一、家庭智慧安防系统之前期项目工作准备

1.所需硬件材料：

a.Raspberry Pi树莓派一台



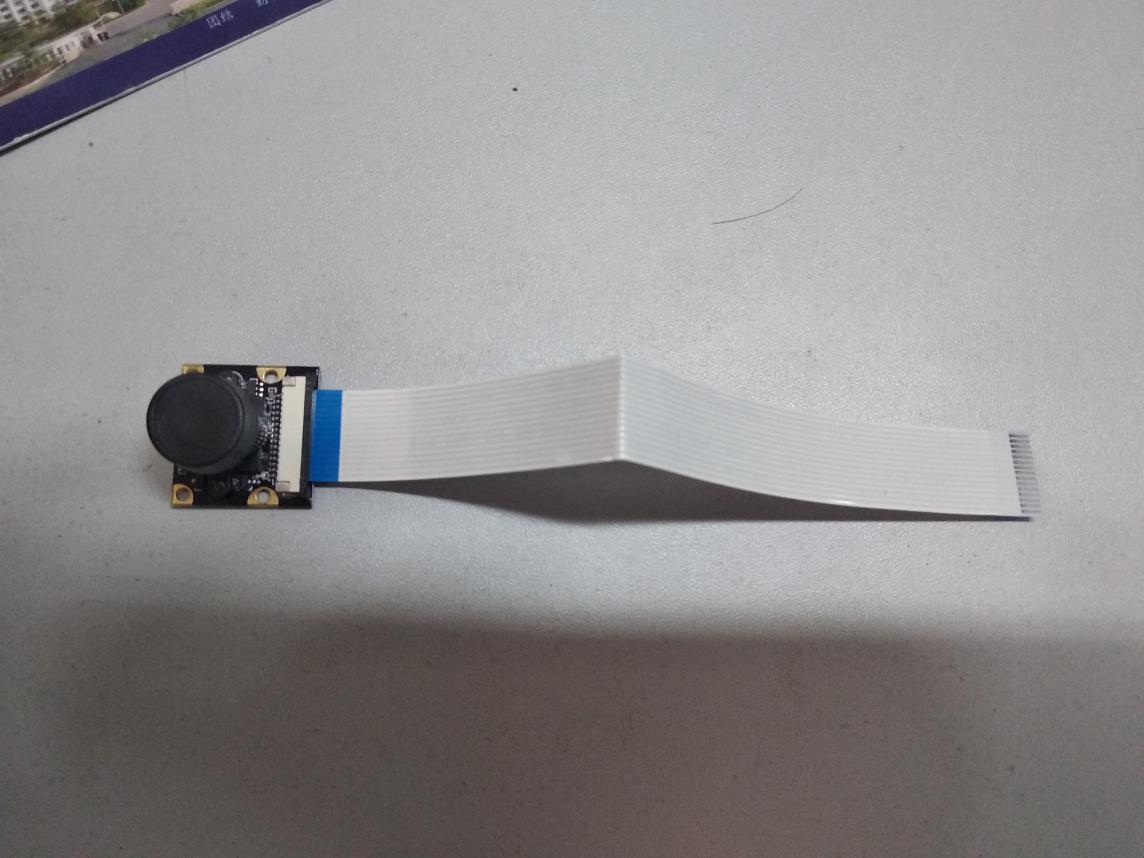
b.MicroSD 卡和读卡器



c.HDMI线、HDMI接口的显示器、USB键盘、USB鼠标



d.树莓派摄像头



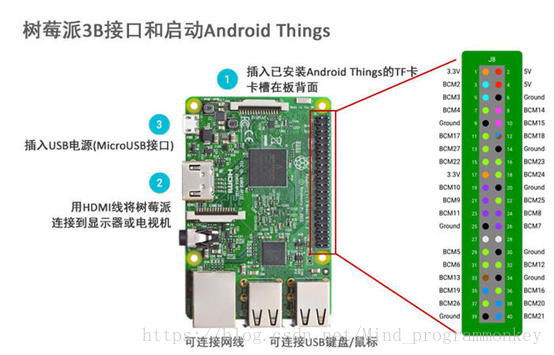
2.所需软件材料：

a.SDFormatter软件：内存卡修复工具用来格式化Micro SD卡

b.win32 Disk Imager软件：用来将树莓派的镜像烧录到Micro SD卡

c.2016-09-23-raspbian-jessie:树莓派镜像

3.启动树莓派



a.烧写镜像

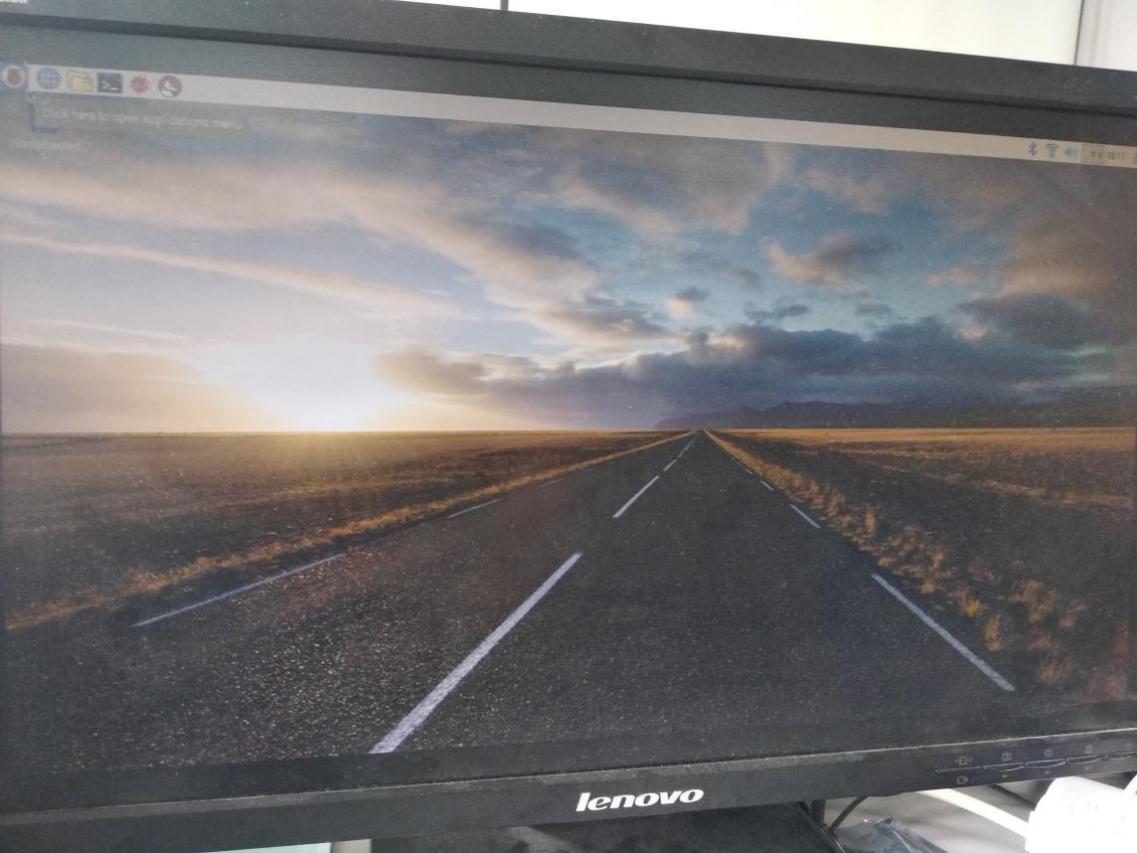
将前面准备好的树莓派镜像通过win32 Disk Imager工具将镜像通过读卡器烧录到sd卡中。

b.在树莓派背后的卡槽中插入SD卡

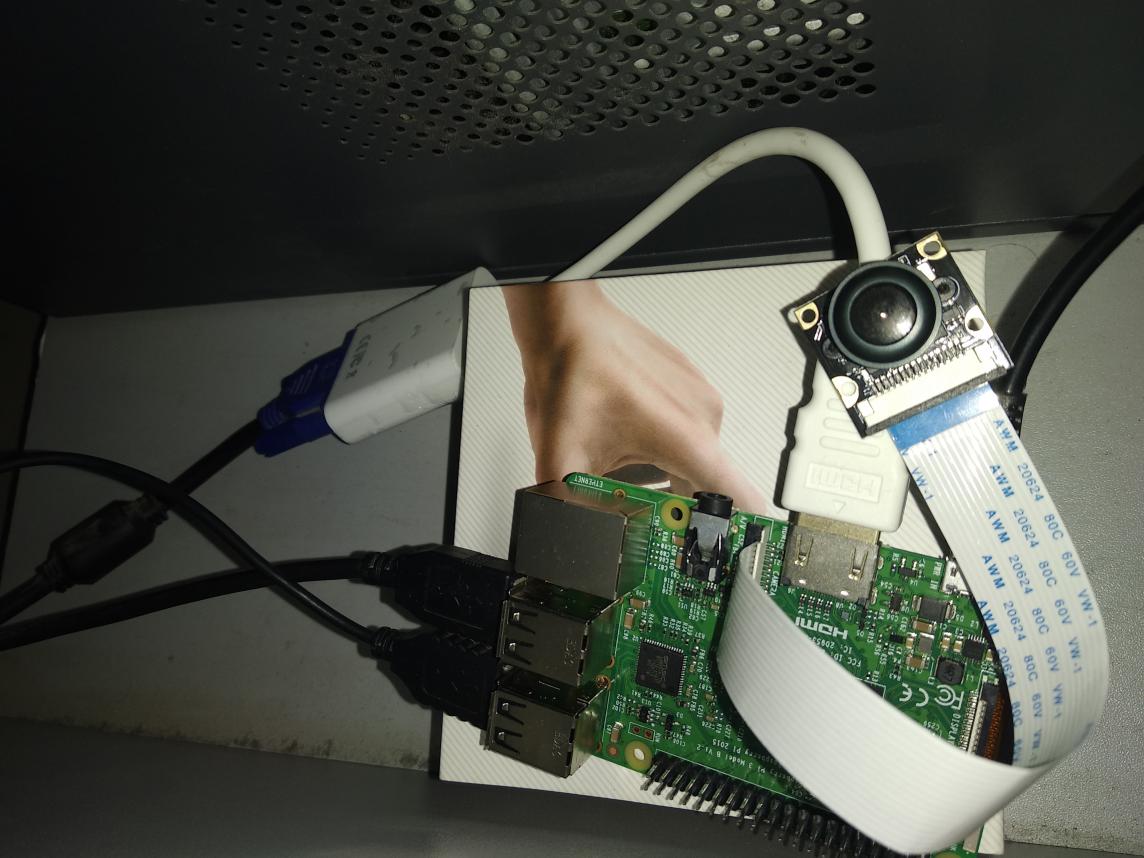
c.用HDMI连接显示器、USB连接键盘、USB连接鼠标

d.供电启动树莓派系统

e.启动画面



f.硬件连接图（注意树莓派的蓝色一端要朝向USB接口处）

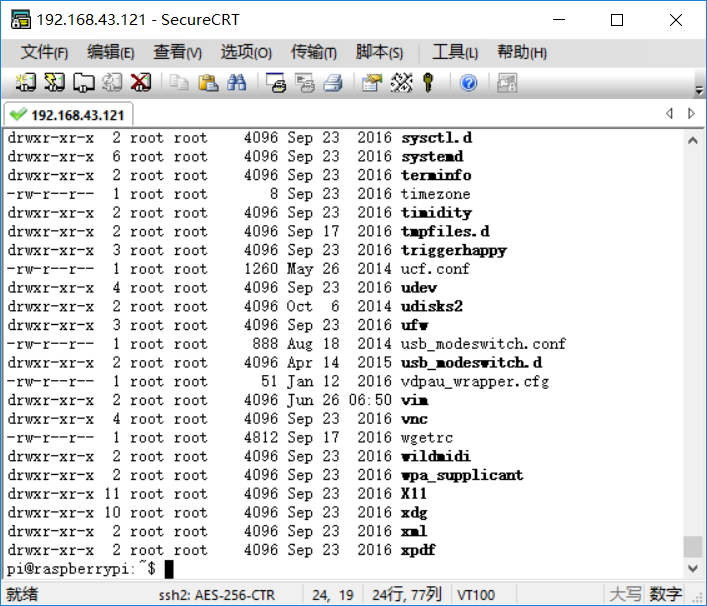


二、家庭智慧安防系统之定时监控

1.开启摄像头的配置

a.首先使用 ls指令来查看是否加载到了对应的video device设备

注意，这里我用的是SecureCRT软件远程连接的树莓派，当然也可以直接在树莓派上操作，如果使用SecureCRT软件远程连接树莓派的话，要注意windows的ip地址跟树莓派的ip地址在一个局域网内，这里我用的是我手机热点。然后在SecureCRT输入树莓派的IP地址以及用户名pi和密码raspberry.

