



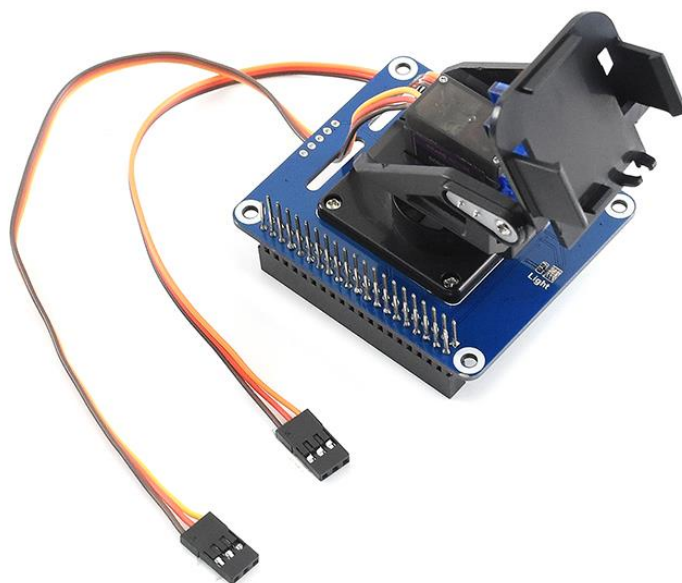
# Pan-Tilt HAT

## 用户手册

### 产品概述

本产品是基于树莓派的云台扩展板，板载 PCA9685 芯片，输出 PWM 控制云台转动。板载 TSL2581 环境光传感器，通过检测光强辅助摄像头工作。通过 I2C 接口控制，无需占用额外的引脚。

提供完善的配套树莓派例程（BCM2835 库，WiringPi 库，以及 python 例程）



### 产品参数

工作电压:	3.3V/5V
控制芯片:	PCA9685
逻辑电压:	3.3V
通信接口:	I2C
产品尺寸:	56.6X65(mm)

## 目录

产品概述 .....	1
产品参数 .....	1
硬件说明 .....	3
控制器 .....	3
通信协议 .....	3
I2C 写数据时序 .....	3
I2C 读数据时序 .....	4
I2C 地址 .....	4
使用指南 .....	5
下载例程 .....	5
例程使用 .....	6
拷贝到树莓派 .....	6
Install libraries .....	7
组装 .....	8
运行测试程序 .....	10
摄像头 .....	11
网络视频远程监控和远程云台控制 .....	12
预期效果 .....	16
常见问题 .....	17

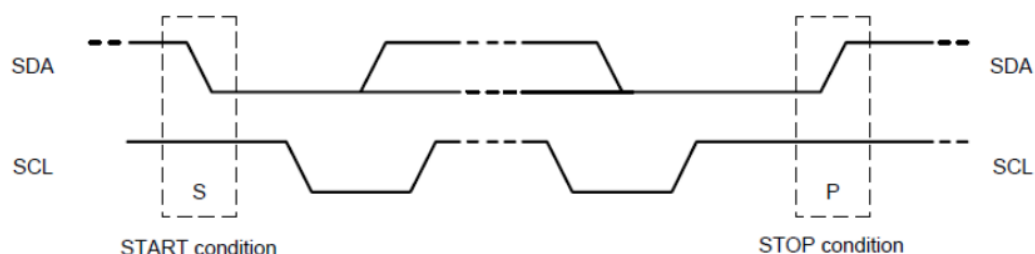
## 硬件说明

### 控制器

本产品控制器为 PCA9685，是一款基于 IIC 总线通信的 12 位精度 16 通道 PWM 波输出的芯片。并且板载 TSL2581 环境光传感器，通过检测光强辅助摄像头工作，同样通过 I2C 接口控制，不会占用太多接口引脚资源。

