RAM directly. Java is written without hardware limitations. Also, functional code can be transferred to any chip or smartphone and server via the JAVA Virtual Machine (JVM).

The reason for using Python in this experiment is that it has a large library of code that is easy to learn, widely used, and has a simpler and cleaner syntax than other programming languages. This experiment was written by calling Python's GPIO code library. The Raspberry Pi also has a dedicated IDLE for Python.

Methods

Experimental Design

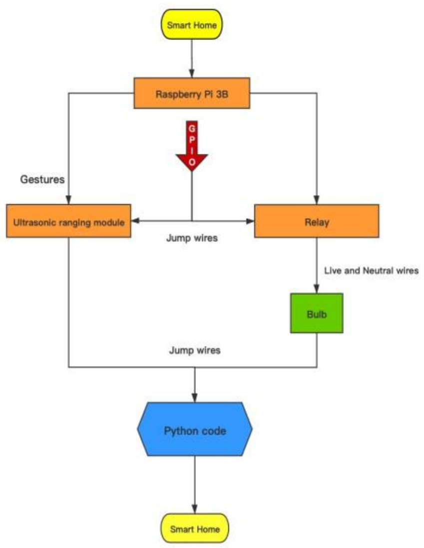

Figure 1. Flow chart of the experimental framework

This experiment aims to wave one hand back and forth to control the light on and off, thus replacing the traditional electric key to control the light.

Above all, the Raspberry Pi has been used to power the modules, while writing code to control them. Two main modules are used in the experiment: an ultrasonic rangefinder and a relay; a test module: LEDs; two electrical appliances: an LED bulb and a CFL bulb. The Raspberry Pi is connected to the ultrasonic rangefinder and relay using jumper wires to power the GPIO pins and transmit electrical signals. The relay and LED bulb are connected using live and neutral wires. The rangefinder and relay are linked to the GPIO pin at VCC for 5V power and the LED bulb is powered at 220v. The programming language uses the Python high-level programming language for altering the individual module parameters and controlling them. The experiment is designed in three main steps. Firstly, the rangefinder recognizes the gesture. Secondly, a relay is used to control the switch of the bulb. And thirdly, the rangefinder and the bulb are combined to achieve the response of the rangefinder recognizing the gesture and the bulb making the switch.

The experimental test bulb was first replaced with a LEDs module.

Experimental Materials

Raspberry Pi

The mainframe Raspberry Pi is a Linux-based microcontroller computer developed by the Raspberry Pi Foundation in the UK. Each generation is powered by an ARM processor from Broadcom and has between 2GB and 8GB of memory, using SD or TF cards as memory cards (Figure 2). The total size of the Raspberry Pi is 85mm*56mm (Figure 3). Raspberry Pi OS is the official operating system of the Raspberry Pi. The Raspberry Pi is relatively rich in modular hardware and has a wide range of built-in programming languages such as Python, C++, C, and more(Richardson & Wallace, 2013). Linux-based systems allow easy access to SSH remote login, SCP remote copy across machines, etc. The Raspberry Pi uses the GPIO pin pins as general-purpose inputs and outputs (Figure 4).

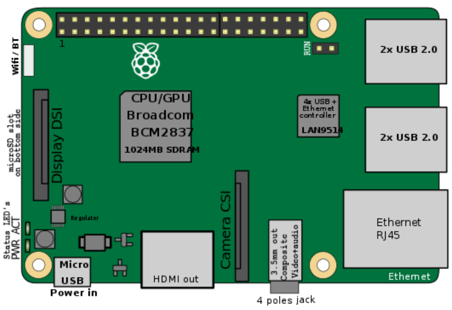

Figure 2. Raspberry Pi panel diagram

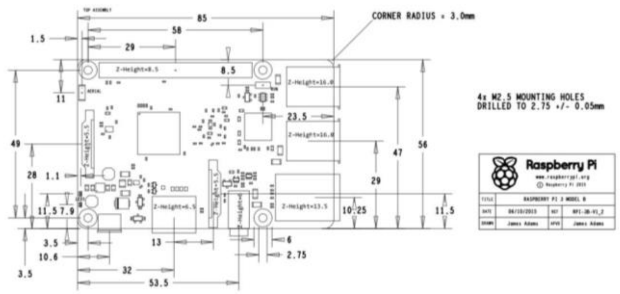

Figure 3. Raspberry Pi parameter diagram

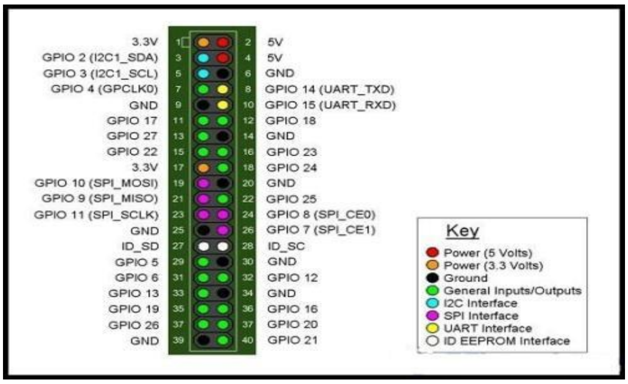

Figure 4. GPIO corresponding pin positions

HC-SR04 Ultrasonic Ranging Module

Ultrasonic ranging module HC-SR04 provides a 2cm - 400cm non-contact measurement function, and the ranging accuracy can reach 3mm. The modules include four pins that are VCC which applies 5V voltage, TRIG which is trigger control signal input, ECHO which is signal output, and GND which is the ground connection pin respectively.