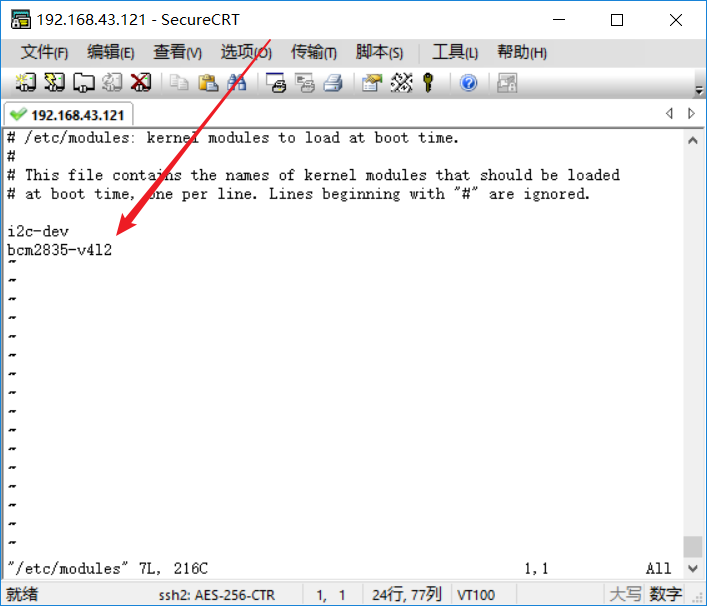


b.添加驱动程序文件

sudo vim /etc/modules

输入上述命令，在这个文件的最后一行输入

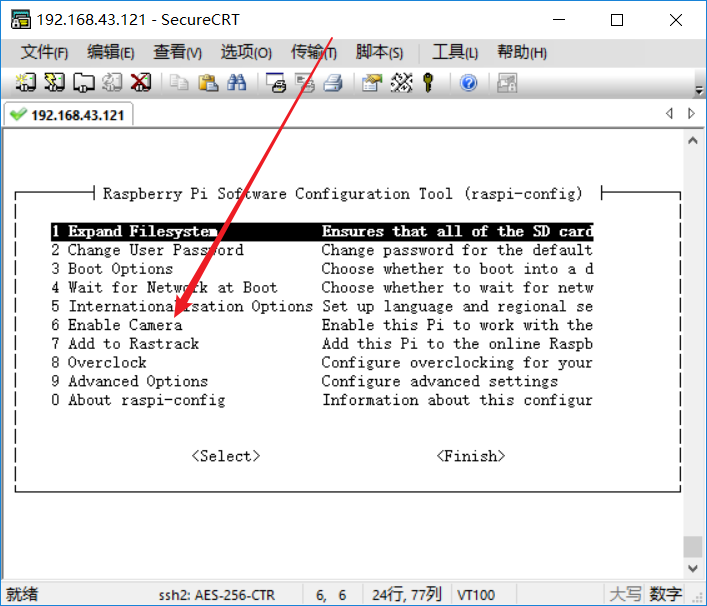
bcm2835-v4l2

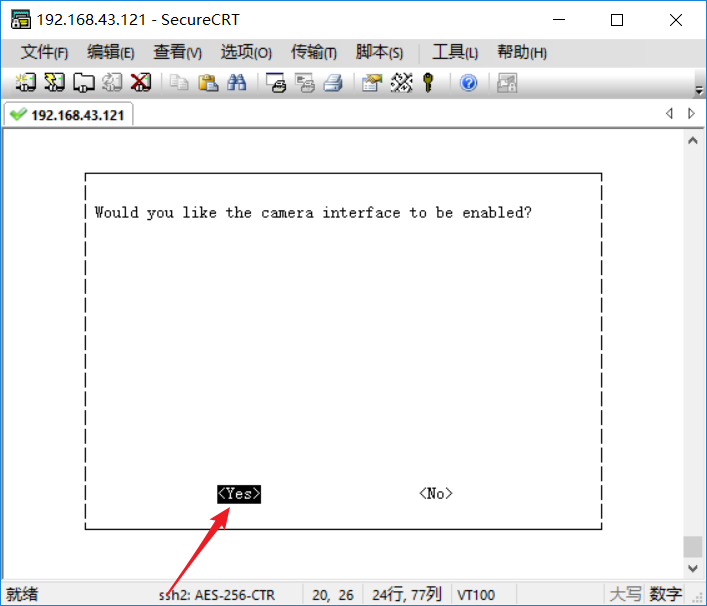


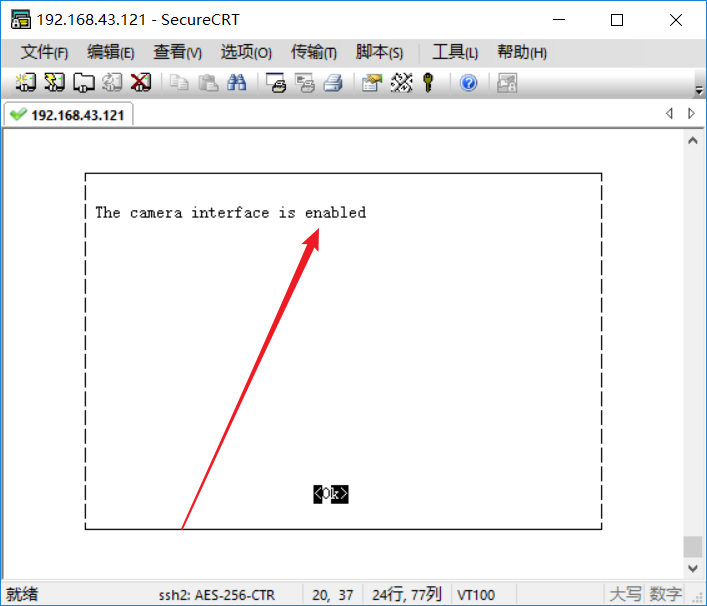
c.修改Raspberry的启动配置使能项

sudo raspi-config

输入上述命令会自动打开下面界面，选择红色Enable Camera所指向的那一行，选择进入，然后使其enable，然后finish，之后选择重启即可。







d.重启之后，来看看/dev设备下是否存在摄像头设备

ls -al /dev/ | grep video

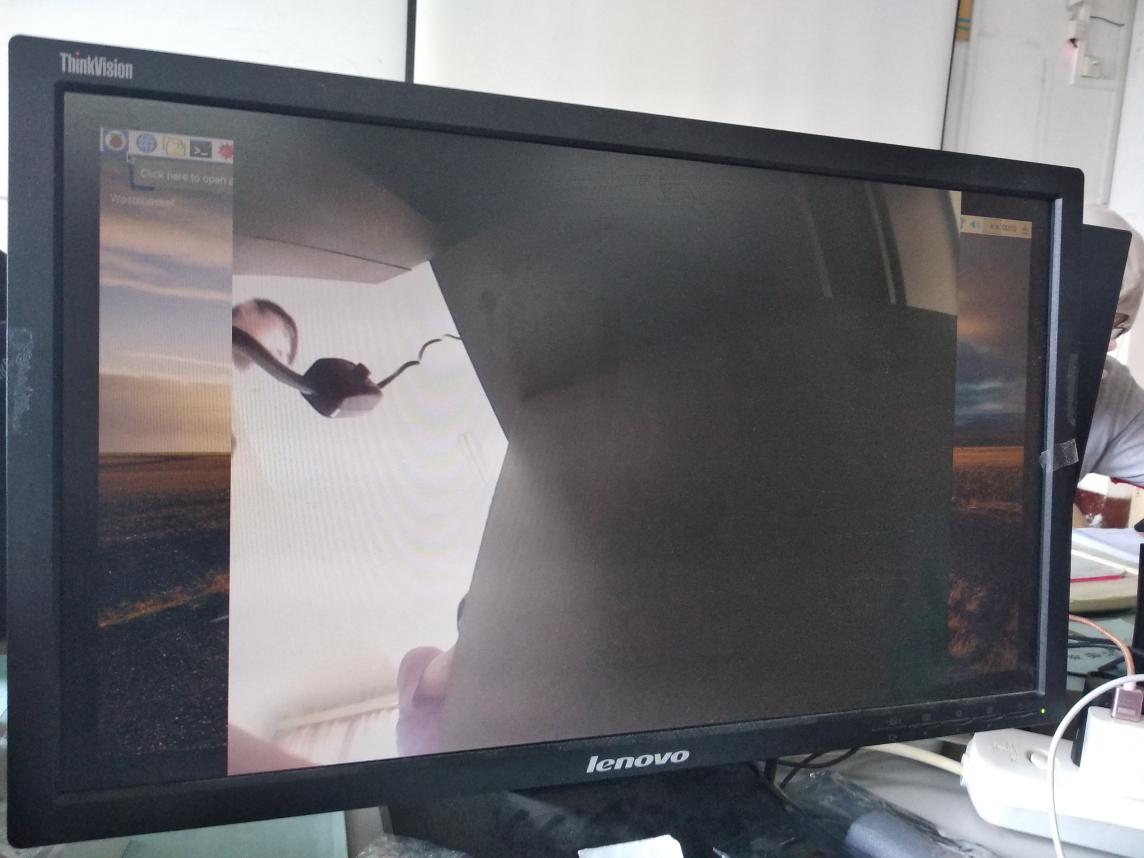
输入上述命令，发现了下图红色箭头所指向的video0的设备，即完成操作。



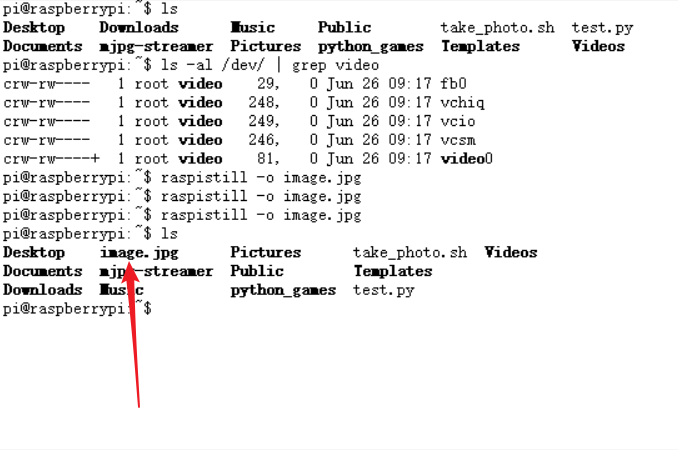
2.测试摄像头的使用情况

raspistill -o image.jpg

通过上述命令会打开摄像头会在屏幕开启摄像头的画面



然后ls显示一下，会发现存在的照片，摄像头测试成功。



3.云服务器的申请

这里我准备将我的照片上传至云服务器上 ，所以我需要申请一个云服务器，在这里我采用的是七牛云服务器。

https://portal.qiniu.com/signup?code=3lmzbyfpvvfbm

首先打开七牛云的官网，点击产品，选择对象存储，立即使用，申请个人账号。



注册账号



注册完成后，还是选择对象存储，进入下方页面，选择新建存储空间，然后填写存储空间的名称，然后立即创建。这里如果你没有实名认证的话，是无法创建成功的，按照提示实名认证之后即可创建成功。



