Raspberry Pi Camera + OpenCV

台灣樹莓派 <sosorry@raspberrypi.com.tw>

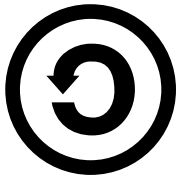
2017/01/21 @Techbang

CC (Creative Commons)

姓名標示 — 非商業性 — 相同方式分享

姓名標示 — 你必須給予 適當表彰、提供指向本授權條款的連結，以及 指出（本作品的原始版本）是否已被變更。你可以任何合理方式為前述表彰，但不得以任何方式暗示授權人為你或你的使用方式背書。

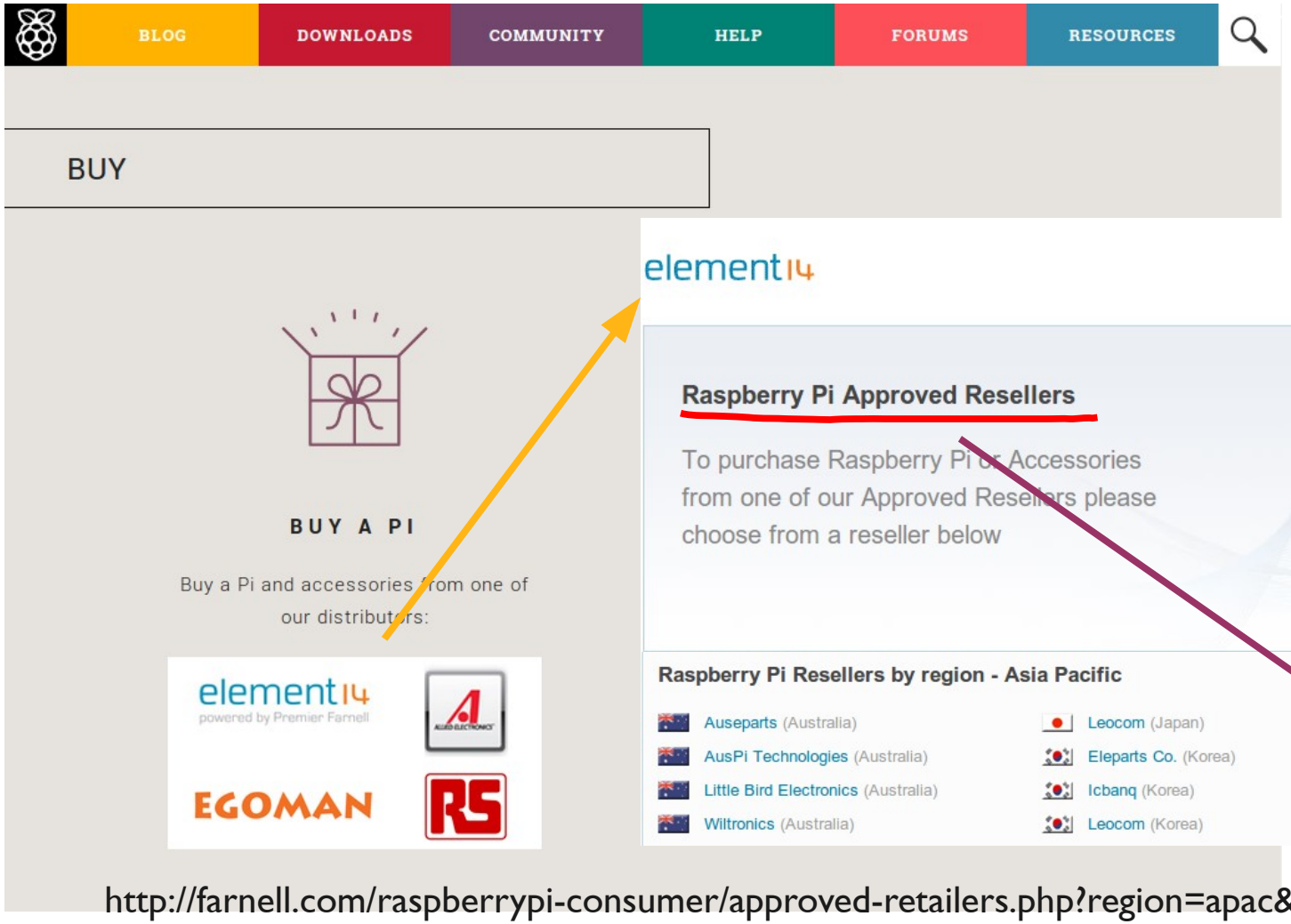
非商業性 — 你不得將本素材進行商業目的之使用。



相同方式分享 — 若你重混、轉換本素材，或依本素材建立新素材，你必須依本素材的授權條款來散布你的貢獻物。

about 台灣樹莓派

- Raspberry Pi 官方經銷商



http://farnell.com/raspberrypi-consumer/approved-retailers.php?region=apac&MER=MER-LM-OB-RPICC-76315

about 台灣樹莓派

- 專注於 Raspberry Pi 應用與推廣
- 舉辦社群聚會 / 工作坊 / 讀書會 / 黑客松
- Website ：
  - https://www.raspberrypi.com.tw/

- Facebook ：
  - 搜尋 RaspberryPi.Taiwan
  - https://www.facebook.com/RaspberryPi.Taiwan

分享 x 社群

- COSCUP,MakerConf,PyCon 講者
- 投影片
  - http://www.slideshare.net/raspberrypi-tw/presentations
- 程式碼
  - https://github.com/raspberrypi-tw

目標

- 了解相機原理
- 學習如何控制 Raspberry Pi Camera
- 學習用 OpenCV 做影像處理

今日環境

- 硬體：Raspberry Pi 3
- 作業系統：2016-09-23-raspbian-jessie.img

- 為了可以使用USB轉TTL傳輸線
  - 修改/boot/config.txt，新增三行
    - dtoverlay=pi3-miniuart-bt
    - core_freq=250
    - enable_uart=1

```
55 # Enable audio (loads snd_bcm
56 dtparam=audio=on
57 dtoverlay=pi3-miniuart-bt
58 core_freq=250
59 enable_uart=1
```

  - 修改/boot/cmdline.txt，將quiet splash的quiet移除

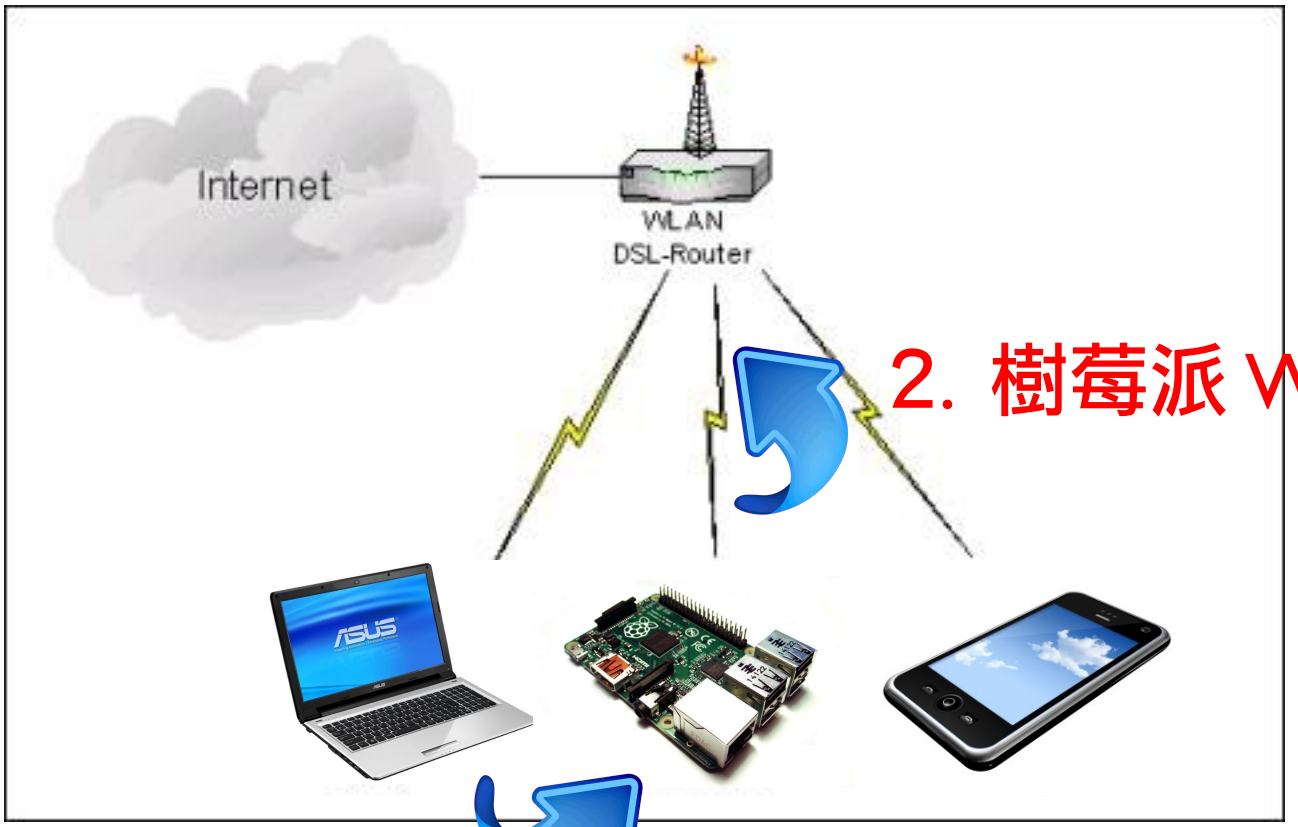
```
1 dwc_otg.lpm_enable=0 console=serial0,115200
  console=tty1 root=/dev/mmcblk0p2 rootfstype=
  ext4 elevator=deadline fsck.repair=yes rootw
  ait quiet splash plymouth.ignore-serial-cons
  oles quiet init=/usr/lib/raspi-config/init_r
  esize.sh
```

刪除 quiet

沒有螢幕與鍵盤如何使用樹莓派？

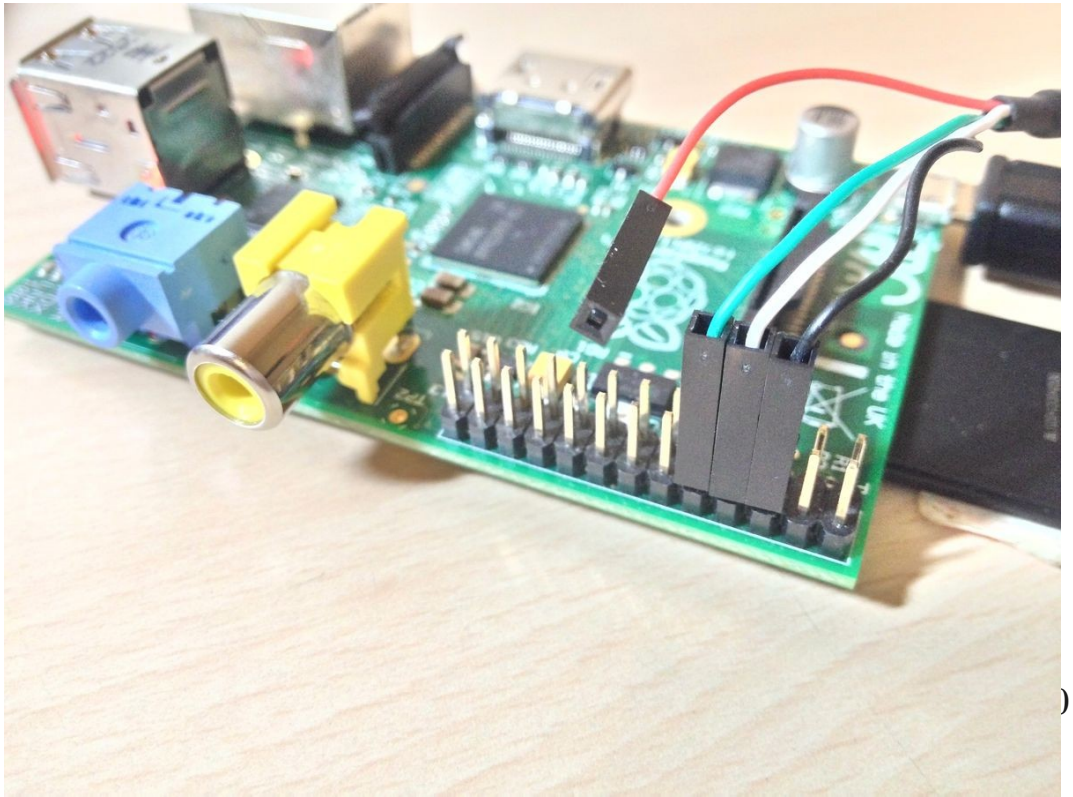
環境設定： Serial + WiFi

用 Serial 來設定 WiFi

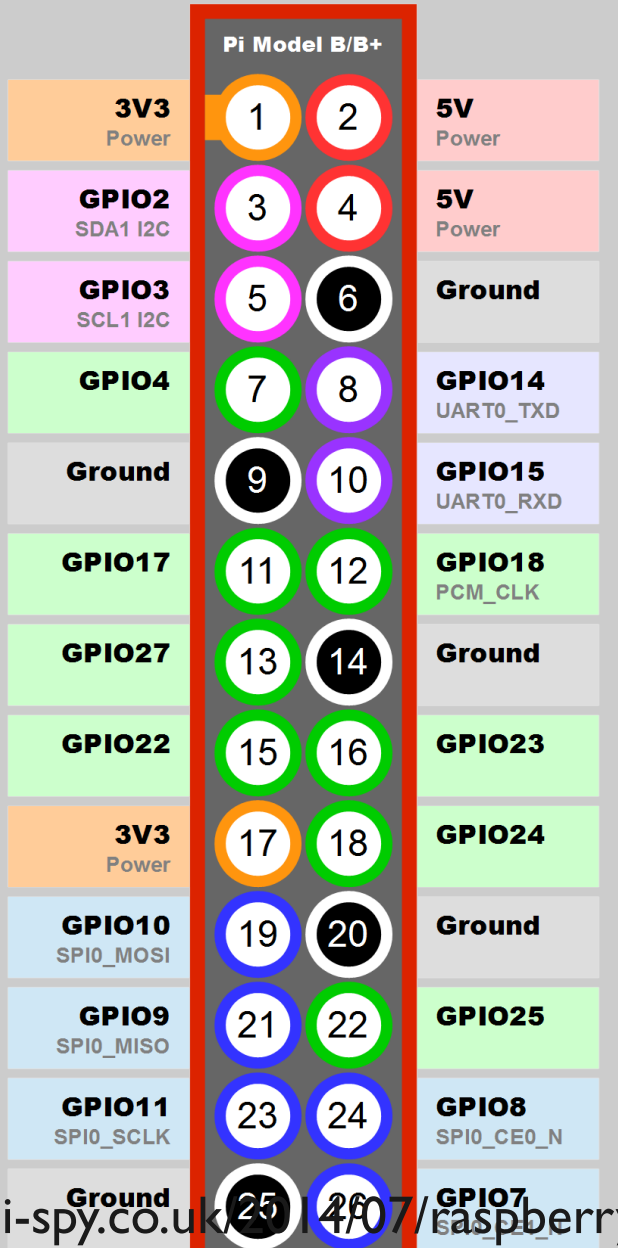


Serial 連線方式

- 以 USB 轉 TTL 傳輸線和 Pi 相連
- 接線方式
  - 黑色 / 白色 / 綠色照圖接
  - 紅色不接



放大圖



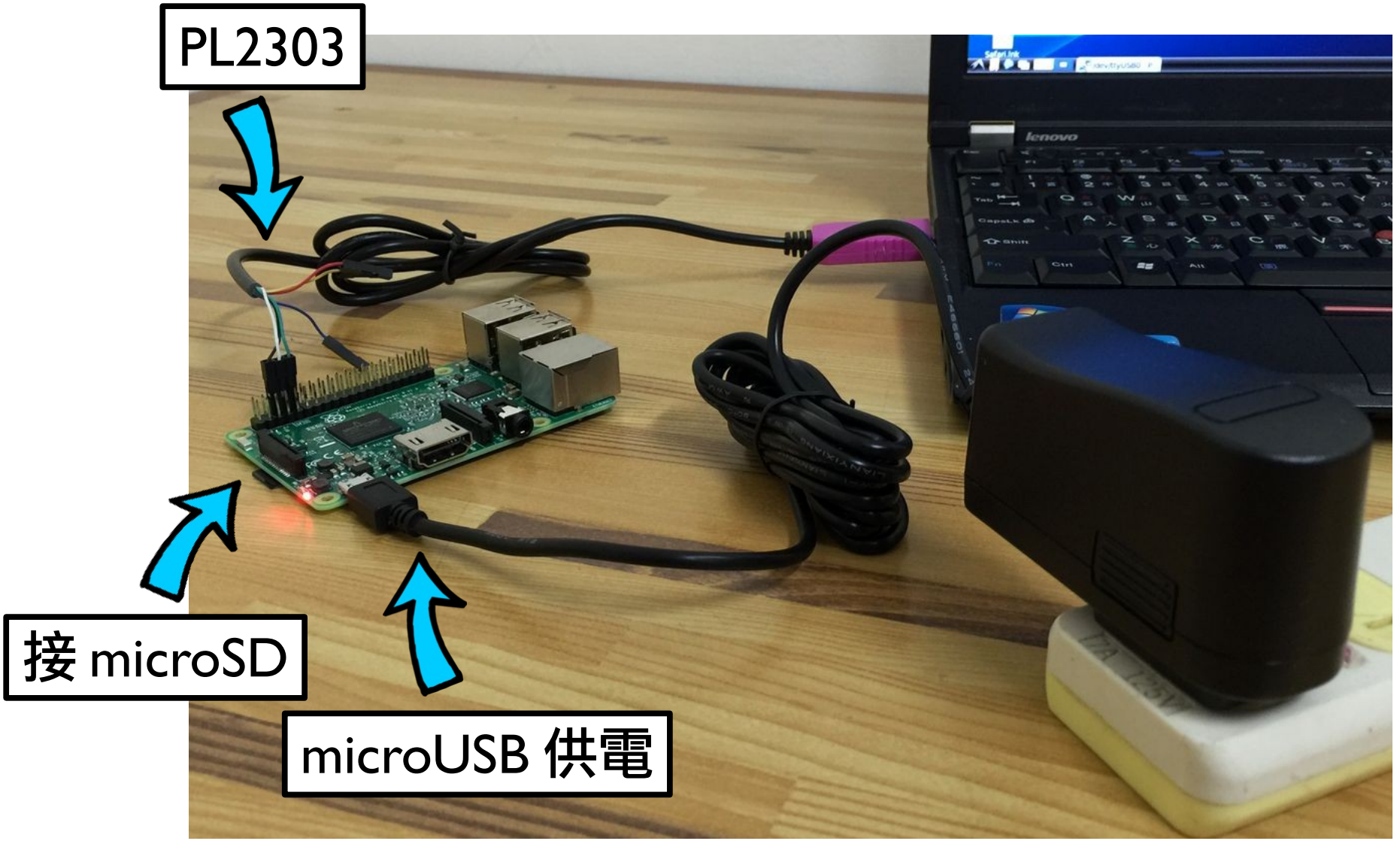
黑色線接 6 號

白色線接 8 號

綠色線接 10 號

http://www.raspberrypi-spy.co.uk/2012/07/raspberry-pi-b-gpio-header-details-and-pinout/

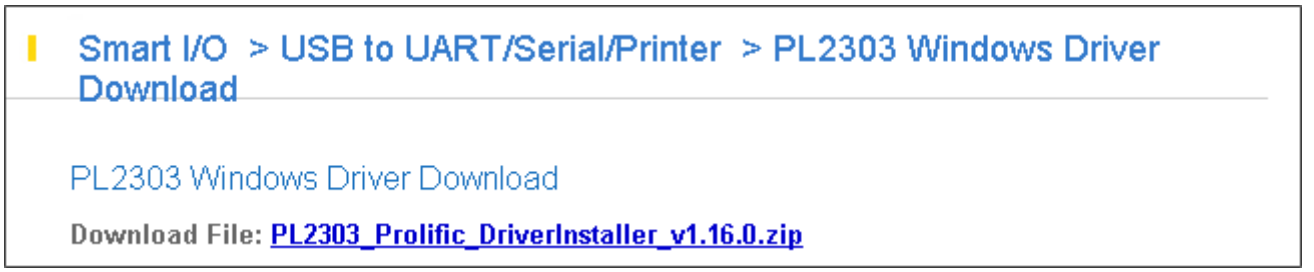
PL2303 接好後上電



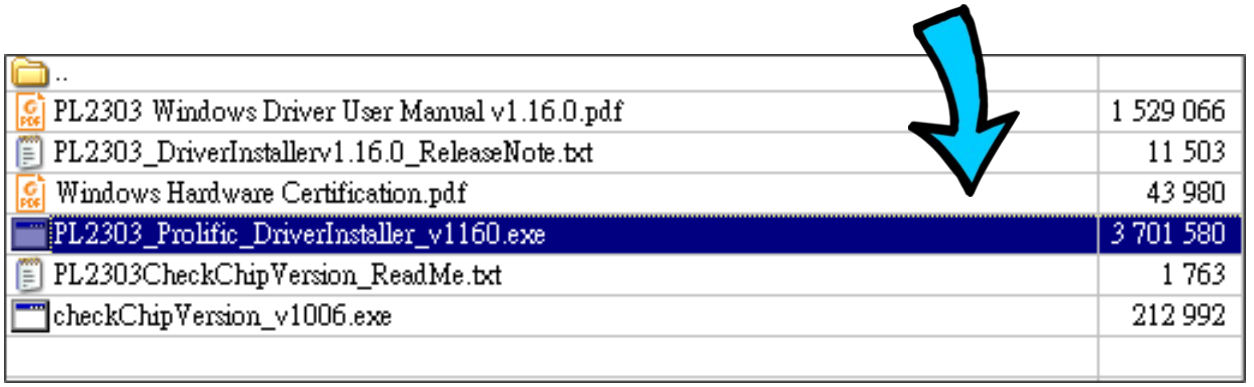
Serial Port in Windows

這是大歐

- 安裝驅動程式，http://goo.gl/QC5Q30

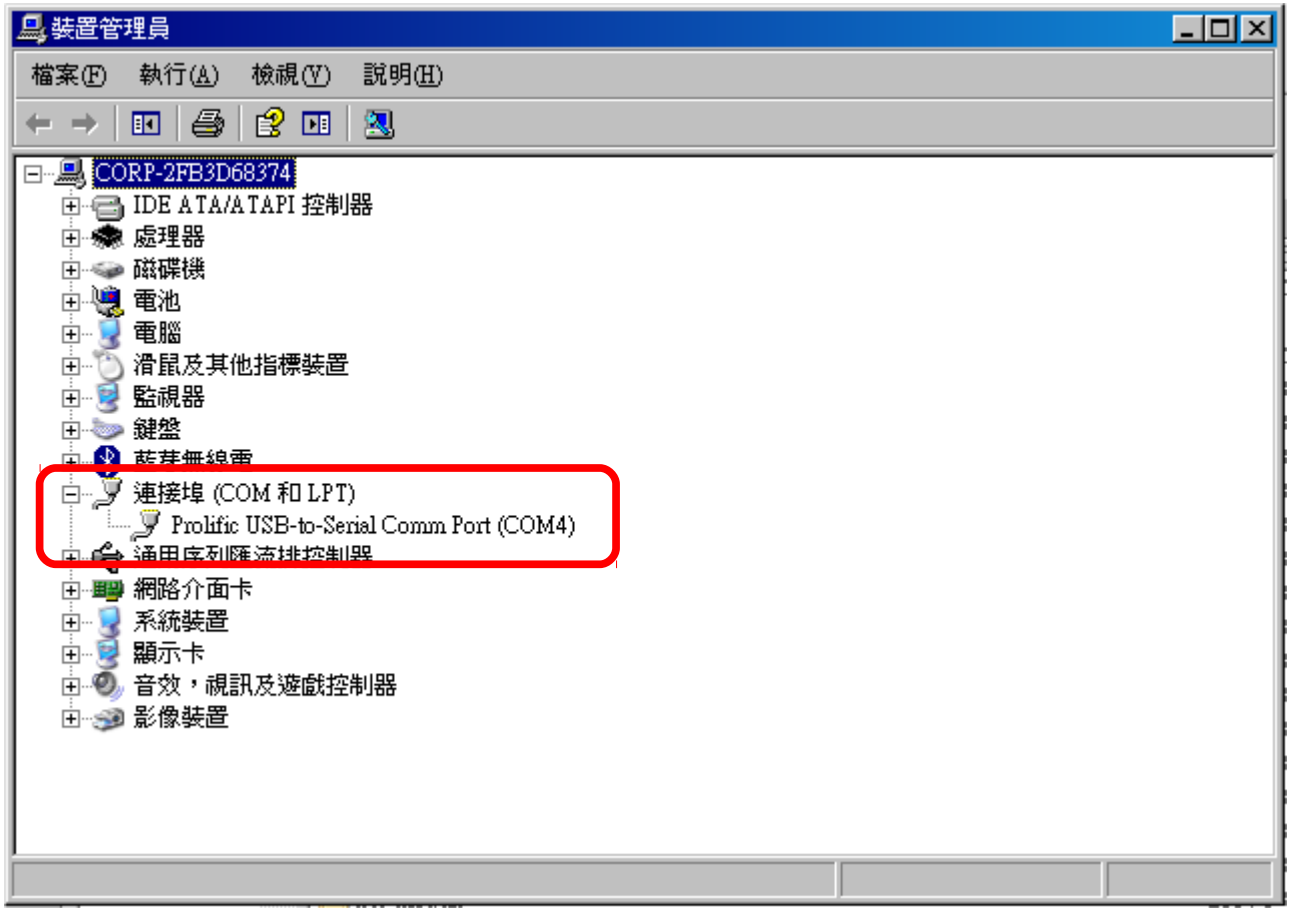


- 解加縮後，執行 DriverInstaller_v1160.exe



Serial Port in Windows

- 從裝置管理員找到 COM 的埠號（本例為 COM4）



Serial Port in Windows - 2

- 下載 putty, http://goo.gl/zdD9G9

Binaries

The latest release version (beta 0.67)

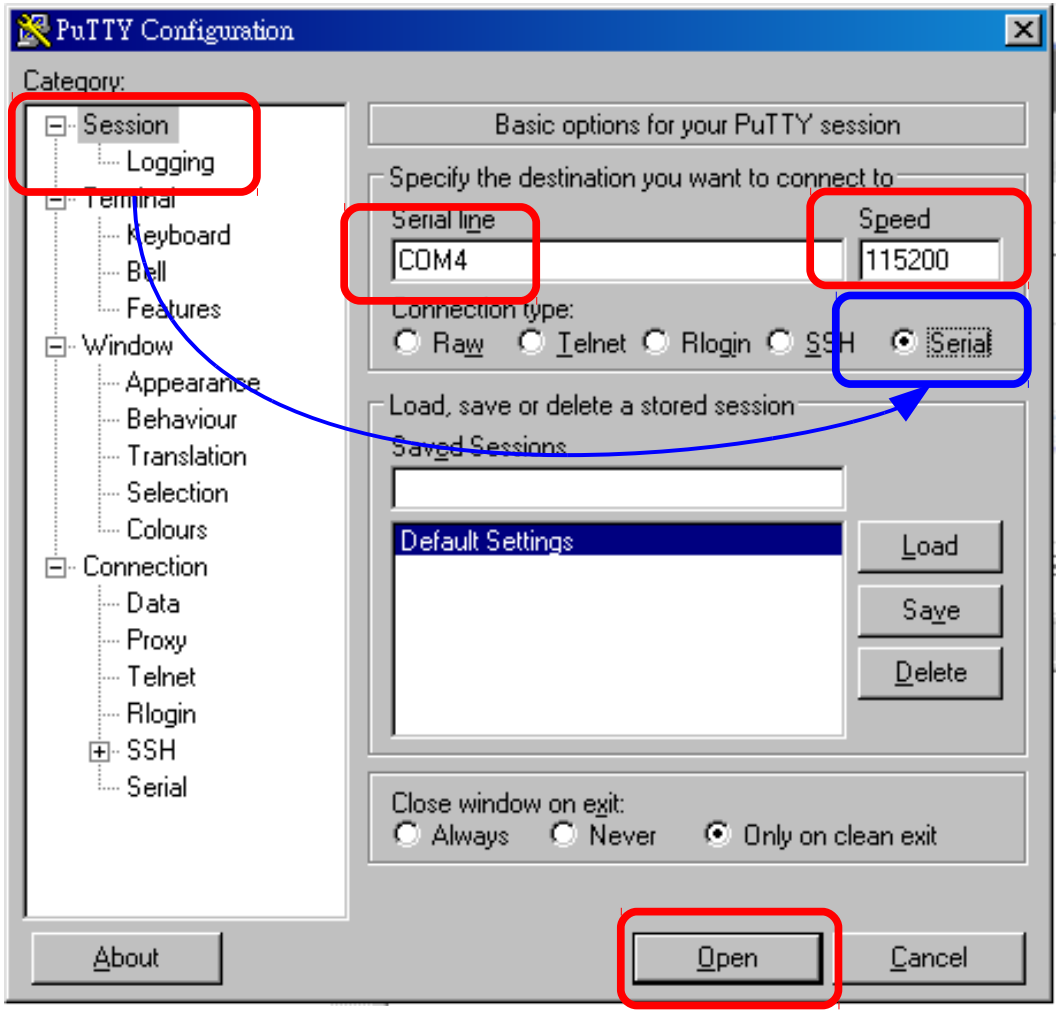
This will generally be a version we think is reasonably likely to work well. If you have a proble
already fixed the bug, before reporting it.