### 通信协议

从上的得知使用的是 I2C 通信，I2C 通信，一条数据线，一条时钟线。I2C 总线在传送数据过程中共有三种类型信号：开始信号、结束信号和应答信号。

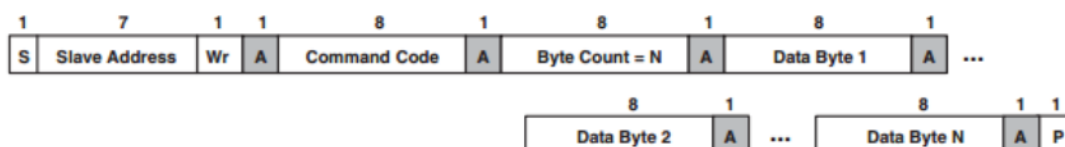


开始信号：SCL 为高电平时，SDA 由高电平向低电平跳变，开始传送数据。

结束信号：SCL 为高电平时，SDA 由低电平向高电平跳变，结束传送数据。

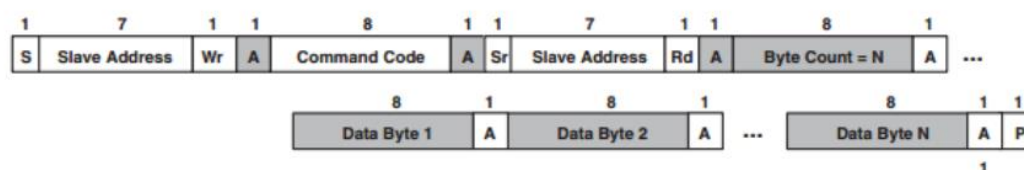
应答信号：接收数据的 IC 在接收到 8bit 数据后，向发送数据的 IC 发出特定的低电平脉冲，表示已收到数据。

### I2C 写数据时序



首先主机（即树莓派，后面统称为主机）会发送一个开始信号，然后将其 I2C 的 7 位地址与写操作位组合成 8 位的数据发送给从机（即 TSL2581 传感器模块，后面统称为从机），从机接收到后会响应一个应答信号，主机此时将命令寄存器地址发送给从机，从机接收到发送响应信号，此时主机发送命令寄存的值，从机回应一个响应信号，直到主机发送一个停止信号，此次 I2C 写数据操作结束

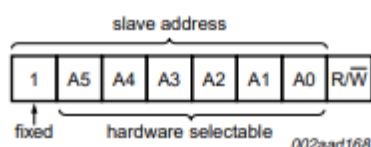
## I2C 读数据时序



首先主机会发送一个开始信号，然后将其 I2C 的 7 位地址与写操作位组合成 8 位的数据发送给从机，从机接收到后会响应一个应答信号，主机此时将命令寄存器地址发送给从机，从机接收到发送响应信号，此时主机重新发送一个开始信号，并且将其 7 位地址和读操作位组合成 8 位的数据发送给从机，从机接收到信号后发送响应信号，再将其寄存器中的值发送给主机，主机端给予响应信号，直到主机端发送停止信号，此次通信结束

## I2C 地址

PCA9685 的 I2C 地址



PCA9685 数据手册第 7 页

TSL2581 的 I2C 地址:

Address SEL Terminal Level	Slave Address	SMB Alert Address
GND	0101001	0001100
Float	0111001	0001100
VDD	1001001	0001100

TSL2581 数据手册第 13 页

注：我们模块默认 PCA9685 的 I2C 地址引脚 A5=A4=A3=A2=A1=A0=0，PCA9685 的 I2C 地址为 0x40，TSL2581 的 I2C 地址引脚浮空（Float），TSL2581 的 I2C 地址为 0x39。如果用户不使用树莓派驱动时候，例如使用 STM32 的时候需要在低位补上 R/W 位。

## 使用指南

### 下载例程

在微雪电子官网上找到对应产品，在产品资料打开下载路径，在 wiki 中下载示例程序：

#### 文档

- [用户手册](#)
- [原理图](#)

#### 程序

- [示例程序](#)

将下载下来的解压包解压，得到如下文件：

 Light Sensor	2019/1/8 10:57	文件夹
 Servo Driver	2019/1/8 10:57	文件夹
 test	2019/1/8 10:57	文件夹
 web_Python	2019/1/8 10:40	文件夹