至此，七牛云服务器创建成功。

4.树莓派实时采集照片传至云服务

a.首先写一个.sh脚本，用来执行所有的命令。

vi take\_photo.sh

在文件中输入下面的内容

|  |  |
| --- | --- |
|  | raspistill -o current\_photo.jpg  python test.py |

b.按照七牛python SDK，执行下面命令

sudo pip install qiniu

c.在.sh脚本的同目录下建立一个test.py文件

注意：在文中替换秘钥对以及你申请的上传空间存储名称

*# -\*- coding: utf-8 -\*-*

**import** time

**from** qiniu **import** Auth, put\_file, etag, urlsafe\_base64\_encode

**import** qiniu.config

**import** os

*#需要填写你的 Access Key 和 Secret Key*

access\_key = ' ' *#这里的密钥填上刚才我让你记住的密钥对*

secret\_key = ' ' *#这里的密钥填上刚才我让你记住的密钥对*

*#构建鉴权对象*

q = Auth(access\_key, secret\_key)

*#要上传的空间*

bucket\_name = ' '*#这里填写你在七牛云创建的空间存储名称*

*#上传到七牛后保存的文件名*

key = '%s\_%s\_%s\_%s\_%s\_%s.jpg'%(time.localtime()[0],time.localtime()[1],time.localtime()[2],time.localtime()[3],time.localtime()[4],time.localtime()[5])

*#生成上传 Token，可以指定过期时间等*

token = q.upload\_token(bucket\_name, key, 3600)

*#要上传文件的本地路径*

localfile = 'current\_photo.jpg'

ret, info = put\_file(token, key, localfile)

filename = 'current\_photo.jpg'

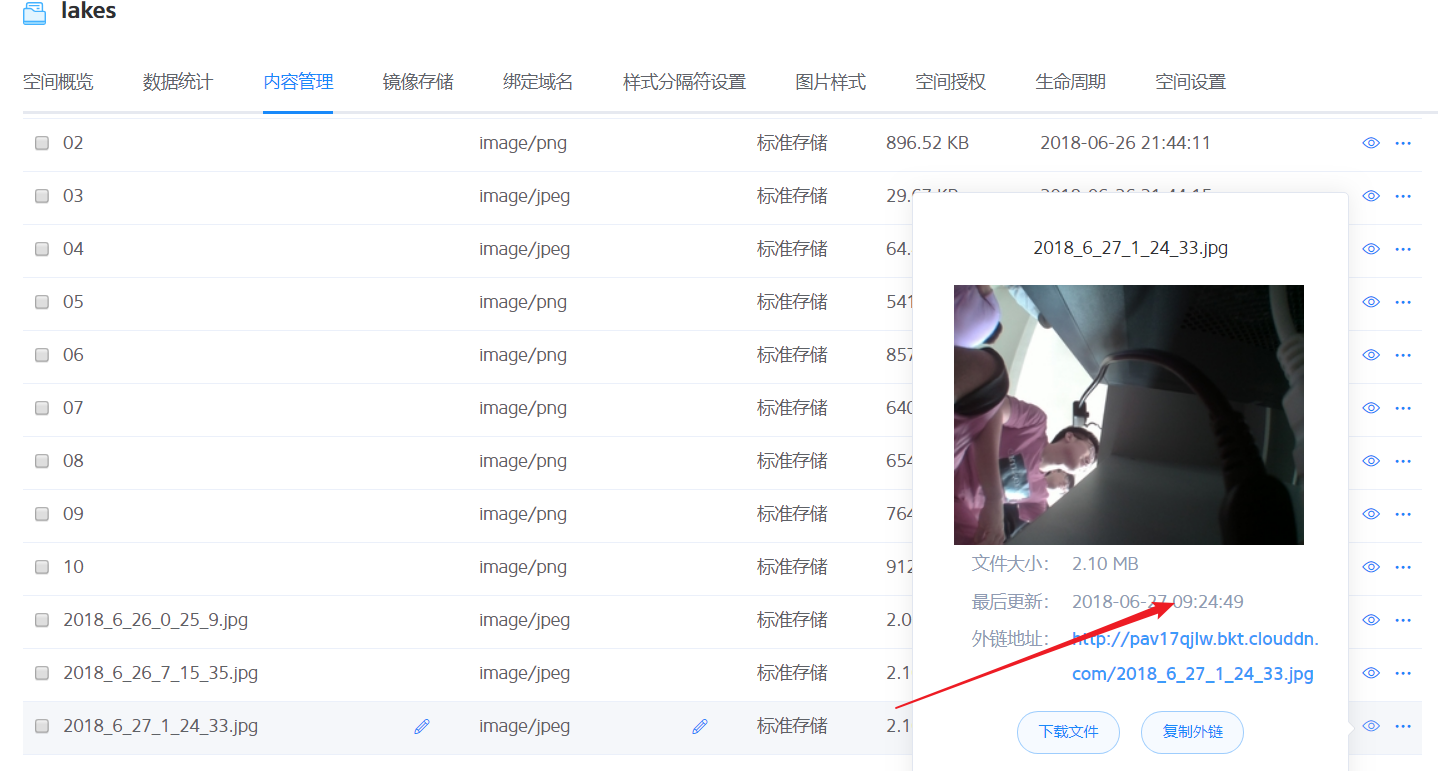
**if** os.path.exists(filename):

    os.remove(filename)

d.执行脚本文件,如果没有报错信息，上传成功

sudo bash take\_photo.sh

e.来到七牛云对象存储空间这里，可以看到刚刚上传成功的图片



至此，我们每次执行take\_photo.sh脚本，都可以让树莓派拍一张图片发送到七牛云上，如果需要自动上传的话，我们可以利用Linux的定时任务crontab来管理这个脚本。

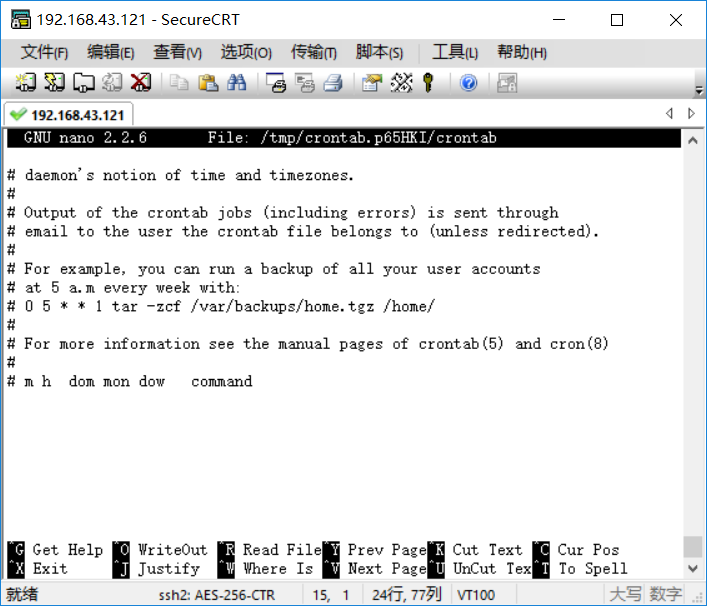
执行下面的命令

crontab -e

在文件的末尾追加下面的内容

|  |  |
| --- | --- |
| 1 | \* \* \* \* \* /home/pi/take\_photo.sh |

然后ctrl+X，按Y保存退出



之后重启cron这个服务

sudo service cron restart

至此，家庭安防的定时监控就完成了，它会每分钟拍下照片并且发送到我们的七牛云上。

三、家庭智慧安防系统之实时监控

这里我用了三种方式来实现实时监控，建议大家用第三种。

1.python

第一种方式是用python来实现的，不过因为不可预知的问题导致未能实现，有兴趣的可以参考下面的文章

<https://blog.csdn.net/bona020/article/details/51034043>

2.motion

第二种方式是用motion来实现的，不过延迟略微有点大。

a.安装motion

sudo apt-get install motion

b.配置motion daemon守护进程

输入命令编辑motion：

sudo nano /etc/default/motion

把里面的no修改成yes，让他可以一直在后台运行：

start\_motion\_daemon=yes

c.修改motion的配置文件

sudo vim /etc/motion/motion.conf

#deamon off 改成 on

deamon on

#关闭localhost的限制

stream\_localhost off

#设置分辨率

width 800

height 600

这里配置文件很长，不容易查找，大家可以用vim的查找命令，如这里我查找width这个单词的话，在文件中按Esc,之后输入

:/width

回车即可，小写n查找下一个，N查找上一个

d.配置启动

输入下面命令启动服务：

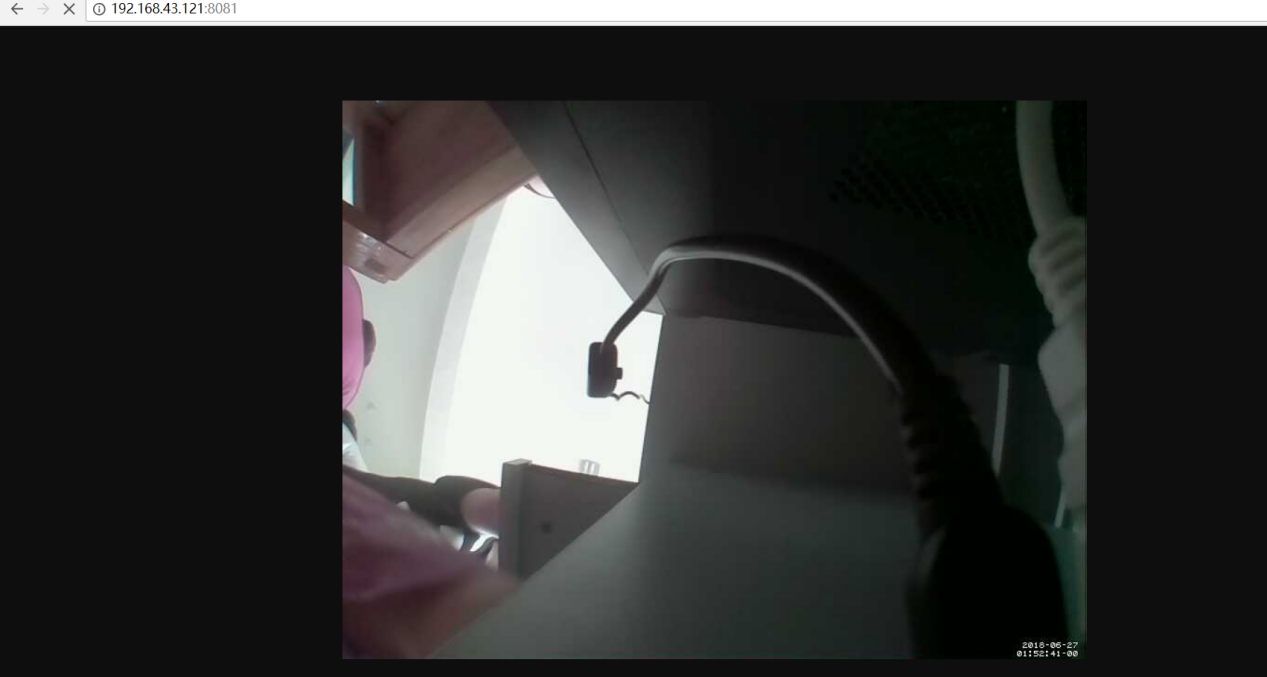
sudo service motion start

输入以下命令开启motion：

sudo motion

 e.查看效果

打开游览器，输入http://树莓派IP:8081即可查看拍到的图像，不过延迟有点高。



# 3.mjpg-streamer

第三种方式安装的工具有点多，不过实时视频很流畅。

a.更新列表

sudo apt-get update    #更新软件列表

b.安装工具

sudo apt-get install subversion  #Subversion是一个自由开源的版本控制系统

sudo apt-get install libjpeg8-dev  #JPEG支持库

sudo apt-get install imagemagick

sudo apt-get install libv4l-dev   #4l是小写"L"