For Windows on Intel x86

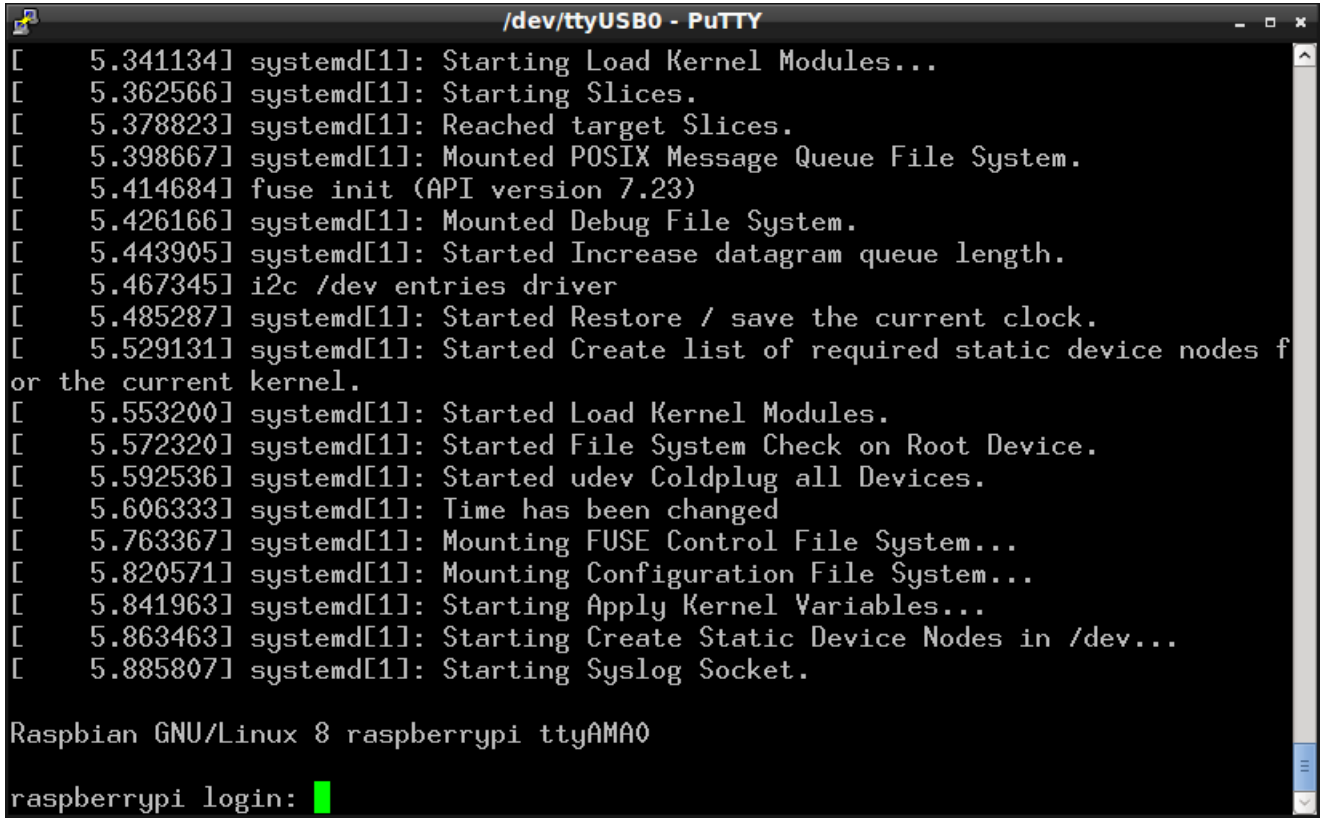
PuTTY:	putty.exe	(or by FTP)	(signature)
PuTTYtel:	puttytel.exe	(or by FTP)	(signature)
PSCP:	pscp.exe	(or by FTP)	(signature)
PSFTP:	psftp.exe	(or by FTP)	(signature)
Plink:	plink.exe	(or by FTP)	(signature)
Pageant:	pageant.exe	(or by FTP)	(signature)
PuTTYgen:	puttygen.exe	(or by FTP)	(signature)

Serial Port in Windows - 3

- 執行 putty
  1. 選擇 Session
  2. 選擇 Serial
  3. Serial line 填 COM4
  4. Speed 填入 115200
  5. Open!
- 沒畫面，先按 ENTER
- 再不行，重插拔電源



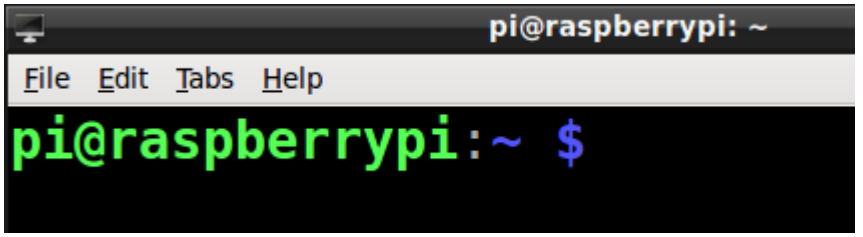
連線成功



- 預設帳號 pi，密碼 raspberry
- 沒畫面，先按 ENTER，再不行，將電源重新插拔
- 如果出現亂碼，確定 speed 為 115200

符號說明

- 登入畫面



- pi 是登入的使用者
- @ 表示”在”
- raspberry 是主機名稱
- ~ 表示在 ' 家目錄 '(home directory)
- $ 表示使用者所使用的 shell(一種文字工具介面)

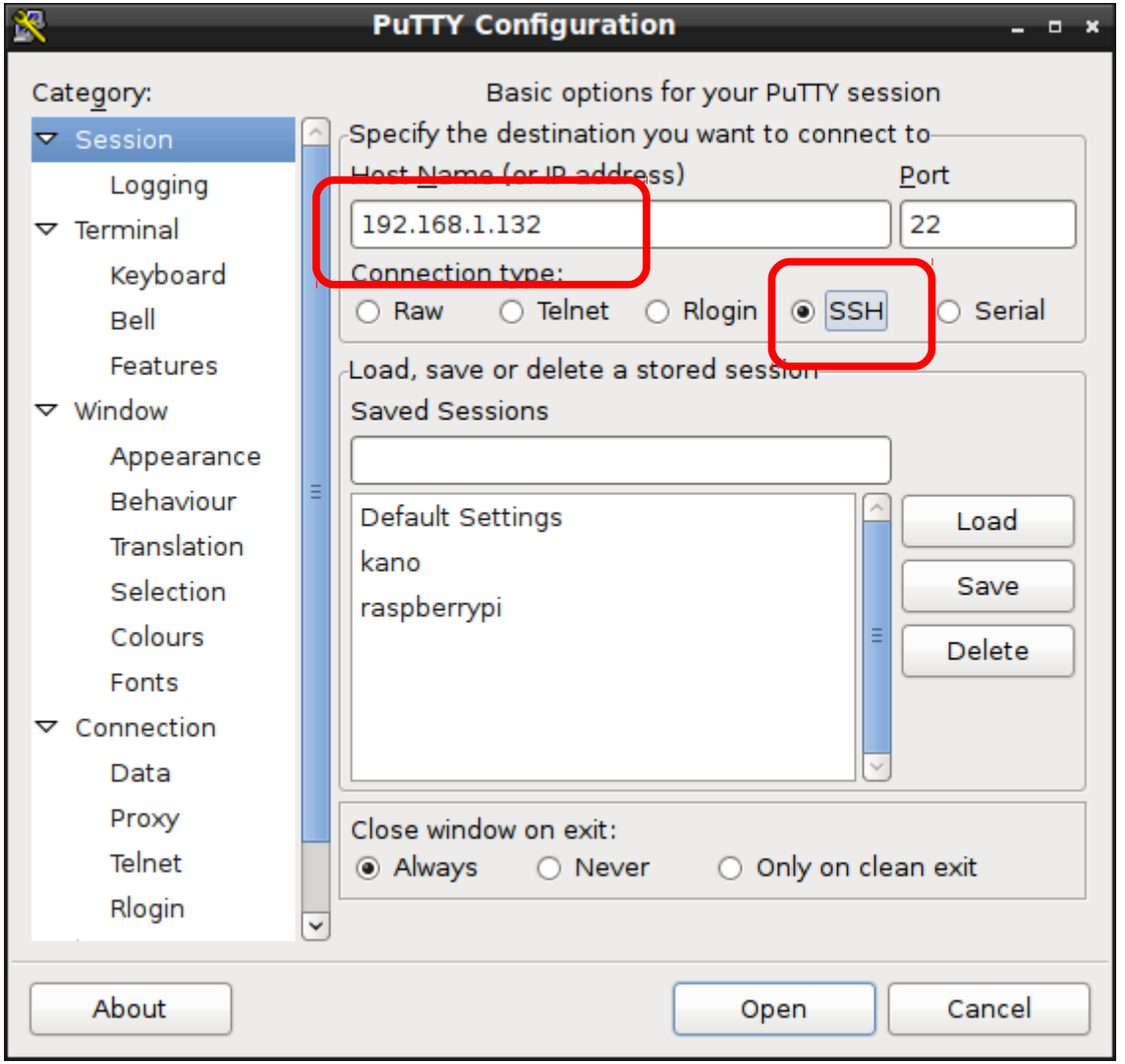
如果連線 WiFi 成功，查詢 IP

```
$ ifconfig wlan0
```

```
pi@raspberrypi:~ $ ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr b8:27:eb:36:31:04
          inet addr:192.168.1.132  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::feb1:a80a:20c8:cc76/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:671 errors:0 dropped:561 overruns:0 frame:0
          TX packets:48 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:176563 (172.4 KiB)  TX bytes:8592 (8.3 KiB)
```

- IP = 192.168.1.132（每個人不同）

就可以使用 SSH 連線 (要同網段)