Servo Driver: 云台舵机程序（BCM2835，WringPi 和 Python 三种例程）

Light Sensor: 环境光测量程序（BCM2835，WringPi 和 Python 三种例程）

test: 舵机安装前的测试程序

web\_Python: 云台摄像头远程控制程序

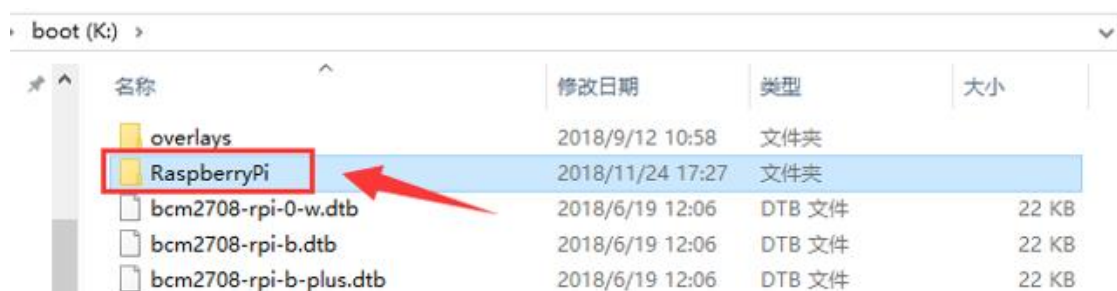
## 例程使用

### 拷贝到树莓派

1. 使用读卡器将 SD 卡插入电脑，将会显示一个名为 Boot 的可移动盘。



2. 将解压文件中 RaspberryPi 文件夹复制到 boot 根目录下



3. 然后弹出 U 盘，将 SD 卡插入树莓派中，插上 USB 上电，查看/boot 目录的文件

```
pi@raspberrypi:~ $ ls /boot/
bcm2708-rpi-0-w.dtb  bcm2710-rpi-3-b.dtb  config.txt  fixup_x.dat  kernel.img  start_cd.elf
bcm2708-rpi-b.dtb   bcm2710-rpi-3-b-plus.dtb  COPYING.linux  FSCK0000.REC  LICENCE.broadcom  start_db.elf
bcm2708-rpi-b-plus.dtb  bcm2710-rpi-cm3.dtb  fixup_cd.dat  FSCK0001.REC  LICENSE.oracle  start_elf
bcm2708-rpi-cm.dtb    bootcode.bin        fixup.dat     issue.txt    overlays      start_x.elf
bcm2709-rpi-2-b.dtb   cmdline.txt         fixup_db.dat  kernel7.img  RaspberryPi  System Volume Information
```

4. 执行如下命令将其复制到用户目录下，并修改其用户权限

```
sudo cp -r /boot/RaspberryPi/ ./
```

```
sudo chmod 777 -R RaspberryPi/
```

```
pi@raspberrypi:~ $ sudo cp -r /boot/RaspberryPi/ ./
pi@raspberrypi:~ $ ls
code libcode RaspberryPi RPiLib ubuntu usbdisk
pi@raspberrypi:~ $ sudo chmod 777 -R RaspberryPi/
pi@raspberrypi:~ $ ls
code libcode RaspberryPi RPiLib ubuntu usbdisk
```

5. 进入目录，查看文件：

```
pi@raspberrypi:~ $ cd RaspberryPi
pi@raspberrypi:~/RaspberryPi $ ls
Light Sensor Servo Driver test web_Python
pi@raspberrypi:~/RaspberryPi $
```

## INSTALL LIBRARIES

需要安装必要的函数库（wiringPi、bcm2835、python 库），否则以下的示例程序可能无法正常工作。安装方法详见：

### 安装 BCM2835 库：

<http://www.airspayce.com/mikem/bcm2835/>

进入 BCM2835 的官网下载并把安装包复制到树莓派上，运行如下：

```
sudo tar zxvf bcm2835-1.xx.tar.gz

cd bcm2835-1.xx

sudo ./configure

make

sudo make check

sudo make install
```

其中 xx 代表的是下载的版本号，例如我下载的 bcm2835-1.52

那么就应该执行：sudo tar zxvf bcm2835-1.52.tar.gz

### 安装 wiringPi 库：

```
sudo apt-get install git

sudo git clone git://git.drogon.net/wiringPi

cd wiringPi

sudo ./build
```

### 安装 python 库：

```
sudo apt-get install python-pip

sudo pip install RPi.GPIO
```

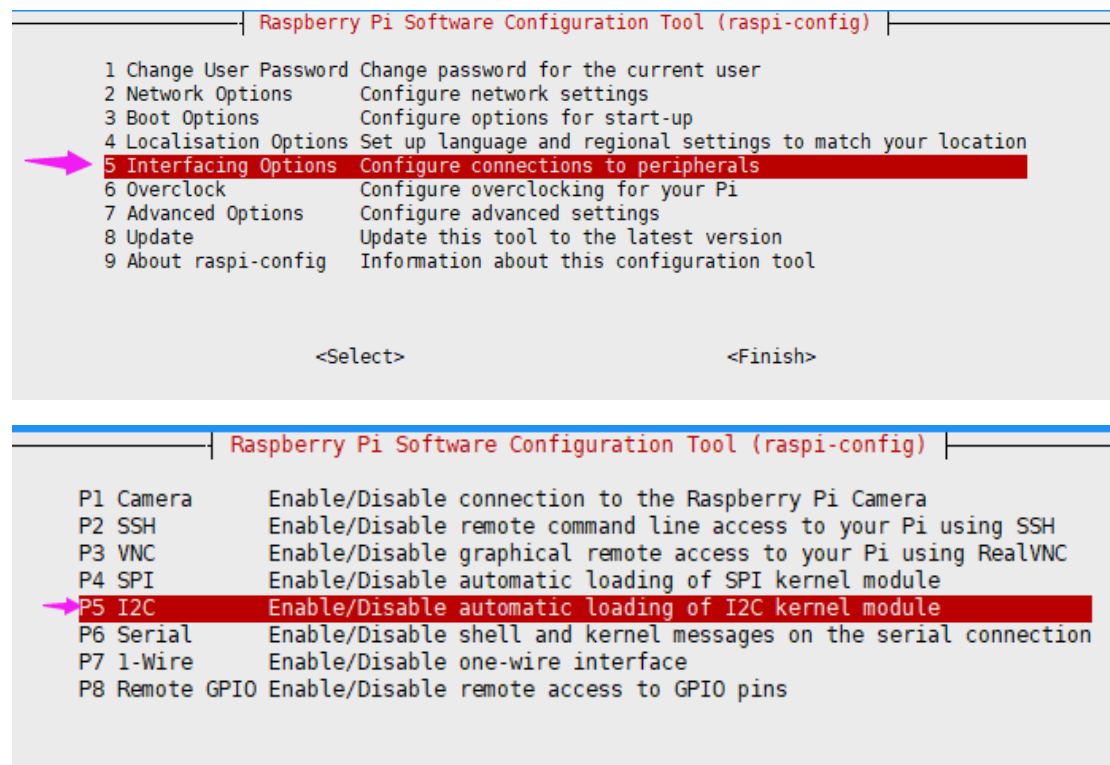
```
sudo pip install spidev

sudo apt-get install python-imaging

sudo apt-get install python-smbus
```