sudo apt-get install cmake  #下载编译工具

c.从github上下载文件

sudo apt-get install git

git clone <https://github.com/jacksonliam/mjpg-streamer.git>

cd mjpg-streamer/mjpg-streamer-experimental #进入下载目录后进入左侧路径

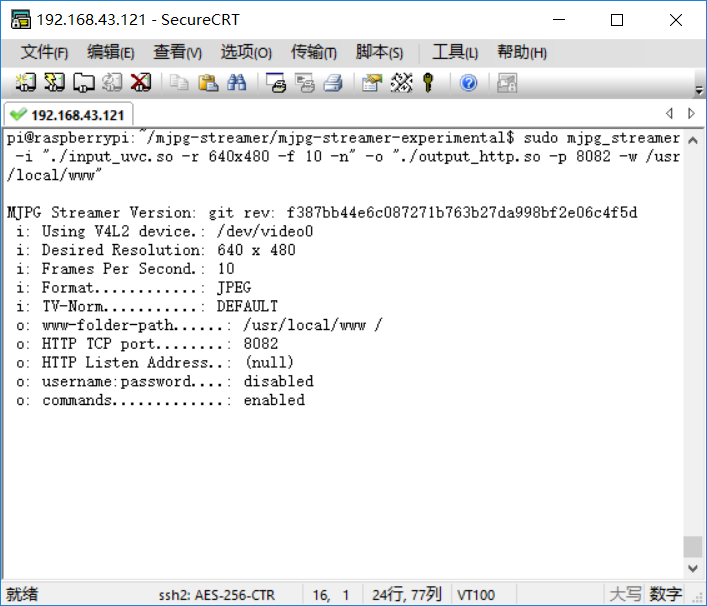
d.对文件进行编译

make all  #编译

sudo make install #安装

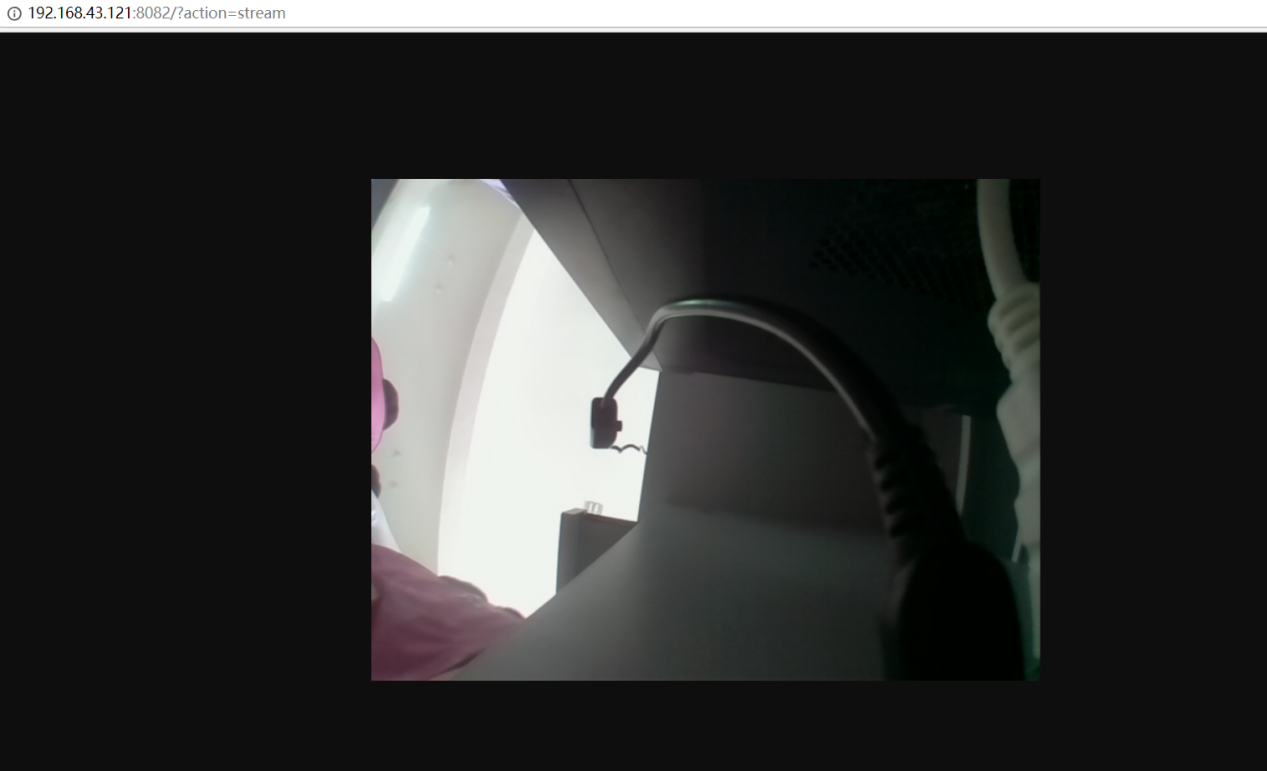
e.运行命令

sudo mjpg\_streamer -i "./input\_uvc.so -r 640x480 -f 10 -n" -o "./output\_http.so -p 8082 -w /usr/local/www"  #此命令尤为重要，如下图所示，输出信息，说明成功！



f.查看效果

游览器输入http://树莓派ID地址:8082/?action=stream可以查看实时视频。http://树莓派ID地址:8082/?action=snapshot可以进行拍照。



四、家庭智慧安防系统之智能温控

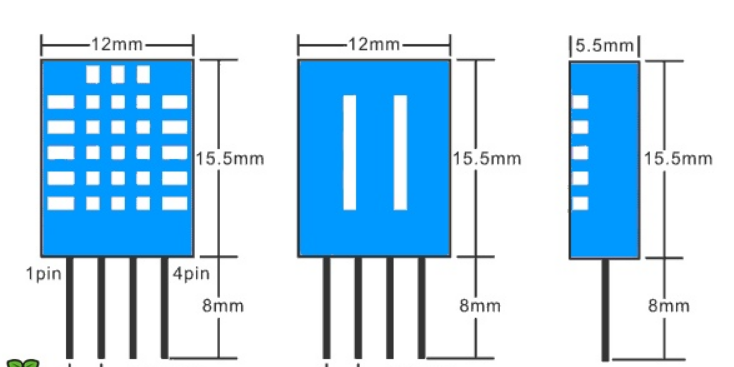
1.所需材料:

树莓派、面包板、温湿度传感器DHT11、直流电机、风扇、L298N电机驱动模块。

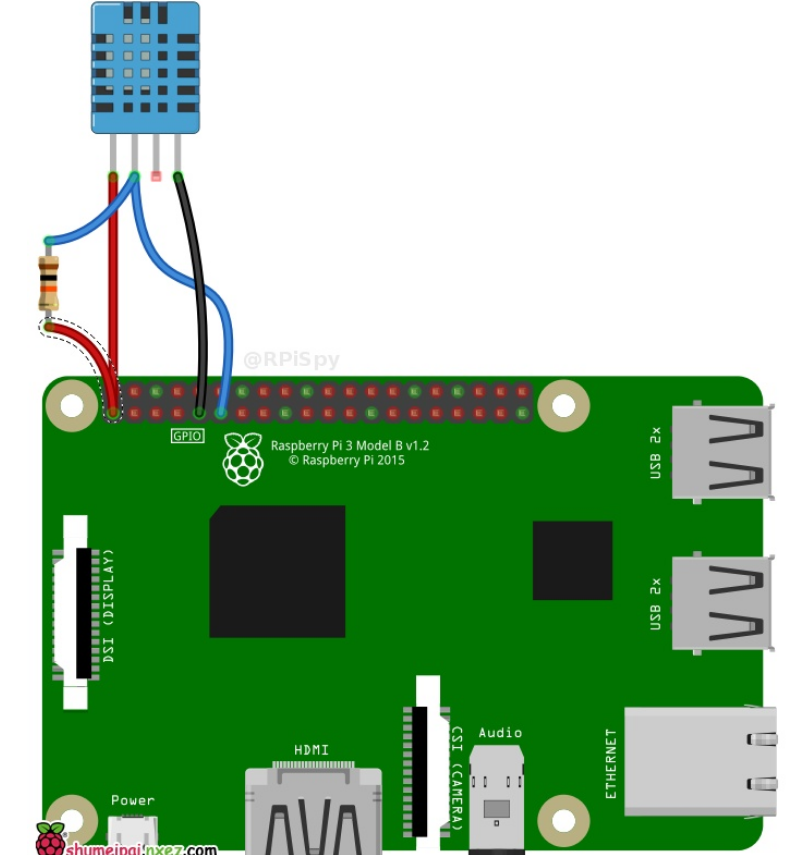
2.树莓派从DHT11温湿度传感器读取数据

（1）DHT11 介绍

DHT11是一款有已校准数字信号输出的温湿度传感器。 其精度湿度+-5%RH， 温度+-2℃，量程湿度20-90%RH， 温度0~50℃。



（2）硬件连接图



（3）具体流程

DHT11的读取需要遵循特定的信号协议完成，为了方便使用Adafruit DHT库。

a.软件安装

sudo apt-get install build-essential python-dev

b.从GitHub获取Adafruit库

|  |  |
| --- | --- |
|  | sudo git clone https://github.com/adafruit/Adafruit\_Python\_DHT.git  cd Adafruit\_Python\_DHT |

c.给python2和python3安装该库

sudo python setup.py install

sudo python3 setup.py install

d.书写dht.py程序引入Adafruit库来读取DHT11的数据

import Adafruit\_DHT

# Set sensor type : Options are DHT11,DHT22 or AM2302

sensor=Adafruit\_DHT.DHT11

# Set GPIO sensor is connected to

gpio=17

# Use read\_retry method. This will retry up to 15 times to

# get a sensor reading (waiting 2 seconds between each retry).

humidity, temperature = Adafruit\_DHT.read\_retry(sensor, gpio)

# Reading the DHT11 is very sensitive to timings and occasionally

# the Pi might fail to get a valid reading. So check if readings are valid.

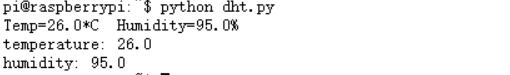
if humidity is not None and temperature is not None:

  print('Temp={0:0.1f}\*C  Humidity={1:0.1f}%'.format(temperature, humidity))

else:

  print('Failed to get reading. Try again!')

e.运行结果图



3.树莓派从DHT11温湿度传感器读取数据写入到数据库中

（1）在树莓派上安装MySQL数据库

a.安装MySQL

使用管理员权限运行获取最新的MySQL及Python编程接口。

$ sudo apt-get install mysql-server python-mysqldb

安装过程中需要输入root管理员的密码，该密码用于之后访问数据库。

b.登录MySQL

mysql -u root -p

Enter password:

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 53

Server version: 5.5.60-0+deb8u1 (Raspbian)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its

affiliates. Other names may be trademarks of their respective

owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

c.查看当前的数据库

mysql> show databases;

+--------------------+

| Database |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

+--------------------+

3 rows in set (0.00 sec)

d.创建一个新的数据库

mysql> CREATE DATABASE jiankong;

Query OK, 1 row affected (0.00 sec)

e.在jiankong数据库下创建dht2的表

mysql> use jiankong;

Database changed  
mysql> create table dht2(id int auto\_increment primary key not null, time char(255),temperature int,humidity int);

Query OK, 0 rows affected (0.03 sec)

f.查看表是否创建成功。

mysql> describe dht2;

+-------------+-----------+------+-----+---------+----------------+

| Field | Type | Null | Key | Default | Extra |

+-------------+-----------+------+-----+---------+----------------+

| id | int(11) | NO | PRI | NULL | auto\_increment |

| time | char(255) | YES | | NULL | |

| temperature | int(11) | YES | | NULL | |

| humidity | int(11) | YES | | NULL | |

+-------------+-----------+------+-----+---------+----------------+

4 rows in set (0.00 sec)

（2）在树莓派上通过python来对mysql数据库进行存储操作

书写测试文件mysql\_test.py来测试是否成功

import MySQLdb

sql='show databases';

conn=MySQLdb.connect(host='localhost',user='root',passwd='root',db='jiankong',port=3306)

cur=conn.cursor(

cur.execute(sql)

print cur.fetchone()

测试成功结果图

（3）将树莓派读取到的温湿度信息写入到我们mysql数据库，对dht.py文件进行修改

#coding: utf8

import Adafruit\_DHT

import MySQLdb

import datetime

#引入gpio的模块

import RPi.GPIO as GPIO

import time

# Set sensor type : Options are DHT11,DHT22 or AM2302

sensor=Adafruit\_DHT.DHT11

# Set GPIO sensor is connected to

gpio=4

dt=datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

# Use read\_retry method. This will retry up to 15 times to

# get a sensor reading (waiting 2 seconds between each retry).

humidity, temperature = Adafruit\_DHT.read\_retry(sensor, gpio)