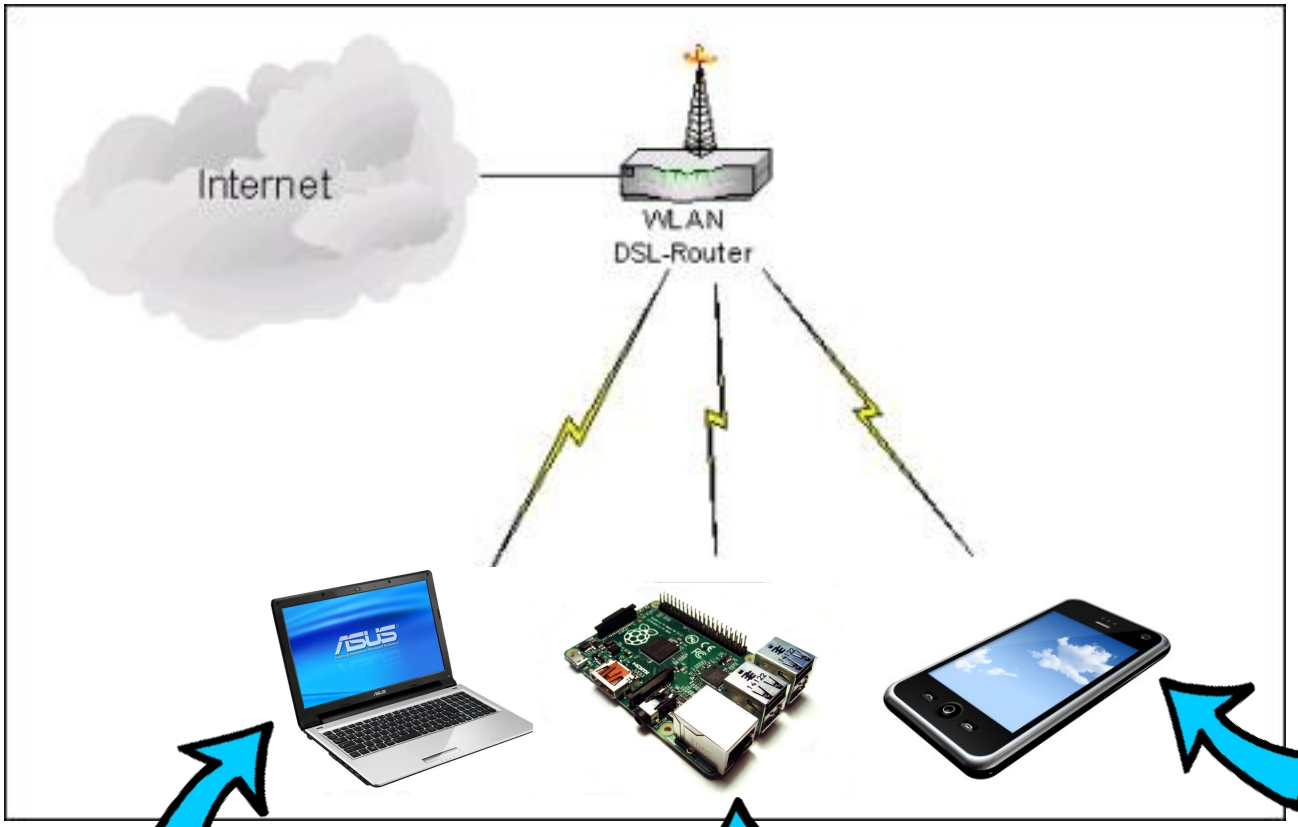
我的 Pi 連到那台 AP ？

```
$ iwconfig wlan0
```

```
pi@raspberrypi:~ $ iwconfig wlan0
wlan0     IEEE 802.11bgn  ESSID:"sosorry2.4G"
          Mode:Managed  Frequency:2.462 GHz  Access Point: 74:DA:38:31:32:5C
          Bit Rate=24 Mb/s   Tx-Power=31 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Power Management:on
          Link Quality=70/70  Signal level=-35 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

- SSID = sosorry2.4G（每個人不同）

同網段的意思



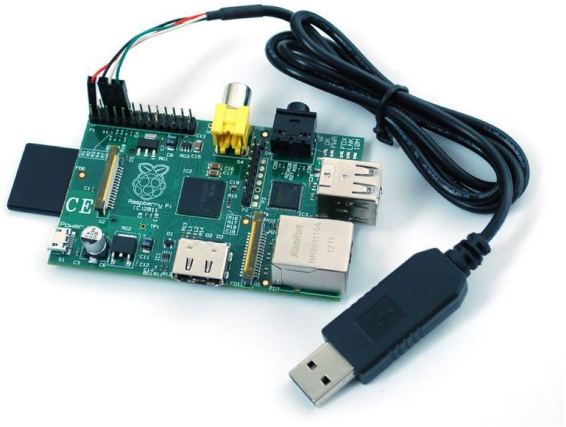
連到 sosorry2.4G

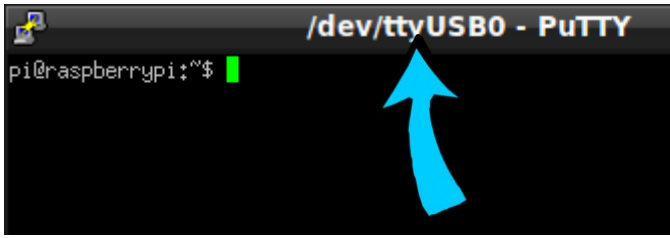
連到 sosorry2.4G

連到 sosorry2.4G

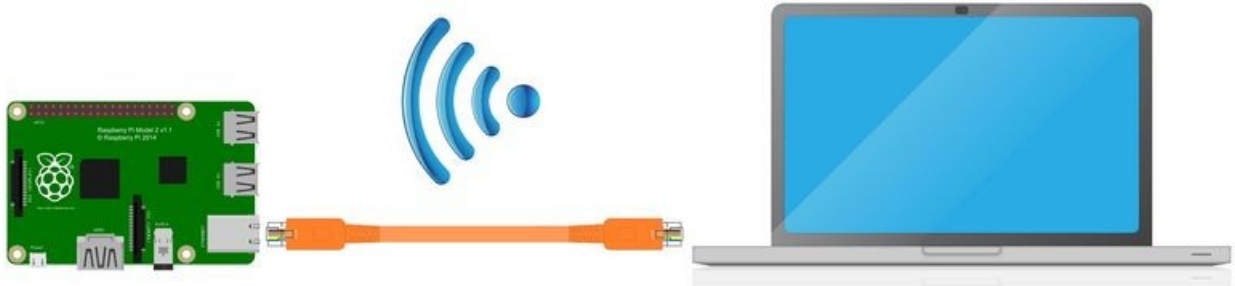
Serial 連線和 SSH 連線有什麼不同？

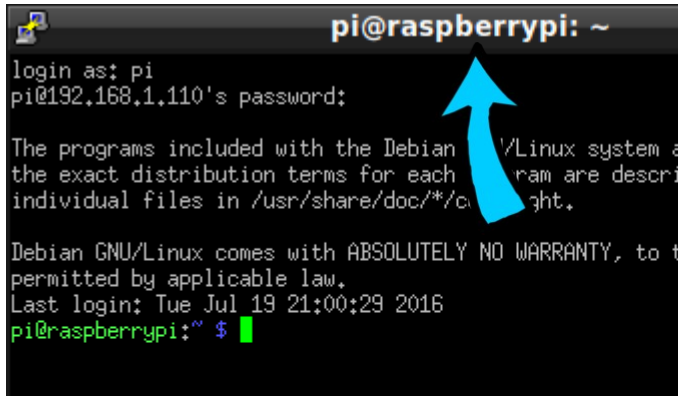
- Serial 以實體線路相連，純文字，是獨占式的連線





- SSH 是 TCP/IP 通訊協定，透過 Ethernet 或 WiFi 連線



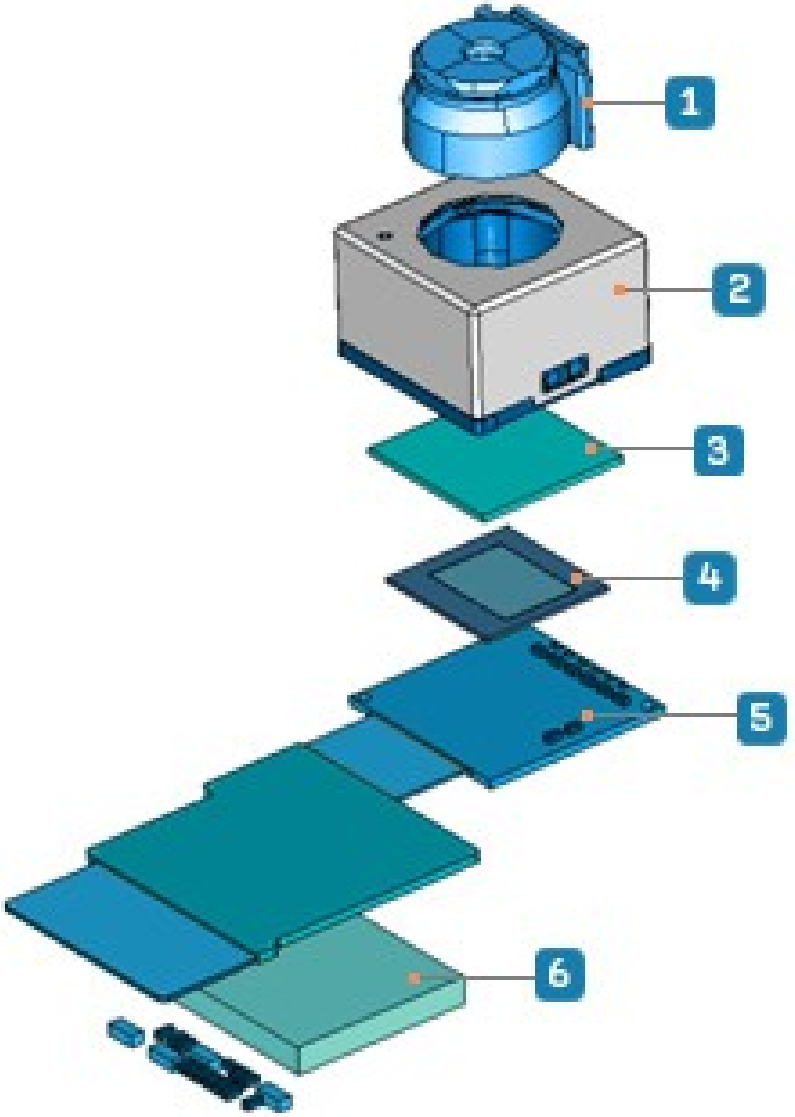


安裝今日所需軟體 (已安裝)

- `$ sudo apt-get update`
- `$ sudo apt-get install -y festival python-opencv python-pip`
- `$ sudo pip install request`

Raspberry Pi Camera 簡介

從手機相機模組講起



1. Lens(透鏡)

2.VCM(音圈馬達)

3. IR-Cut(紅外光濾片)

4. Sensor(感光元件)

5. PCB(印刷電路板)

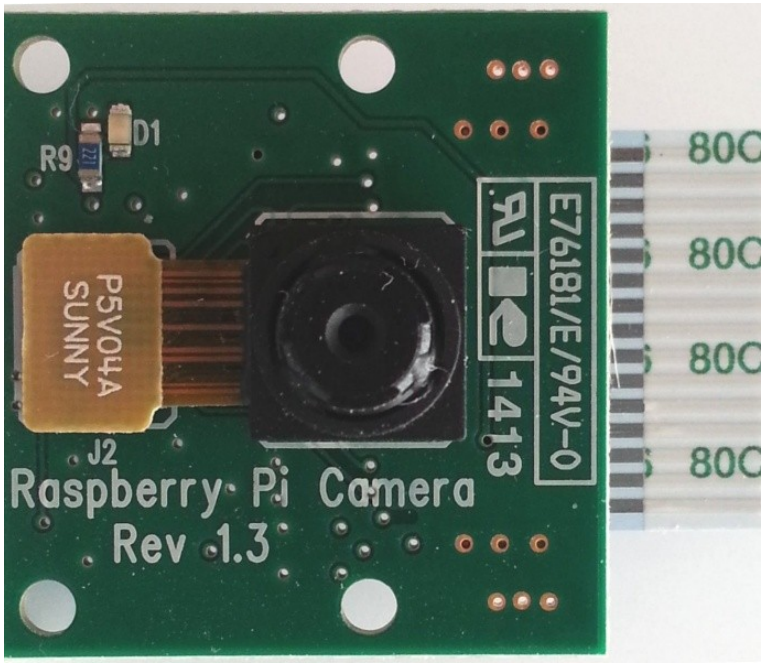
6. ISP(影像訊號處理器)

http://www.phonearena.com/news/13MP-camera-tipped-for-Samsung-Galaxy-S-IV_id35168

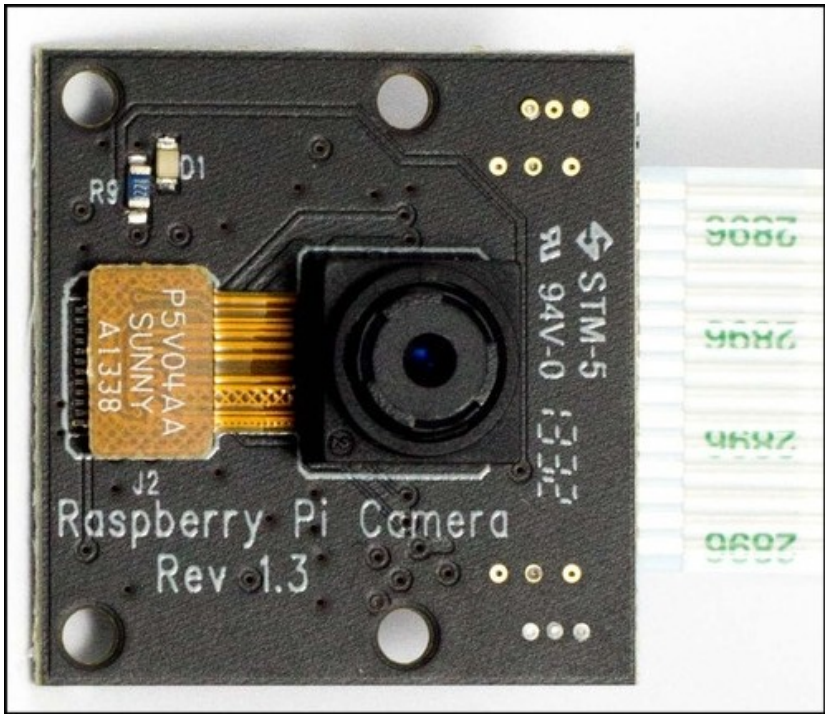
技術規格 (v1)

- Sensor:OmniVision OV5647 (5MP)
- 靜態拍照最高解析度:2592 x 1944 pixel
- Pixel Size:1.4 x 1.4 μm
- Lens:f=3.6 mm, f/2.9
- Angle of View:54 x 41 degrees
- Field of View:2.0 x 1.33 m at 2 m
- Fixed Focus:1m to infinity
- 動態攝影最高解析度:1080p@30 FPS with H.264/AVC

Type of Raspberry Pi Camera



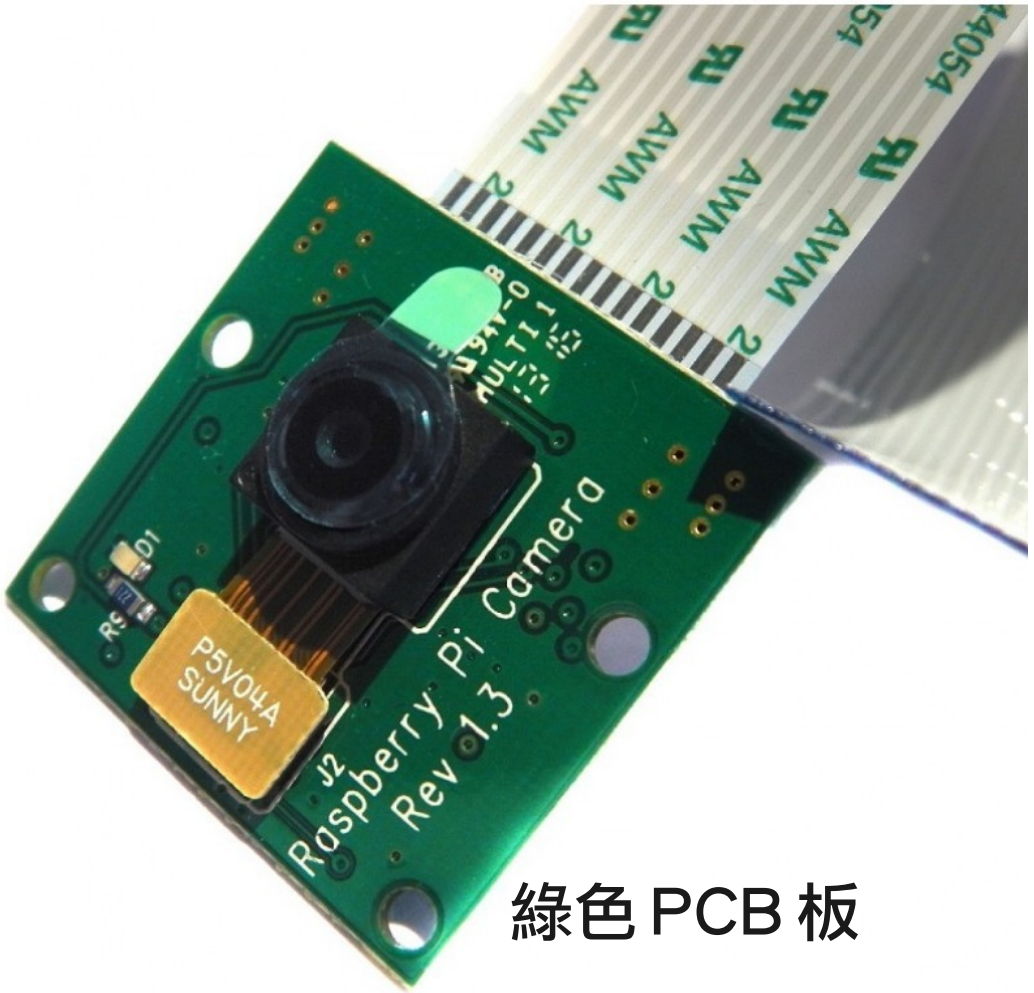
Raspberry Pi Camera Module



NoIR Camera Module

http://elinux.org/Rpi_Camera_Module

Raspberry Pi Camera Module(v1)



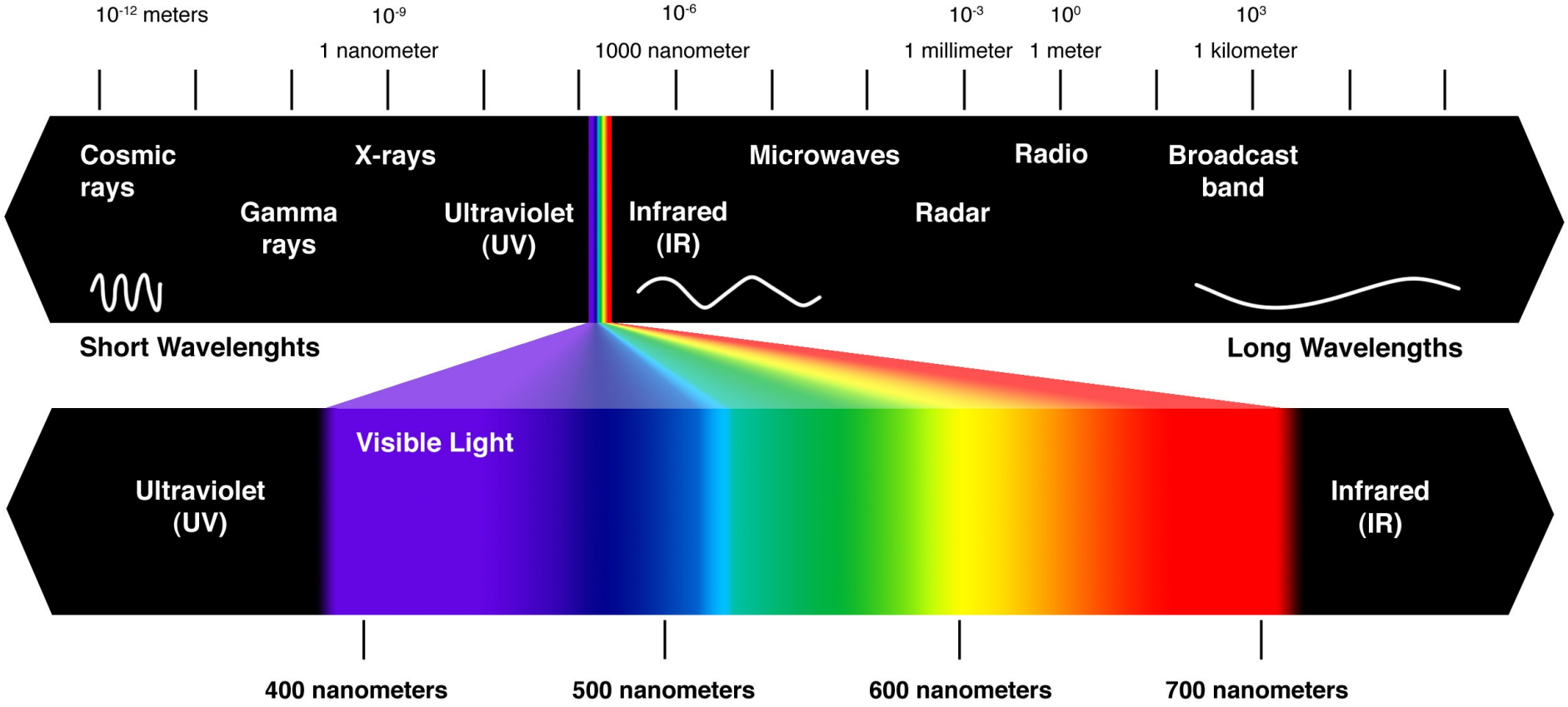
15-Pins, CSI 介面

綠色 PCB 板

https://www.modmypi.com

基礎光學原理

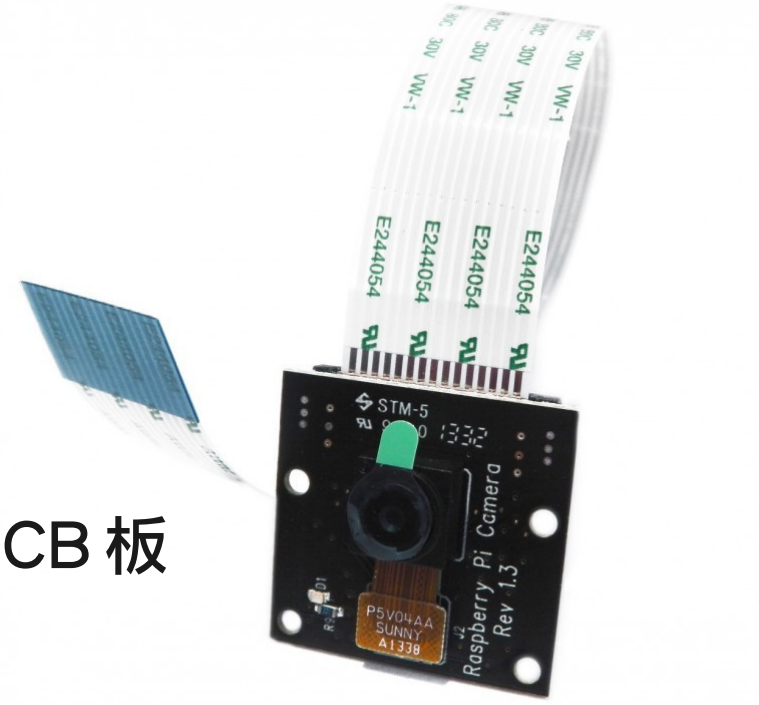
- 問：樹葉為什麼看起來是綠色的？
- 答：因為樹葉吸收了大部分可見光，只反射綠色光



https://www.raspberrypi.org/learning/infrared-bird-box/worksheet/

No IR Camera

- No IR = No 'IR cut filter' installed
- 因此 CMOS 可吸收到不可見光 (Infrared)
- No IR 相機 + 紅外線發光源 = 夜視相機



黑色 PCB 板

https://www.buyapi.ca/product/raspberry-pi-noir-camera-module-v2-8mp/

兩種相機效果比較

1. 非 NoIR 相機

2. NoIR 相機

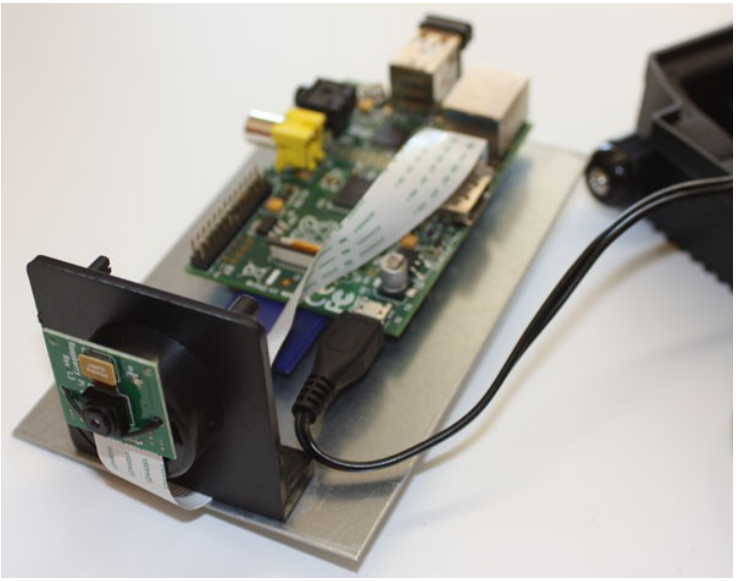
3. NoIR 相機

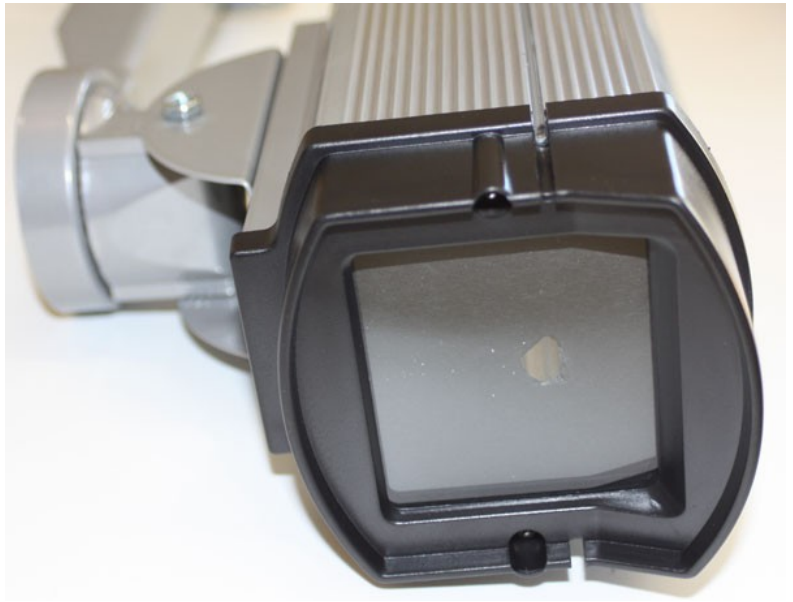
4. NoIR 相機 + 藍色濾光片

https://www.raspberrypi.org/magpi/issues/18/

Raspberry Pi Camera 應用介紹

IP Camera

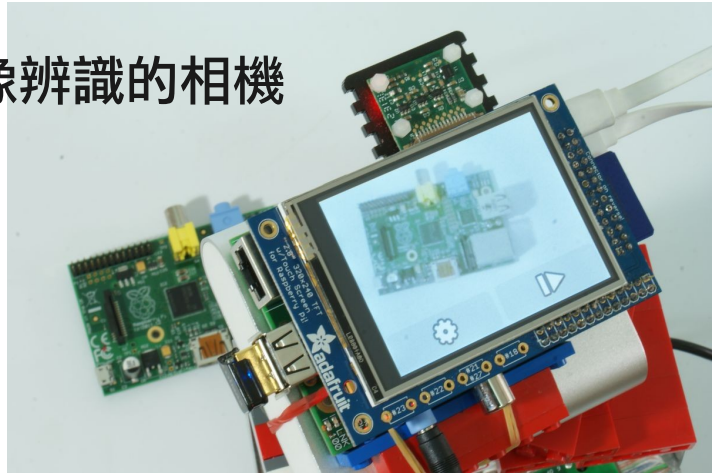


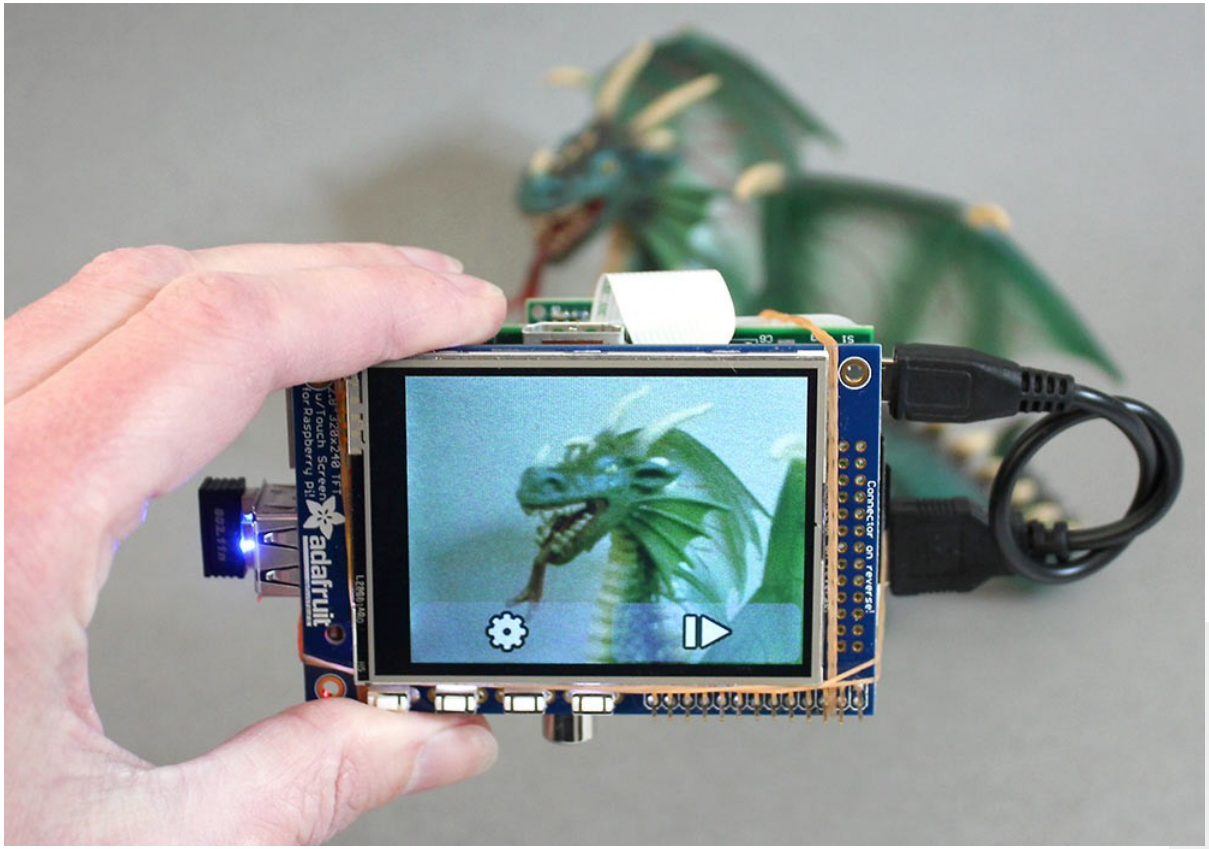


http://www.codeproject.com/Articles/665518/Raspberry-Pi-as-low-cost-HD-surveillance-camera

雲端相機

可做影像辨識的相機





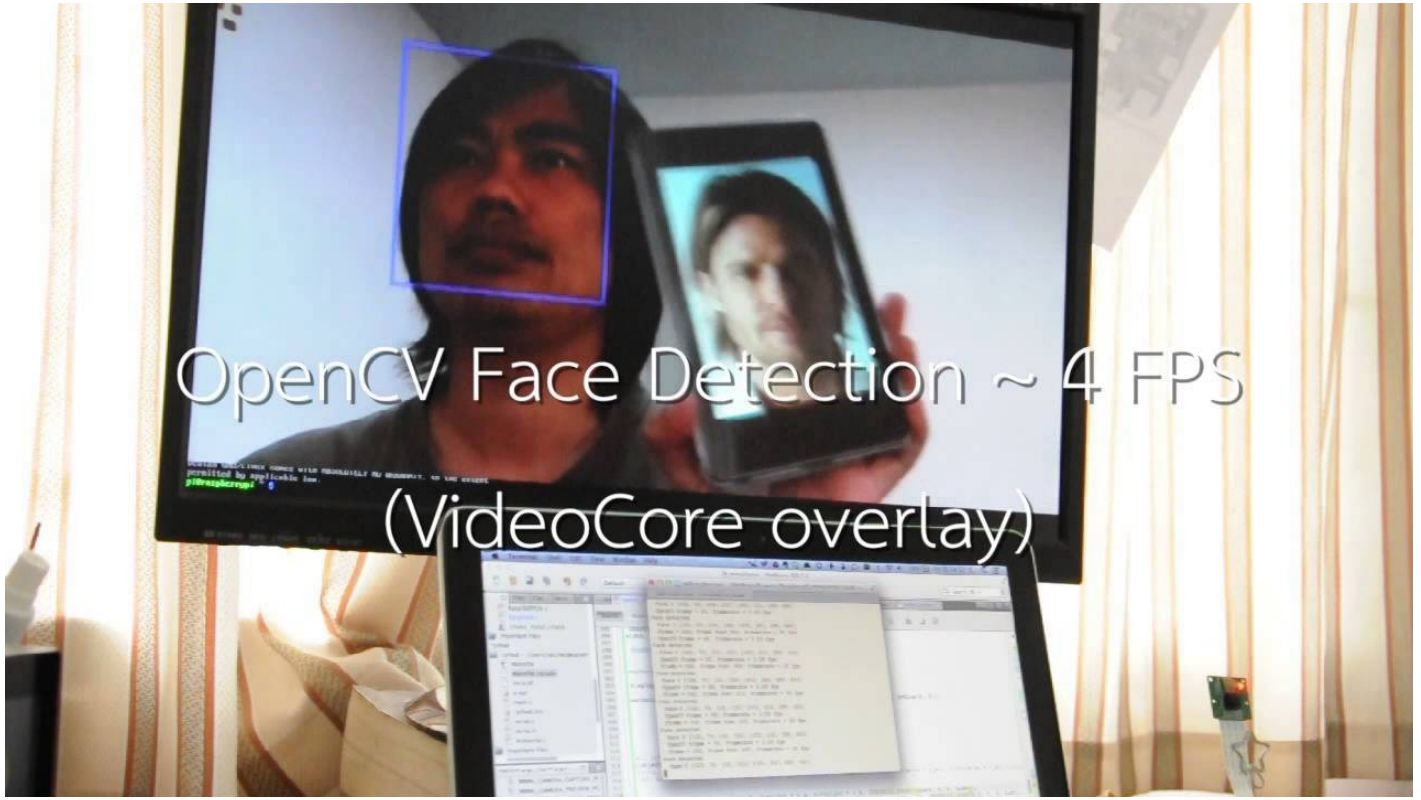
imagga

can you imaggine!

http://learn.adafruit.com/diy-wifi-raspberry-pi-touch-cam

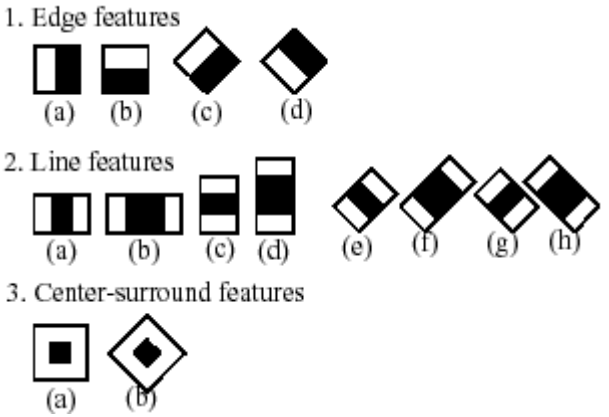
Pi 立得

人臉偵測與追蹤



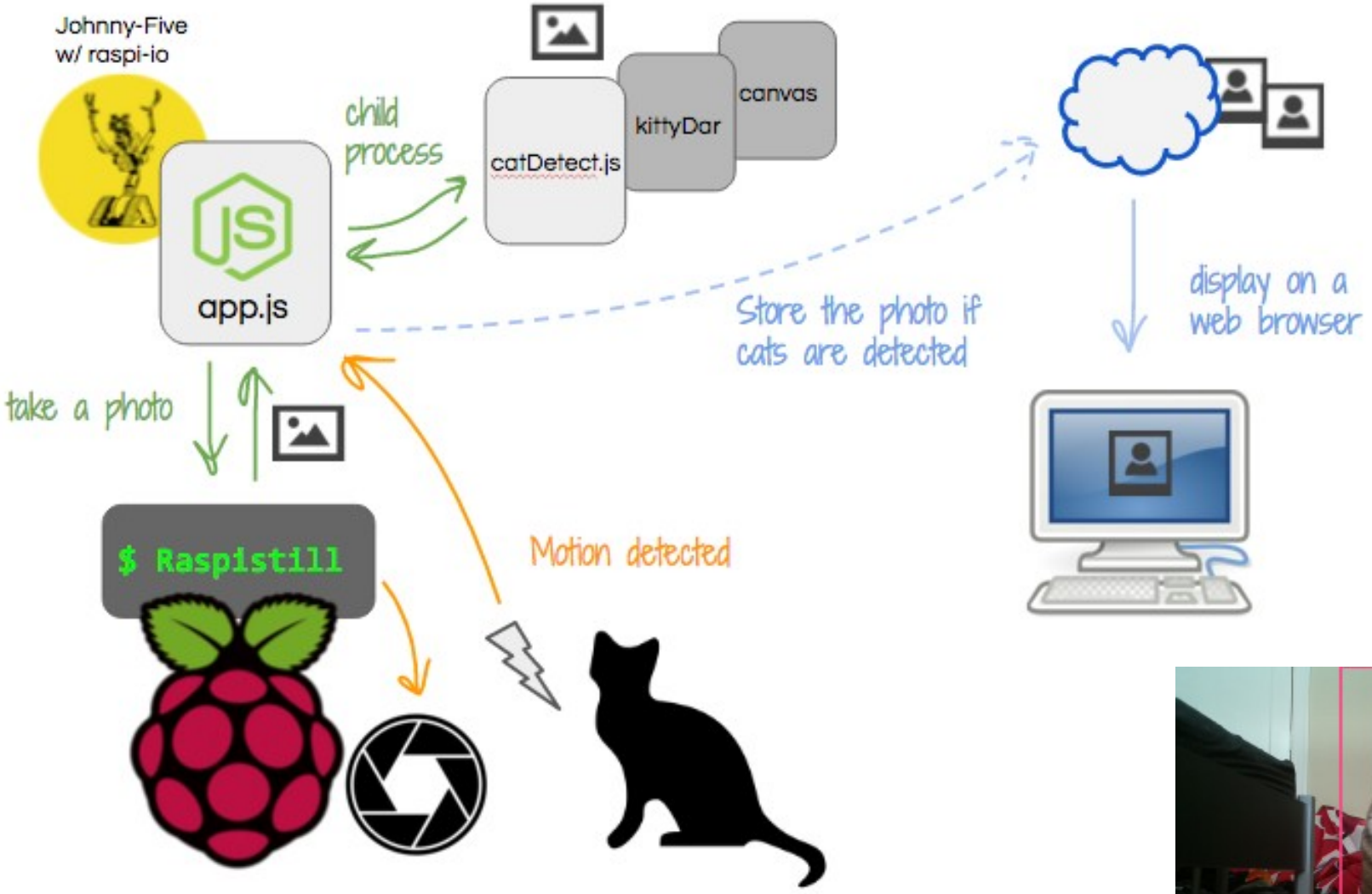
OpenCV

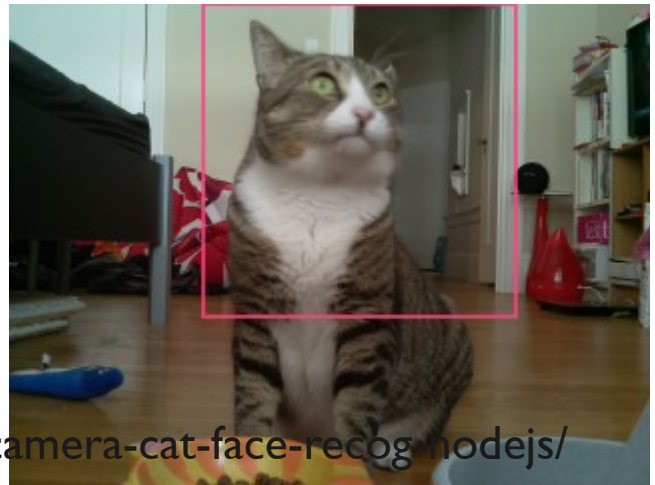
Haar	Daubechies-4 (D4)
H = {1 / sqrt(2), 1 / sqrt(2) } G = {1 / sqrt(2), -1 / sqrt(2) }	H = { (1 + sqrt(3)) / (4 * sqrt(2)), (3 + sqrt(3)) / (4 * sqrt(2)), (3 - sqrt(3)) / (4 * sqrt(2)), (1 - sqrt(3)) / (4 * sqrt(2)) } G[0] = H[3], G[1] = -H[2], G[2] = H[1], G[3] = -H[0]



https://github.com/tasanakorn/rpi-mmal-demo

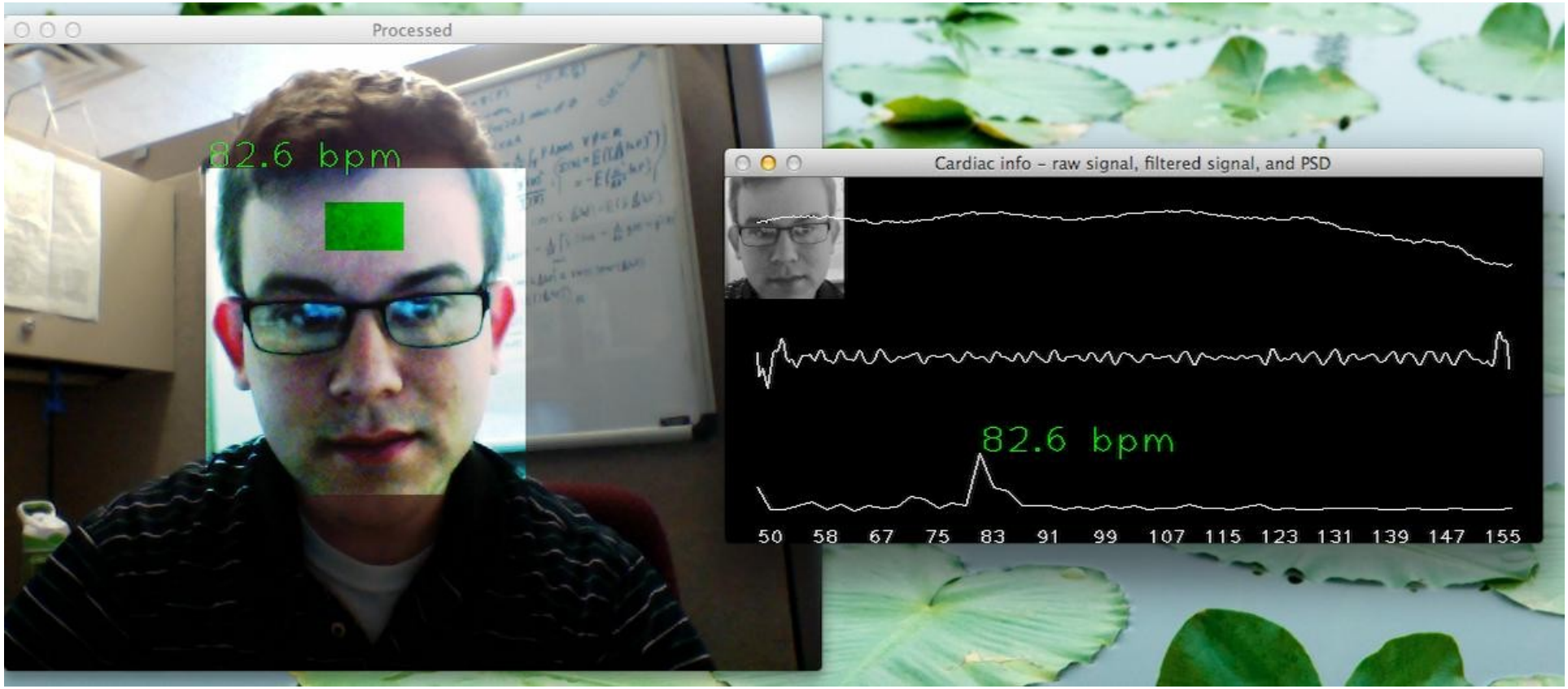
貓臉偵測





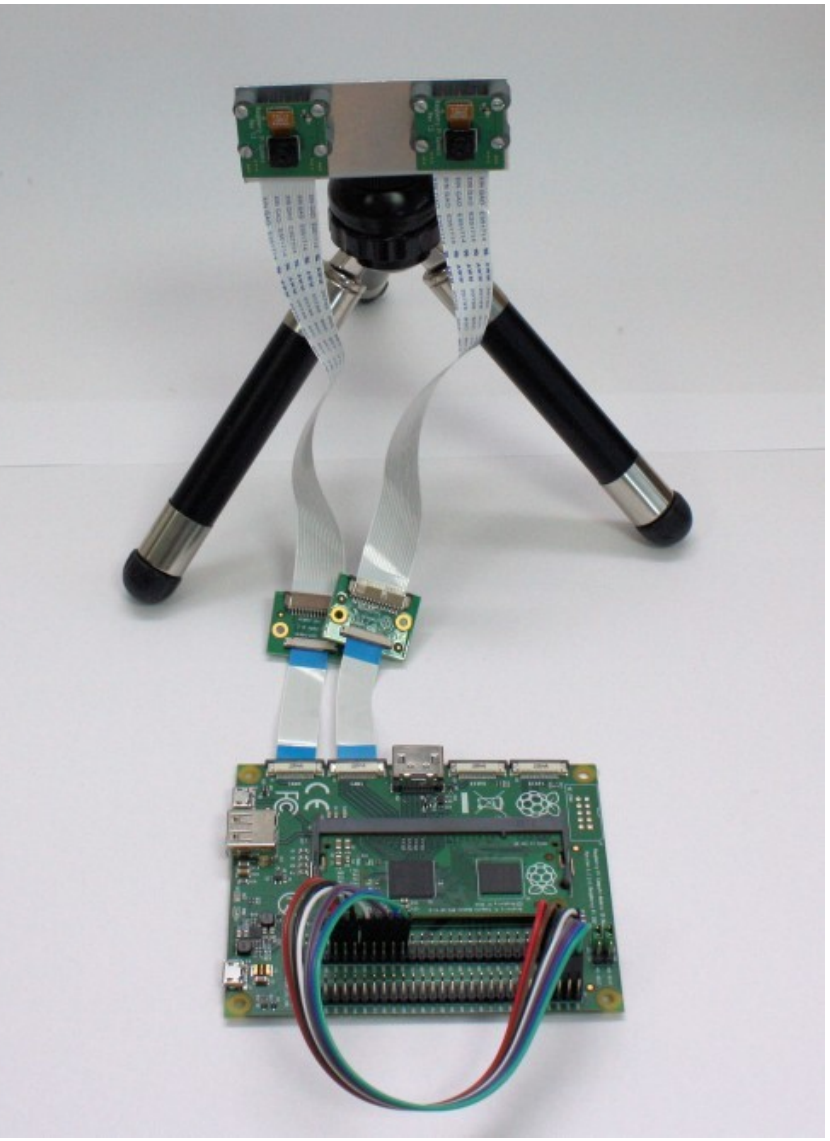
http://www.girliemac.com/blog/2015/12/25/kittycam-raspberrypi-camera-cat-face-recog-nodejs/

脈博辨識



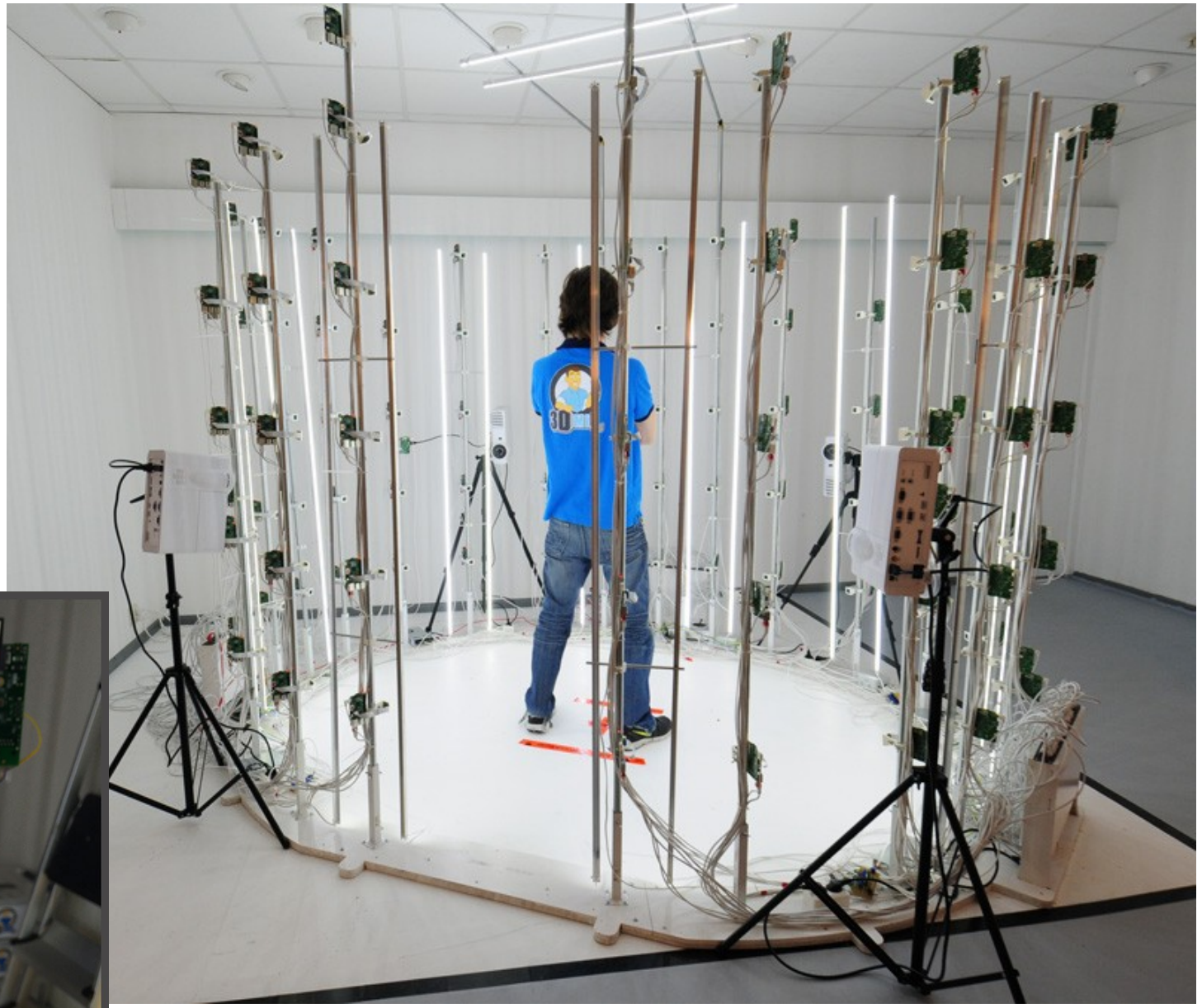
https://github.com/thearn/webcam-pulse-detector