## 开启 I2C 接口:

```
sudo raspi-config
```



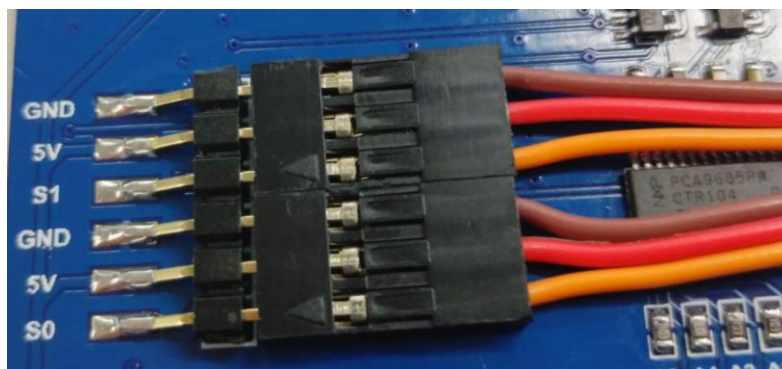
## 组装

注意：组装前进行调试，舵机初始角度不是在起始位置，舵机旋转时可能会卡死，所以建议在第一次使用时先不要组装云台，先单独测试舵机转的角度，防止舵机意外损坏。

## 组装前调试

连接：





棕色线	GND
红色线	5V
黄色	信号线 (S1/S0)

运行测试代码前一定要注意倾斜舵机安装位置，安装位置不好可能会导致舵机发热烧毁，先不组装舵机，保证舵机可以旋转 360 不会有障碍，连接硬件，**平移舵机连接 S1，倾斜舵机连接 S0**。然后打开程序文件夹中 test 文件夹

 Light Sensor	2019/1/8 10:57	文件夹
 Servo Driver	2019/1/8 10:57	文件夹
 <b>test</b>	2019/1/8 10:57	文件夹
 web_Python	2019/1/8 10:40	文件夹