# Reading the DHT11 is very sensitive to timings and occasionally

# the Pi might fail to get a valid reading. So check if readings are valid.

if humidity is not None and temperature is not None:

print('Temp={0:0.1f}\*C Humidity={1:0.1f}%'.format(temperature, humidity))

print "temperature:",temperature

print "humidity:",humidity

conn=MySQLdb.connect(

host='localhost',

port=3306,

user='root',

passwd='root',

db='jiankong',

)

sql="insert into dht2(time,temperature,humidity) values('%s','%d','%d')"%(dt,temperature,humidity)

cur=conn.cursor()

cur.execute(sql)

cur.close()

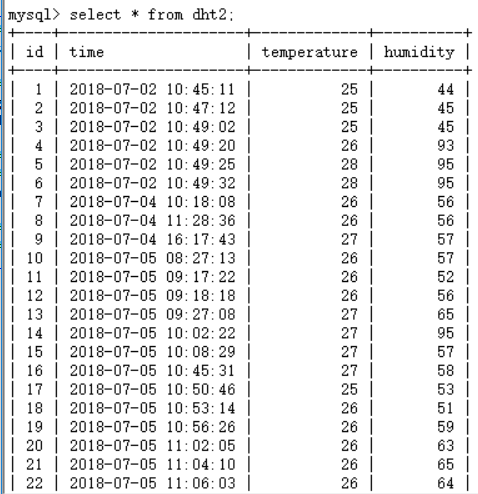
conn.commit()

conn.close()

else:

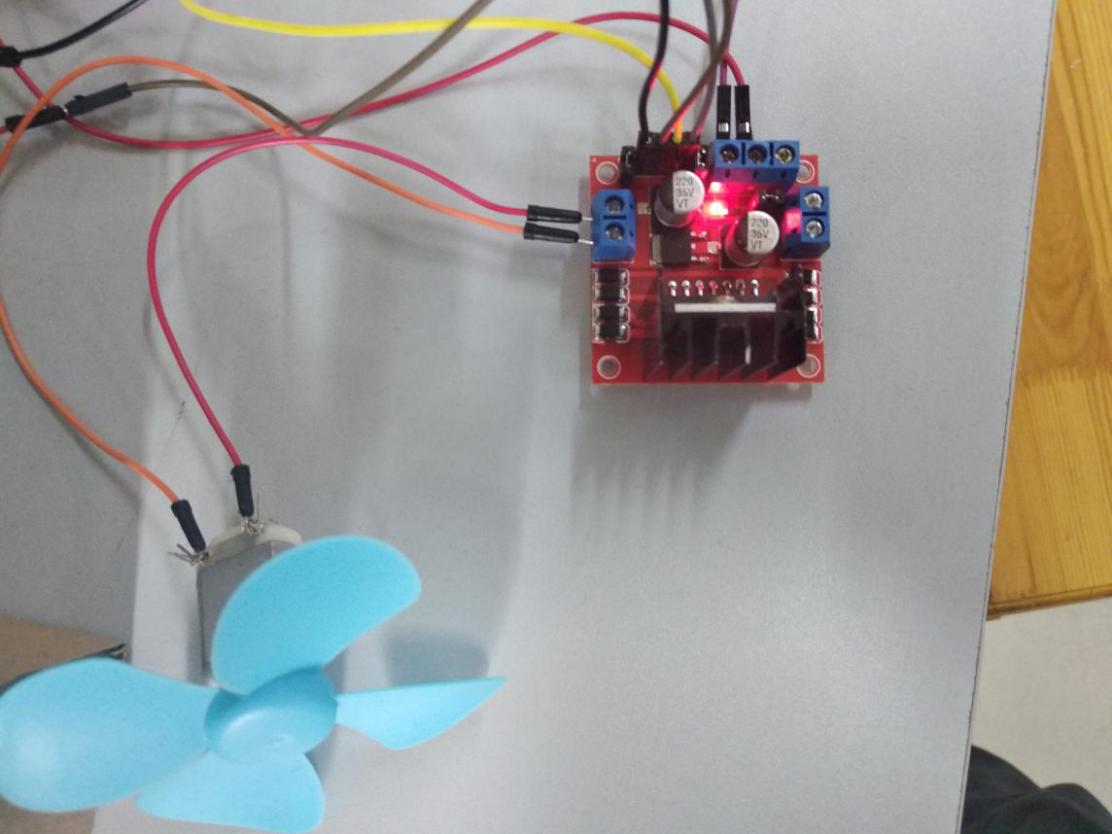
print('Failed to get reading. Try again!')

查看一下数据表dht2中的数据



4.树莓派结合L298N模块驱动直流电机

（1）L298N模块的介绍



接口说明：INT1、INT2、INT3、INT4分别接了树莓派的GPIO13、GPIO15、GPIO16、GPIO18。+5V接树莓派5V，GND接地。

（2）Python代码：

#!/usr/bin/python

#coding: utf8

#引入gpio的模块

import RPi.GPIO as GPIO

import time

#设置GPIO模式

GPIO.setmode(GPIO.BOARD)

#设置in1到in4接口

IN1 = 13

IN2 = 15

IN3 = 16

IN4 = 18

#初始化接口

def init():

GPIO.setup(IN1,GPIO.OUT)

GPIO.setup(IN2,GPIO.OUT)

GPIO.setup(IN3,GPIO.OUT)

GPIO.setup(IN4,GPIO.OUT)

#前进的代码

def qianjin(sleep\_time):

GPIO.output(IN1,GPIO.HIGH)

GPIO.output(IN2,GPIO.LOW)

GPIO.output(IN3,GPIO.HIGH)

GPIO.output(IN4,GPIO.LOW)

time.sleep(sleep\_time)

GPIO.cleanup()

#后退

def cabk(sleep\_time):

GPIO.output(IN1,GPIO.LOW)

GPIO.output(IN2,GPIO.HIGH)

GPIO.output(IN3,GPIO.LOW)

GPIO.output(IN4,GPIO.HIGH)

time.sleep(sleep\_time)

GPIO.cleanup()

#左转

def left(sleep\_time):

GPIO.output(IN1,False)

GPIO.output(IN2,False)

GPIO.output(IN3,GPIO.HIGH)

GPIO.output(IN4,GPIO.LOW)

time.sleep(sleep\_time)

GPIO.cleanup()

#右转

def right(sleep\_time):

GPIO.output(IN1,GPIO.HIGH)

GPIO.output(IN2,GPIO.LOW)

GPIO.output(IN3,False)

GPIO.output(IN4,False)

time.sleep(sleep\_time)

GPIO.cleanup()

init()#调用初始化方法初始化接口

cabk(10)#调用后退方法，并且10秒后停止

运行该程序，直流电机可运行10s。

5.树莓派智能调温综合

对dhp.py程序进行修改，使其测温度、湿度的时候，如果温度超过25℃，会自启动风扇降温。

#coding: utf8

import Adafruit\_DHT

import MySQLdb

import datetime

#引入gpio的模块

import RPi.GPIO as GPIO

import time

# Set sensor type : Options are DHT11,DHT22 or AM2302

sensor=Adafruit\_DHT.DHT11

# Set GPIO sensor is connected to

gpio=4

dt=datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

#设置GPIO模式

GPIO.setmode(GPIO.BOARD)

#设置in1到in4接口

IN1 = 13

IN2 = 15

IN3 = 16

IN4 = 18

#初始化接口

def init():

GPIO.setup(IN1,GPIO.OUT)

GPIO.setup(IN2,GPIO.OUT)

GPIO.setup(IN3,GPIO.OUT)

GPIO.setup(IN4,GPIO.OUT)

#后退

def cabk(sleep\_time):

GPIO.output(IN1,GPIO.LOW)

GPIO.output(IN2,GPIO.HIGH)

GPIO.output(IN3,GPIO.LOW)

GPIO.output(IN4,GPIO.HIGH)

time.sleep(sleep\_time)

GPIO.cleanup()

# Use read\_retry method. This will retry up to 15 times to

# get a sensor reading (waiting 2 seconds between each retry).

humidity, temperature = Adafruit\_DHT.read\_retry(sensor, gpio)

# Reading the DHT11 is very sensitive to timings and occasionally

# the Pi might fail to get a valid reading. So check if readings are valid.

if humidity is not None and temperature is not None:

print('Temp={0:0.1f}\*C Humidity={1:0.1f}%'.format(temperature, humidity))

print "temperature:",temperature

print "humidity:",humidity

conn=MySQLdb.connect(

host='localhost',

port=3306,

user='root',

passwd='root',

db='jiankong',

)

sql="insert into dht2(time,temperature,humidity) values('%s','%d','%d')"%(dt,temperature,humidity)

cur=conn.cursor()

cur.execute(sql)

cur.close()

conn.commit()

conn.close()

if temperature > 25:

init()

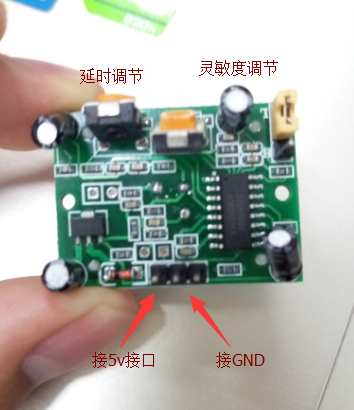
cabk(30)

else:

print('Failed to get reading. Try again!')

五、家庭智慧安防系统之智能监控

1.所需材料：人体感应模块、蜂鸣器以及树莓派摄像头

人体感应模块接线：中间数据引脚接的是pin12，剩下两个引脚分别接电源和接地。

蜂鸣器接线：一根接电源线，一根接pin11

2.树莓派结合人体感应模块进行蜂鸣器报警

当人体感应模块感应到有人体活动时，蜂鸣器进行报警，同时将报警信息写入到MySQL数据库中。

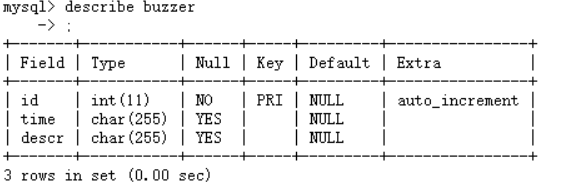
（1）数据库设计

这里的步骤同上一章，在jiankong数据库进行建立buzzer表操作

mysql> use jiankong;

Database changed  
mysql> create table buzzer(id int auto\_increment primary key not null, time char(255),descr char(255));

Query OK, 0 rows affected (0.03 sec)



（2）buzzer.py文件

import RPi.GPIO as GPIO

import MySQLdb

import time

def init():

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BOARD)

GPIO.setup(12, GPIO.IN)

GPIO.setup(11,GPIO.OUT)

pass

def beep():

for i in range(1,6):

GPIO.output(11, GPIO.LOW)

time.sleep(0.5)

GPIO.output(11, GPIO.HIGH)

time.sleep(0.5)

print "the Buzzer will make sound"

def detct():

for i in range(1, 31):

if GPIO.input(12) == True:

print time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time()))+ " Someone is closing!"

conn=MySQLdb.connect(

host='localhost',

port=3306,

user='root',

passwd='root',

db='jiankong',

)

sql="insert into buzzer(time,descr) values('%s','%s')"%(time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time())),'someone is closing')

cur=conn.cursor()

cur.execute(sql)

cur.close()

conn.commit()

conn.close()

beep()

else:

GPIO.output(11, GPIO.HIGH)

print time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time()))+" Noanybody!"