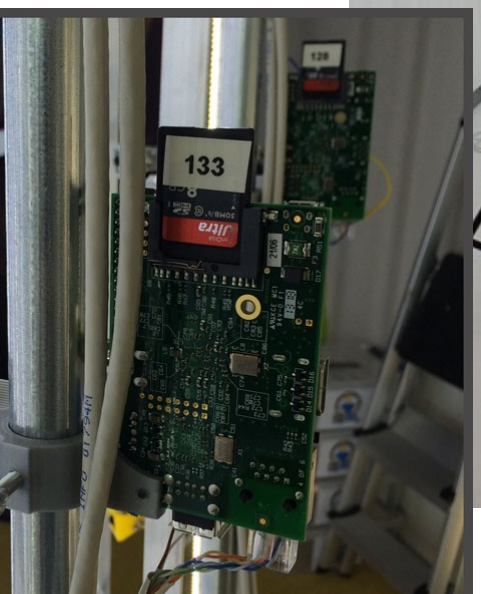
3D 建模計算深度 /2 Cameras



http://www.raspberrypi.org/real-time-depth-perception-with-the-compute-module/

3D 掃描 /50 Cameras





http://www.pi3dscan.com/

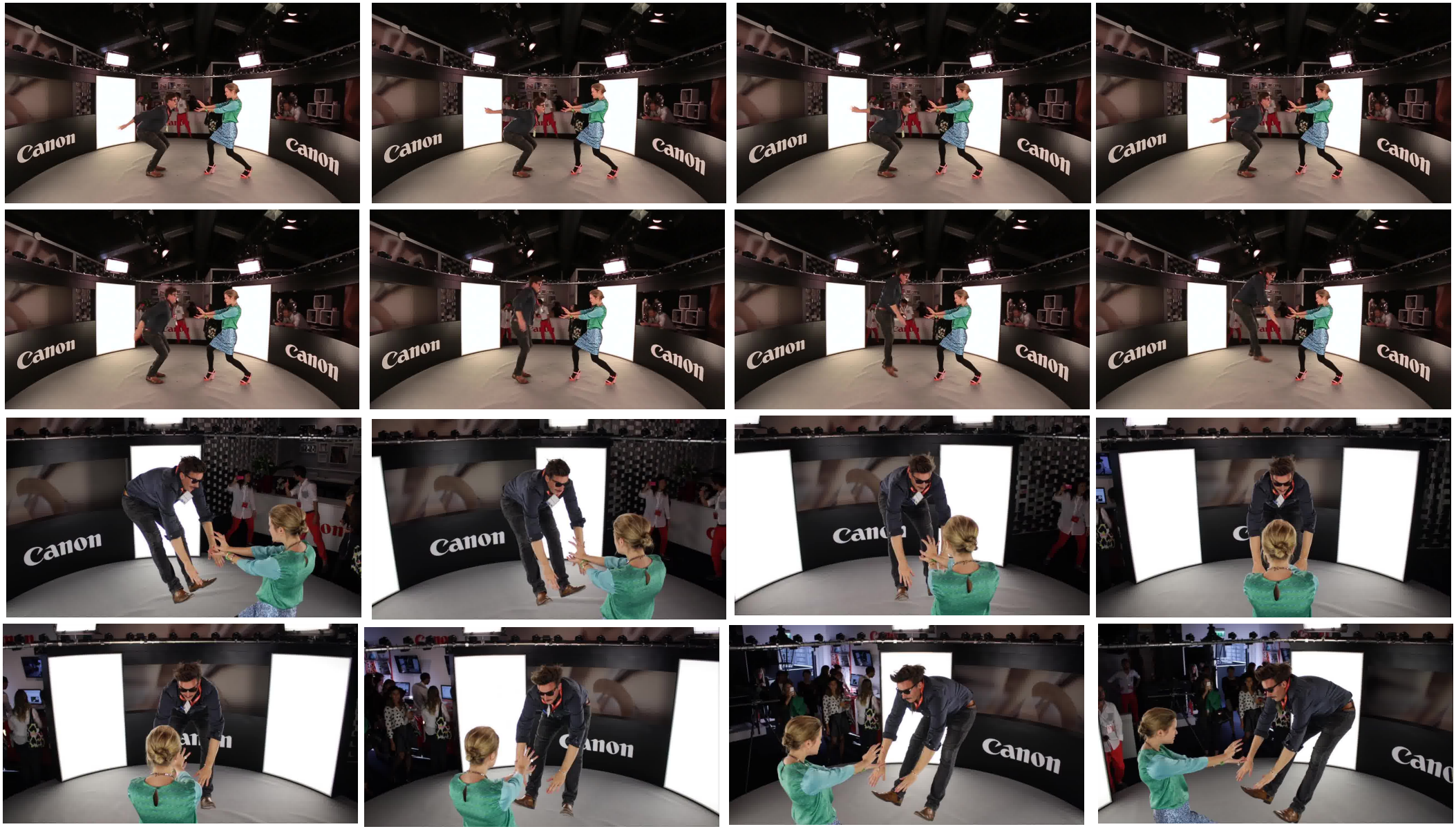
效果 + Autodesk Recap

http://www.pi3dscan.com/

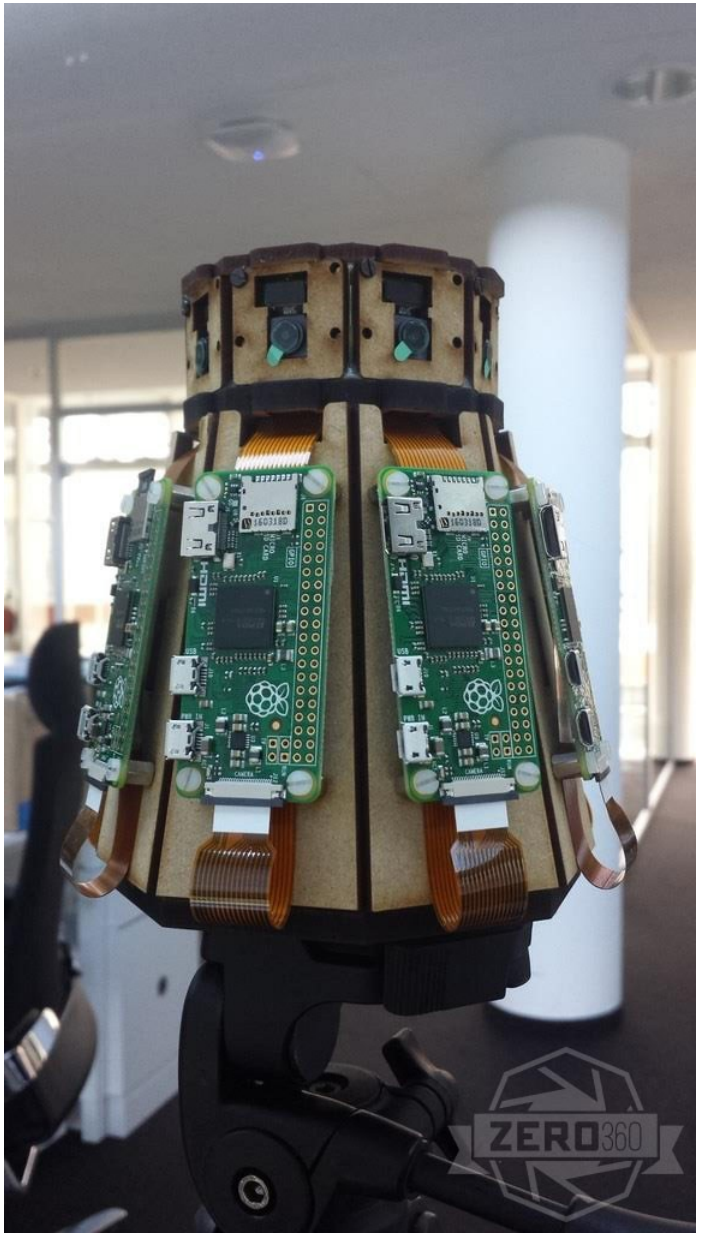
360 度照片

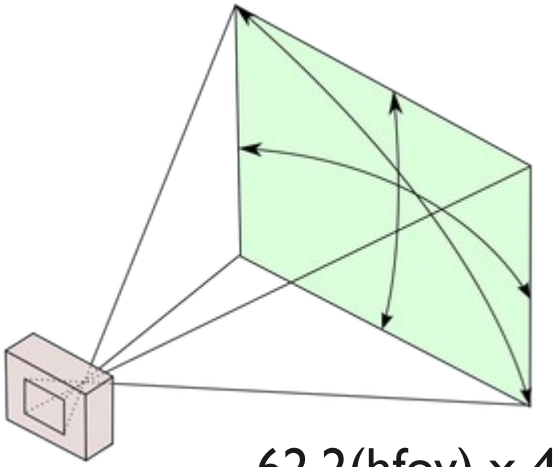
https://vimeo.com/77218985

效果

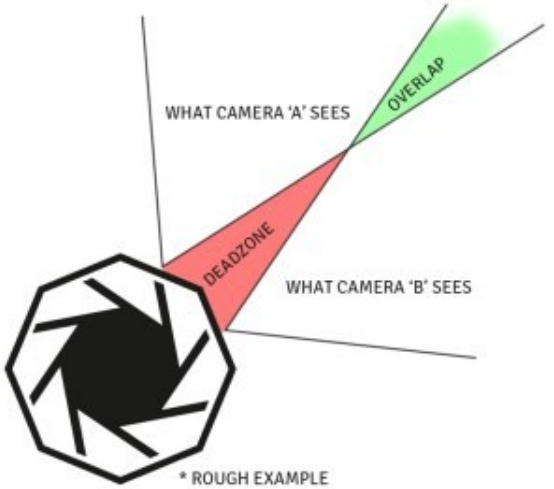


ZERO360



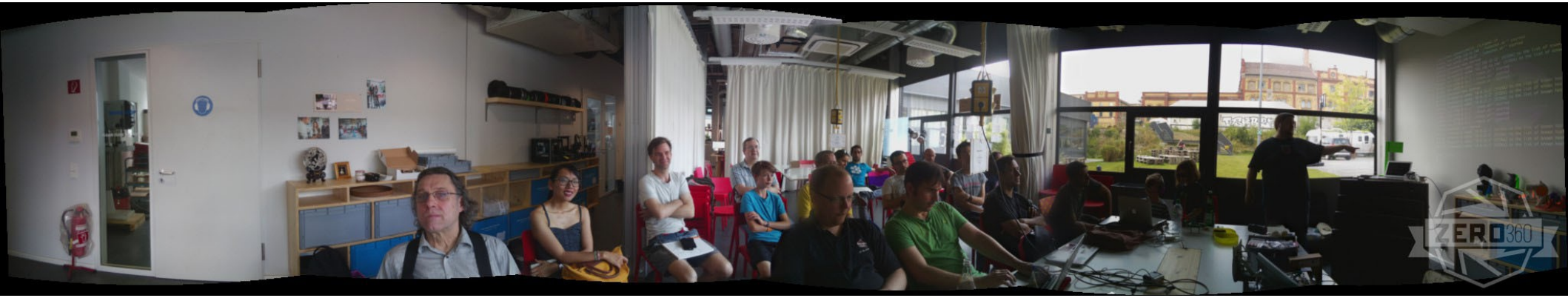


62.2(hfov) x 48.8(vfov)



http://raspberryjamberlin.de/introducing-zero360/

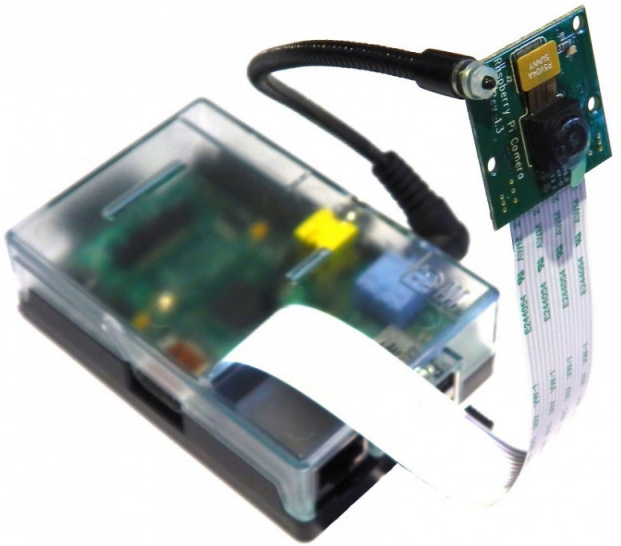
八台相機拍出另一種全景照片

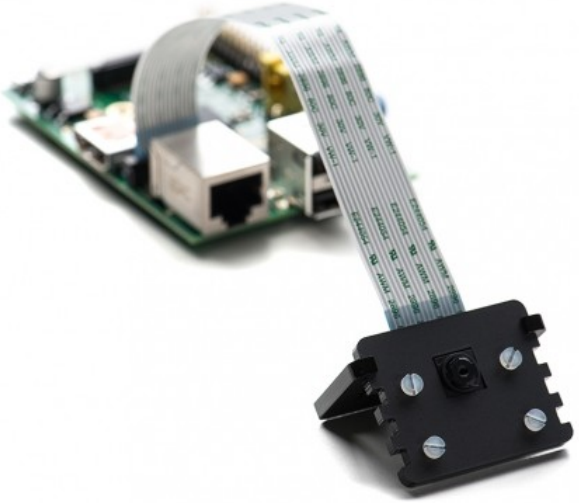


http://raspberryjamberlin.de/introducing-zero360/

Camera 改裝套件

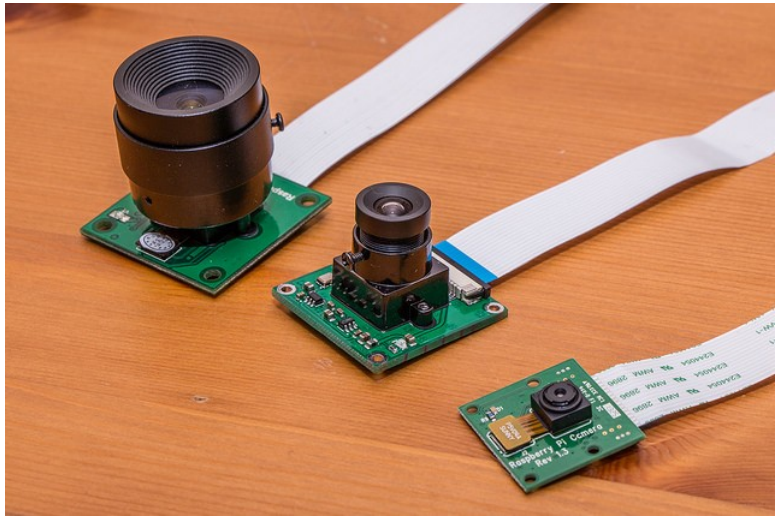
固定的機構





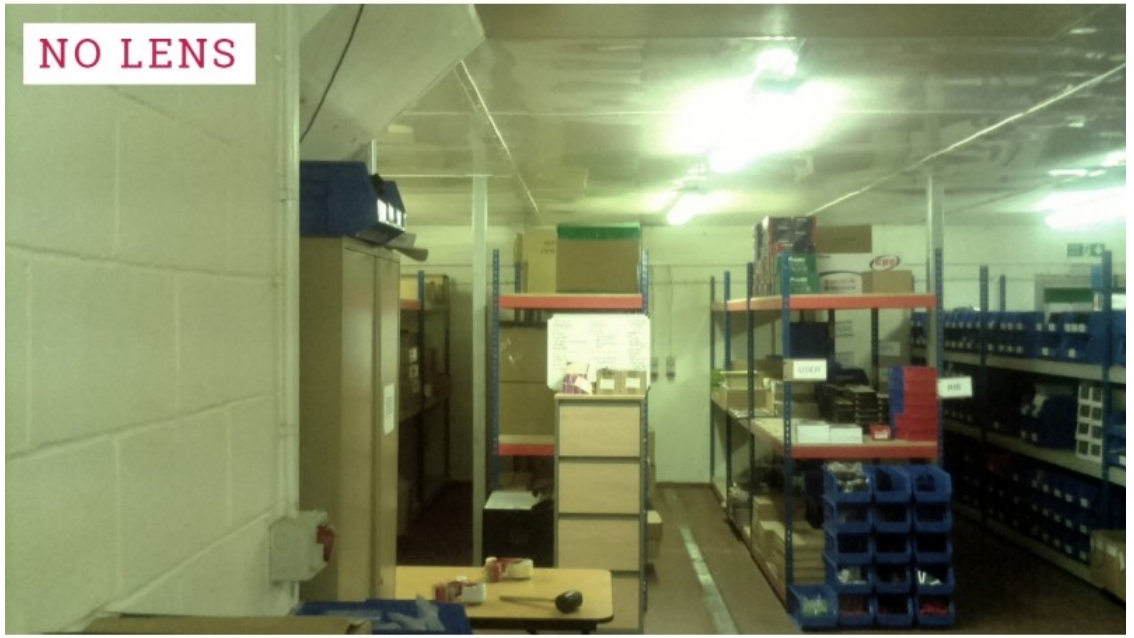
http://www.modmypi.com.

相機模組改裝



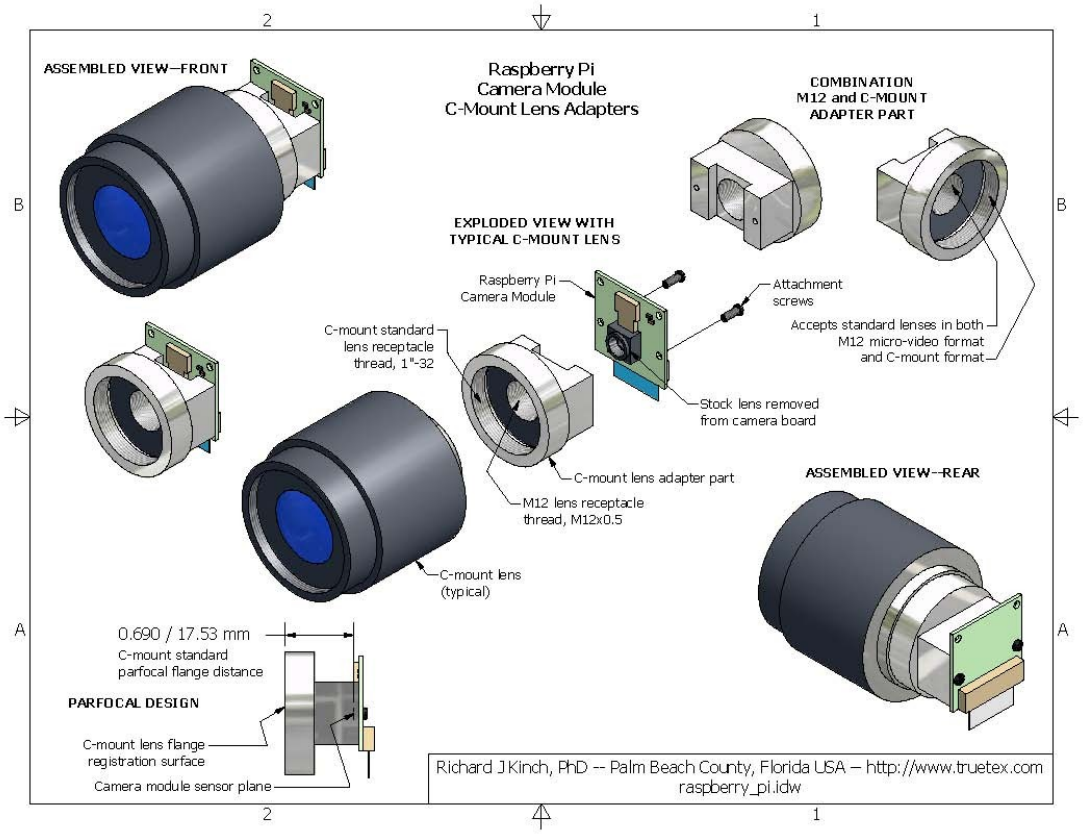
https://www.adafruit.com/products/1937
https://www.flickr.com/people/100320847@N06/

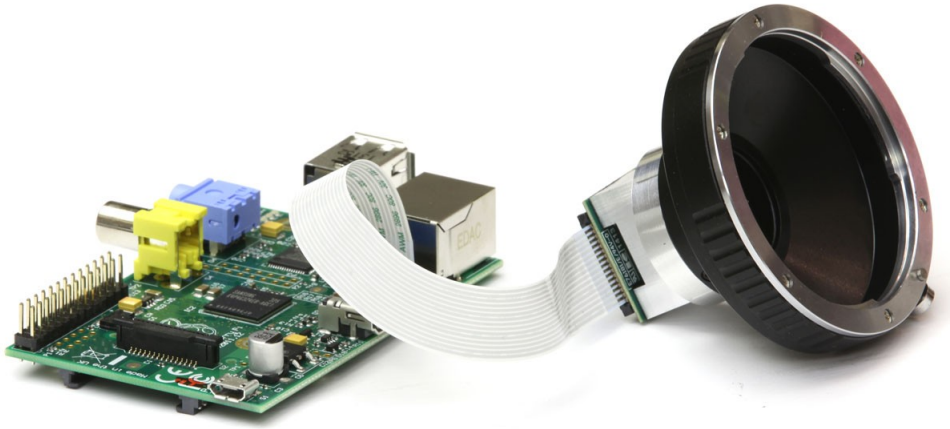
鏡頭改裝



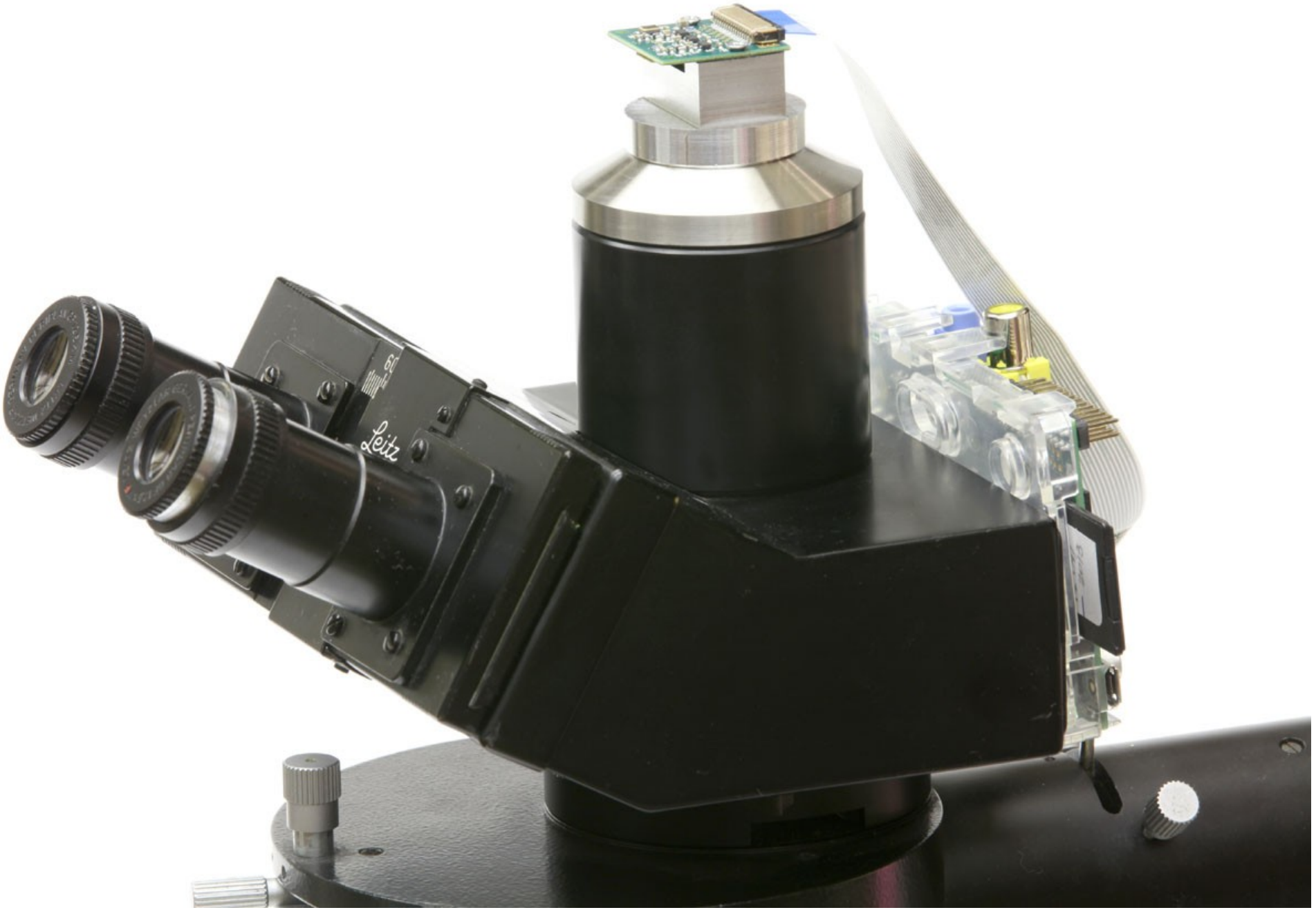
http://www.modmypi.com/

惡改鏡頭



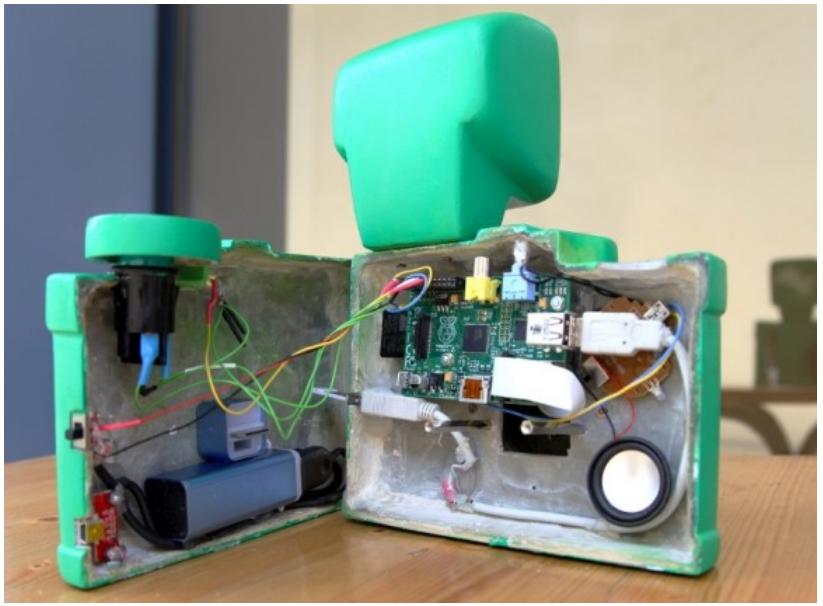


http://www.truetex.com/raspberrypi



http://www.truetex.com/raspberrypi

外殼改裝



http://www.modmypi.com/
http://blog.pi3g.com/2013/11/coming-soon-raspberry-with-case-mounted-camera/

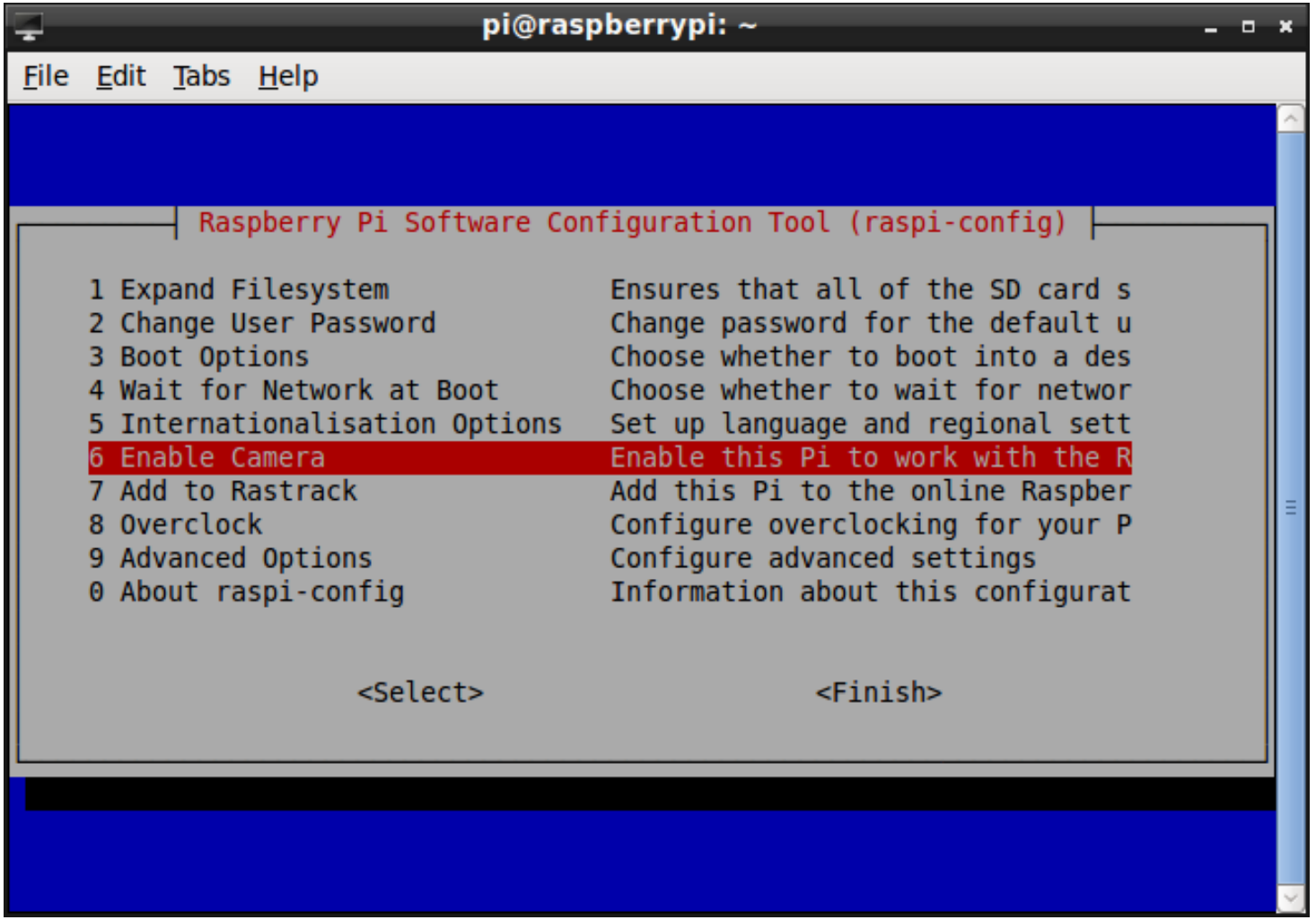
Camera 安装

在關機的狀態下安裝 Camera

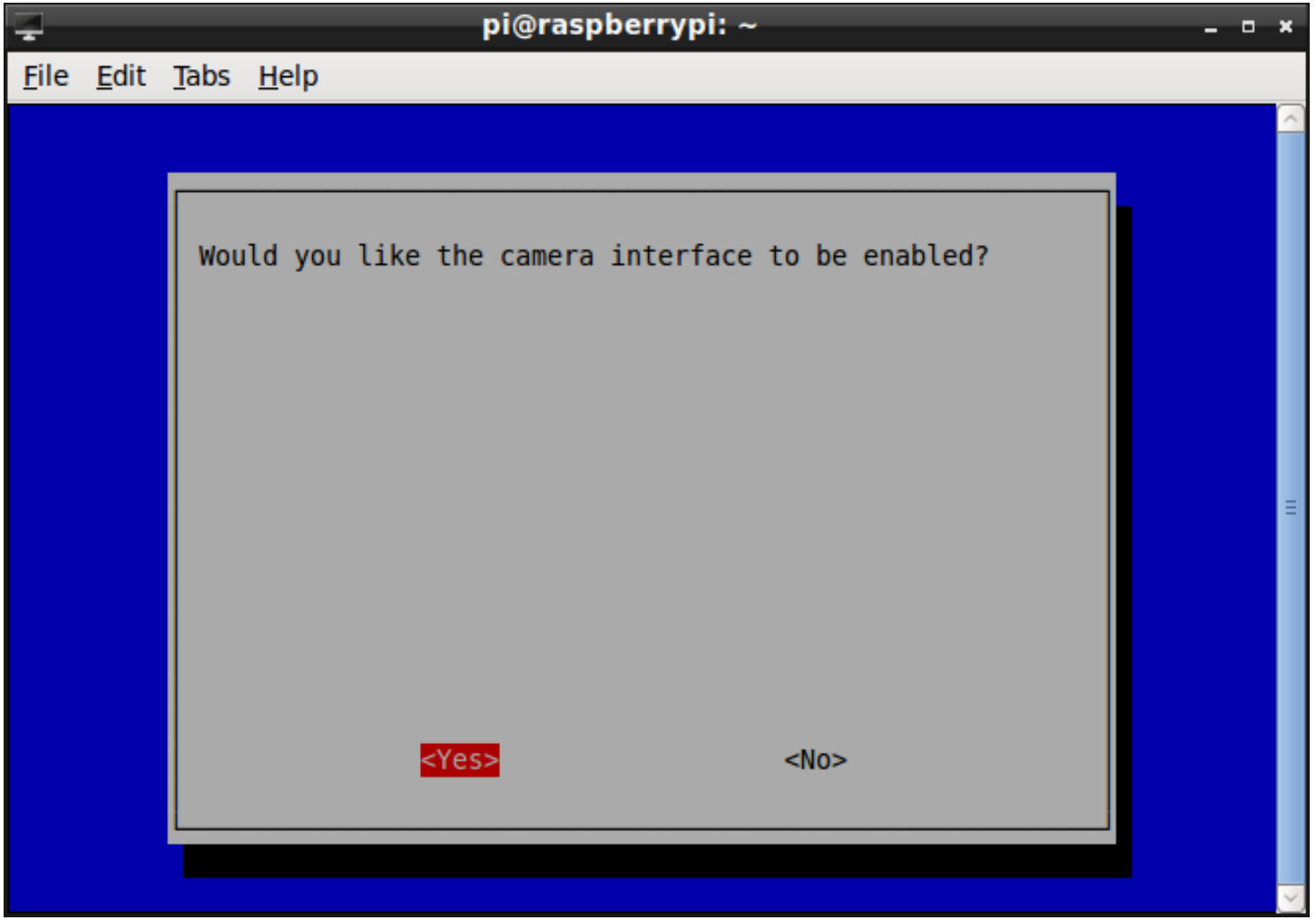
關機指令： $ sudo poweroff

http://goo.gl/7LqyMY

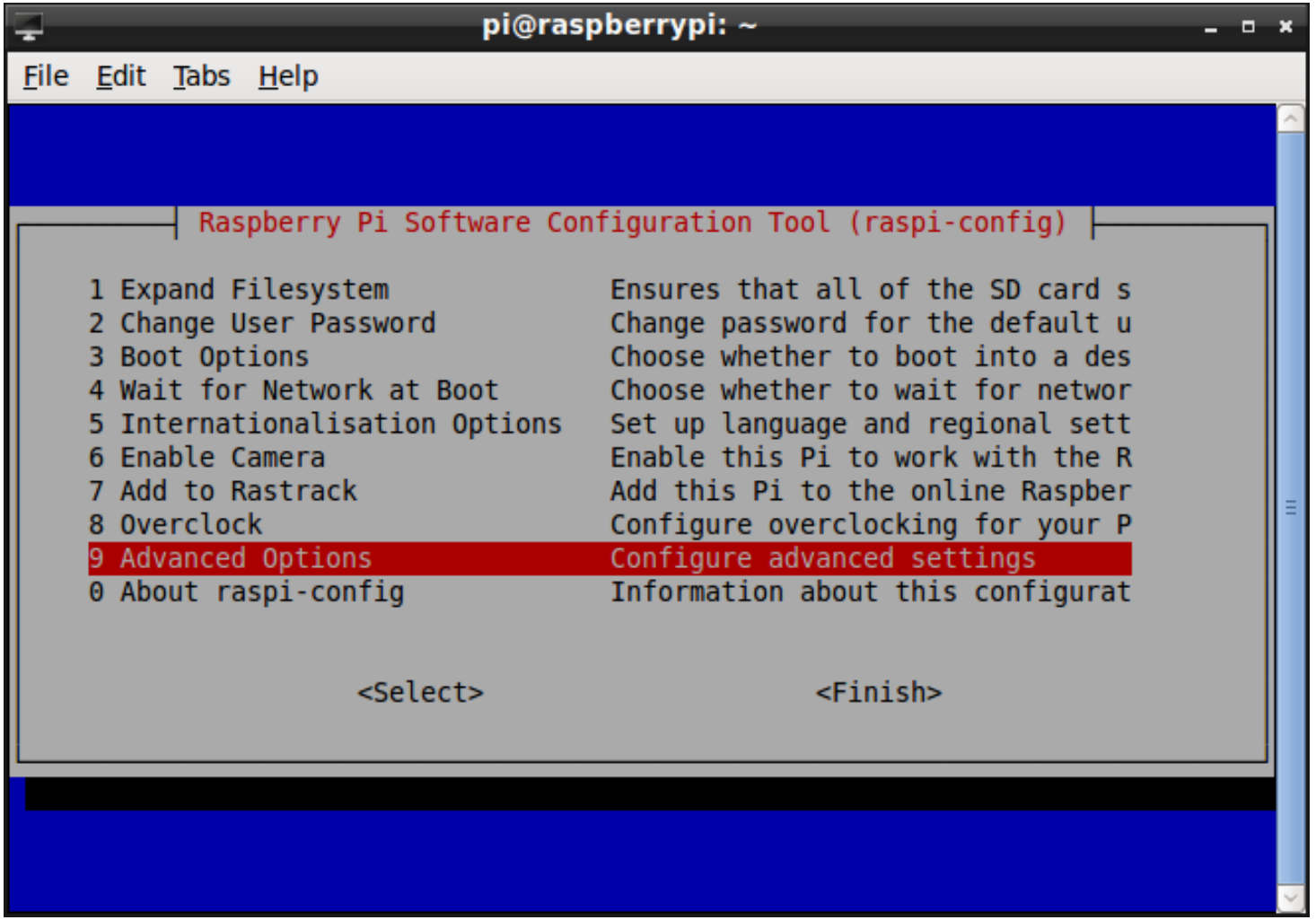
$ sudo raspi-config



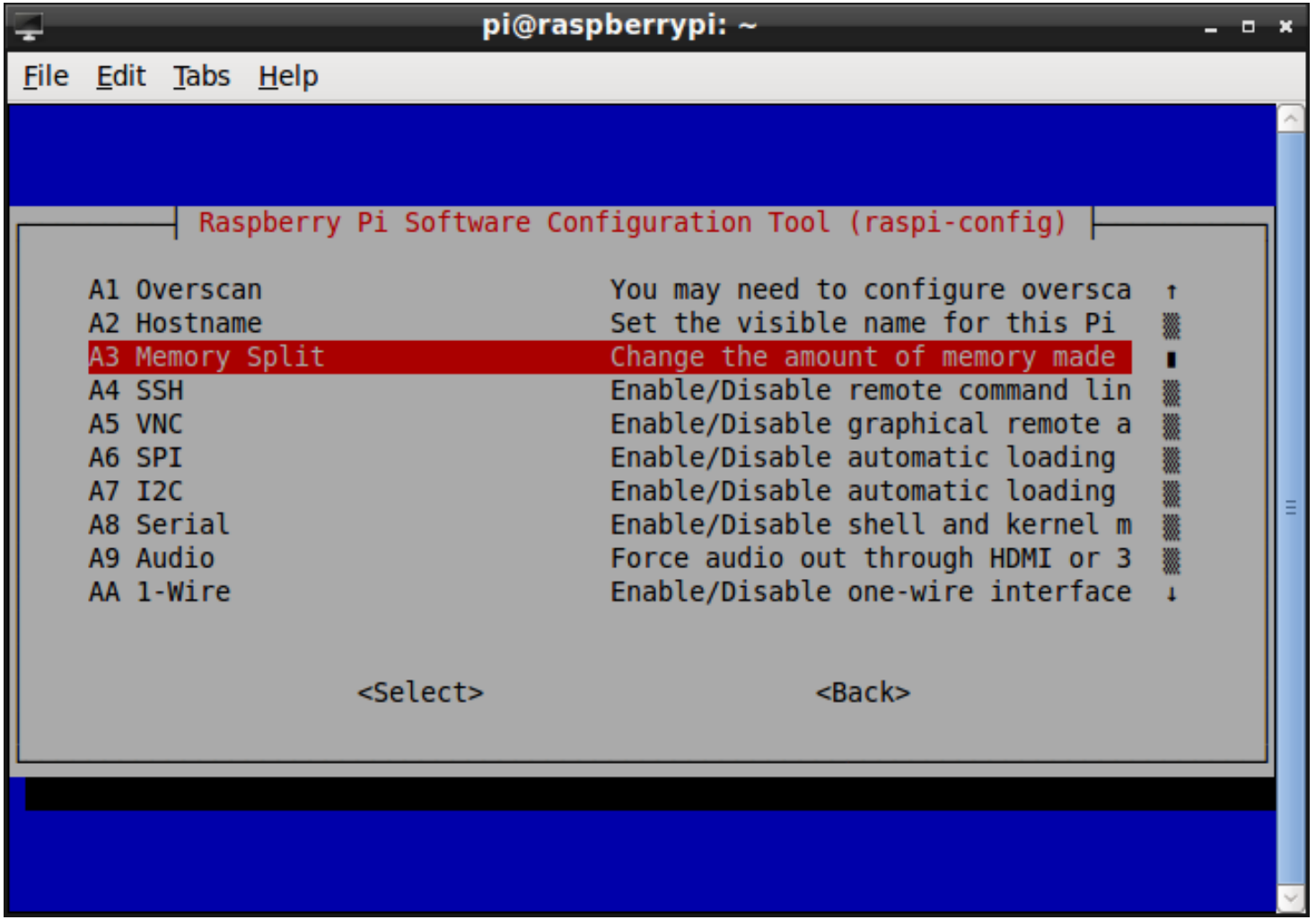
啟用 Raspberry Pi Camera



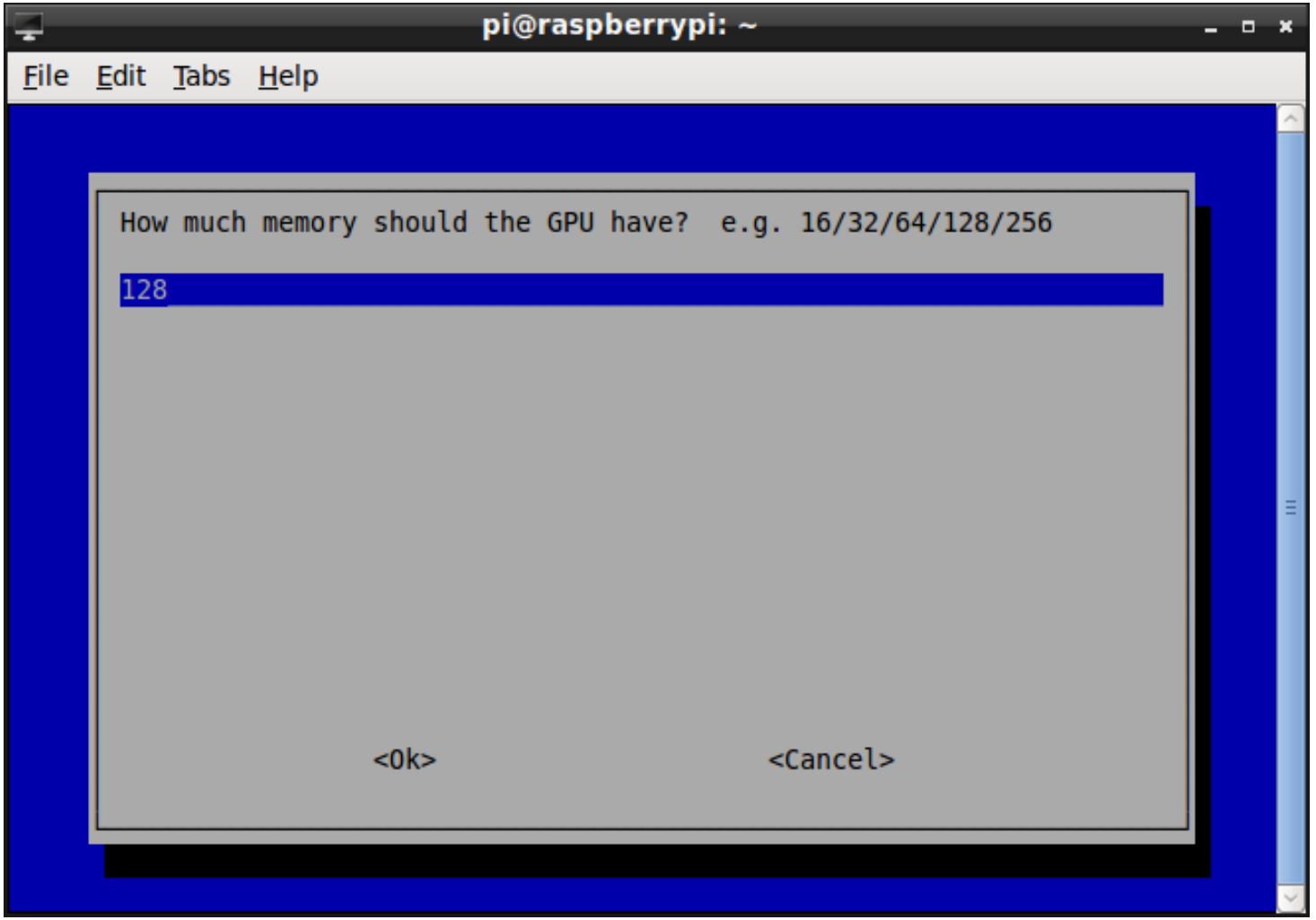
進階選項


pi@raspberrypi: ~
File Edit Tabs Help
Raspberry Pi Software Configuration Tool (raspi-config)
1 Expand Filesystem Ensures that all of the SD card s
2 Change User Password Change password for the default u
3 Boot Options Choose whether to boot into a des
4 Wait for Network at Boot Choose whether to wait for networ
5 Internationalisation Options Set up language and regional sett
6 Enable Camera Enable this Pi to work with the R
7 Add to Rastrack Add this Pi to the online Raspber
8 Overclock Configure overclocking for your P
9 Advanced Options Configure advanced settings
0 About raspi-config Information about this configurat
<Select>
<Finish>

設定記憶體分配


pi@raspberrypi: ~
File Edit Tabs Help
Raspberry Pi Software Configuration Tool (raspi-config)
A1 Overscan
You may need to configure oversca
A2 Hostname
Set the visible name for this Pi
A3 Memory Split
Change the amount of memory made
A4 SSH
Enable/Disable remote command lin
A5 VNC
Enable/Disable graphical remote a
A6 SPI
Enable/Disable automatic loading
A7 I2C
Enable/Disable automatic loading
A8 Serial
Enable/Disable shell and kernel m
A9 Audio
Force audio out through HDMI or 3
AA 1-Wire
Enable/Disable one-wire interface
<Select>
<Back>

>128M


pi@raspberrypi: ~
File Edit Tabs Help
How much memory should the GPU have? e.g. 16/32/64/128/256
128
<Ok>
<Cancel>

實戰 Camera 使用

使用 Camera 前先消除靜電吧

http://www.wikihow.com/Remove-Static-Electricity

實驗 1:Hello Camera

目的 : 練習照相和攝影的指令

RaspiStill

- 只預覽 2 秒 (-t), 不存檔
  - `$ raspistill -t 2000`

- 5 秒後拍照 (預設), 檔案 test.jpg(-o)
  - `$ raspistill -o test.jpg`

- 3 秒後拍照，並編碼成 png 格式 (-e), 長 640x 寬 480
  - `$ raspistill -t 3000 -o test.png -e png -w 640 -h 480`

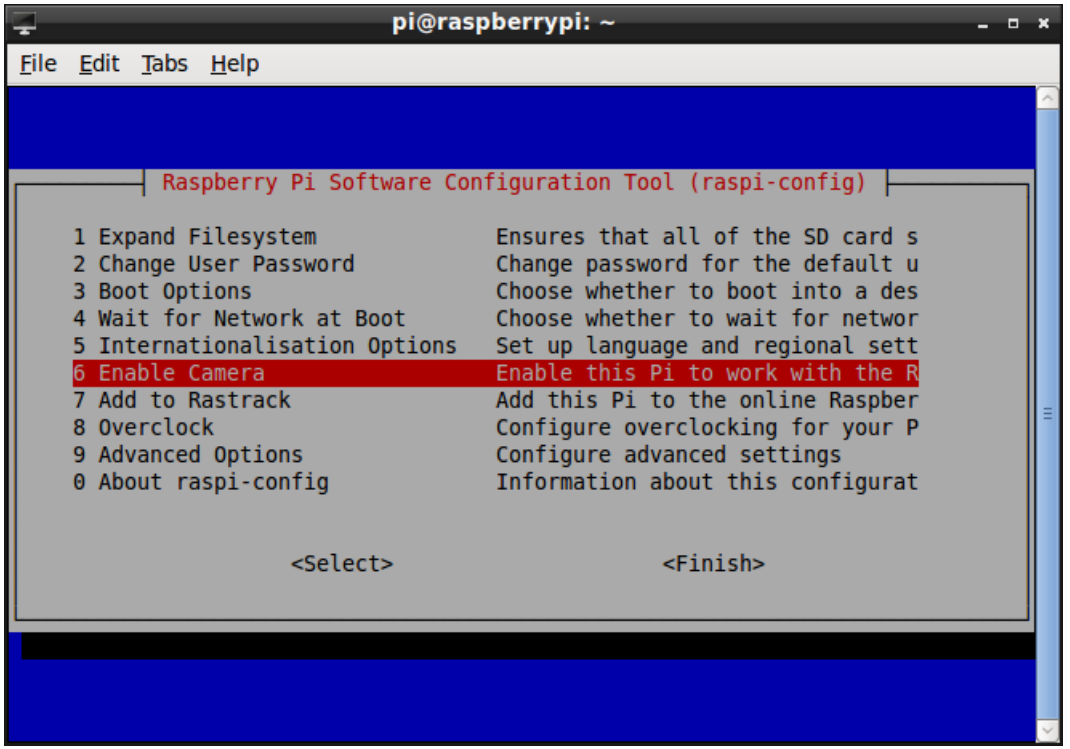
https://www.raspberrypi.org/documentation/raspbian/applications/camera.md

常見 Camera 問題？

- 訊息：Camera is not enabled in this build