请先检查，倾斜舵机是安装在 S0 通道，再运行程序：

```
make
```

```
sudo ./main
```

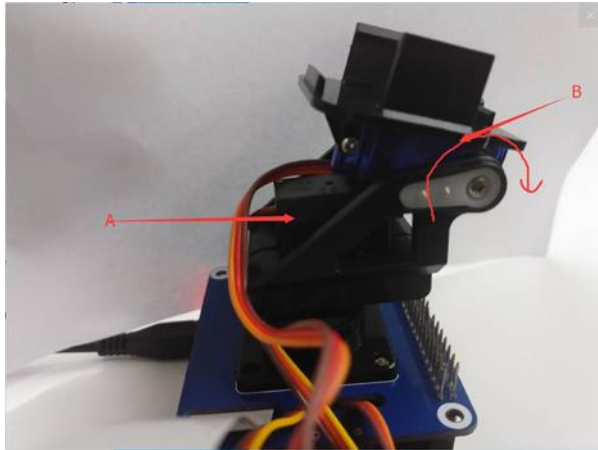
注意不要修改 test 里面的文件。

运行后，舵机会全部旋转到 0 度（舵机旋转角度为 0 到 180，也就是旋转到初始位置）。然后

切断电源组装，注意不要选择倾斜舵机，将倾斜舵机安装如下：

具体组装视频可以参考：<http://www.waveshare.net/wiki/AlphaBot2-PiZero-Video> 1 分

19 秒开始



A: 倾斜舵机

B: 平移舵机

倾斜舵机初始位置一定要如图所示，平行与底部，箭头方向为倾斜舵机旋转方向。舵机旋转方向，如图所示，倾斜舵机

---

### 运行测试程序

在 RaspberryPi 目录下，云台舵机程序 Servo Driver、环境光强测量程序 Light Sensor，进入对应的目录下运行：

BCM2835 例程：

```
cd bcm2835  
sudo ./main
```

wiringPi 例程：

```
cd wiringpi  
sudo ./main
```

python 例程：

```
sudo python main.py
```

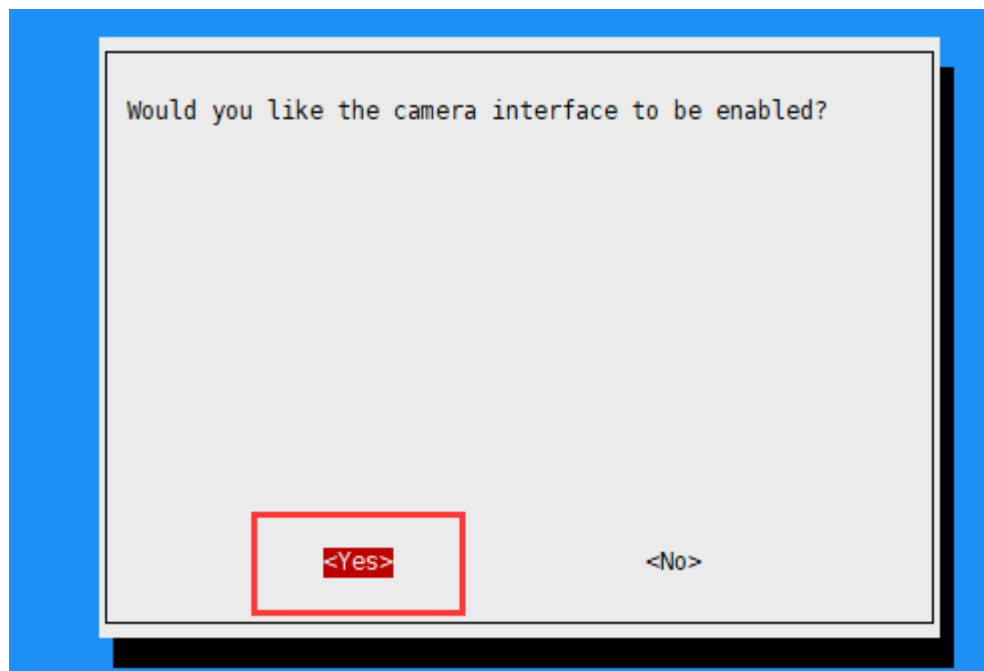
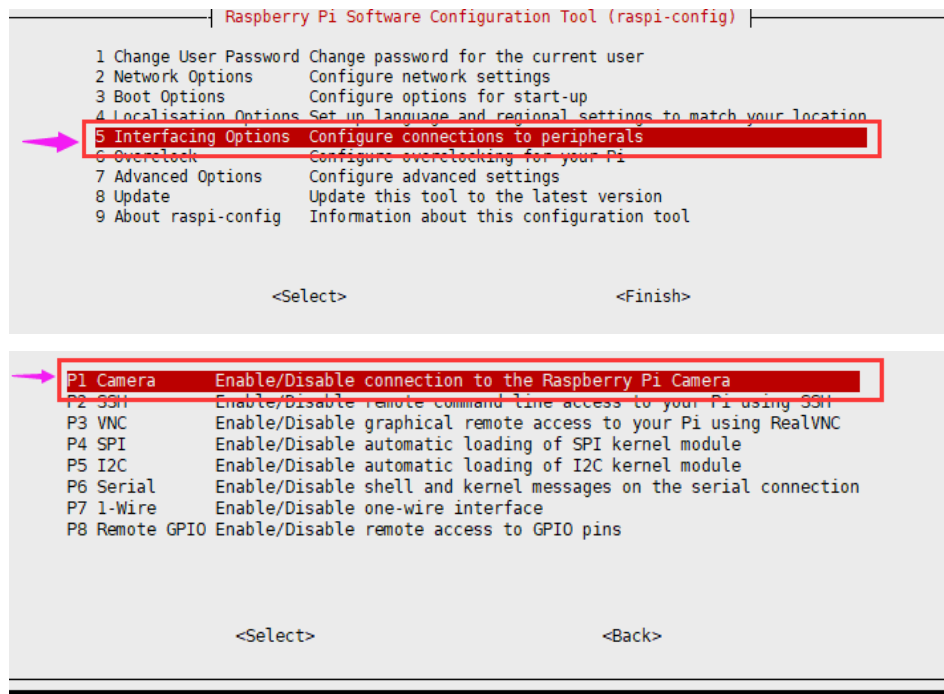
注：BCM2835、wiringpi 例程运行如果提示找不到文件，执行 make 即可