The timing diagram needs to supply a 10uS pulse to the trigger input, the module will emit an 8-cycle burst of ultrasound at 40kHz and raise its echo (Figure 5). The Echo is a distance object that is pulse width and range in proportion.

Distance = high level time* velocity(340M/S)/2

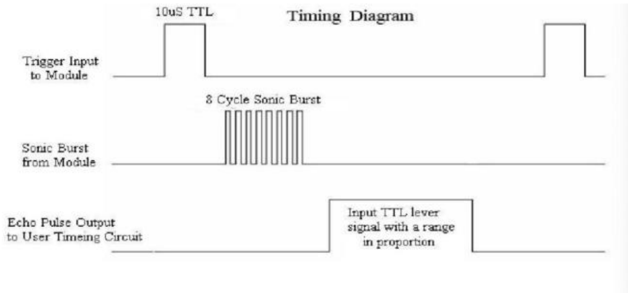

Figure 5. Ultrasonic ranging module schematic

Relay

Relay is just like a switch, which is automatic operation instead of manual operation with an electrical signal. Relays are used wherever it is necessary to control a high power or high voltage circuit with a low power circuit. Relays play a role in the circuit for automatic regulation and safety protection of the circuit. There are three types of contacts: normally open, common, and normally close.

One of the circuits is connected to the coil and the other includes the switch contact A1 and A2. When a small current passes through the coil of the relay, the iron core attracts a movable arm, and the contacts are connected to A1 and A2 close, completing the second circuit (Figure 6) (Sang, Jones, Chadha & Woodside, 2014).

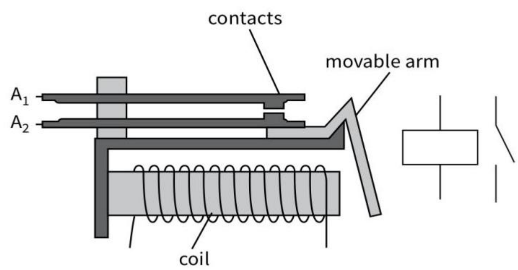

Figure 6. A relay and its circuit symbol

Luminous Materials

The test module is seven colour flashing LEDs with 5mm round (Figure 7). That standard forward voltage is 2.5-6v. The test lamp is three primary colour electronic energy-saving lamps, whose power is 5 Watt, length is 105mm, diameter is 35mm, frequency is 50HZ, colour temperature is 6500K, screw mouth is E27, and 2U thick tube.

The experience lamp is LED whose power is 5W, colour temperature is 6500K, voltage is 220-240v, the lumen is 500lm, frequency is 50HZ - 60HZlength is 108mm, diameter is 60mm, and screw mouth is E27.

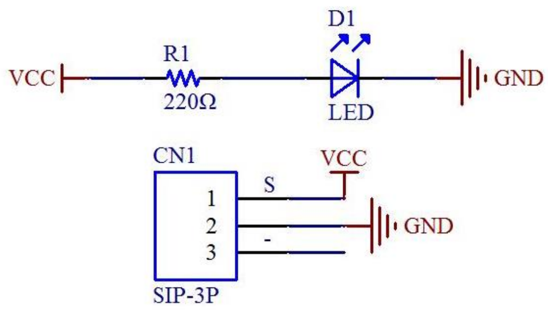

Figure 7. LEDs module diagram

Jumper Wire

A jumper wire is an electrical wire that normally interconnects the modules and the pins of the raspberry pi. It has three versions: male to male, male to female, and female to female. The difference between males and females is that the ends of males have a pin protruding and plugging into other modules or sensors, but the ends of females do not have that.

Live and Neutral

For the power supply line of civil power, the AC power supply voltage of the city power is 220 volts (V) (different countries are different, China is 220V). It includes a neutral wire [N] and a live wire [L].

Experimental Specifics

The experiment aimed to control the switch of a small light bulb by waving the hand twice in succession.

Testing the Use of Ultrasonic Ranging Module

The VCC pin of the ultrasonic sensor is connected to the Raspberry Pi GPIO 5V voltage pin. The GND pin is connected to the Raspberry Pi GPIO GND pin. The TRIG and ECHO are linked to GPIO17 (Output) and GPIO18 (Input) respectively as written in python code (corresponding to PINs 11 and 12). One of the transmitters of the ultrasonic distance measurement module emits ultrasonic waves and records the time at the same time. When an obstacle is encountered, the reflected wave is received at the other end and the time is recorded again, thus calculating the distance to the obstacle.