```
mmal: mmal_vc_component_create: failed to create component 'vc.ril.camera' (1:ENOMEM)
mmal: mmal_component_create_core: could not create component 'vc.ril.camera' (1)
mmal: Failed to create camera component
mmal: main: Failed to create camera component
mmal: Camera is not enabled in this build. Try running "sudo raspi-config" and ensure that "camera" has been enabled
```

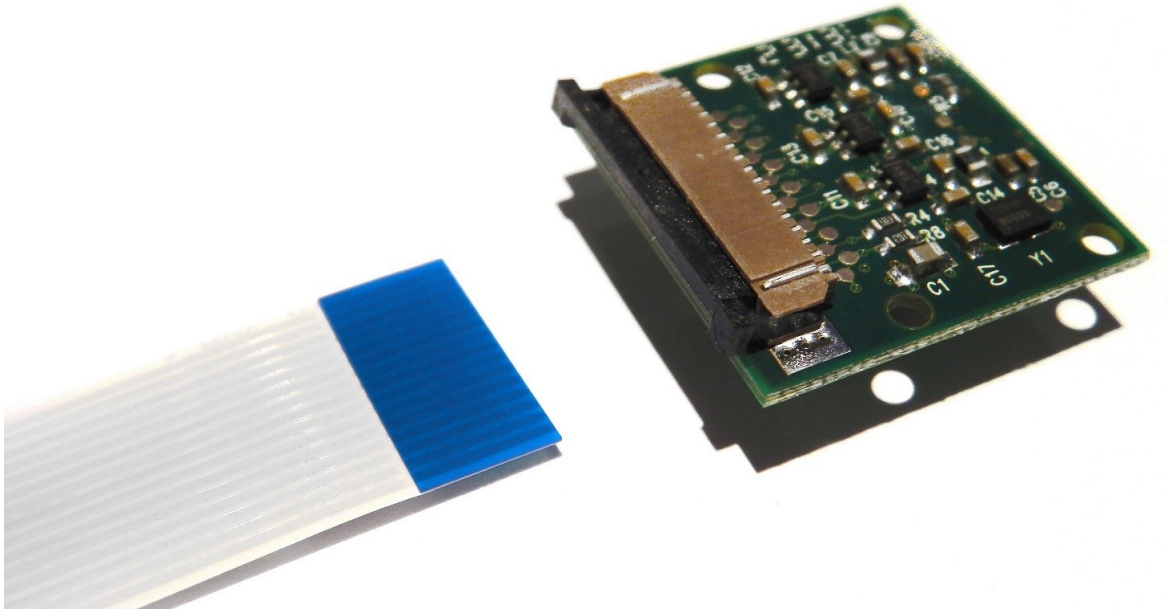
- 解法：進 raspi-config 重新 enable camera
  - $ sudo raspi-config



- 訊息：Camera is not detected