## 摄像头

1. 将树莓派摄像头连接到树莓派
2. 开启摄像头

输入 `sudo raspi-config`

按下图操作：



### 3. 重启树莓派

```
sudo reboot
```

### 4. 测试摄像头

用 raspistill 指令拍摄一张.jpg 格式的图片，并保存在当前目录下：

```
raspistill -o image.jpg
```

具体 raspistill 的使用可以在终端输入 raspistill -h 查看

### 5. 录像：

```
raspivid -o video.h264 -t 1000
```

-t 1000 表示录制 1000ms。视频保存在当前目录下，视频格式为.H264，播放可以把文件复制到 windows 环境下使用视频播放软件播放

---

## 网络视频远程监控和远程云台控制

如果你想通过网络远程控制云台、摄像头，下面为你介绍：

### 1. 安装 mjpg-streamer 网络视频监控

参考微雪课堂：<http://www.waveshare.net/study/article-764-1.html>

### 2. 打开摄像头

参考上一节操作

### 3. 添加设备

```
sudo nano /etc/modules
```

在文件最后加上 bcm2835-v4l2 （注意 4l2 中的 l 是字母 L 小写而不是数字 1）

重启树莓派之后，在 /dev 目录下会多出一个 video0 的设备节点

sudo reboot

```
pi@raspberrypi:~$ ls /dev/
autofs      initctl      queue        ram5         tty11        tty29        tty46        tty63        vcs6
block       input        net          ram6         tty12        tty3          tty47        tty7         vcs7
btrfs-control kmsg        network_latency ram7         tty13        tty30        tty48        tty8         vcsa
bus         log          network_throughput ram8         tty14        tty31        tty49        tty9         vcsa1
cachefiles  loop0        null         ram9         tty15        tty32        tty5         ttyAMA0      vcsa2
char        loop1        ppp          random       tty16        tty33        tty50        ttyprintk    vcsa3
console     loop2        ptmx         raw          tty17        tty34        tty51        ttyS0        vcsa4
cpu_dma_latency loop3        pts          rkill        tty18        tty35        tty52        uhid         vcsa5
cuse        loop4        ram0         serial0      tty19        tty36        tty53        uinput       vcsa6
disk        loop5        ram1         serial1      tty2         tty37        tty54        urandom      vcsa7
fb0         loop6        ram10        shm          tty20        tty38        tty55        vchiq        vcsm
fd          loop7        ram11        snd          tty21        tty39        tty56        vcio         vhci
full        loop-control ram12        stderr       tty22        tty4         tty57        vc-mem       video0
fuse        mapper       ram13        stdin        tty23        tty40        tty58        vcs          watchdog
gpiochip0   mem          ram14        stdout       tty24        tty41        tty59        vcs1         watchdog0
gpiochip1   memory_bandwidth ram15        tty          tty25        tty42        tty6         vcs2         zero
gpiochip2   mmcblk0     ram2         tty0         tty26        tty43        tty60        vcs3
gpiomem     mmcblk0p1   ram3         tty1         tty27        tty44        tty61        vcs4
hwrng       mmcblk0p2   ram4         tty10        tty28        tty45        tty62        vcs5
```

#### 4. 安装依赖库

```
sudo apt-get install libv4l-dev libjpeg8-dev
sudo apt-get install subvert
```

#### 5. 运行程序

进入 web\_Python 目录: `cd RaspberryPi/web_Python`

```
pi@raspberrypi:~$ cd RaspberryPi/web_Python/
pi@raspberrypi:~/RaspberryPi/web_Python$
```

输入命令 `pwd` 得到当前目录信息, 复制目录:

```
pi@raspberrypi:~/RaspberryPi/web_Python$ sudo nano
pi@raspberrypi:~/RaspberryPi/web_Python$ pwd
~/RaspberryPi/web_Python
pi@raspberrypi:~/RaspberryPi/web_Python$
```

#### 6. 修改程序

输入命令 `sudo nano main.py` 找到 `os.chdir('')` 函数, 将红箭的目录改成你刚刚复制

的目录, 然后在后面加上 `/mjpg`

```

-*- coding: UTF-8 -*-
import threading
import SocketServer
import RPi.GPIO as GPIO
from PCA9685 import PCA9685
from SocketServer import StreamRequestHandler as SRH
from time import ctime
import time

import thread
import os
import sys

o_path = os.getcwd()
sys.path.append(o_path)
os.chdir('.../mjpg') ←
from mjpg import camera
# import mod1

pwm = PCA9685()
pwm.setPWMFreq(50)
pwm.setRotationAngle(0, 0)
pwm.setRotationAngle(1, 0)

```

然后保存退出

## 7. 查看树莓派当前 IP: ifconfig

如果你是直接连接的网线，查看的是 eth0 的 IP 地址

```

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 1... netmask 255.255.255.0 broadcast 192.168.1.
    inet6 fe80::495c:b3c2:30d8:38ff prefixlen 64 scopeid 0x20<lin
    ether b8:27:eb:26:26:8b txqueuelen 1000 (Ethernet)
    RX packets 13599 bytes 1179076 (1.1 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0