The program code is as follows：

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
import time
TRIG = 11
ECHO = 12

def setup():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(TRIG, GPIO.OUT)
    GPIO.setup(ECHO, GPIO.IN)
def distance():
    GPIO.output(TRIG, 0)
    time.sleep(0.000002)

    GPIO.output(TRIG, 1)
    time.sleep(0.00001)
    GPIO.output(TRIG, 0)

    while GPIO.input(ECHO) == 0:
        a = 0
    time1 = time.time()
    while GPIO.input(ECHO) == 1:
        a = 1
    time2 = time.time()

    during = time2 - time1
    return during * 340 / 2 * 100

def loop():
    while True:
        dis = distance()
        print dis, 'cm'
        print ''
        time.sleep(0.3)

```

```
def destroy():
    GPIO.cleanup()

if __name__ == "__main__":
    setup()
    try:
        loop()
    except KeyboardInterrupt:
        destroy()

```

Figure 8. Ultrasonic sensor code

“#! /user/bin/python” is telling the operating system to call the Python interpreter under /user/bin when executing this script;

“import RPi.GPIO as GPIO” is to import the GPIO library;

“import time” is the import time library;

“TRIG = 11” for the TRIG corresponding to the pin on the Raspberry Pi; “ECHO = 12” for the ECHO corresponding to the pin on the Raspberry Pi; “def setup()” for defining a function to act as an initialization of the ultrasonic module; “GPIO.setmode(GPIO.BOARD)” to adopt the actual physical pin to the GPIO port; “GPIO.setup(TRIG, GPIO.OUT)” TRIG set to output mode;

“GPIO.setup(ECHO, GPIO.IN)” ECHO set to input mode;

“def distance()” as ultrasonic distance calculation function;

“GPIO.output(TRIG, 0)” initial start;

“time.sleep(0.000002)” delay time of 2us;

“GPIO.output(TRIG, 1)” ultrasonic start signal;

“time.sleep(0.00001)” delay 10us;

“GPIO.output(TRIG, 0)” set to low;

“GPIO.input(ECHO) == 0:” wait for return signal;

“time1 = time.time()” get current time;

“while GPIO.input(ECHO) == 1:” return signal cutoff information;

“time2 = time.time()” to get the time at the time;

“during = time2 - time1” to calculate the time difference;

“return during * 340 / 2 * 100” to find the distance;

“def loop():” define the loop function to output the distance;

“dis = distance()” to get the ultrasound calculate distance;

“print dis, ‘cm’”output distance value cm as a unit;

“print ‘ ‘ “newline;

“time.sleep(0.3)” delay 300ms;

“def destroy():” release the resource function;

“GPIO.cleanup()” frees resources;

“if __name__ == “__main__”:”Program entry;

“setup()” calls the initialization function;

“loop()” Calling loop functions;

“except KeyboardInterrupt: destroy()” When Ctrl+C is pressed, the destroy() subroutine will be execute.

Test the Ultrasonic Sensor and LEDs Module Connection

The LEDs replace CFLs first. The LEDs were controlled by code to test whether this method could be used in the final experiment to connect the rangefinder and CFL. Connect the ultrasonic rangefinder to GPIO 17 and GPIO 18 (corresponding to Pins 11 and 12) and the LEDs to GPIO 22 and GPIO 23 (corresponding to Pins 15 and 16). The effect is to switch on when the hand is less than 50cm from the LEDs and to switch off when it is less than 50cm.

Explanation of the main code: If the wave is less than 50cm, the count variable is c+1; if the count variable is an odd number, the output is “light initial” and the light is switched on; if the light is kept on, the output is “light”; if the count variable is an even number, the output is “close” and the light is switched off. The emitted frequency is 0.5s (Figure 10).

```
def loop():
    global a
    global b
    a=0
    b=3
    c=0
    while True:
        dis = distance()
        time.sleep(0.1)
        if dis<=50 :
             c=c+1
             if c%2!=0:
                b=1
                print "light initial"
                print ''
                setColor(colors[0])
             elif c%2==0:
                b=2

        if b==1:
            print "light"
            a=a+1
        elif b==2:
            print"close"
            setColor(colors[2])
        time.sleep(0.5)
```

Figure 9. The main code of control ultrasonic and LEDs

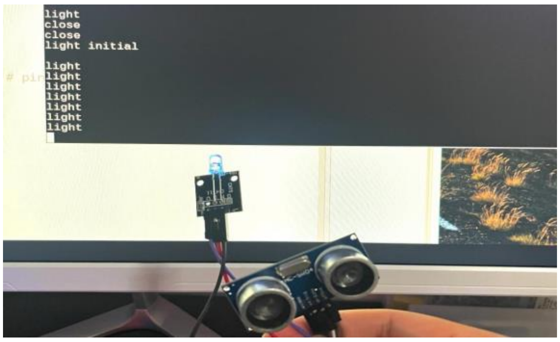

Figure 10. The effect of running the code

The code adds a control program for the LEDs based on the sensor code.

Here is the code added:

```
colors = [0xFF00, 0x00FF, 0x0FF0, 0xF00F]

for i in pins:
    GPIO.setup(pins[i], GPIO.OUT)
    GPIO.output(pins[i], GPIO.HIGH)

p_R = GPIO.PWM(pins['pin_R'], 2000)
p_G = GPIO.PWM(pins['pin_G'], 2000)

p_R.start(0)
p_G.start(0)

def map(x, in_min, in_max, out_min, out_max):
    return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min

def setColor(col):
    R_val = (col & 0x1100) >> 8
    G_val = (col & 0x0011) >> 0