```
mmal: mmal_vc_component_create: failed to create component 'vc.ril.camera' (1:ENOMEM)
mmal: mmal_component_create_core: could not create component 'vc.ril.camera' (1)
mmal: Failed to create camera component
mmal: main: Failed to create camera component
mmal: Camera is not detected. Please check carefully the camera module is installed correctly
```

- 解法：重新安裝 camera，或是更換排線
  或是檢查 camera module 是否鬆脫



https://www.modmypi.com/blog/how-to-replace-the-raspberry-pi-camera-cable

RaspiVid

- 錄 5 秒 (-t) 1080p30 影片，名稱 video.h264
  - $ raspivid -t 5000 -o video.h264
- framerate 為 5(fps 從 2-30)，長 640x 寬 480
  - $ raspivid -t 5000 -o video.h264 -f 5 -w 640 -h 480
- 錄 5 秒的影片並導到標準輸出 (stdout)
  - $ raspivid -t 5000 -o -

https://www.raspberrypi.org/documentation/raspbian/applications/camera.md

更多參數或用法請看文件

http://goo.gl/V4klcZ

https://github.com/raspberrypi/documentation/blob/master/raspbian/applications/camera.md

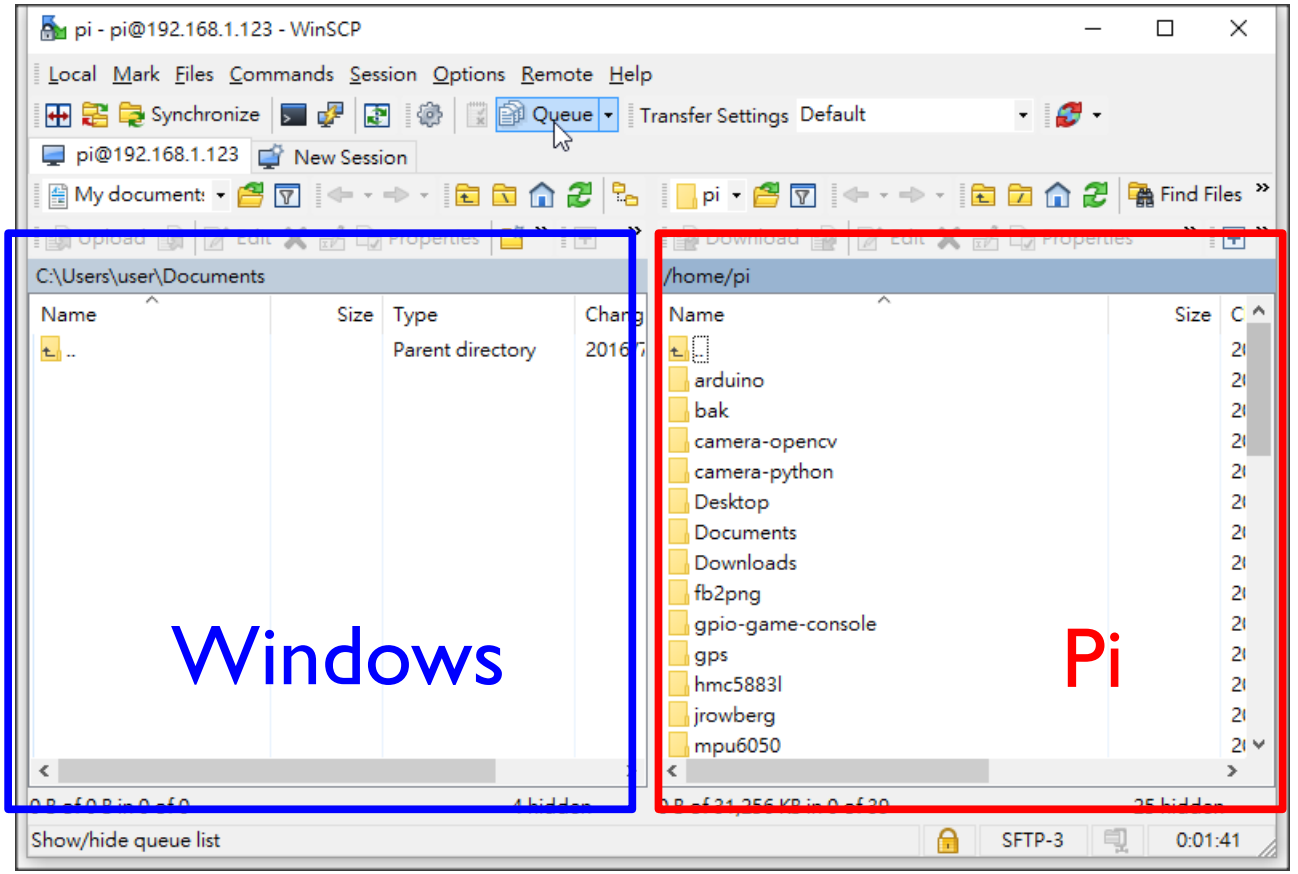
如何看照片和影片？

方法一：將 Pi 的檔案傳回本機端

在 Windows 上安裝 WinSCP

- **下載網址**

  `http://winscp.net/eng/download.php`



http://winscp.net/

在 linux 或是 Mac OS 上

- 使用 scp
  - 一個實做 SCP(Secure Copy Protocol) 的應用程式
  - 透過 SSH(Secure Shell) 傳輸資料

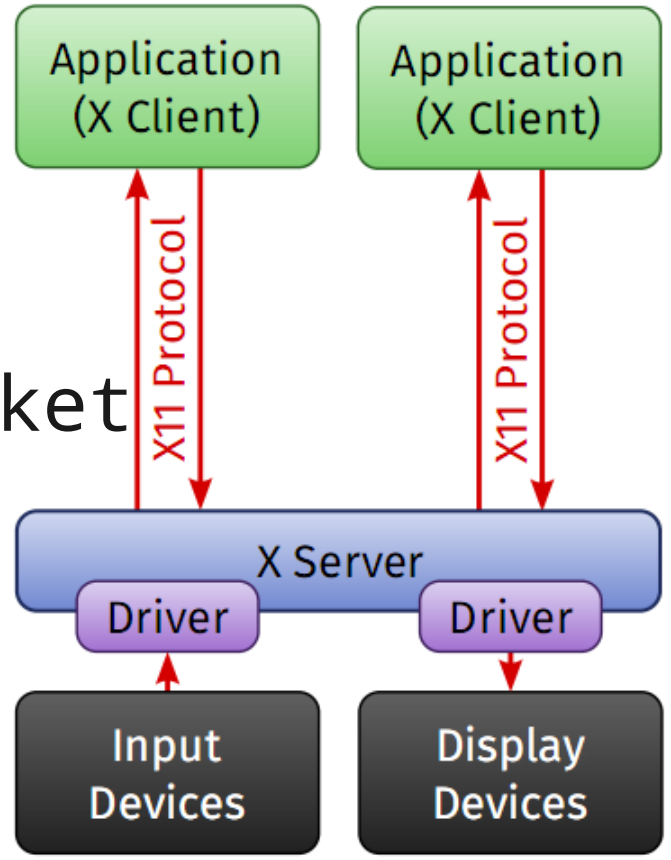
- 從 Pi 複製檔案到 PC 上的兩種寫法
  - `$ scp pi@x.x.x.x:/home/pi/file.txt .`
  - `$ scp file.txt pi@x.x.x.x:/home/pi`

表示當前目錄

方法二：使用 X11 Forwarding

X Window System

- 是一種圖形應用標準
- Client/Server 架構
  - X Client: 應用程式
  - X Server: 管理硬體輸入 / 輸出
- 可透過網路傳輸
  - TCP/IP 或是 Unix Domain Socket
- X11 是通訊協定名稱



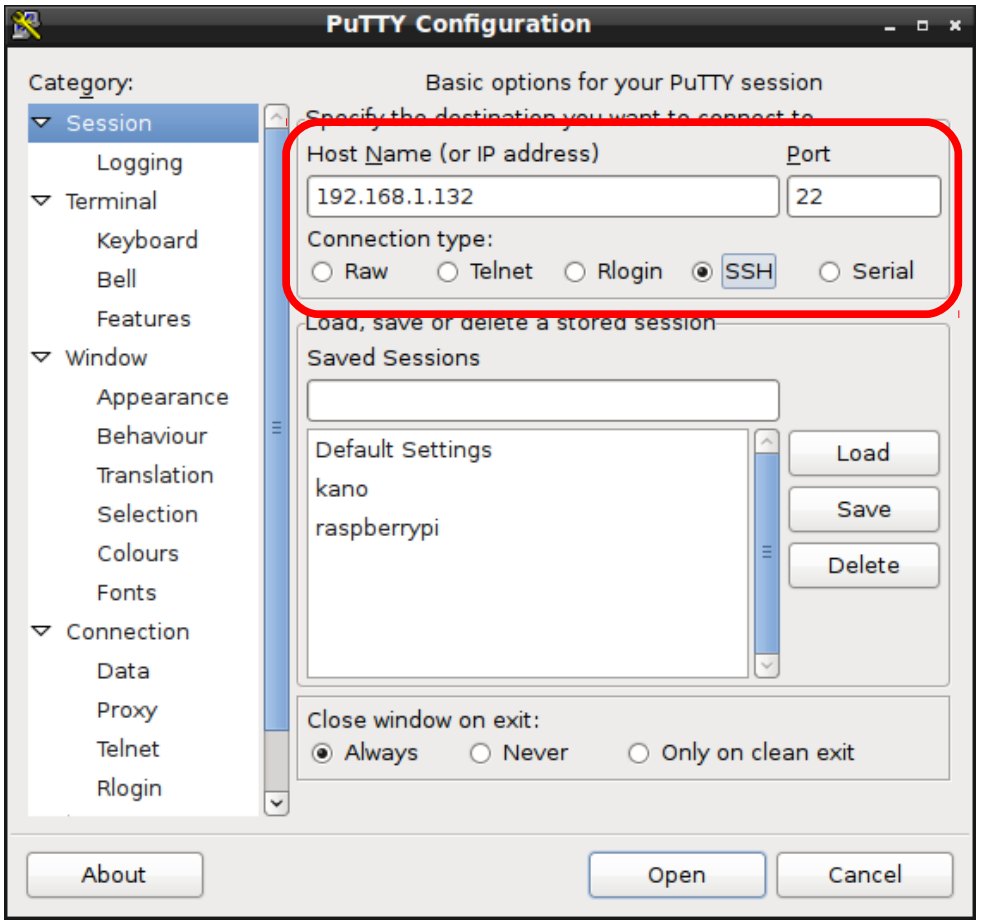
http://keyj.emphy.de/files/linuxgraphics_en.pdf

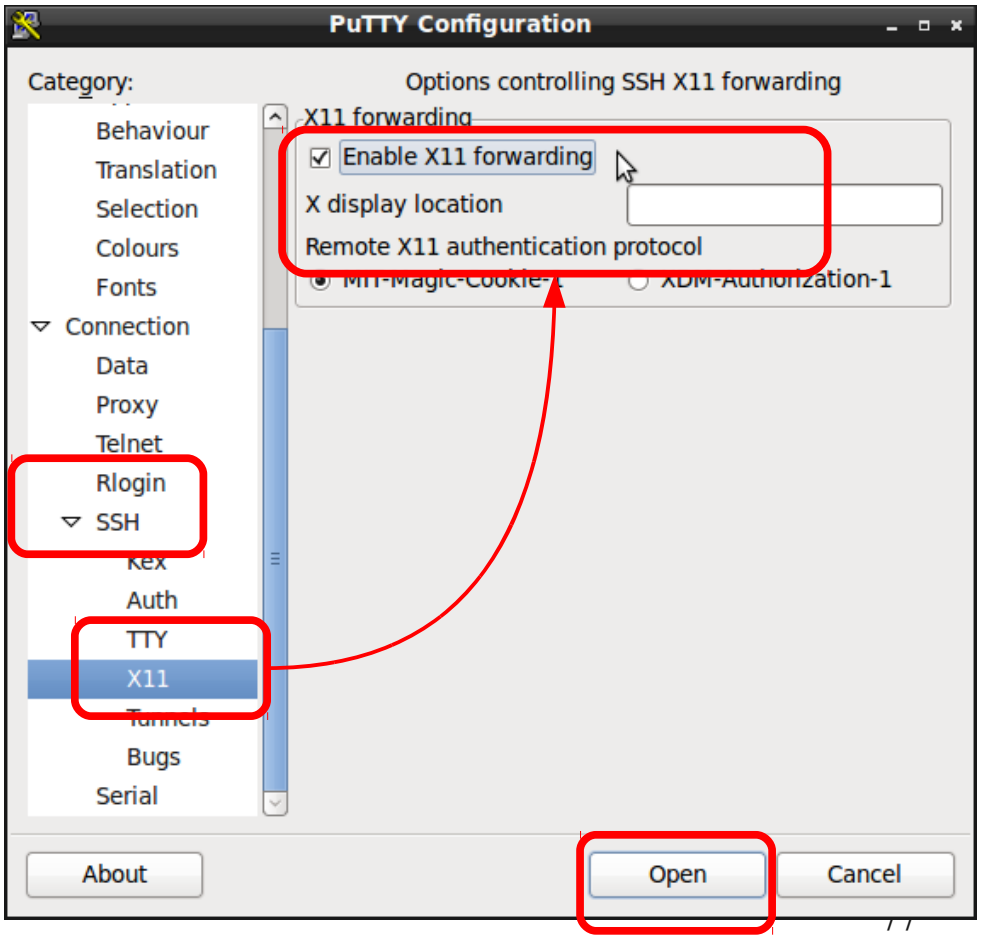
在 Windows 安裝 X Server

- 安裝 Xming，下一步到底
  http://sourceforge.net/projects/xming/

在 Windows 設定 X11 Forwarding

- SSH > X11 > Enable X11 forwarding





如果是 Linux 或是 Mac OS

開啟終端機，ssh -X pi@PI 的 IP

“Can not open display” on Mac

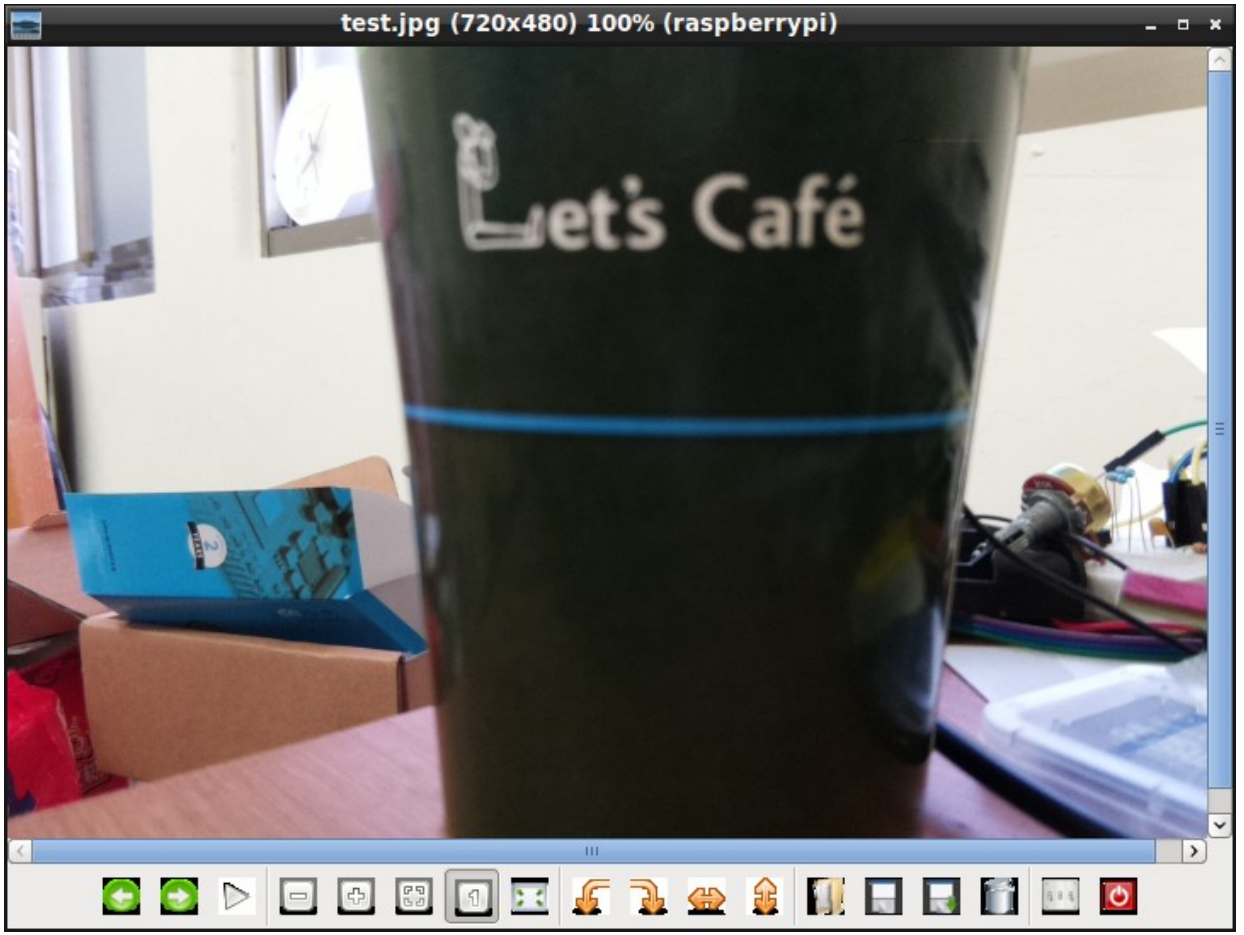
- 第一步：在 Mac 編輯 /etc/sshd_config
  (或是 /etc/ssh/sshd_config)
  修改這行 # X11Forwarding no
  把 no 改成 yes 並且把註解拿掉
- 第二步：下載安裝 XQuartz 並重開機
  http://xquartz.macosforge.org/landing/

- 感謝 Dami 和 YUN-TAO CHEN 的貢獻

https://hackpad.com/X11-Forwarding-FcyKHioKxmW

X11 Forwarding 連線成功後

- 看照片
  - `$ gpicview test.jpg`

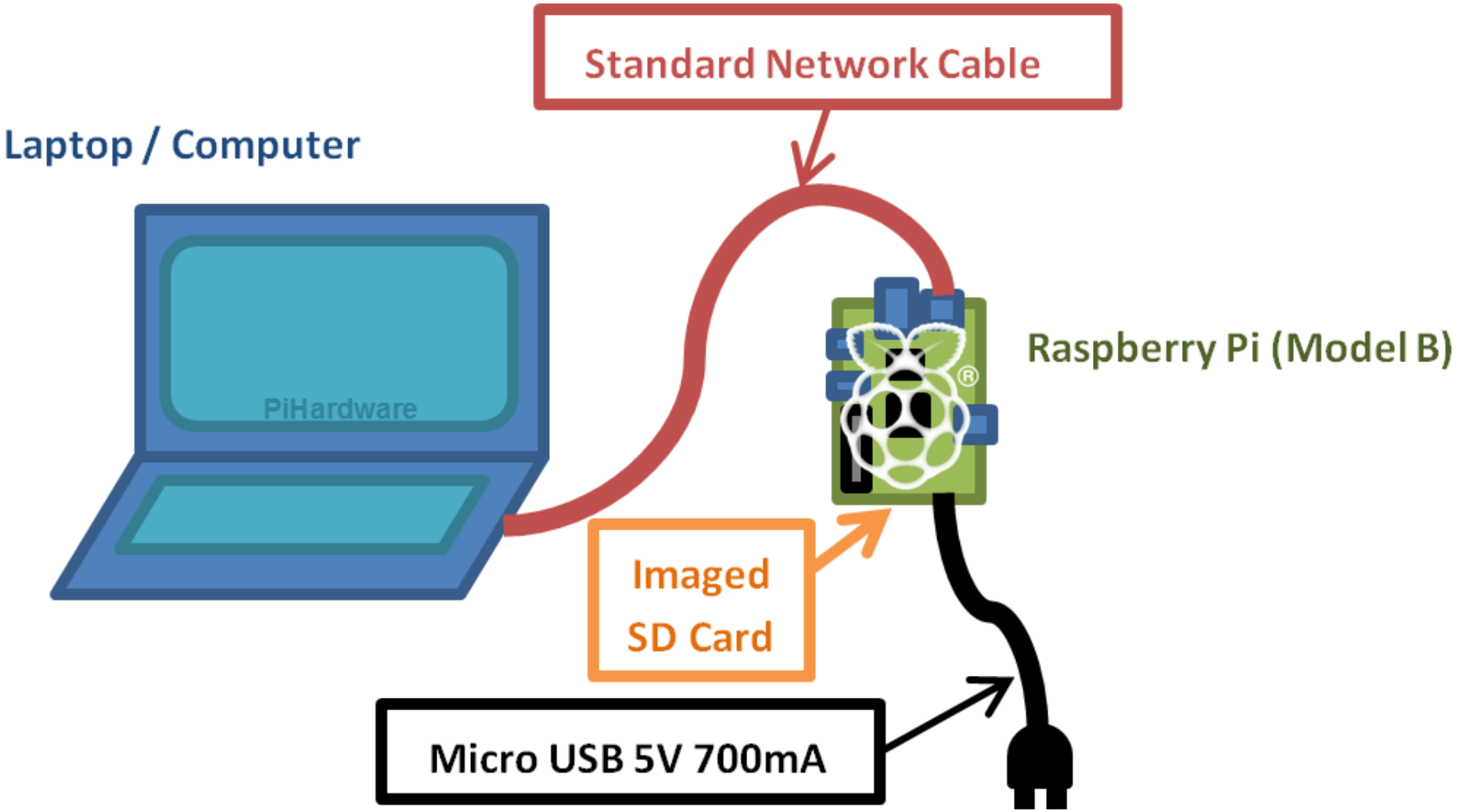


X11 Forwarding FAQ

- Q: 什麼時候可以使用 X11 Forwarding ?
- A: 當 GUI toolkit 或是 library 是建構在 X 的時候
  - 例如 Qt,Gtk,Tkinter 等等適合
  - SDL,Framebuffer,GPU 直接輸出等不能使用

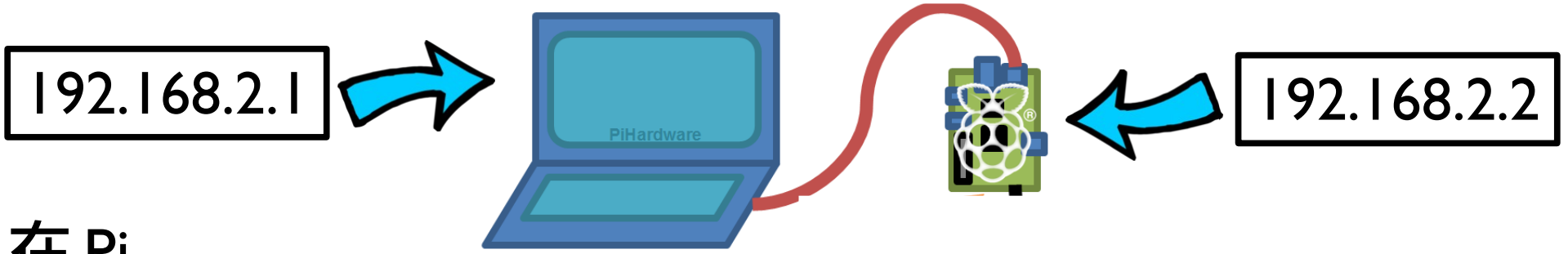
- Q: 什麼時候要使用 sudo 什麼時候不用 ?
- A: 因為 X11 Forwarding 需要認證 (authorization), 所以如果要畫面回傳時都不能使用 sudo 執行

嫌慢的話直接用網路線對接

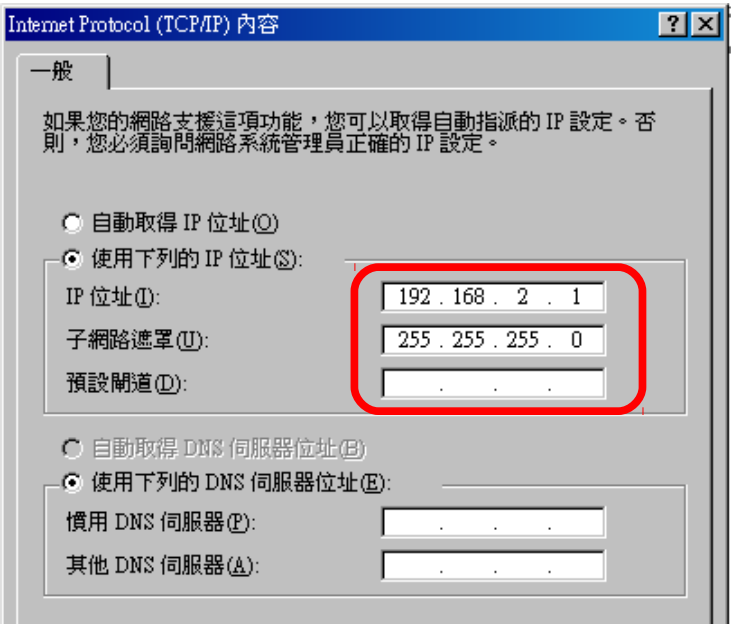


https://pihw.wordpress.com/guides/direct-network-connection/

網路線對接後的設定

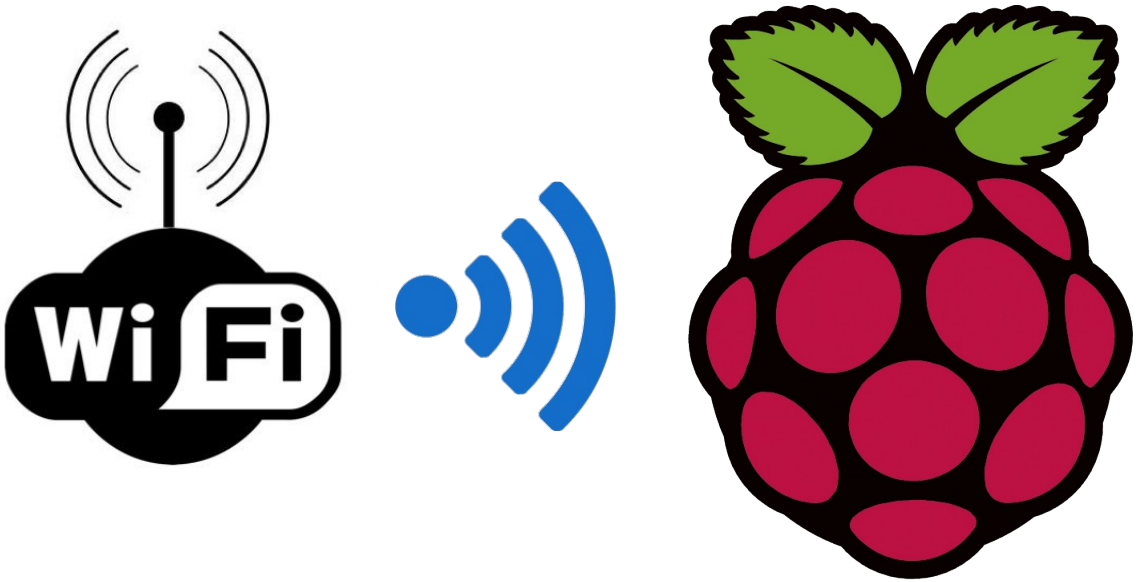


- 在 Pi
  - $ sudo ifconfig eth0 192.168.2.2 netmask 255.255.255.0
- 在 Windows

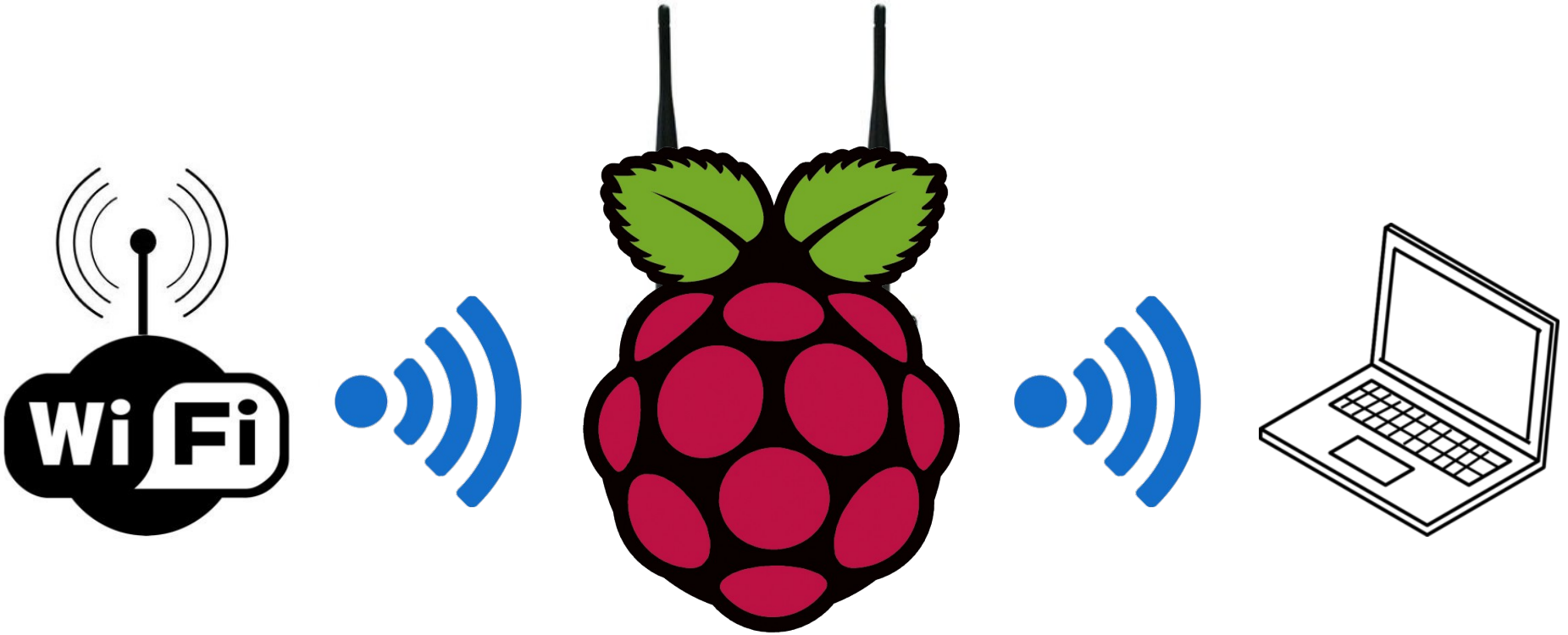


或者試試看雙模設定吧

WiFi Client



WiFi Client



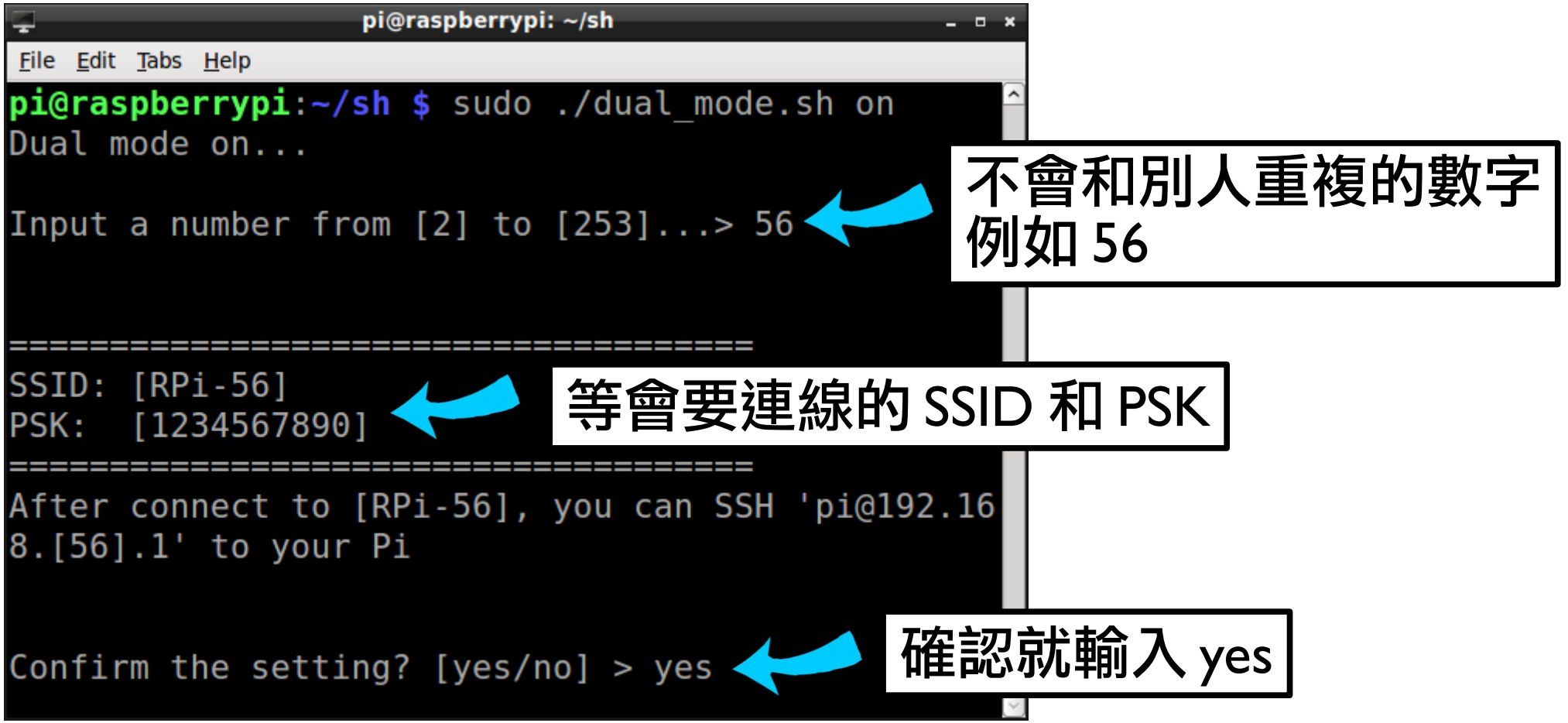
WiFi AP

下載 dual_mode.sh

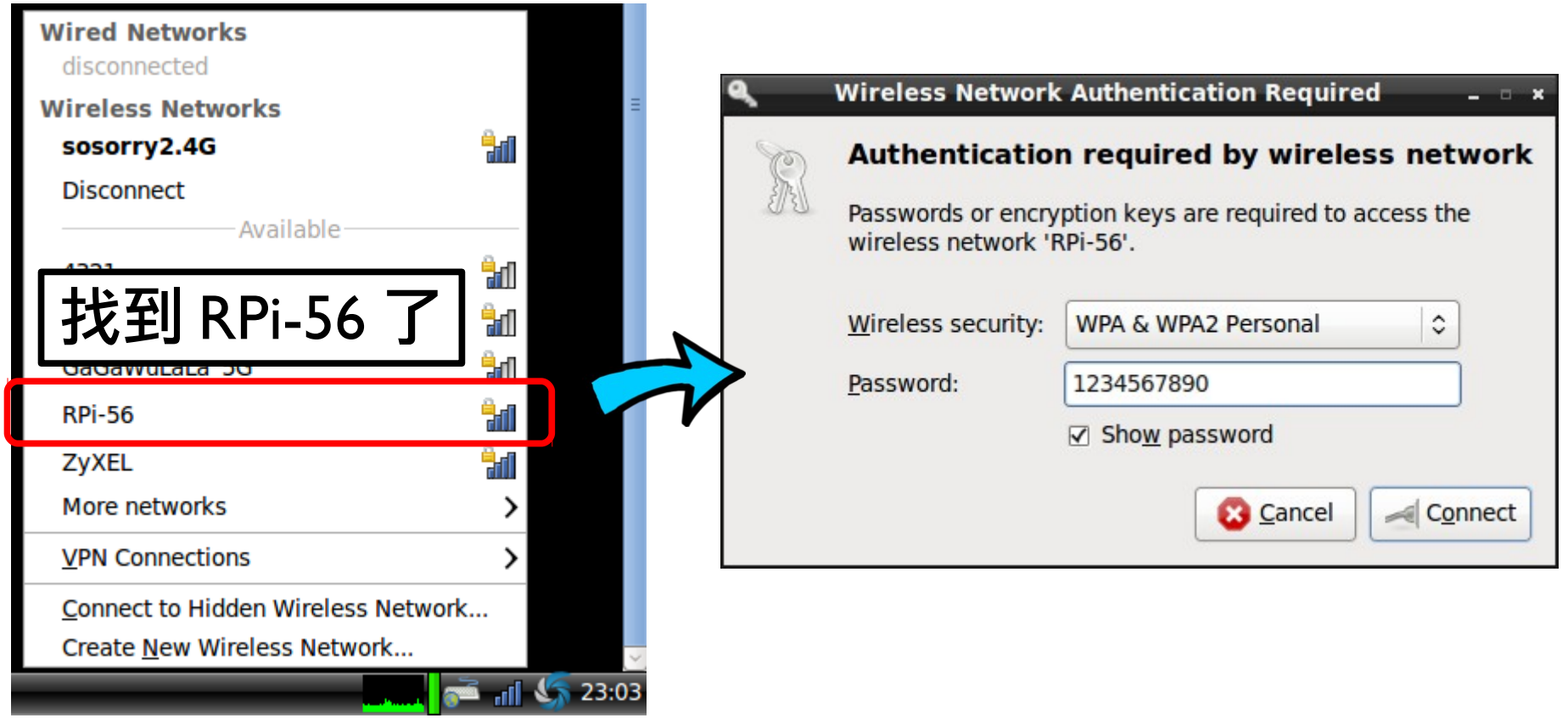
- `$ cd ~`
- `$ git clone https://github.com/raspberrypi-tw/sh`
- `$ cd sh`

啟用網卡雙模功能 (Pi 3 限定)

- `$ sudo ./dual_mode.sh on`

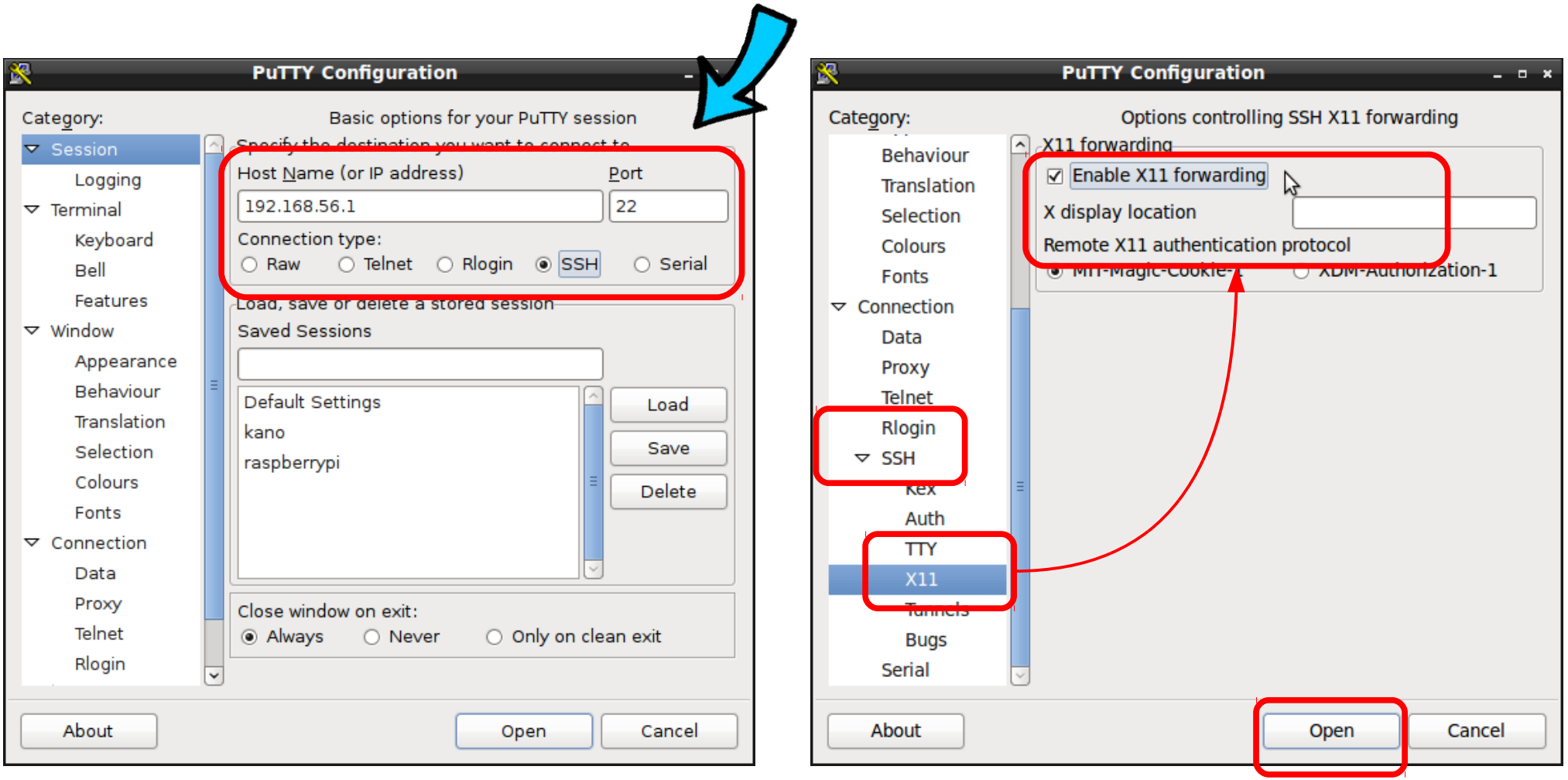


重開機以後就可以搜尋到 SSID



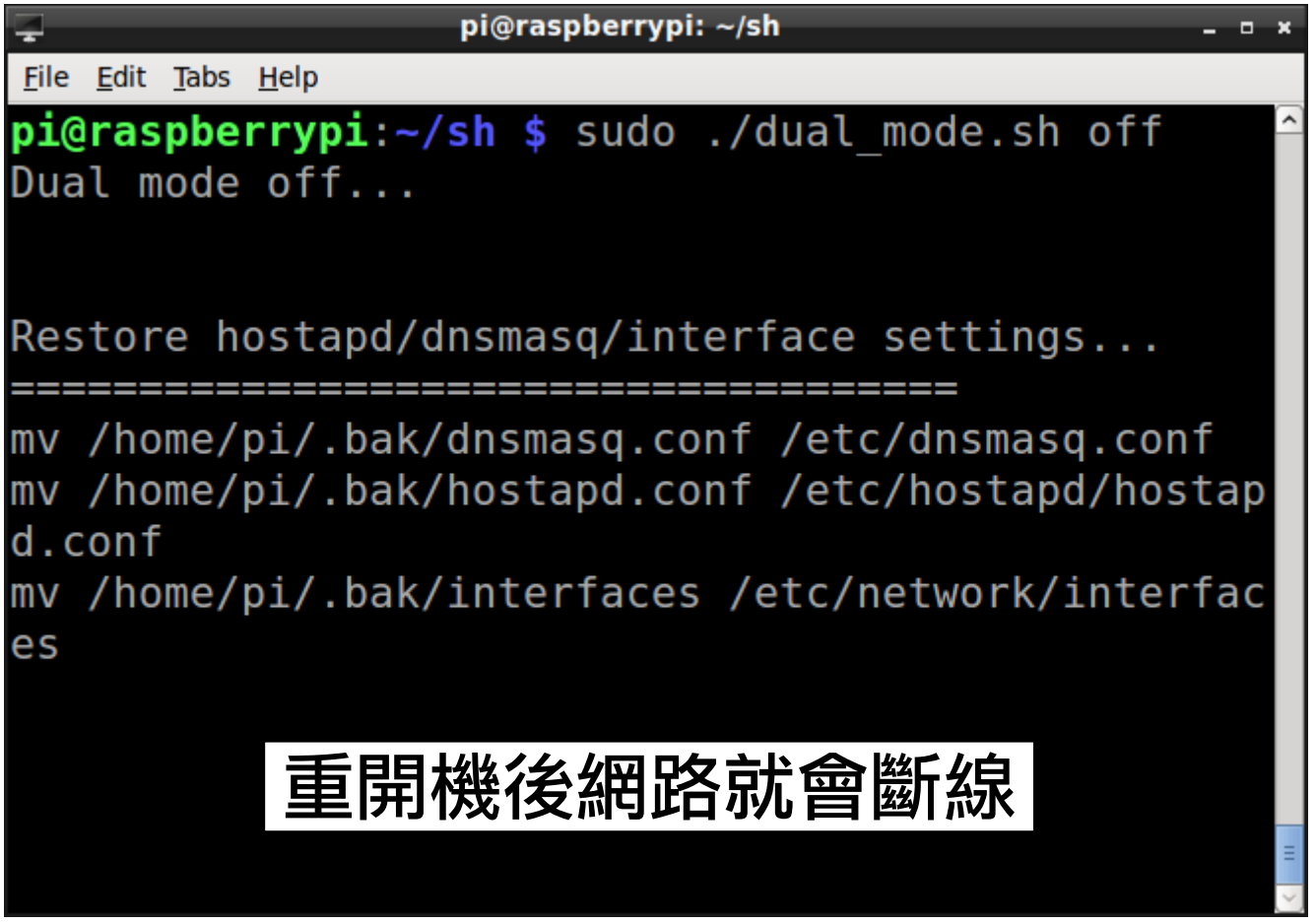
連線成功後，從電腦直接連到 Pi

- Pi 的 IP 為 192.168.56.1



停用網卡雙模功能 (回復原始設定)

- $ cd ~/sh
- $ sudo ./dual_mode.sh off



實驗 2: 寫程式控制 Camera

目的 : 自己的 Camera 自己做

使用picamera (Python library)

Python2 五分鐘速成

- 變數，物件，型別，註解
- 模組
- 縮排
- 迴圈
- 條件判斷
- 函式

變數，物件，型別，註解

- 動態型別(dynamic typing)