```

如果你是通过 wifi 连接的，查看的则是 wlan0 的 IP 地址

```

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 3 netmask 255.255.255.0 broadcast 192.1
    inet6 fe80::ceba:ae60:3947:7a06 prefixlen 64 scopeid 0x2
    ether b8:27:eb:73:73:de txqueuelen 1000 (Ethernet)

```

```

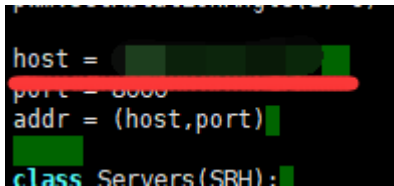
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.218 netmask 255.255.255.0 broadcast 192.168.1.
    inet6 fe80::ceba:ae60:3947:7a06 prefixlen 64 scopeid 0x20<lin
    ether b8:27:eb:73:73:de txqueuelen 1000 (Ethernet)
    RX packets 9274 bytes 1197461 (1.1 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 37 bytes 5696 (5.5 KiB)

```

复制 IP

## 8. 再次编辑 main.py 文件: `sudo nano main.py`

将 host=后面的值改成树莓派的当前 IP



```

host = 192.168.1.1
port = 8080
addr = (host,port)






class Servers(SRH):

```

## 9. PC 端下载微雪树莓派智能小车软件（只支持 Windows 系统）。

[http://www.waveshare.net/w/upload/a/a9/AlphaBot\\_Qt.7z](http://www.waveshare.net/w/upload/a/a9/AlphaBot_Qt.7z)

## 10. 安装软件（解压后直接打开即可）

	platforms	2017/3/22 14:52	文件夹	
	translations	2017/3/22 14:52	文件夹	
	AlphaBot.exe	2017/3/22 14:41	应用程序	298 KB
	D3Dcompiler_47.dll	2015/7/10 19:00	应用程序扩展	3,606 KB
	libEGL.dll	2017/1/19 4:44	应用程序扩展	22 KB

## 11. 在 IP 输入框中输入树莓派的当前 IP，然后树莓派端运行程序: `sudo python main.py`

程序运行后，舵机湖全部恢复在 0 度（其实位置），屏幕出现以下提示（忽略）

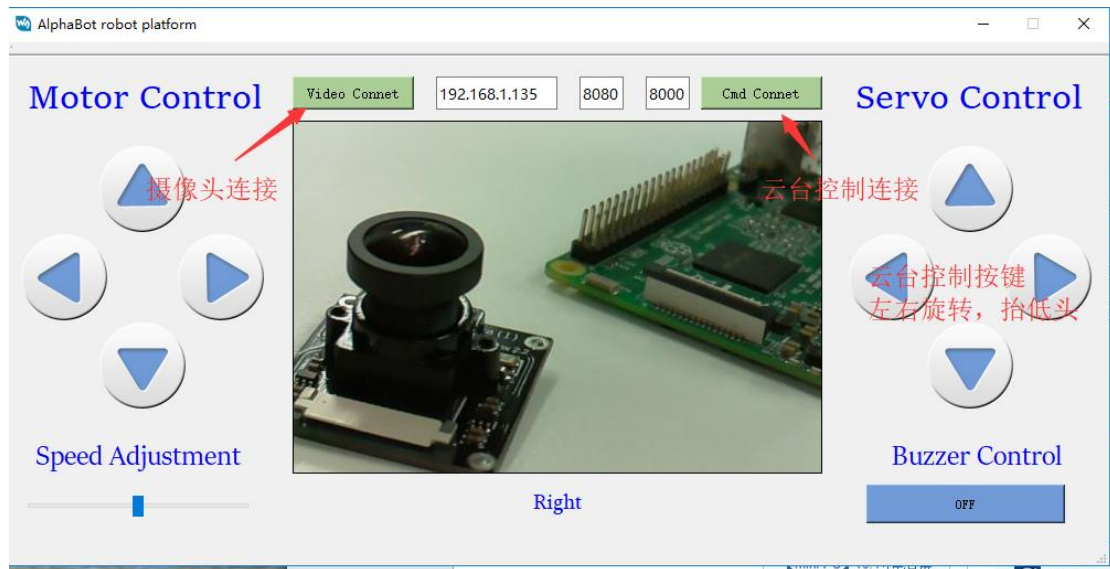


```

pi@raspberrypi:~/RaspberryPi/web_Python $ sudo python main.py
server is running....
MJPG Streamer Version: svn rev: Unversioned directory
i: Using V4L2 device.: /dev/video0
i: Desired Resolution: 640 x 480
i: Frames Per Second.: 10
i: Format.....: YUV
i: JPEG Quality.....: 80
Adding control for Pan (relative)
UVCIOC_CTRL_ADD - Error: Inappropriate ioctl for device
Adding control for Tilt (relative)
UVCIOC_CTRL_ADD - Error: Inappropriate ioctl for device
Adding control for Pan Reset
UVCIOC_CTRL_ADD - Error: Inappropriate ioctl for device
Adding control for Tilt Reset
UVCIOC_CTRL_ADD - Error: Inappropriate ioctl for device
Adding control for Pan/tilt Reset
UVCIOC_CTRL_ADD - Error: Inappropriate ioctl for device
Adding control for Focus (absolute)
UVCIOC_CTRL_ADD - Error: Inappropriate ioctl for device
mapping control for Pan (relative)
UVCIOC_CTRL_MAP - Error: Inappropriate ioctl for device
mapping control for Tilt (relative)
UVCIOC_CTRL_MAP - Error: Inappropriate ioctl for device
mapping control for Pan Reset
UVCIOC_CTRL_MAP - Error: Inappropriate ioctl for device
mapping control for Tilt Reset
UVCIOC_CTRL_MAP - Error: Inappropriate ioctl for device
mapping control for Pan/tilt Reset
UVCIOC_CTRL_MAP - Error: Inappropriate ioctl for device
mapping control for Focus (absolute)
UVCIOC_CTRL_MAP - Error: Inappropriate ioctl for device
mapping control for LED1 Mode
UVCIOC_CTRL_MAP - Error: Inappropriate ioctl for device
mapping control for LED1 Frequency
UVCIOC_CTRL_MAP - Error: Inappropriate ioctl for device
mapping control for Disable video processing
UVCIOC_CTRL_MAP - Error: Inappropriate ioctl for device
mapping control for Raw bits per pixel
UVCIOC_CTRL_MAP - Error: Inappropriate ioctl for device
o: www-folder-path....: ./www/
o: HTTP TCP port.....: 8080
o: username:password.: disabled
o: commands.....: enabled