    R_val = map(R_val, 0, 255, 0, 100)
    G_val = map(G_val, 0, 255, 0, 100)

    p_R.ChangeDutyCycle(R_val)
    p_G.ChangeDutyCycle(G_val)
```

Figure 11. Additional code of ultrasonic and LEDs

“pins = {‘pin_R’:15, ‘pin_G’:16}” Set the LEDs pins are 15 and 16;

“colours = [0xFF00, 0x00FF, 0x0FF0, 0xF00F]” Put the ASCII of the colour in the list;

“GPIO.setup(pins[i], GPIO.OUT)” Set pins’ mode is output;

"GPIO.output(pins[i], GPIO.HIGH)" Set pins to high(+3.3V) to off led;

"p_R = GPIO.PWM(pins['pin_R'], 2000)" Set frequency to 2KHz;

"p_R.start(0)" Initial duty Cycle = 0(LEDs off);

"def map(x, in_min, in_max, out_min, out_max):

return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min Scaling functions;

def setColour(col)" The function of set colours;

"R_val = map(R_val, 0, 255, 0, 100)" Reduces the range from 0-255 in the same proportion to between 0 and 100;

"p_R.ChangeDutyCycle(R_val)" Change duty cycle;

Construction of Lamp Connection Relay

Energy-saving lamps are high voltage appliances, so a relay which is a low voltage module is selected to control high voltage appliances. The prepared materials include a live wire, a neutral wire, a 220v 5w energy-saving lamp, an energy-saving lamp holder (screw mouth is E27), a plug, and an electric pen. Connect the plug to the lamp holder with the live and neutral wires first (Figure 12). Record the position of the fire end of the plug on the bulb. Then cut the firewire in half. The live wire to the small lamp is connected to the NO (Normal Open) terminal of the relay, and the live wire at the end of the connector is connected to COM (Common) (Figure 13). Test and record the fire end of the socket with an electric pen, the fire end of the plug corresponding to the fire end of the socket. Relay "dc+" is connected to Raspberry Pi VCC pin, "dc-" is connected to Raspberry Pi GND pin, and "in" is connected to pin 7. Use Raspberry Pi GPIO 5V power supply, 5V voltage control 220V voltage to switch on and off the bulb.

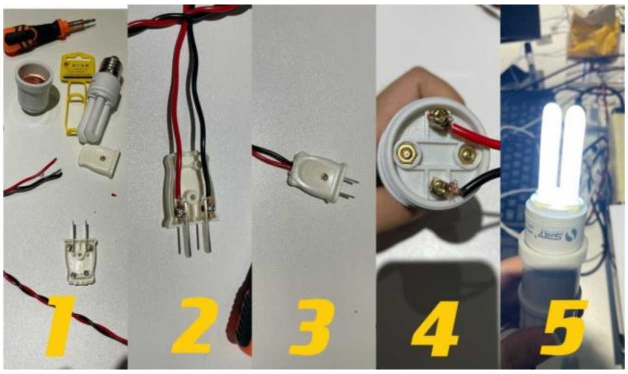

Figure 12. The process of the lamp connected to live and neutral

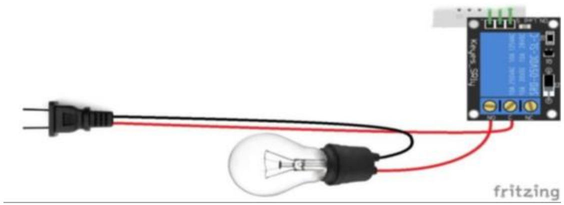

Figure 13. The way of Lamp connected to relay

Final Connection

The energy-saving lamp changes to an LED lamp. The purpose of the experiment is to wave the light on and off in one round trip. A round trip is defined as a user wave as the hand passes the sensor from its original position being sensed once and returning to its original position being sensed twice. Continuous-wave signals are emitted over a period of time, with a time interval between the first and second detections. A round trip measurement is defined as the rangefinder scanning the first wave, not scanning the second wave, and scanning the third wave again. The distance set to be measured is less than or equal to 50cm as the distance detected by the sensor for one valid wave of the hand when the distance between the measured object and the sensor is greater than 50cm as the distance away from the sensor for one valid wave.

Full working code:

```python
#!/usr/bin/env python

import RPi.GPIO as GPIO
import time

TRIG = 11
ECHO = 12
RelayPin = 7
def setup():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(TRIG, GPIO.OUT)
    GPIO.setup(ECHO, GPIO.IN)

def setuprelay():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(RelayPin, GPIO.OUT)
    GPIO.output(RelayPin, GPIO.HIGH)

def distance():
    GPIO.output(TRIG, 0)
    time.sleep(0.000002)

    GPIO.output(TRIG, 1)
    time.sleep(0.00001)
    GPIO.output(TRIG, 0)

    while GPIO.input(ECHO) == 0:
        a = 0
    time1 = time.time()
    while GPIO.input(ECHO) == 1:
        a = 1
    time2 = time.time()

    during = time2 - time1
    return during * 340 / 2 * 100

```

Figure 14. Experience code(1)