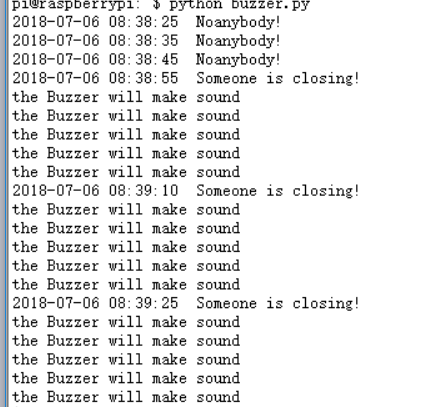
time.sleep(10)

time.sleep(2)

init()

detct()

（3）运行结果



3.短信预警

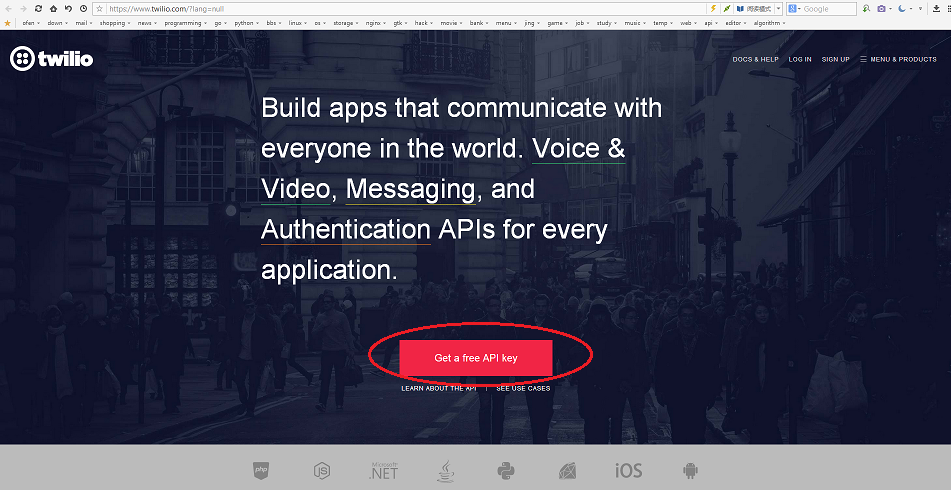
这里采用Twilio免费给手机发短信。

（1）在树莓派上通过命令行下载安装twilio

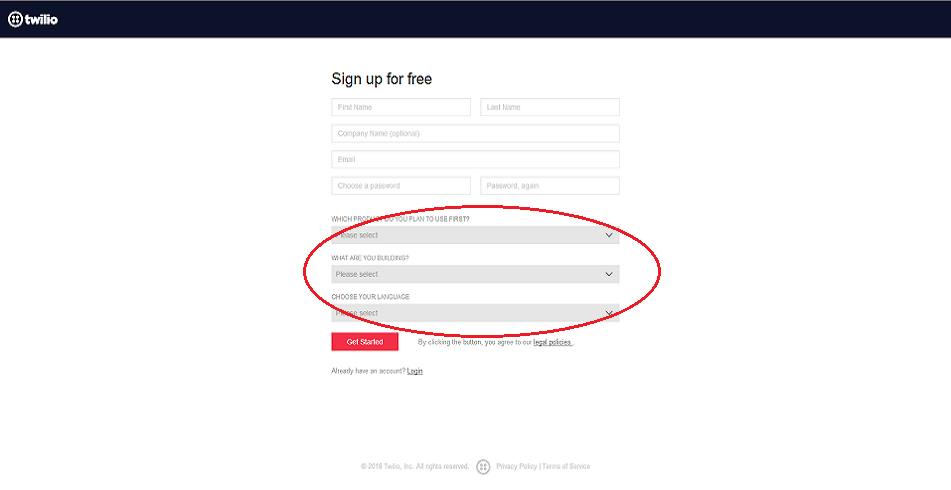
sudo pip3 install twilio

（2）在twilio注册账号

在官网 [https://www.twilio.com](https://www.twilio.com/" \t "https://blog.csdn.net/fengfengdiandia/article/details/_blank)



选择Get a free APIkey，然后注册账号即可。



这三个问题我分别选的是SMS、Phone Verification和Python。

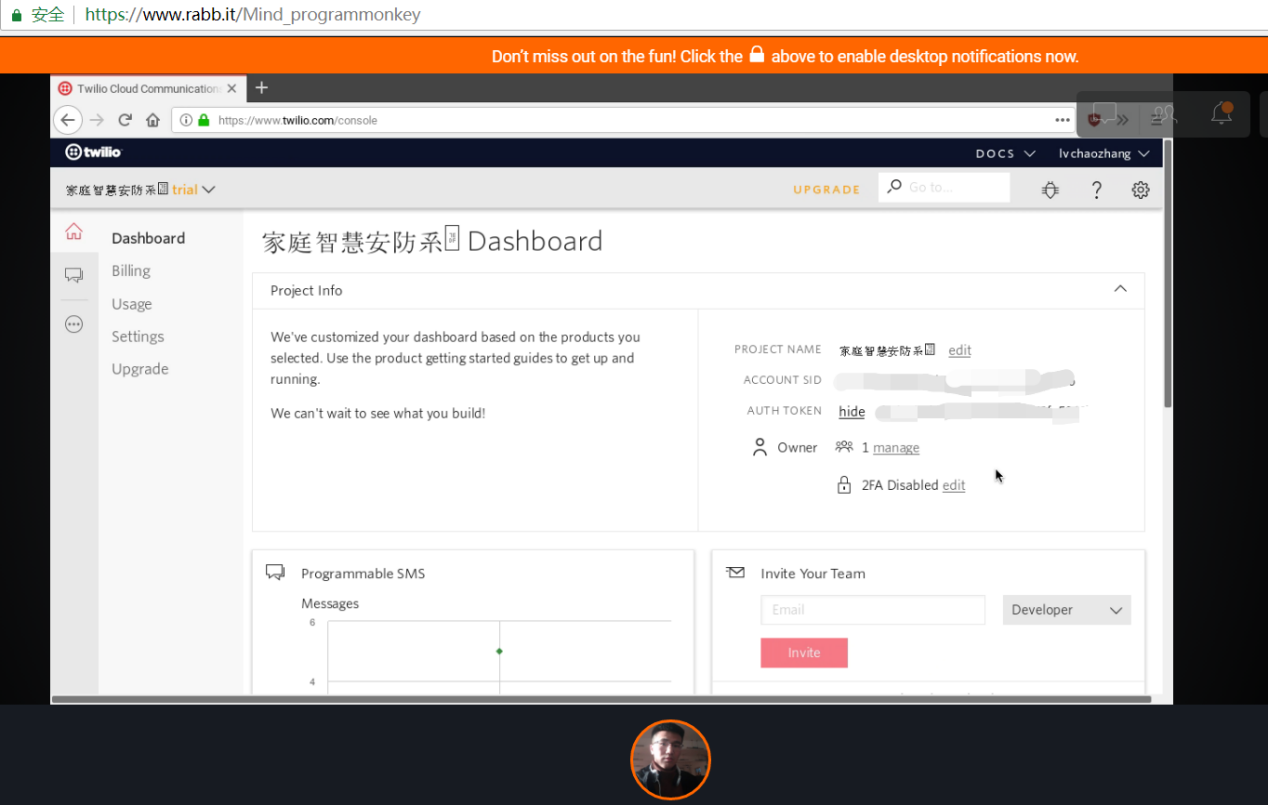
这里可能注册失败，因为注册的时候需要验证码，而在国内验证码是无法显示的，所以这里给大家提供两个办法，这两个办法都是基于谷歌游览器的，第一个办法是给谷歌游览器安装谷歌上网助手插件，另外一个办法在最新版谷歌游览器上访问https://www.rabb.it/,然后通过这个网站提供的谷歌游览器来注册账号。

最后账号还需要手机号的验证。



注册成功登陆之后，选择Get Started，会免费得到一个twilio电话号码。

此时，新建一个项目，这里我新建了一个家庭安防系统的项目，这里记住你的ACCOUNT SID和AUTH token.



（3）python结合twilio向用户发送短信

from twilio.rest import Client

# Your Account SID from twilio.com/console

account\_sid = "ACXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"

# Your Auth Token from twilio.com/console

auth\_token = "your\_auth\_token"

client = Client(account\_sid, auth\_token)

message = client.messages.create(

to="+15558675309",

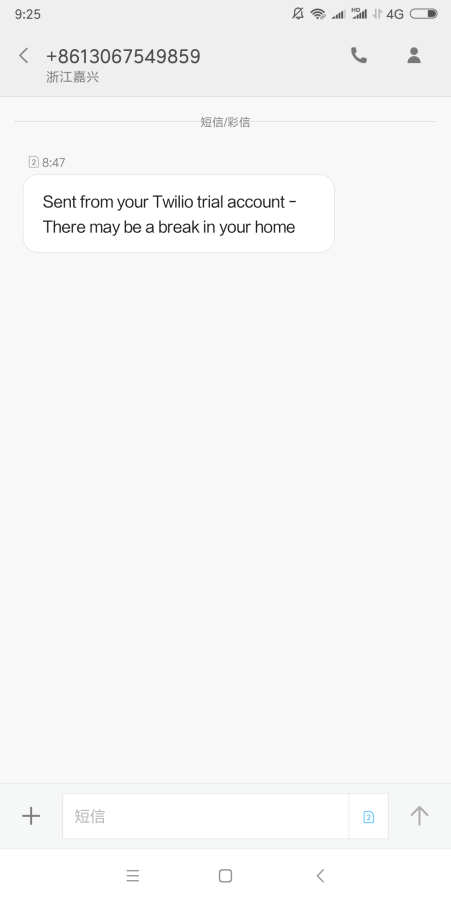
from\_="+15017250604",

body="Hello from Python!")

print(message.sid)

注意这里to中的电话号码是你注册twilio的号码，from是你注册账号时免费得到的twilio电话号码。

通过上述python代码便可以向用户发送短信，下面是接收的短信情况。



具体的代码情况可参考：<https://www.twilio.com/docs/libraries/python>

4.智能监控结合

在第二章我们做了定时监控模块，这里我们将定时监控模块加入进来，重新进行整合。

当人体感应模块感应到人体活动时，蜂鸣器警报报警，与此同时树莓派摄像头模块抓拍照片传到七牛云服务器，并且还会通过上述的twilio向用户发送一条示警短信。

这里提供jiankong.py的代码

# -\*- coding: utf-8 -\*-

import RPi.GPIO as GPIO

import MySQLdb

import time

import picamera

from qiniu import Auth, put\_file, etag, urlsafe\_base64\_encode

import qiniu.config

import os

#需要填写你的 Access Key 和 Secret Key

access\_key = '' #这里的密钥填上刚才我让你记住的密钥对

secret\_key = '' #这里的密钥填上刚才我让你记住的密钥对

#构建鉴权对象

q = Auth(access\_key, secret\_key)

#要上传的空间

bucket\_name = 'jiankong'

#上传到七牛后保存的文件名

key = '%s\_%s\_%s\_%s\_%s\_%s.jpg'%(time.localtime()[0],time.localtime()[1],time.localtime()[2],time.localtime()[3],time.localtime()[4],time.localtime()[5])

#生成上传 Token，可以指定过期时间等

token = q.upload\_token(bucket\_name, key, 3600)

#

camera=picamera.PiCamera()

def uploadfile():

camera.capture("current\_photo.jpg")

#要上传文件的本地路径

localfile = 'current\_photo.jpg'

ret, info = put\_file(token, key, localfile)

filename = 'current\_photo.jpg'

if os.path.exists(filename):

os.remove(filename)

def init():

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BOARD)

GPIO.setup(12, GPIO.IN)

GPIO.setup(11,GPIO.OUT)

pass

def beep():

for i in range(1,6):

GPIO.output(11, GPIO.LOW)

time.sleep(0.5)

GPIO.output(11, GPIO.HIGH)

time.sleep(0.5)

print "the Buzzer will make sound"

def detct():

for i in range(1, 31):

if GPIO.input(12) == True:

print time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time()))+ " Someone is closing!"

conn=MySQLdb.connect(

host='localhost',

port=3306,

user='root',

passwd='root',

db='jiankong',

)

sql="insert into buzzer(time,descr) values('%s','%s')"%(time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time())),'someone is closing')

cur=conn.cursor()

cur.execute(sql)

cur.close()

conn.commit()

conn.close()

beep()

uploadfile()

else:

GPIO.output(11, GPIO.HIGH)

print time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time()))+" Noanybody!"

time.sleep(10)

time.sleep(2)

init()

detct()

六、家庭智慧安防系统之网页端

1.Tomcat的操作

（1）tomcat2的安装

sudo apt-get install apache2

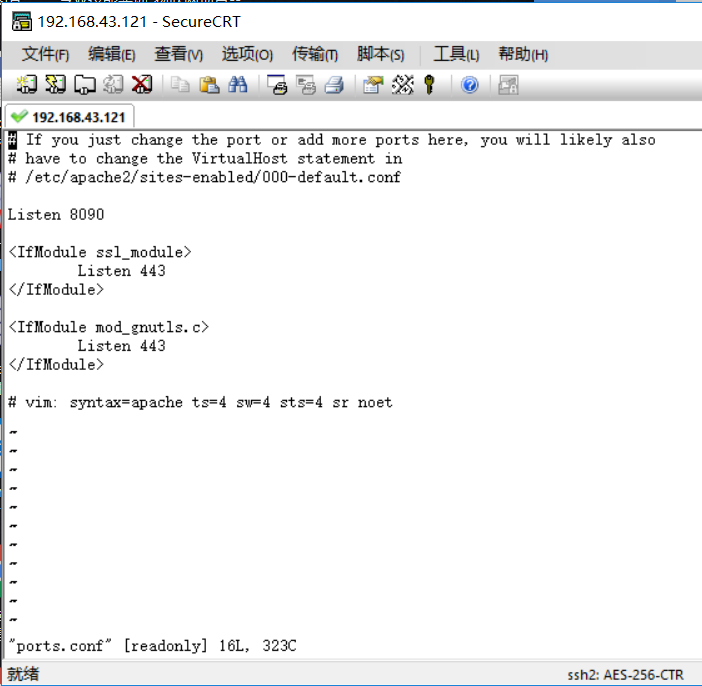
（2）tomcat2的端口的修改

cd /etc/apache2

打开端口配置文件进行修改

vim ports.conf

将Listen的端口修改，这里我修改为8090.