```

正常连接后软件效果图：



如果需要退出程序，注意先断开云台控制，如果不断开云台控制直接使用 ctrl+c 强制退出

的话可能导致下次无法连接，为了避免不必要的麻烦，建议你再退出程序前先断开连接，

点击一下 cmd Connt

参考微雪课堂：<http://www.waveshare.net/study/article-768-1.html>

## 预期效果

### Servo Driver:

控制云台左旋转 倾斜舵机慢慢低头

控制云台右旋转 倾斜舵机慢慢抬头

### Light Sensor:

先输出设备 ID （这个 ID 不是 IIC 地址）

然后输出环境光光强数值

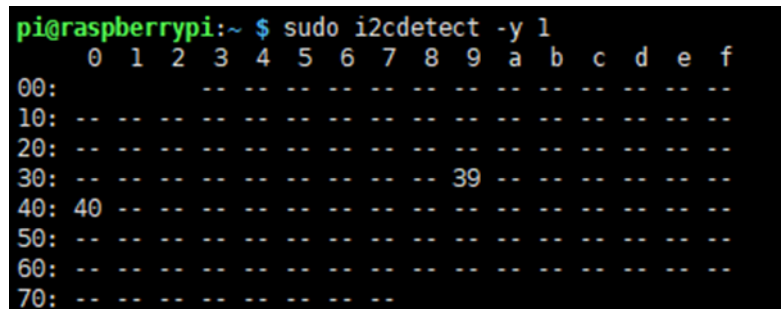
```
pi@raspberrypi:~/RaspberryPi/Light Sensor/bcm2835 $ sudo ./main
bcm2835 init success !!!
READ ID = 0x90
---- i2c sensor init ----
lux = 114
lux = 114
lux = 113
lux = 107
lux = 115
lux = 107
lux = 103
lux = 96
lux = 98
lux = 99
lux = 97
lux = 101
lux = 102
lux = 102
lux = 102
lux = 102
lux = 102
lux = 100
lux = 110
lux = 111
lux = 110
```



## 常见问题

1. 环境光强检测程序输出 ID 为 0xf0 或者 0x00, 光强输出也为 0, 为什么?

答: 可能是不正确的使用树莓派控制可能会导致 (详情看下面的), 如果重启后还是不正常那么你尝试重启后输入命令 `sudo i2cdetect -y 1` 下会有两个地址



```
pi@raspberrypi:~$ sudo i2cdetect -y 1
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  39  --  --  --  --  --
40: 40  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
```

检查 I2C 是否更改过 I2C 地址或者没有打开 I2C 功能, 默认情况 39 为环境光传感器 I2C 地址, 40 为 PWM 芯片 PCA9685 的 I2C 地址

2. 不正确的使用树莓派控制可能会导致?

答: 如果运行 wiringPi 例程正常, 再运行 python 或者 BCM2835 可能会屏幕无法正常刷新, 因为 bcm2835 库是树莓派 cpu 芯片的库函数, 底层是直接操作寄存器, 而 wiringPi 库和 python 的底层都是通过读写 linux 系统的设备文件操作设备, 可能导致 GPIO 口异常, 重启树莓派可完美解决。