```python
def loop():
    global a
    global b
    a=0
    b=0
    c=0
    f=0
    s=0
    while True:
        dis = distance()
        if dis<50:
            a=1
            print "dis1=",dis
        if a==1 and dis>=50:
            a=0
            f=f+1
            print "dis2=",dis

        if f==2:
            f=0
            s=s+1
            print s
            GPIO.output(RelayPin, GPIO.HIGH)
        if s%2==0 and s!=0:
            GPIO.output(RelayPin, GPIO.HIGH)
        elif s%2==1:
            GPIO.output(RelayPin, GPIO.LOW)
        time.sleep(0.05)
```

Figure 15. Experience code(2)

```python

def destroy():
    GPIO.cleanup()
def destroyrelay():
    GPIO.output(RelayPin, GPIO.HIGH)
    GPIO.cleanup()
if __name__ == "__main__":
    setup()
    setuprelay()
    try:
        loop()
    except KeyboardInterrupt:
        destroyrelay()
        destroy()
```

Figure 16. Experience code(3)

Explanation of the subject code (Figure 15):

“a” is the variable that determines whether the waving distance is less than 50cm. “f” is the variable that determines the number of times the waving distance is greater than 50cm. “s” is the variable that calculates the total number of times the hand is waved. The third “if” judgment is to control the switch of the light bulb. If the “f” value is 2, it means that the distance away from the measurement twice can be proved to wave twice then the “s” variable count is increased by one. When “s” is an odd number, the light bulb lights up. When “s” is an even number, the light bulb goes out to prevent the ultrasonic measurement frequency from being too high. The computer lag can adjust the “time. sleep ( )” value in brackets. Currently, 0.05 is a more appropriate value without stuttering while the frequency is moderate.

Variables

Variable 1: Since the transmitting end of the ultrasonic rangefinder cannot be refracted to the receiving end, the hand cannot be less than 2cm from the ultrasonic rangefinder. Otherwise, it cannot be measured (Figure 17).

Solution: Try to wave your hand between 5cm and 50 cm from the sensor (Figure 18).

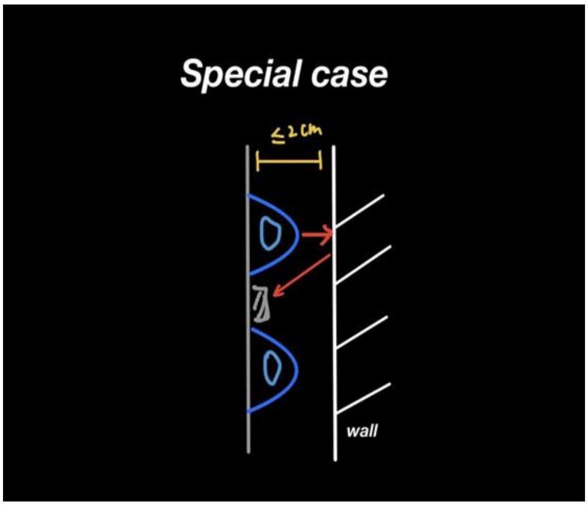

Figure 17. Distance less than 2cm

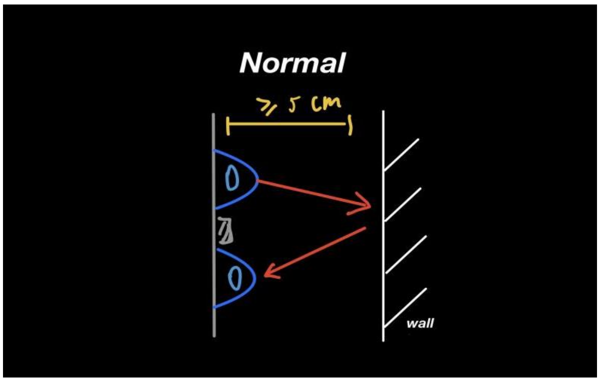

Figure 18. Distance between 5cm and 50cm

Variable 2: When there is no object in the detection range of the sensor, a number larger than 1000 cm will be output in most cases, which will not affect the experimental results.

Variable 3: If other objects are present with the measured object when the sensor is measuring, it may interfere with the sensor's ability to correctly identify the gesture.

Solution: The ultrasonic rangefinder can be pointed at the ceiling with the hand waving above the sensor, so as to ensure that no other objects interfere with the correct operation of the rangefinder.

Variable 4: As the bulb is powered by 220v, while the copper wire of the small bulb connected to the relay port is not completely wrapped by insulating tape, it is easy to produce the risk of electric shock. There are contacts exposed at the low end of the relay. If you touch the NC and COM port contacts with your hand, it is easy to form a pathway and get an electric shock.

Solution: Try not to touch the small light and the area where the small light is connected to the relay with 220v connected. If you need to touch it, be sure to act when it is disconnected from 220v. Insulate the low end of the relay with a block of wood and also secure it with insulating tape.

Academic Ethics

This experiment will not involve the privacy and safety of others, and the results of this experiment for society will be beneficial rather than harmful, so this study will not violate academic ethics.

Results

The primary stage of this experiment was done 60 times and was divided into 6 groups. The first group had a wave rate of approximately 17-20s and a wave height of approximately 18cm from the sensor. the second group had a wave rate of approximately 10-12s and a wave height of approximately 18cm from the sensor. the third group had a wave rate of approximately 17-20s and a wave height of approximately 13cm from the sensor. the fourth group had a wave rate of approximately 10-12s and a wave height of 1cm from the sensor. the fifth group had a wave rate of approximately 17-20s and a wave height of 5cm from the sensor. The fifth group is a back and forth waving rate of approximately 17-20s and a waving distance of 5cm from the sensor. the fifth group is a back and forth waving rate of approximately 10-12s. the fifth group is a back and forth waving rate of approximately 17-20s and a waving distance of 5cm from the sensor.

The number of samples was 10 for all groups except group 2 and group 5 (the data of the first sample in group 2 was lost and the video of the ninth sample recorded in group 5 was lost), the actual number of valid samples for the 60 experiments was 58 groups.

The evaluation of the distance measured by the sensor was first analyzed. Using python, the 58 sets of data were summed one to ten and divided by the total number of waves in each group to calculate the average wave height for each group, while the distance measured by the sensor for each measurement was recorded and then compared to the actual wave height (Figure 19 and 20). Each set of experiments was tested by finding the height of a reference.