（3）进入tomcat2的工作目录

cd /var/www/html/

将写好的网页放在这里即可通过tomcat来访问。

2.监控网页的编写

安装相关软件依赖

sudo apt-get install php5 libapache2-mod-php5

这里通过php来跟数据库进行连接，获取数据库中的数据，并显示在网页上。网页这里通过了bootstrap来修饰一下，引用bootstrap的代码段。

!-- Bootstrap -->

<link href="https://cdn.bootcss.com/bootstrap/3.3.7/css/bootstrap.min.css" rel="stylesheet">

<!-- jQuery (Bootstrap 的所有 JavaScript 插件都依赖 jQuery，所以必须放在前边) -->

<script src="https://cdn.bootcss.com/jquery/1.12.4/jquery.min.js"></script>

<!-- 加载 Bootstrap 的所有 JavaScript 插件。你也可以根据需要只加载单个插件。 -->

<script src="https://cdn.bootcss.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

jiankong.php的网页

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>家庭实时监控系统</title>

<!-- Bootstrap -->

<link href="https://cdn.bootcss.com/bootstrap/3.3.7/css/bootstrap.min.css" rel="stylesheet">

<!-- jQuery (Bootstrap 的所有 JavaScript 插件都依赖 jQuery，所以必须放在前边) -->

<script src="https://cdn.bootcss.com/jquery/1.12.4/jquery.min.js"></script>

<!-- 加载 Bootstrap 的所有 JavaScript 插件。你也可以根据需要只加载单个插件。 -->

<script src="https://cdn.bootcss.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</head>

<body>

<center>

<table class="table table-striped">

<h1>家庭实时监控系统</h1>

<br>

<h3 align="left" class="bg-primary">温湿度信息</h3>

<tr><th>时间(yyyy-mm-dd)</th><th>温度(°C)</th><th>湿度(%RH)</th></tr>

<?php

$servername = "localhost";

$username = "root";

$password = "root";

$dbname = "jiankong";

// 创建连接

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection

if ($conn->connect\_error) {

die("连接失败: " . $conn->connect\_error);

}

$sql = "select \* from dht2 order by id desc limit 3";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

// 输出数据

while($row = $result->fetch\_assoc()) {

echo"<tr><td>".$row["time"]."</td><td>".$row["temperature"]."</td><td>".$row["humidity"]."</td><tr>";

}

} else {

echo "0 结果";

}

$conn->close();

?>

</table>

<table class="table table-striped">

<h3 align="left" class="bg-primary">安防情况</h3>

<tr><th>序号</th><th>时间</th><th>描述</th></tr>

<?php

$servername = "localhost";

$username = "root";

$password = "root";

$dbname = "jiankong";

// 创建连接

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection

if ($conn->connect\_error) {

die("连接失败: " . $conn->connect\_error);

}

$sql = "select \* from buzzer order by id desc limit 3";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

// 输出数据

while($row = $result->fetch\_assoc()) {

echo"<tr><td>".$row["id"]."</td><td>".$row["time"]."</td><td>".$row["descr"]."</td><tr>";

}

} else {

echo "0 结果";

}

$conn->close();

?>

</table>

</center>

<h3 align="left" class="bg-primary">实时监控情况</h3>

<div>

<img style="-webkit-user-select: none;cursor: zoom-in;" src="http://192.168.43.121:8082/?action=stream" width="400" height="300"/>

</div>

</body>

</html>

结果展示图



3.将IP地址内网穿透，可外网访问

内网穿透的前期背景知识，可参考我的这篇博客

<https://blog.csdn.net/mind_programmonkey/article/details/80285596>

具体步骤：

（1）在这个网站下载python版本的

<https://www.ngrok.cc/download.html>

（2）在这个网址中开通免费隧道，然后将网址的IP地址和端口映射上去（具体的步骤可参考上述博客地址）。这里注意因为一共是两个地址，一个是在线监控的地址，一个是网页的地址，端口不一致，所以我们需要开通两个隧道。这里我采取的办法是用两个账号，分别注册，来映射端口



（3）运行

将（1）下载到的python版本运行，粘贴上隧道id，即可成功开启。

成功运行之后即可通过域名来访问我们家庭的监控情况。



七、家庭智慧安防系统之人脸验证

主要是使用OpenCV和Python进行人脸面部的检测。

1.在linux上安装OpenCV等软件

sudo apt-get update

sudo apt-get install vim build-essential python-software-properties

sudo apt-get install python-opencv python-numpy python-scipy

sudo apt-get install libopencv-dev libopencv-contrib-dev

sudo apt-get install python-picamera python-pil python-tk

测试opencv的状态

pi@raspberrypi:~$ python

Python 2.7.9 (default, Sep 17 2016, 20:26:04)

[GCC 4.9.2] on linux2

Type "help", "copyright", "credits" or "license" for more information.

>>> import cv2

>>> print cv2.\_\_version\_\_

2.4.9.1

2.下载人脸检测的Haar特征分类器

$ wget http://eclecti.cc/files/2008/03/haarcascade\_frontalface\_alt.xml

人脸的Haar特征分类器就是一个XML文件，该文件会描述人脸的Haar的特征值。Haar特征的用途可不止可以用来描述人脸这一种，用来描述眼睛，嘴唇或是其它物体也是可以的。

当然OpenCV已经自带了人脸的Haar特征分类器。OpenCv安装目录中的\data\haarcascades目录下的haarcascade\_frontalface\_alt.xml与haarcascade\_frontalface\_alt2.xml都是用来检测人脸的Haar分类器。这个haarcascades目录下还有人的全身，眼睛，嘴唇的Haar分类器。

3.检查图片中的人脸图像

（1）首先我们新建一个face目录用来存放整个项目的数据和源码



（2）获取原始图像

raspistill -o image.jpg



（3）对image.jpg原始图像进行人脸识别

jiance.py代码

import cv2

def detect(path):

img = cv2.imread(path)

cascade = cv2.CascadeClassifier("/home/pi/haarcascade\_frontalface\_alt.xml")

rects = cascade.detectMultiScale(img, 1.3, 4, cv2.cv.CV\_HAAR\_SCALE\_IMAGE, (20,20))

if len(rects) == 0:

return [], img

rects[:, 2:] += rects[:, :2]

return rects, img

def box(rects, img):

for x1, y1, x2, y2 in rects:

cv2.rectangle(img, (x1, y1), (x2, y2), (127, 255, 0), 2)

cv2.imwrite('/home/pi/face/handle\_data/image.jpg', img);

rects, img = detect("/home/pi/face/data/image.jpg")

box(rects, img)

（4）识别出的图片存储在handle\_data中



4.对检测出人脸的图片进行裁剪，裁剪出人脸图像

（1）face.py代码

#-\*-coding:utf8-\*-

import os

import cv2

import time

import shutil

def getAllPath(dirpath, \*suffix):

PathArray = []

for r, ds, fs in os.walk(dirpath):

for fn in fs:

if os.path.splitext(fn)[1] in suffix:

fname = os.path.join(r, fn)

PathArray.append(fname)

return PathArray

#从源路径中读取所有图片放入一个list，然后逐一进行检查，把其中的脸扣下来，存储到目标路径中

def readPicSaveFace(sourcePath,targetPath,invalidPath,\*suffix):

try:

ImagePaths=getAllPath(sourcePath, \*suffix)

#对list中图片逐一进行检查,找出其中的人脸然后写到目标文件夹下

count = 1

# haarcascade\_frontalface\_alt.xml为库训练好的分类器文件，下载opencv，安装目录中可找到

face\_cascade = cv2.CascadeClassifier('/home/pi/haarcascade\_frontalface\_alt.xml')

for imagePath in ImagePaths:

img = cv2.imread(imagePath)

if type(img) != str:

faces = face\_cascade.detectMultiScale(img, 1.1, 5)

if len(faces):

for (x, y, w, h) in faces:

# 设置人脸宽度大于128像素，去除较小的人脸

if w>=128 and h>=128:

# 以时间戳和读取的排序作为文件名称

listStr = [str(int(time.time())), str(count)]

fileName = ''.join(listStr)

# 扩大图片，可根据坐标调整

X = int(x\*1)

W = min(int((x + w)\*1),img.shape[1])

Y = int(y\*1)

H = min(int((y + h)\*1),img.shape[0])

f = cv2.resize(img[Y:H, X:W], (W-X,H-Y))

cv2.imwrite(targetPath+os.sep+'%s.jpg' % fileName, f)

count += 1

print imagePath + "have face"

else:

shutil.move(imagePath, invalidPath)

except IOError:

print "Error"

else:

print 'Find '+str(count-1)+' faces to Destination '+targetPath

if \_\_name\_\_ == '\_\_main\_\_':

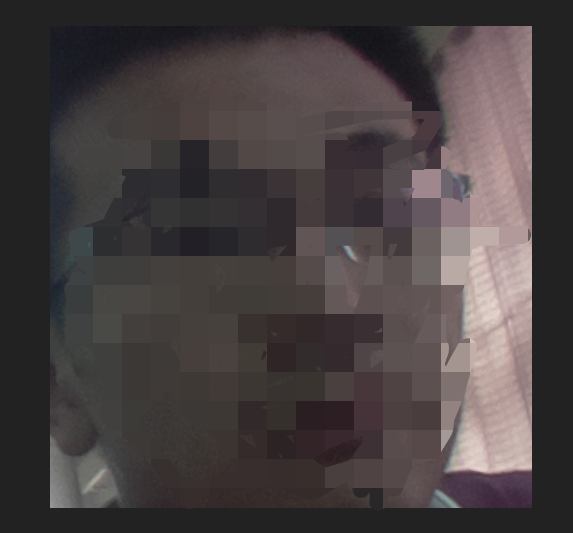
invalidPath = r'/home/pi/face/haveNoPeople'

sourcePath = r'/home/pi/face/data'

targetPath = r'/home/pi/face/faceOfPeople'

readPicSaveFace(sourcePath,targetPath,invalidPath,'.jpg','.JPG','png','PNG')

（2）裁剪出的人脸图像



5.在树莓派上实现人脸识别验证操作

在这里我们利用Face++视觉服务平台，将树莓派作为硬件载体，使用Face++人脸识别云平台搭建一个人脸识别验证的登录系统。

（1）face++的注册与配置

首先登入face++的官方网站

[https://www.faceplusplus.com.cn/](https://www.faceplusplus.com.cn/" \t "https://blog.csdn.net/obuzuopiqi/article/details/_blank)

点击右上角的注册按钮进入注册界面，按步骤完成注册信息填写并进行邮箱验证。 在概览界面可以看到快速开始栏，点击“填写开发者资料”完善相关资料。之后按步骤进行API Key的创建，这是非常关键的一步。