```
# 這是註解
i = 3                          # 變數 i 指到數字物件 3
i = [1, 2, 3, 4, 5]  # 變數 i 指到串列物件
print i[2]                     # 印出串列中第三個元素
i = “abcde”                    # 變數 i 指到字串物件
print i[2]                     # 印出字串中第三個元素
```

模組

```
# import MODULE
import picamera
# import MODULE as ALIAS
import RPi.GPIO as GPIO
```

縮排

- 用縮排取代大括號
- 程式碼的區塊是用縮排分隔
- 不使用 tab，使用空白鍵
- 常見縮排為 4 個空白鍵

迴圈

- 自動迭代(iterator)

```
for i in xrange( start, stop[, step] ) :
    process

for i in xrange( 0, 11, 5 ) :
    process
```

條件判斷

```
if condition1:
    process1
elif condition2:
    process2
else:
    process3
process4
```

函式

```
def function_name():
    process

def function_name( param_name ):
    process

def function_name( param_name = 3):
    process
```

例外處理

- 程式受到中斷停止

```
file = open('/etc/issue', 'r')
try:
    for line in file:
        print line
except:
    print 'Open file error'
finally:
    file.close()
```

with... as...

- 使用with as來簡化try exception程式

```
with open('/etc/issue', 'r') as file:
    for line in file:
        print line
```

照相

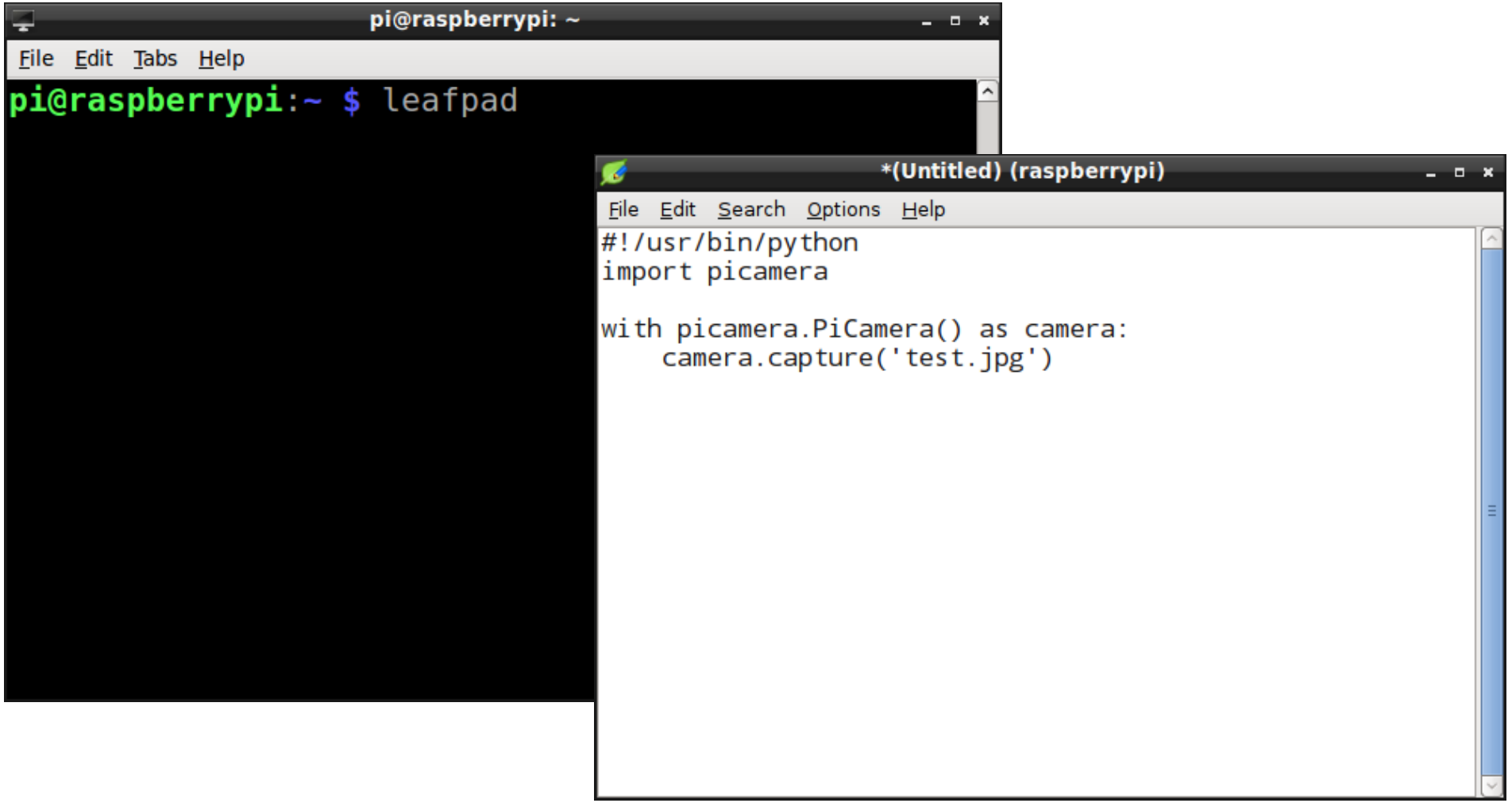
```
#!/usr/bin/python
import picamera

with picamera.PiCamera() as camera:
    camera.capture('test.jpg')
```

- 預設相片解析度為 720x480

http://picamera.readthedocs.org/en/release-1.8/recipes1.html

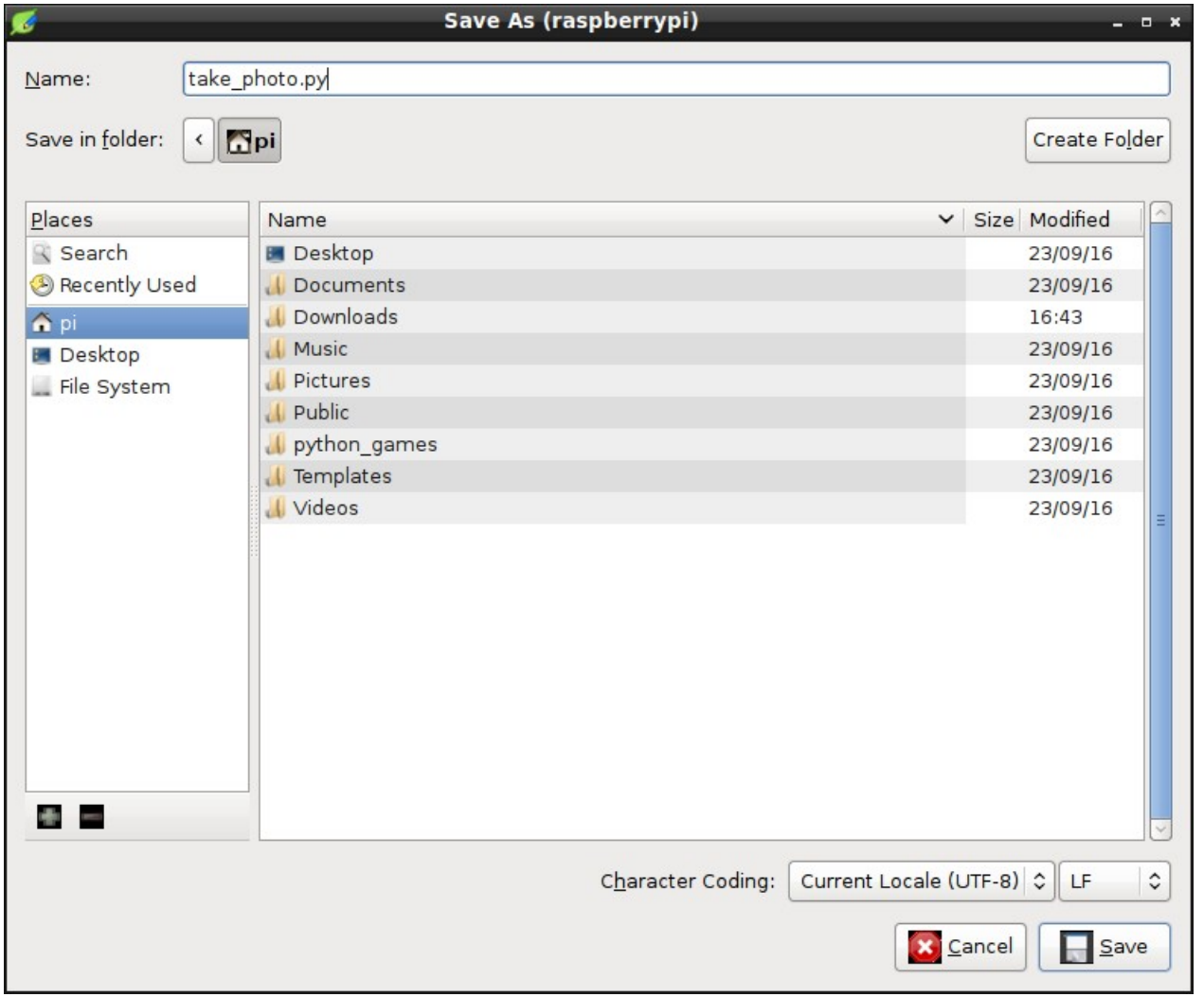
leafpad 編輯器使用

- 在 X11 forwarding 連線成功下執行 leafpad



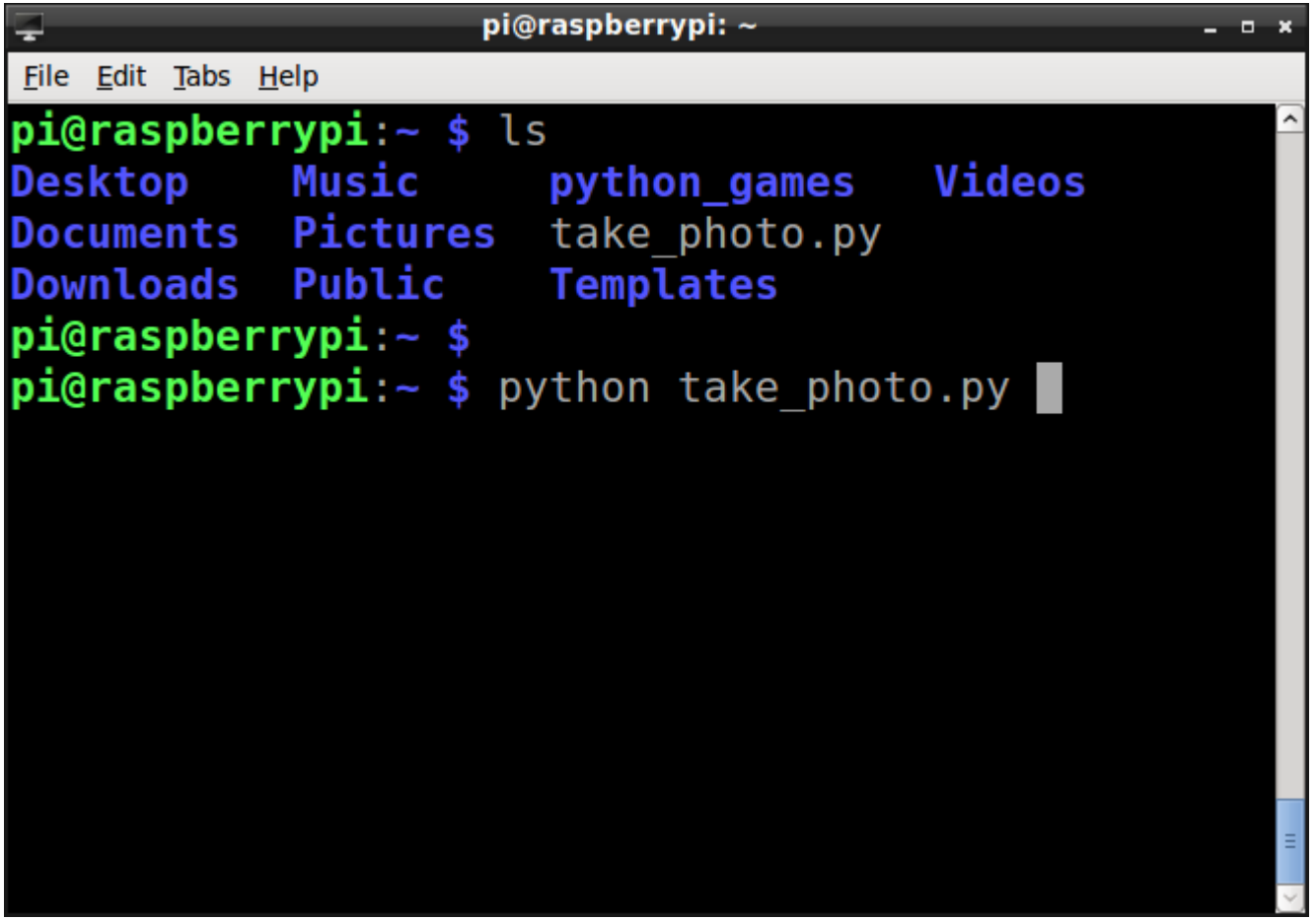
存檔

- 存檔 `take_photo.py`



執行

- `$ python take_photo.py`



下載範例程式

```
$ cd ~
$ curl
http://sosorry.s3.amazonaws.com/raspberry
pi/code/camera-opencv.tar.gz | tar xvz
```

DEMO
take_photo.py

$ cd ~/camera-opencv/02-picamera

$ python take_photo.py

你必須知道的 Linux 指令

- `$ cd <目錄>` # 跳到 <目錄>
  - `$ cd ..` # 回上一層
  - `$ cd ~` # 回 <家目錄>
- `$ pwd` # 查看目前工作目錄
- `$ ls` # 列出檔案與目錄
- `$ sudo reboot` # 重開機

錄影

```
#!/usr/bin/python
import picamera

with picamera.PiCamera() as camera:
    camera.start_recording('video.h264')
    camera.wait_recording(3)
    camera.stop_recording()
```

- 錄 3 秒鐘影像，儲存到檔案 video.h264
- 預設錄影格式為 H.264/AVC 壓縮， 解析度 1280x800

http://picamera.readthedocs.org/en/release-1.8/recipes1.html

DEMO
record_video.py

```
$ cd ~/camera-opencv/02-picamera
$ python record_video.py
```

低光源拍照

```
import picamera
import time
from fractions import Fraction

with picamera.PiCamera() as camera:
    camera.resolution = (640, 480)
    camera.framerate = Fraction(1, 3)
    camera.shutter_speed = 3000000
    camera.exposure_mode = 'off'
    camera.iso = 800
    time.sleep(10)
    camera.capture('dark.jpg')
```

http://picamera.readthedocs.org/en/release-1.8/recipes1.html

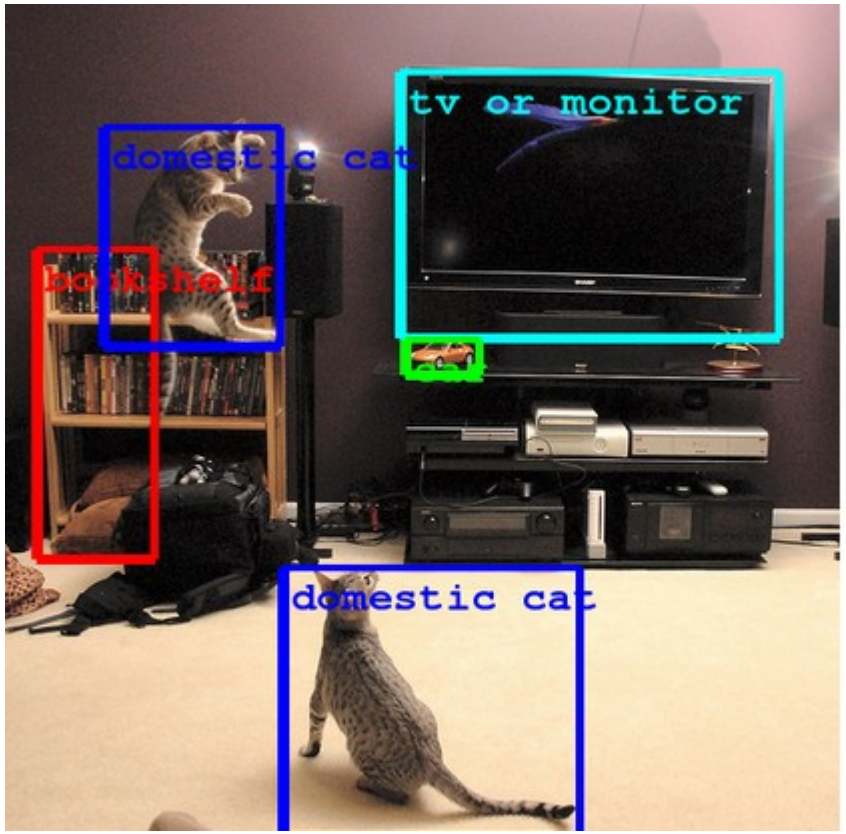
DEMO
low_light.py

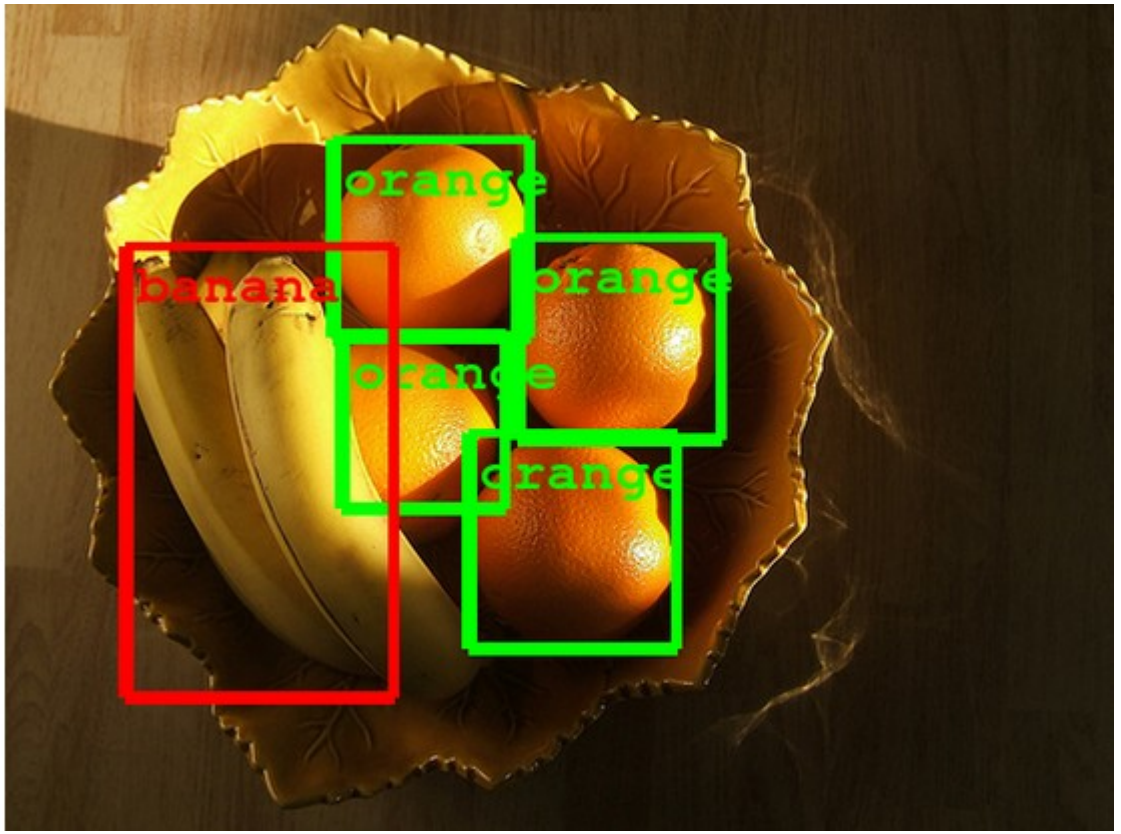
```
$ cd ~/camera-opencv/02-picamera
$ python low_light.py
```

實驗 3: 會認東西的 Camera

目的 : 串接網路服務

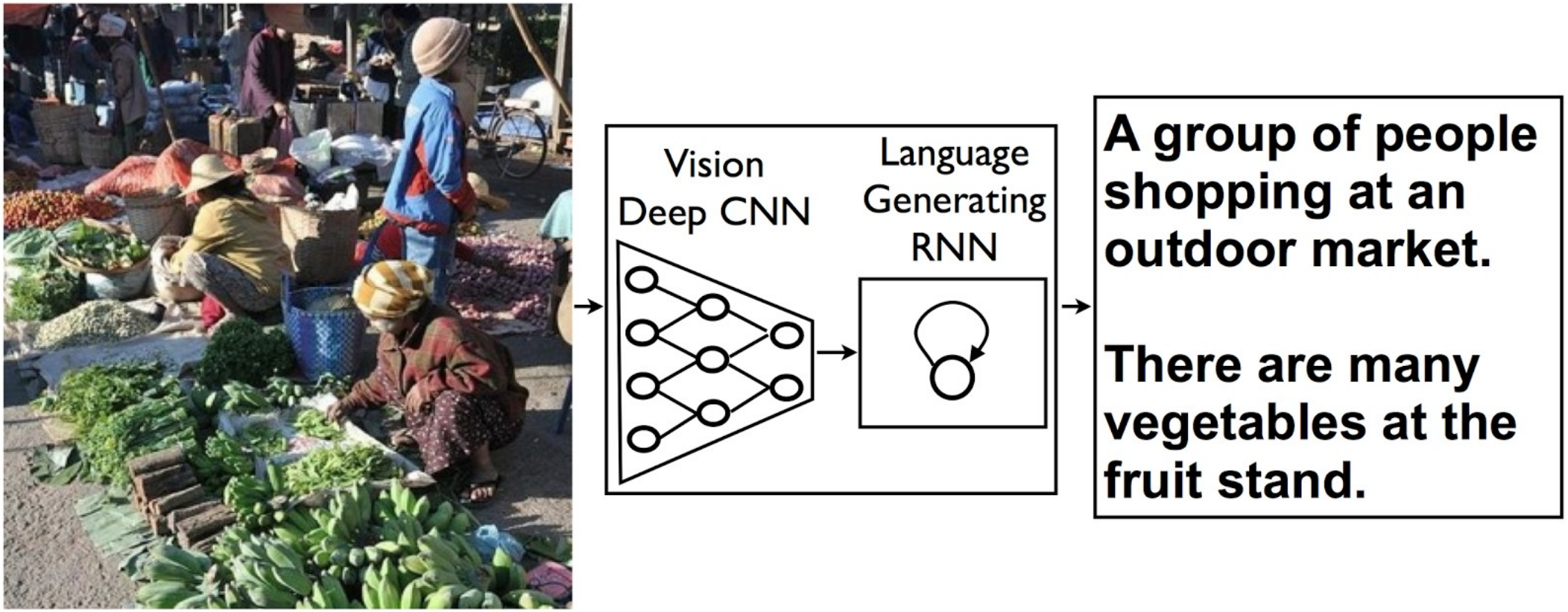
影像辨識





http://googleresearch.blogspot.tw/2014/09/building-deeper-understanding-of-images.html

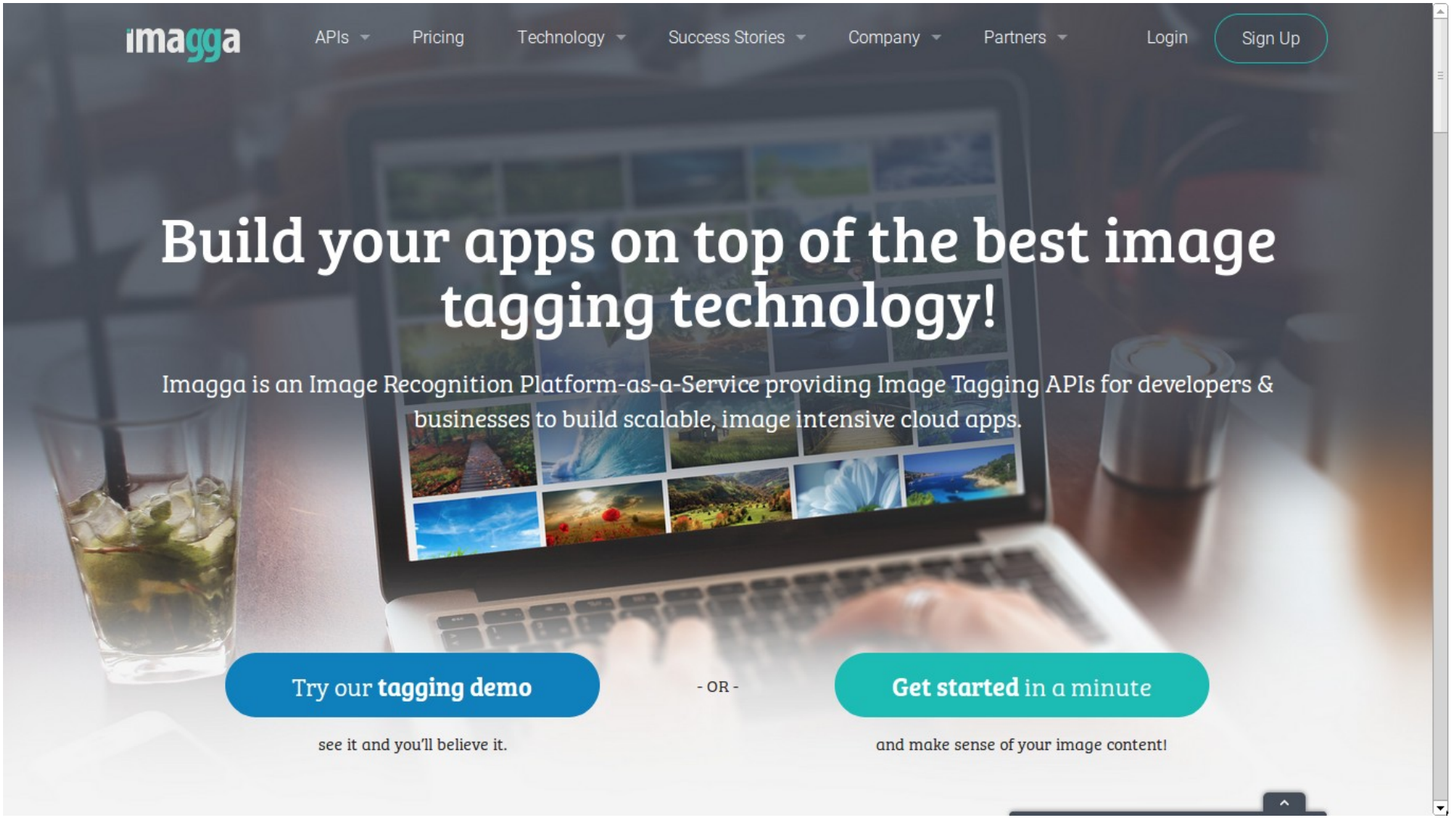
更強大的看圖說故事



CNN : Convolutional Neural Network(捲積式類神經網路)
RNN : Recurrent Neural Network(遞迴式類神經網路)

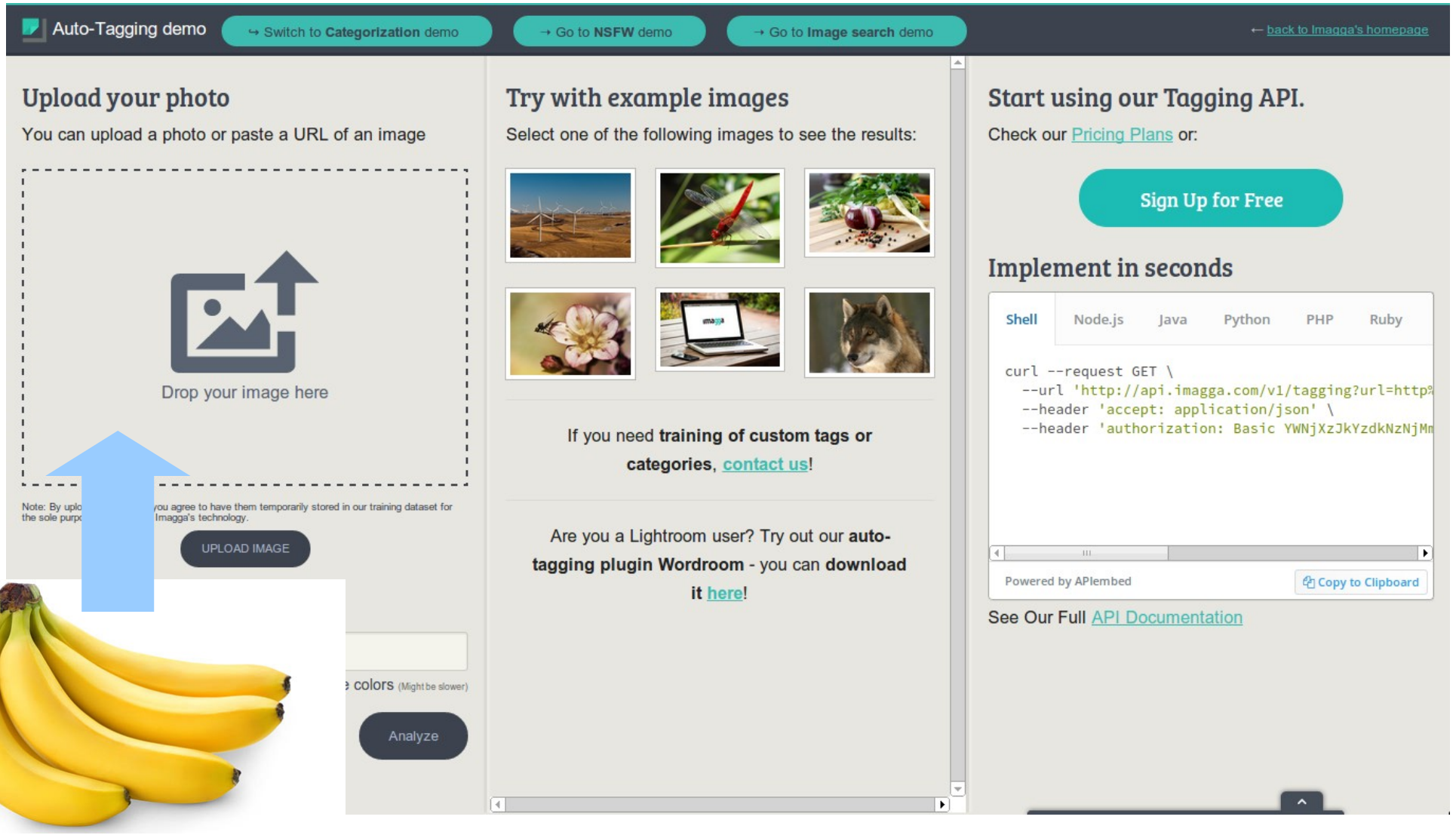
http://googleresearch.blogspot.tw/2014/11/a-picture-is-worth-thousand-coherent.html

影像分類服務



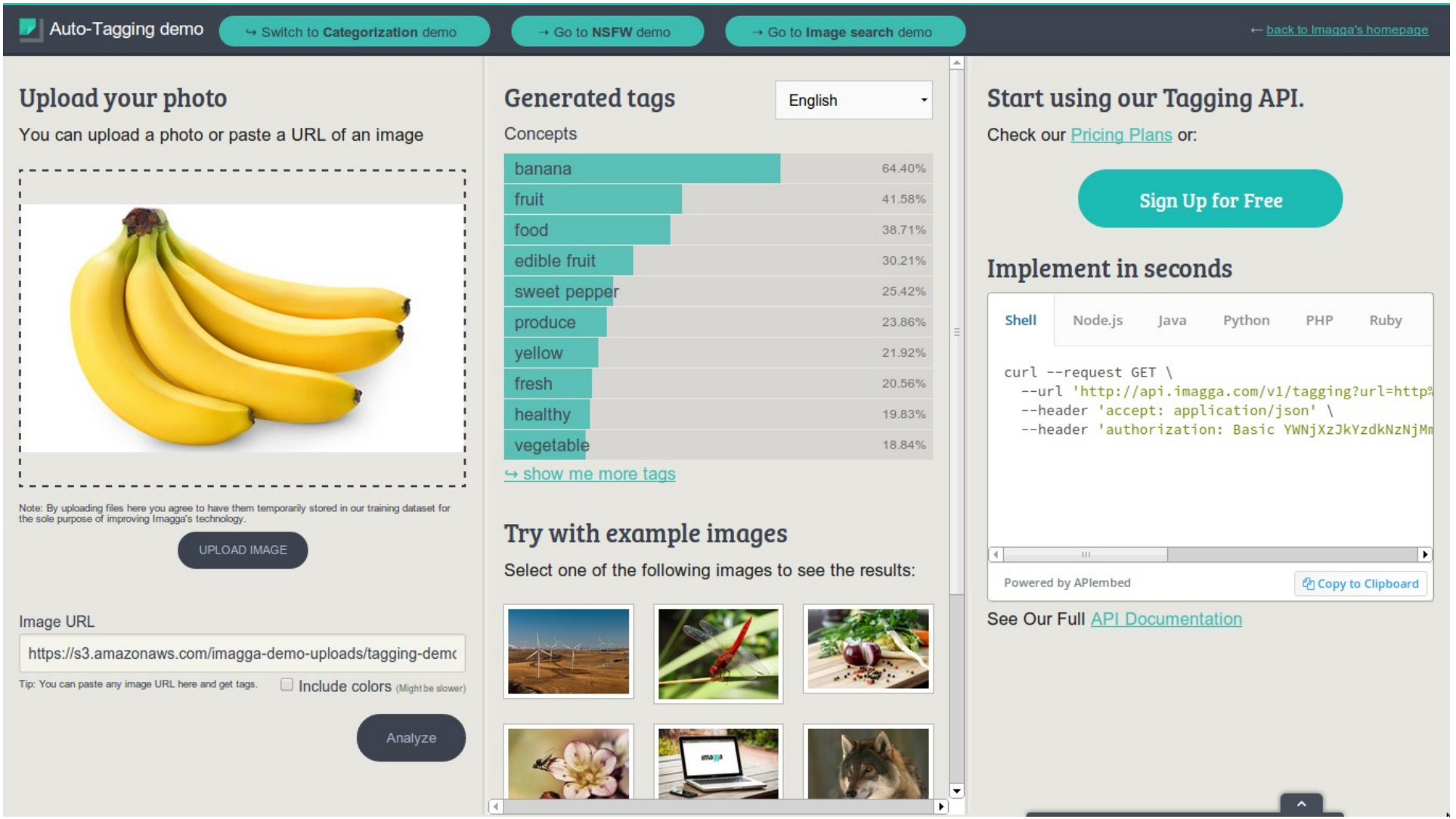
https://imagga.com/

看 DEMO



http://imagga.com/auto-tagging-demo/?key=123#

Auto-Tagging



http://imagga.com/auto-tagging-demo/?key=123#

如何開始使用服務？

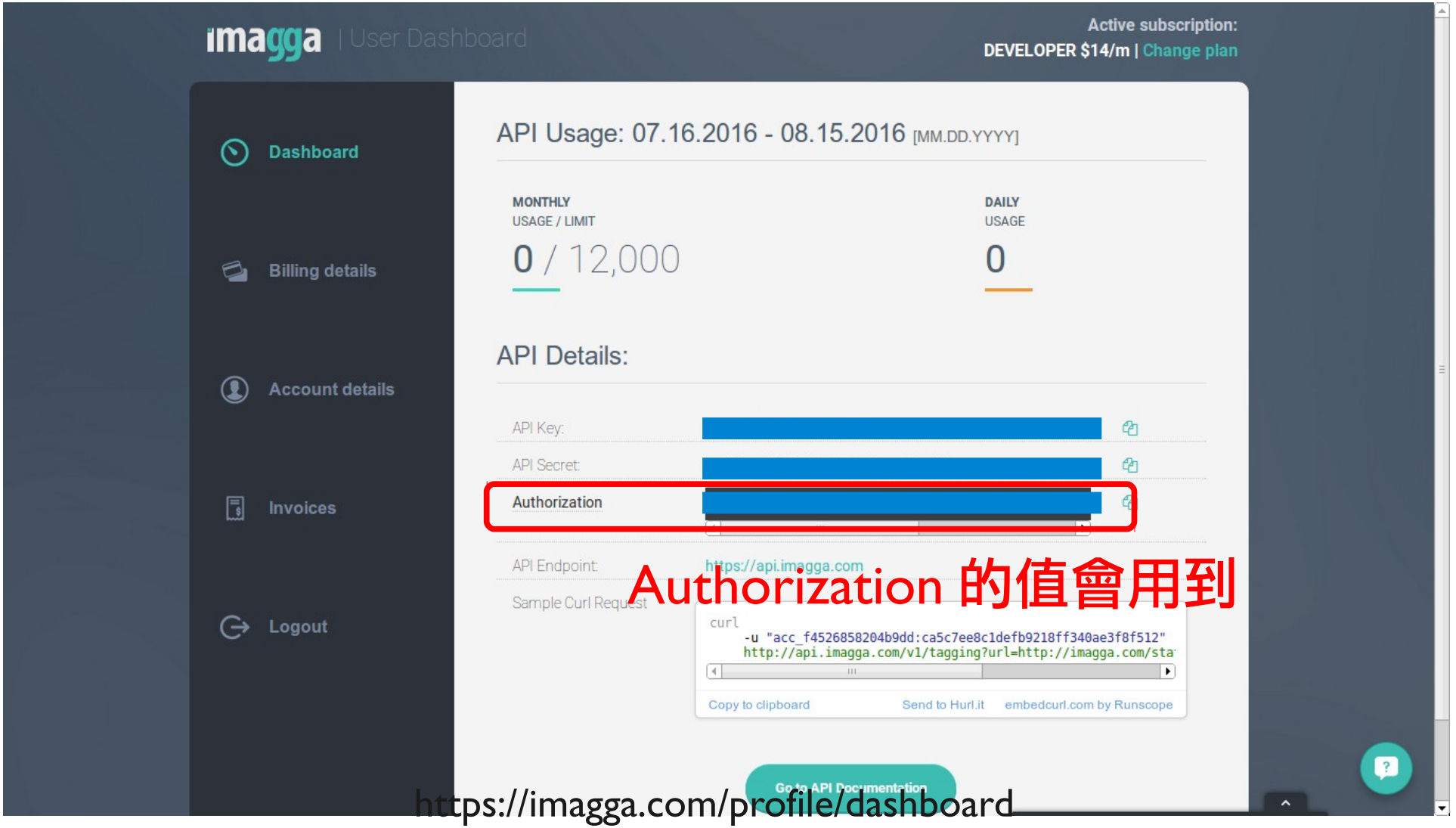
1. 註冊與認證
- https://imagga.com/auth/signup

2. 取得 Authorization
- https://imagga.com/profile/dashboard

3. 串接
- input: 程式指定圖片的 URL 或是上傳圖檔
- output: JSON 字串

Dashboard

- Basic authentication 使用 authorization value



https://imagga.com/profile/dashboard

HTTP 兩三件事

GET & POST Method

- RFC 2616 / Hypertext Transfer Protocol - HTTP/1.1

- HTTP method
  - OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT

- GET 像明信片
  - 資料(query string)在URL 傳送

- POST 像信紙 + 信封
  - 資料可以在message-header
  - 也可以在message-body

Python 程式串接 (File URL)

tagging API 的 URL