```
while 1:
    p=p+1
    q=0
    line=f.readline()
    for i in range(5,len(line)-1):
        if (line[i]== ","):
            break
        q=q+int(line[i])
        q = q * 10
    q=q/10
   # print(p,q)
    if(q<=100 and q>0):

        min=min+1
        sum1=sum1+q
        list_min.append(q)
        #print(q, min)
    elif(q>100):

        max=max+1
        sum2=sum2+q
        list_max.append(q)
        #print(q,max)
    #print(sum1,min)
    #print(line[5:len(line) - 1])
    if not line:
        break
```

Figure 19. Code to automatically calculate the average distance (1)

```
f.close()
min_average=sum1/min
max_average=sum2/max

file.write("min="+str(min_average)+" "+"number="+ str(min))
file.write("\n")
file.write("max="+str(max_average)+" "+"number="+ str(max))
file.write("\n")
file.write('Min_are:')
for i in range (min):
    file.write(str(list_min[i])+",")
file.write("\n")
file.write('Max_are:')
for i in range (max):
    file.write(str(list_max[i])+",")
#file.write(list_min)
#file.write(list_max)
```

Figure 20. Code to automatically calculate the average distance (2)

The data is then converted into CSV format to make a bar chart, which gives a more visual representation of the overall data trend and allows a comparison of each group of data.

Results of the test heights were 18cm (Figure 21):

In terms of averages: the average of the four sampled groups was 17cm and 18cm, with the average of all measured heights having an error of no more than 2cm from the actual. Occasionally, there are 1-2 extremely large values.

From each group of data individuals: the difference between the maximum and minimum values for the first group is 7cm, the difference between the maximum and minimum values for the second group is 5cm, and the difference between the maximum and minimum values for the third group is 7cm (minus the extreme values) and the difference between the maximum and minimum values for the fourth group is 4cm.

From the four groups as a whole: group one has the greatest variation, group two has a smooth change in the front with a slight change in magnitude at the back, and group three has a large amplitude change at the front and a smooth variation at the back. The fourth group has the smoothest change.

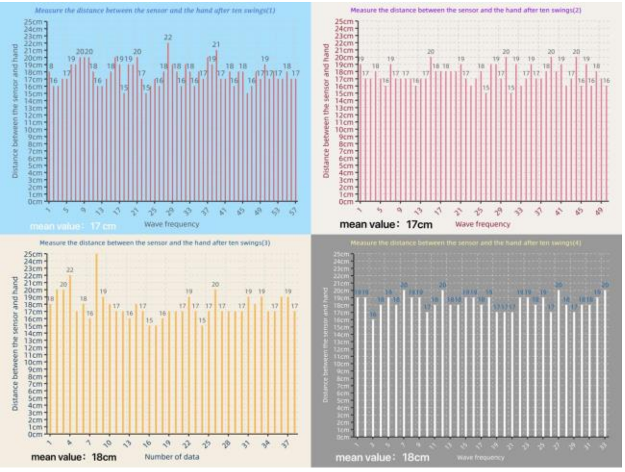

Figure 21. The distance collection chart of 18cm

The results of the height test were 13cm (Figure 22):

In terms of averages: the average height of the four groups is 16cm, 14cm, 14cm, and 14cm respectively, the average is within 3cm of the actual height, but occasionally there are 1 or 2 extreme figures.

From each set of data individuals: the difference between the maximum and minimum values of the first set of data is 5cm (minus the extreme values) the difference between the maximum and minimum values of the second set is 3cm, the difference between the maximum and minimum values of the third set is 7cm (minus the extreme values) and the difference between the maximum and minimum values of the fourth set is 8cm (minus the extreme values).

From the four groups as a whole: groups one and three show a slight fluctuating trend, group two has the smoothest data and group four has a larger fluctuating trend.

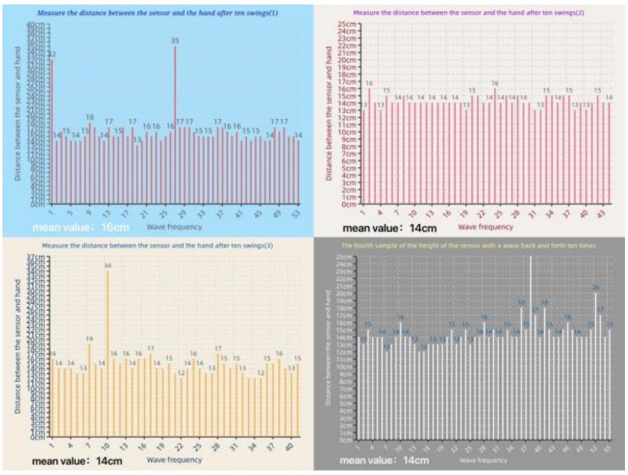

Figure 22. The distance collection chart of 13cm

Description of successful and failed gesture recognition: The video review of the 58 sets of experiments showed that only 12 sets of experiments were identified in all ten waves back and forth, with a 21% success rate. The highest number of successes reached 4 out of 10 groups with a height of 13cm and a rate of 10 to 12s, with an even distribution of missed multiple measurements. The main reason for the failure of the experiment with a wave rate of 17-20s at a height of 18cm was that the sensor missed and over-measured during the first and second wave rounds, with the majority of over-measurements occurring, and also during the fifth round. The failed experiments with a wave