其中类型一定要填写试用，只有创建试用API Key才能使用 API 免费服务，并且一个用户只能有一个免费API Key。免费服务是有一些限制的：

Face 存储有上限：每个用户使用免费服务只能创建 1000个 FaceSet，总计最多存储 100 万个人脸。

一个用户只能有一个 API Key 使用免费服务，而且该 API Key 不能转为正式 API Key。SDK 不提供免费使用或免费测试。

之后在应用管理-API Key界面就可以看到两个非常关键的参数，API Key和API Secret，在所有的POST调用当中均需要这两个参数。

（2）人脸识别的模块

文档中心https://console.faceplusplus.com.cn/documents/4888373

传入图片进行人脸检测和人脸分析。

可以检测图片内的所有人脸，对于每个检测出的人脸，会给出其唯一标识 face\_token，可用于后续的人脸分析、人脸比对等操作。对于正式 API Key，支持指定图片的某一区域进行人脸检测。

如果您需要将检测出的人脸用于后续的分析、比对等操作，建议将对应的 face\_token 添加到 FaceSet 中。如果一个 face\_token 在 72 小时内没有存放在任一 FaceSet 中，则该 face\_token 将会失效。

如果对同一张图片进行多次人脸检测，同一个人脸得到的 face\_token 是不同的。

调用URL：[https://api-cn.faceplusplus.com/facepp/v3/detect](https://api-cn.faceplusplus.com/facepp/v3/detect" \t "https://blog.csdn.net/obuzuopiqi/article/details/_blank)

## 请求参数

|  |  |  |  |
| --- | --- | --- | --- |
| **是否必选** | **参数名** | **类型** | **参数说明** |
| 必选 | api\_key | String | 调用此API的API Key |
| 必选 | api\_secret | String | 调用此API的API Secret |
| 必选（三选一） | image\_url | String | 图片的 URL。  注：在下载图片时可能由于网络等原因导致下载图片时间过长，建议使用 image\_file 或 image\_base64 参数直接上传图片。 |
| image\_file | File | 一个图片，二进制文件，需要用post multipart/form-data的方式上传。 |
| image\_base64 | String | base64 编码的二进制图片数据  如果同时传入了 image\_url、image\_file 和 image\_base64 参数，本API使用顺序为 image\_file 优先，image\_url 最低。 |
| 可选 | return\_landmark | Int | 是否检测并返回人脸关键点。合法值为：   |  |  | | --- | --- | | 2 | 检测。返回 106 个人脸关键点。 | | 1 | 检测。返回 83 个人脸关键点。 | | 0 | 不检测 |   注：本参数默认值为 0 |
| 可选 | return\_attributes | String | 是否检测并返回根据人脸特征判断出的年龄、性别、情绪等属性。合法值为：   |  |  | | --- | --- | | none | 不检测属性 | | * gender * age * smiling * headpose * facequality * blur * eyestatus * emotion * ethnicity * beauty * mouthstatus * eyegaze * skinstatus | 希望检测并返回的属性。  需要将属性组成一个用逗号分隔的字符串，属性之间的顺序没有要求。  关于各属性的详细描述，参见下文“返回值”说明的 "attributes" 部分。  注：由于提供了 eyestatus，所以 glass 参数将被移除（计划时间 2017-9-30），请尽快修改程序使用 eyestatus |   注：本参数默认值为 none |
| 可选（仅正式 API Key 可以使用） | calculate\_all | Int | 是否检测并返回所有人脸的人脸关键点和人脸属性。如果不使用此功能，则本 API 只会对人脸面积最大的五个人脸分析人脸关键点和人脸属性。合法值为：   |  |  | | --- | --- | | 1 | 是 | | 0 | 否 |   注：本参数默认值为 0 |
| 可选（仅正式 API Key 可以使用） | face\_rectangle | String | 是否指定人脸框位置进行人脸检测。  如果此参数传入值为空，或不传入此参数，则不使用此功能。本 API 会自动检测图片内所有区域的所有人脸。  如果使用正式 API Key 对此参数传入符合格式要求的值，则使用此功能。需要传入一个字符串代表人脸框位置，系统会根据此坐标对框内的图像进行人脸检测，以及人脸关键点和人脸属性等后续操作。系统返回的人脸矩形框位置会与传入的 face\_rectangle 完全一致。对于此人脸框之外的区域，系统不会进行人脸检测，也不会返回任何其他的人脸信息。  参数规格：四个正整数，用逗号分隔，依次代表人脸框左上角纵坐标（top），左上角横坐标（left），人脸框宽度（width），人脸框高度（height）。例如：70,80,100,100 |

代码face\_token.py用来获取图片的face\_token的值

#coding=utf8

import cv2

import requests

import json

url = 'https://api-cn.faceplusplus.com/facepp/v3/detect'

files = {'image\_file':open('owner.jpg', 'rb')}

payload = {'api\_key': '\*\*填入你的api值\*\*\*',

'api\_secret': '\*\*填写你的api值\*\*',

'return\_landmark': 0,

'return\_attributes':'gender,age,glass'}

r = requests.post(url,files=files,data=payload)

data=json.loads(r.text)

print r.text

width = data['faces'][0]['face\_rectangle']['width']

top = data['faces'][0]['face\_rectangle']['top']

height = data['faces'][0]['face\_rectangle']['height']

left = data['faces'][0]['face\_rectangle']['left']

img = cv2.imread("owner.jpg")

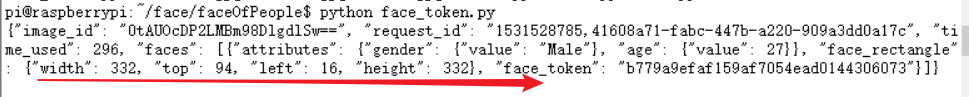
vis = img.copy()

cv2.rectangle(vis, (left, top), (left+width, top+height),(0, 255, 0), 2)

cv2.imshow("Image", vis)

cv2.waitKey (0)

获取的face\_token值，需保存，后续要用。



（3）人脸集合建立的模块

文档中心：https://console.faceplusplus.com.cn/documents/4888391

创建一个人脸的集合FaceSet，用于存储人脸标识face\_token。一个FaceSet能够存储1,000个face\_token。

调用URL：[https://api-cn.faceplusplus.com/facepp/v3/faceset/create](https://api-cn.faceplusplus.com/facepp/v3/faceset/create" \t "https://blog.csdn.net/obuzuopiqi/article/details/_blank)

## 请求参数

|  |  |  |  |
| --- | --- | --- | --- |
| **是否必选** | **参数名** | **类型** | **参数说明** |
| 必选 | api\_key | String | 调用此 API 的 API Key |
| 必选 | api\_secret | String | 调用此 API 的 API Secret |
| 可选 | display\_name | String | 人脸集合的名字，最长256个字符，不能包括字符^@,&=\*'" |
| 可选 | outer\_id | String | 账号下全局唯一的 FaceSet 自定义标识，可以用来管理 FaceSet 对象。最长255个字符，不能包括字符^@,&=\*'" |
| 可选 | tags | String | FaceSet 自定义标签组成的字符串，用来对 FaceSet 分组。最长255个字符，多个 tag 用逗号分隔，每个 tag 不能包括字符^@,&=\*'" |
| 可选 | face\_tokens | String | 人脸标识 face\_token，可以是一个或者多个，用逗号分隔。最多不超过5个 face\_token |
| 可选 | user\_data | String | 自定义用户信息，不大于16 KB，不能包括字符^@,&=\*'" |
| 可选 | force\_merge | Int | 在传入 outer\_id 的情况下，如果 outer\_id 已经存在，是否将 face\_token 加入已经存在的 FaceSet 中  0：不将 face\_tokens 加入已存在的 FaceSet 中，直接返回 FACESET\_EXIST 错误  1：将 face\_tokens 加入已存在的 FaceSet 中  默认值为0 |

face\_tokens.py所需的代码

#coding=utf8

import requests

url = 'https://api-cn.faceplusplus.com/facepp/v3/faceset/create'

payload = {'api\_key': '\*\*\*申请的值\*\*\*\*',

'api\_secret': '\*\*\*申请的值\*\*\*',

'display\_name':'jiankong\_lcz',

'outer\_id':'jiankong\_lcz',

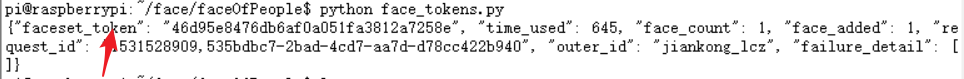
'face\_tokens':'上一步图片的face\_token的值'

}

r = requests.post(url,data=payload)

print r.text

返回的faceset\_token值，也需要保存



（4）人脸搜索对比模块

文档中心：<https://console.faceplusplus.com.cn/documents/4888381>

在Faceset中找出与目标人脸最相似的一张或多张人脸。支持传入face\_token或者直接传入图片进行人脸搜索。使用图片进行比对时会选取图片中检测到人脸尺寸最大的一个人脸。

调用URL：[https://api-cn.faceplusplus.com/facepp/v3/search](https://api-cn.faceplusplus.com/facepp/v3/search" \t "https://blog.csdn.net/obuzuopiqi/article/details/_blank)

## 请求参数

|  |  |  |  |
| --- | --- | --- | --- |
| **是否必选** | **参数名** | **类型** | **参数说明** |
| 必选 | api\_key | String | 调用此 API 的 API Key |
| 必选 | api\_secret | String | 调用此 API 的 API Secret |
| 必选（四选一） | face\_token | String | 进行搜索的目标人脸的 face\_token，优先使用该参数 |
| image\_url | String | 目标人脸所在的图片的 URL |
| image\_file | File | 目标人脸所在的图片，二进制文件，需要用 post multipart/form-data 的方式上传。 |
| image\_base64 | String | base64 编码的二进制图片数据  如果同时传入了 image\_url、image\_file 和 image\_base64 参数，本 API 使用顺序为 image\_file 优先，image\_url 最低。 |
| 必选（二选一） | faceset\_token | String | 用来搜索的 FaceSet 的标识 |
| outer\_id | String | 用户自定义的 FaceSet 标识 |
| 可选 | return\_result\_count | Int | 控制返回比对置信度最高的结果的数量。合法值为一个范围 [1,5] 的整数。默认值为 1 |
| 可选（仅正式 API Key 可以使用） | face\_rectangle | String | 当传入图片进行人脸检测时，是否指定人脸框位置进行检测。  如果此参数传入值为空，或不传入此参数，则不使用此功能。本 API 会自动检测图片内所有区域的所有人脸。  如果使用正式 API Key 对此参数传入符合格式要求的值，则使用此功能。需要传入一个字符串代表人脸框位置，系统会根据此坐标对框内的图像进行人脸检测，以及人脸关键点和人脸属性等后续操作。系统返回的人脸矩形框位置会与传入的 face\_rectangle 完全一致。对于此人脸框之外的区域，系统不会进行人脸检测，也不会返回任何其他的人脸信息。  参数规格：四个正整数，用逗号分隔，依次代表人脸框左上角纵坐标（top），左上角横坐标（left），人脸框宽度（width），人脸框高度（height）。例如：70,80,100,100  注：只有在传入 image\_url、image\_file 和 image\_base64 三个参数中任意一个时，本参数才生效。 |

distinguish.py所需的代码：

#coding=utf8

import cv2

import cv2.cv as cv

import requests

import json

import picamera

import os

camera=picamera.PiCamera()

camera.capture("current\_photo.jpg")

filename='current\_photo.jpg'

url = 'https://api-cn.faceplusplus.com/facepp/v3/search'

payload = {'api\_key': '\*\*\*填入你申请的key值\*\*\*\*',

'api\_secret': '\*\*\*填入你申请的key值\*\*\*\*',

'faceset\_token':'\*\*上一步你获取对的faceset值\*\*',

}

files = {'image\_file':open('current\_photo.jpg', 'rb')}

r = requests.post(url,files=files,data=payload)

data=json.loads(r.text)

print r.text

if os.path.exists(filename):

os.remove(filename)

if data["results"][0]["face\_token"] == "\*\*\*owner图片的face\_token值\*\*\*" and data["results"][0]["confidence"]>=data["thresholds"]["1e-5"]:

print'\n主人'

else:

print '\n闯入者'

首先这个会调用树莓派的摄像头获取当前图像，并截图，截图成功之后会将这张人脸图片上传至云端进行对比并返回结果。

验证结果：