rate of 10-12s at a height of 18 cm also focused on the first to third wave back and forth, with 4-9 back and forth being more accurate compared to the 17-20 s at the same height. The failures for a height of 13cm waving rate of 17-20s continued to be concentrated in the first two waving round trips, with an average number of over-and under-testing and a predominance of over-testing. Failures for the 5cm height waving rate of 17-20s were concentrated in the third to sixth waving rounds. The height was 5cm and the waving rate was 10-12s. The main failures were in the second to sixth rounds, all of which were missed.

The test was then changed to 10 groups of 13cm in height with a wave frequency of 10-12s. The experiment started with a wave of one back and forth to turn off the light, then the second wave of 5s, 3s, and 2s to turn on the light, and the number of successes and failures was counted in ten groups. The first group tested was 5s and all ten groups were successful. The first set of tests was 5s with all ten sets of experiments successful. The second set of tests was 3s, with the tenth set failing because the second back and forth sensor was missed. The third set of tests was 2s apart, with the sixth and seventh sets failing, the sixth set failing because the first wave back and forth was missed and the seventh set failing because the second wave back and forth was missed.

Conclusion

The designed effect was achieved in the experiment (Figure 23). The analysis of the 8 sets of samples yielded relatively accurate distance measurements by the sensor. This experiment also led to the valuable conclusion that the device worked best at a height of 13cm with a hand wave frequency of 10-12s. This time the test was conducted with only two waves, and the intervals between the two waves were 2s, 3s, and 5s. The success rate was 100% for 5s intervals, 90% for 3s, and 80% for 2s.

Limitation

To test the success and failure probability of waving the light on, 10 consecutive waves were first performed, with a very low success rate. However, this does not reflect the success rate of the whole experimental set-up, as the number of tests is limited and the continuous waving back and forth of the test requires a high degree of sensor accuracy. If one of the sensors misses a measurement, the result will be a direct failure.

Overall, the accuracy of the ultrasonic rangefinder is still very high with an 80% probability that the small light will recognize the gesture and light up successfully, but there are very few cases where the sensor will fail to recognize the gesture and there will be multiple measurements and missed measurements. The error rate is higher when measuring with a continuous wave of the hand, but the accuracy is greatly improved when waving the hand back and forth individually.

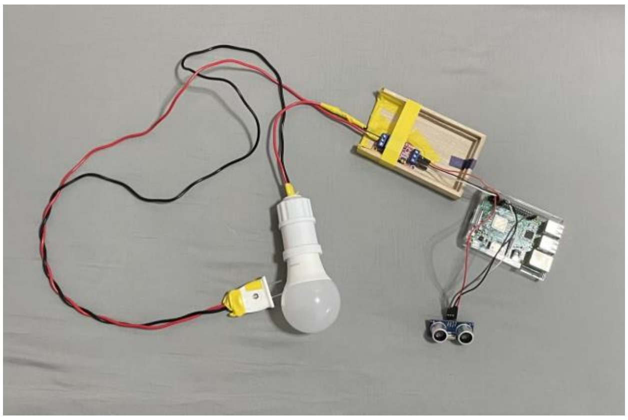

Figure 23. The final connected device

Improvement Options

Ultrasonic rangefinders are sometimes missed and multiple measurements. In fact, there are many modules available, such as digital temperature sensors (Figure 24), and u-type photoelectric sensors(Figure 25). The digital temperature sensor controls the switching on and off of the light bulb by measuring the temperature with a thermistor. The u-shaped photoelectric sensor controls the switching of the light bulb by blocking and conducting the infra-red emitted light, both sensors increase the probability of successful switching. But neither sensor is as convenient and efficient as waving a hand over the control, and the ultrasonic rangefinder does not require touching the sensor. So each sensor has its characteristics and advantages.

As for the experimental problem of relays being susceptible to electric shock, wooden blocks have been used in the experiments to pad the low end of the relay to prevent the contacts from being exposed. This is a simple solution to the safety problem of electric shock but loses aesthetic appeal and portability. Special plastic relays cases are now also available on the market and can be wrapped in a specific plastic case if appearance is a concern.

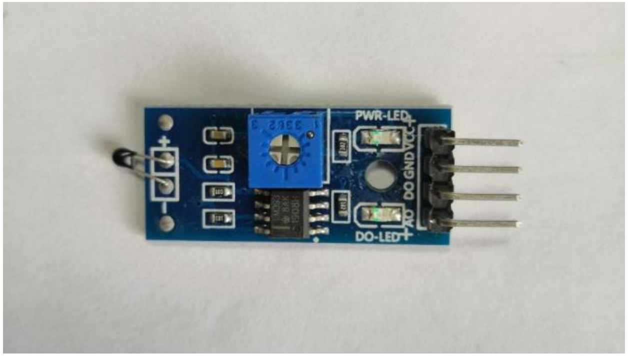

Figure 24. Digital temperature sensors

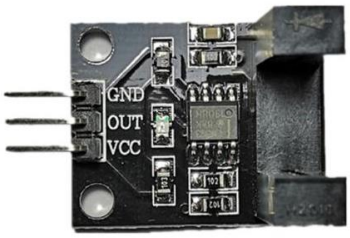

Figure 25. U-type photoelectric sensors

Contributions

With the continuous development of technology, smart homes are gradually emerging and taking over the market. This experiment can serve as a reference for scholars who want to understand or design and build smart homes. Background information on the Raspberry Pi and the corresponding functions and introduction of the panel are presented. The role and principle of LED, ultrasonic rangefinder, and relay modules are explained. Write code in Python to connect the various sensors and explain them. Understand the principles of Live wire and neutral and build CFL and LED lights. Implement and explain how to control the ultrasonic rangefinder measurements in Python and then control the LED lights via relays. Present safety issues to be aware of when using relay modules -- Touching the low side of the relay can easily cause an electric shock due to the circuit path. To avoid electric shock, place a wooden block or plastic box under the low end of the module and use insulating tape to bind the part of the relay that is connected. Propose a quick solution using python to organize the data.

Conflict of Interests: the author has claimed that no conflict of interests exists.

References

1. An Junlin. (2018). Research on Intelligent Lighting Design based on Interactive experience. (Doctoral dissertation, Hubei University of Technology).
2. Compact fluorescent lamp https://en.wikipedia.org/wiki/Compact_fluorescent_lamp.
3. HOW TO SET UP A 5V RELAY ON THE ARDUINO. https://www.circuitbasics.com/setting-up-a-5v-relay-on-the-arduino/
4. Jump wire. https://en.wikipedia.org/wiki/Jump_wire
5. Kang YunChuan. (2020). Smart watering system for home flowers based on Raspberry Pi. (Doctoral dissertation, Chongqing Three Gorges University).
6. Live wire and Neutral wire https://baike.baidu.com/item/零线火线/4717352?fr=aladdin
7. Lv Ning. (2014). On the Pros and Cons of Traditional Chinese Style Furniture in Contemporary Household. (Doctoral dissertation, Hebei Normal University).
8. Raspberry Pi Basic Experiment 11: U-Shaped Photoelectric Sensor Experiment. https://www.jianshu.com/p/487d4a766003.

9. Richardson, M., & Wallace, S. (2013). Getting Started With Raspberry Pi, DC: O'Reilly Media.
10. Raspberry Pi Basic Experiment 17: Temperature Sensor Experiment. https://www.jianshu.com/p/4e120b2786e9.
11. Rong Huaying, & LianGuoEn. (2017). International Smart Home Industry Trade Foresight in the Context of Artificial Intelligence Development. Foreign Trade Practice (10).
12. Sang, D., & Jones, G., & Chadha, G., & Woodside, R. (2014). Cambridge International AS and A level Physics Coursebook Second Edition, DC: Cambridge University. pp. 386-405.
13. The difference between LED and energy-saving lamps. http://www.elecfans.com/d/1178486.html.
14. The difference between a Raspberry Pi and a microcontroller (2020). http://m.elecfans.com/article/1143665.html
15. The three main programming languages for IoT development. https://blog.csdn.net/cpongo1/article/details/89551389.
16. Types of microcontrollers (2010). https://zhidao.baidu.com/question/179925662.html?qbl=relate_question_1.
17. Ultrasonic Ranging Module HC-SR04 https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf.
18. Wang Guojun. (2014). Study of ultrasonic distance measurement sensors. (Doctoral dissertation, Heilongjiang University).
19. Wen-Tsai, Sung, Jia-Syun, & Lin. (2013). Design and implementation of a smart led lighting system using a self-adaptive weighted data fusion algorithm. Sensors, 13(12), 16915-16939.
20. Zhou Zhichao. (2017). The Design of the Intelligent Lighting Control System Based on the Internet of Things. (Doctoral dissertation, Ningbo University).