```
import requests

url = "http://api.imagga.com/v1/tagging"
querystring = {"url":"網路上圖檔的URL 網址"}
headers = {
    'accept':"application/json",
    'authorization':"自己的authorization value(含basic)"
    }
response = requests.request("GET", url, headers=headers,
params=querystring)
```

回傳結果 (JSON)


```
{
    "results": [
        {
            "image": "http://playground.imagga.com/static/img/example_photo.jpg",
            "tags": [
                {
                    "confidence": 100,
                    "tag": "shore"
                },
                {
                    "confidence": 52.19003265126093,
                    "tag": "sea"
                },
                {
```


- JSON(JavaScript Object Notation)是一種資料結構
  - 物件(object)以 { } 表示
  - 鍵 / 值(collection)以 : 表示
  - 陣列(array)以 [ ] 表示

http://docs.imagga.com/

解析 JSON

```
import json

url = "http://api.imagga.com/v1/tagging"
querystring = {"url":"網路上圖檔的URL網址"}
headers = {
    'accept':"application/json",
    'authorization':自己的authorization value(含basic)"
    }
response = requests.request("GET", url,
headers=headers, params=querystring)
data = json.loads(response.text.encode("ascii"))
print data["results"][0]["tags"][0]
["tag"].encode("ascii")
```

解析 JSON


```
{
    "results": [
        {
            "image": "http://playground.imagga.com/static/img/example_photo.jpg",
            "tags": [
                {
                    "confidence": 100,
                    "tag": "shore"
                },
                {
                    "confidence": 52.19003265126093,
                    "tag": "sea"
                },
                {
```


```
data = json.loads(response.text.encode("ascii"))
print data["results"][0]["tags"][0]
["tag"].encode("ascii")
```

DEMO
tag_file_url.py

```
$ cd ~/camera-opencv/03-imagga_web_service
$ python tag_file_url.py
```

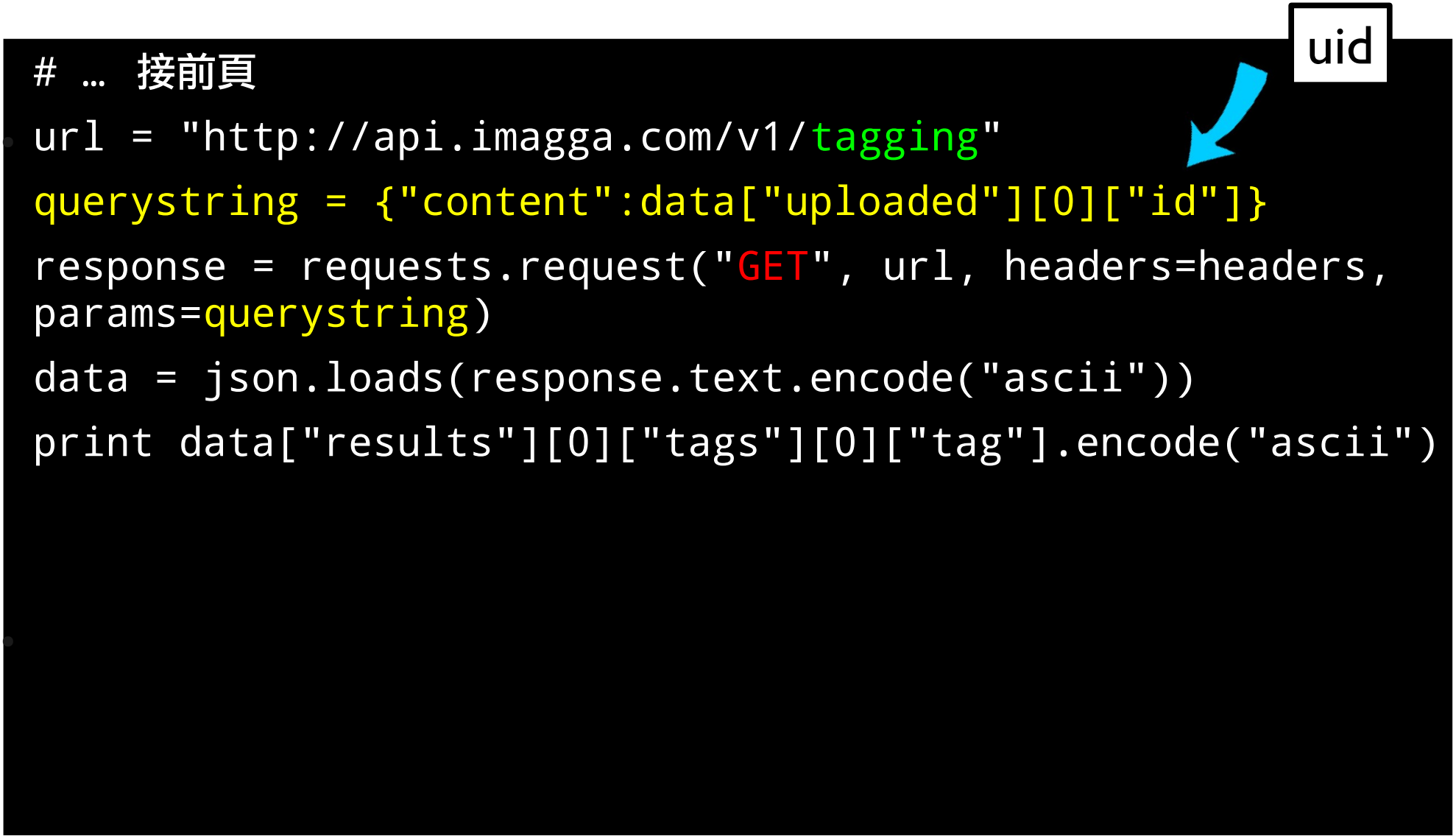
Python 程式串接 (Upload File)

- 先拍張照片
- 根據文件得知查詢上傳的檔案需要兩個步驟
  - 上傳檔案後取得檔案 uid
  - 將 uid 以參數方式送出查詢

上傳檔案取得檔案 uid

```
import requests
import json

url = “http://api.imagga.com/v1/content”
files = {“file”: open(“/home/pi/test.jpg”,”rb”)}
headers = {
    'accept':"application/json",
    'authorization':"自己的authorization value(含basic)"
    }
response = requests.post(url, files=files, headers=headers)
print(response.text)
data = json.loads(response.text.encode("ascii"))
Print data["uploaded"][0]["id"]
```

將 uid 以參數方式送出查詢



DEMO
tag_upload_file.py

```
$ cd ~/camera-opencv/03-imagga_web_service
$ python tag_upload_file.py
```

實做雲端相機

拆解功能

- 執行 python 程式後的步驟
  1. 拍照 & 存檔
  2. 將檔案上傳後取得 uid
  3. 再將 uid 以參數方式送出查詢
  4. 將查詢結果用 TTS(Text To Speech) 發聲

```
$ echo tag | festival --tts
```

由 imagga 回傳的 tag

拍照後的連續動作

```
with picamera.PiCamera() as camera:
    camera.capture("test.jpg")
    files = {"file": open("test.jpg", "rb")}

    url = "http://api.imagga.com/v1/content"
    response = …

    url = "http://api.imagga.com/v1/tagging"
    querystring = ...
    data = json.loads(...)

    obj = data[...][“tag”]
    print "<< " + obj + " >>"
    cmd = "echo " + obj + " | festival --tts"
    os.system(cmd)
```

上傳圖檔

查詢圖檔 tag

喇叭發聲 (TTS)

DEMO
smart_camera.py

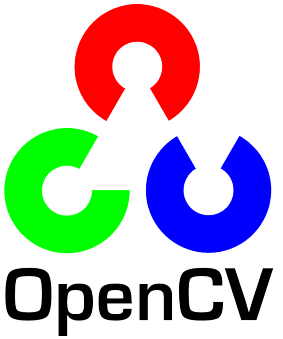
```
$ cd ~/camera-opencv/03-imagga_web_service
$ python smart_camera.py
```

實驗 4: 色彩空間轉換

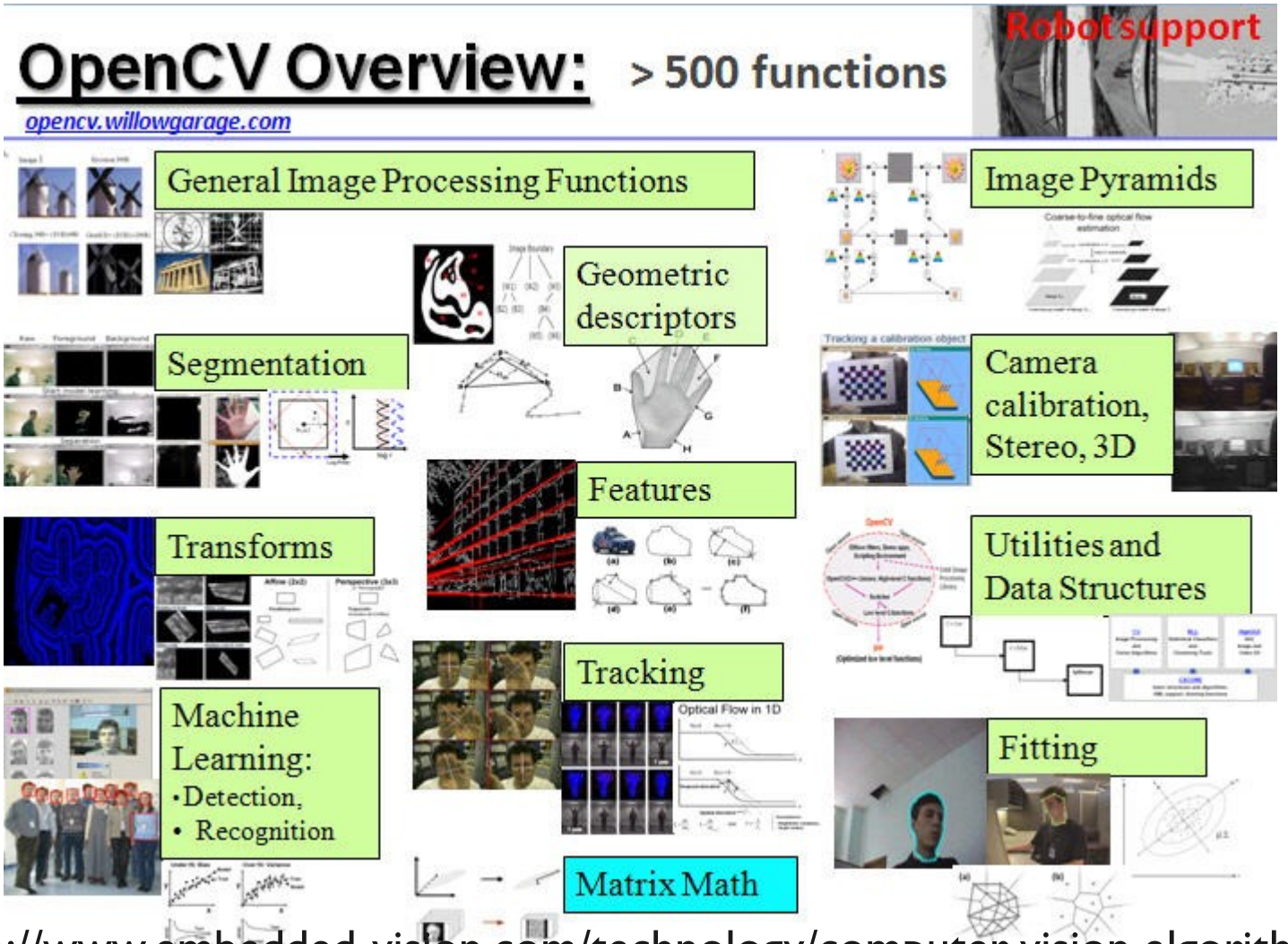
目的 : 數位影像處理入門

OpenCV

- Open Source Computer Vision Library

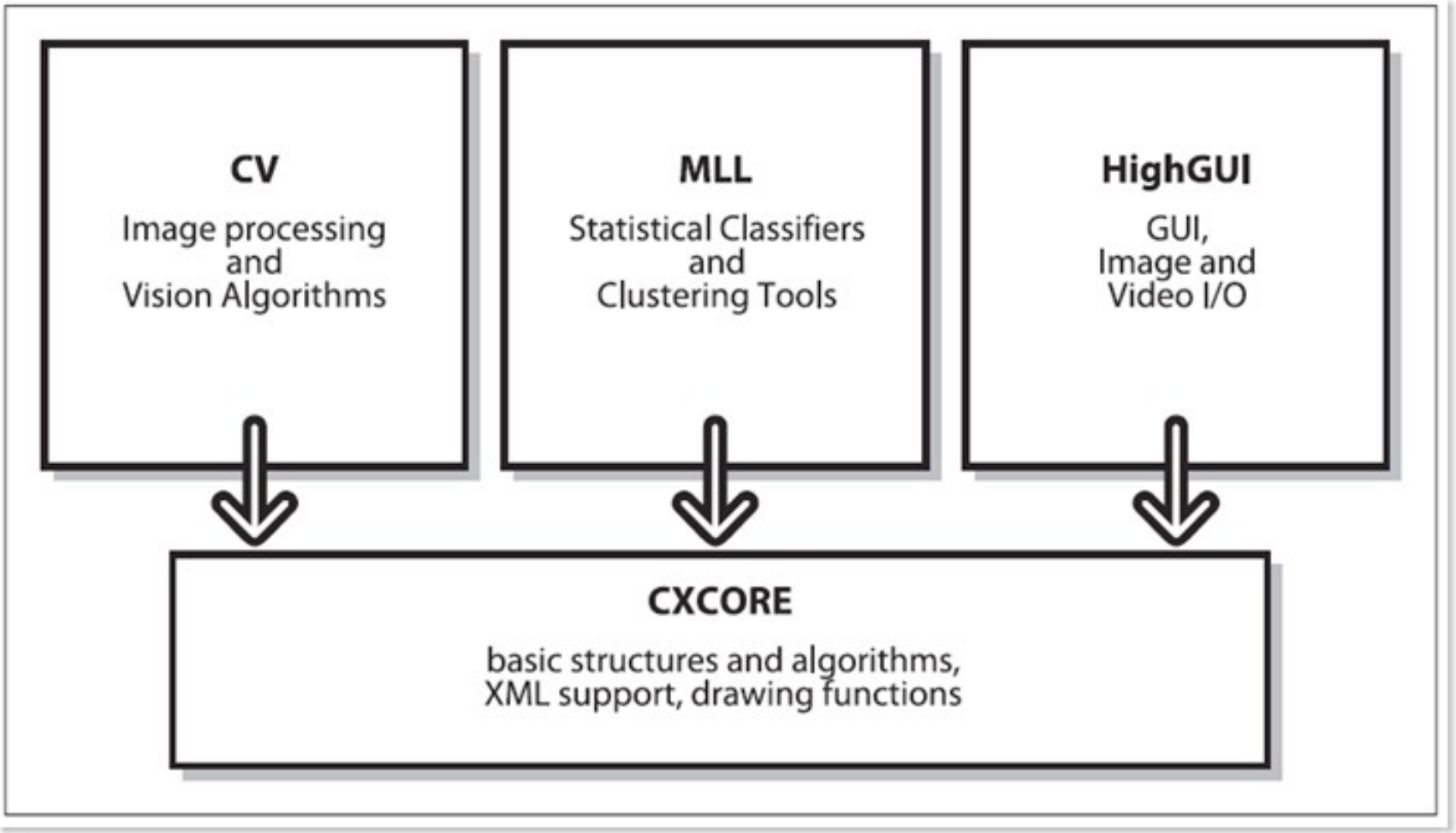


- 跨平台的計算機函式庫，主要由 C/C++ 撰寫



http://www.embedded-vision.com/technology/computer-vision-algorithms

OpenCV 的架構



http://www.cse.iitk.ac.in/users/vision/dipakmj/papers/OReilly%20Learning%20OpenCV.pdf

載入圖檔並顯示

```
import cv2
import sys

imagePath = sys.argv[1]
image = cv2.imread(imagePath)

cv2.imshow("preview", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

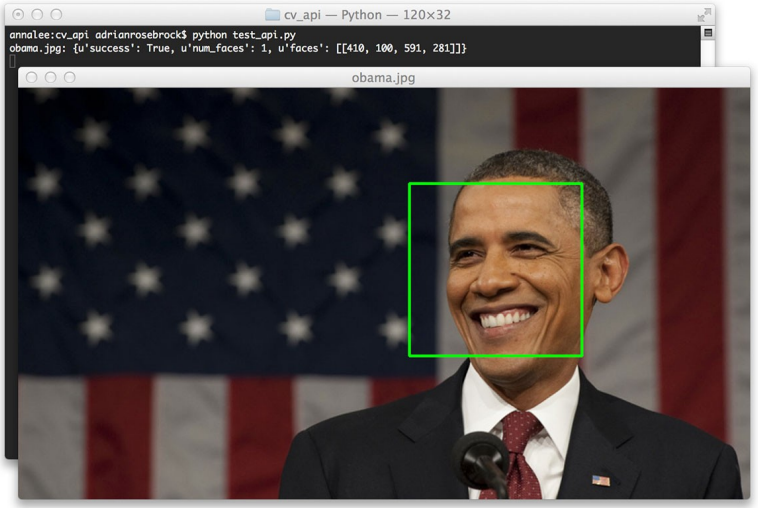
DEMO
image_load.py

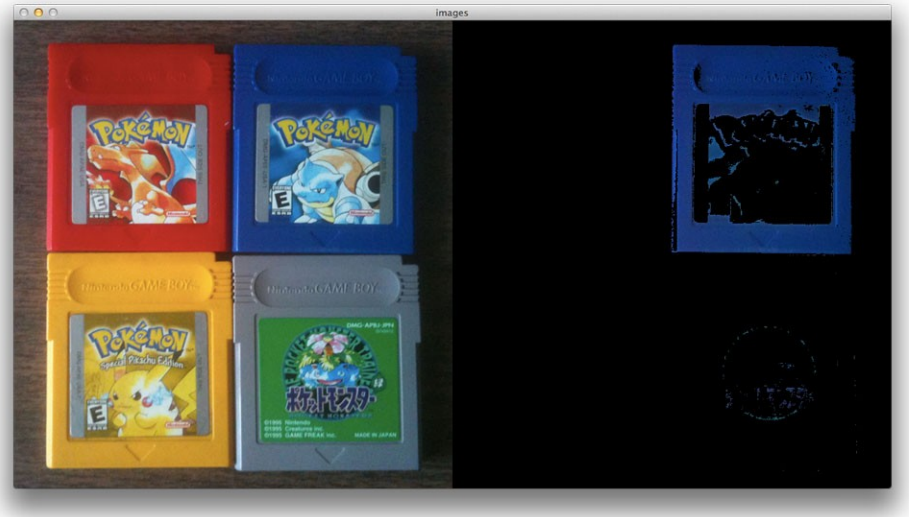
```
$ cd ~/camera-opencv/04-image_processing
$ python image_load.py  /home/pi/test.jpg
```

電腦如何分辨東西？

- 顏色
- 形狀
- 特徵

. . .

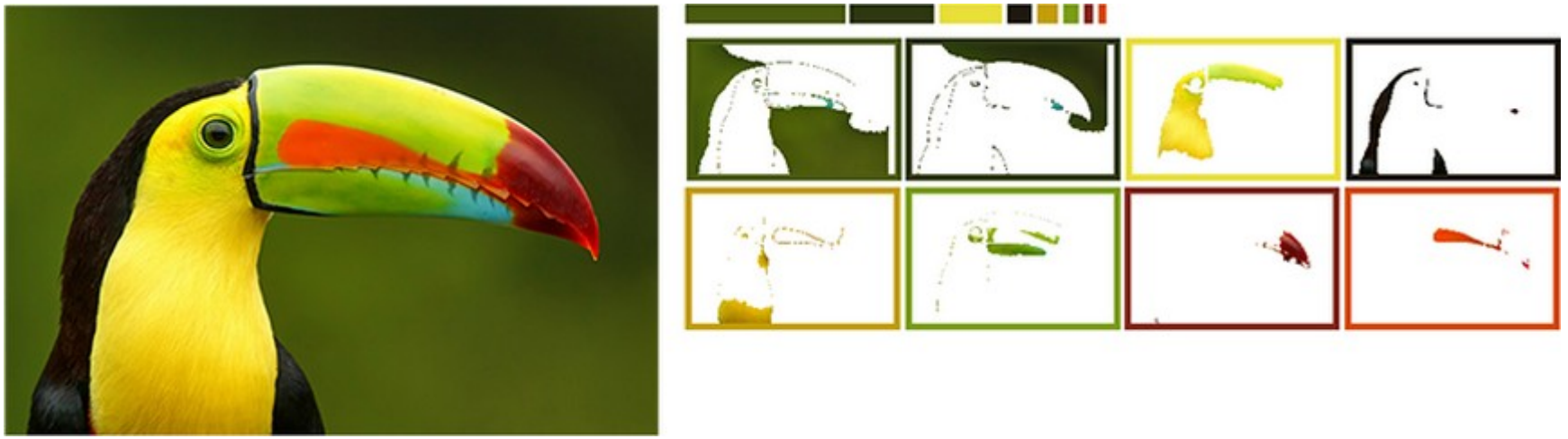




http://www.pyimagesearch.com/

影像處理的意義

- 分離與萃取資訊
- 增強或平滑訊號
- 作為分群、特徵識別等應用的前處理



http://mkweb.bcgsc.ca/color-summarizer/

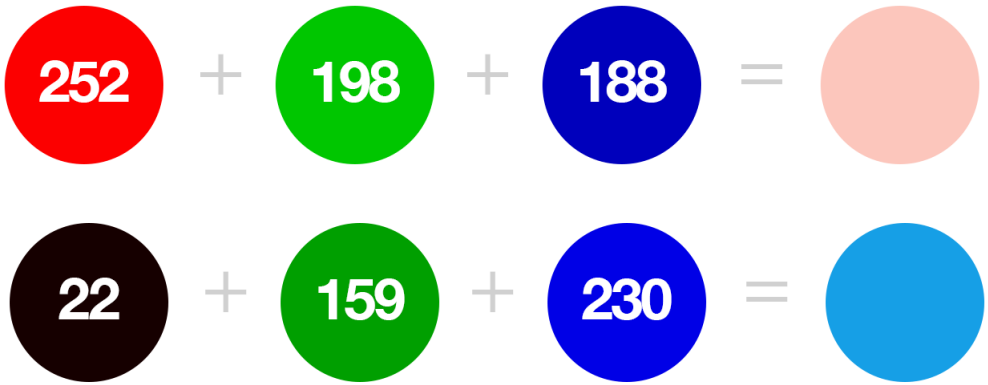
色彩空間 (Color Space)

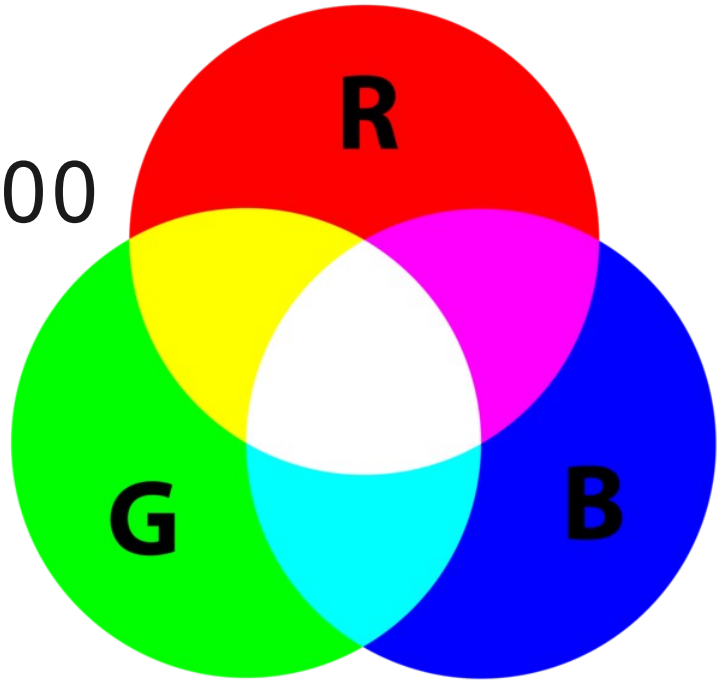
- RGB
- CMY(K)
- HSV
- YUV

RGB

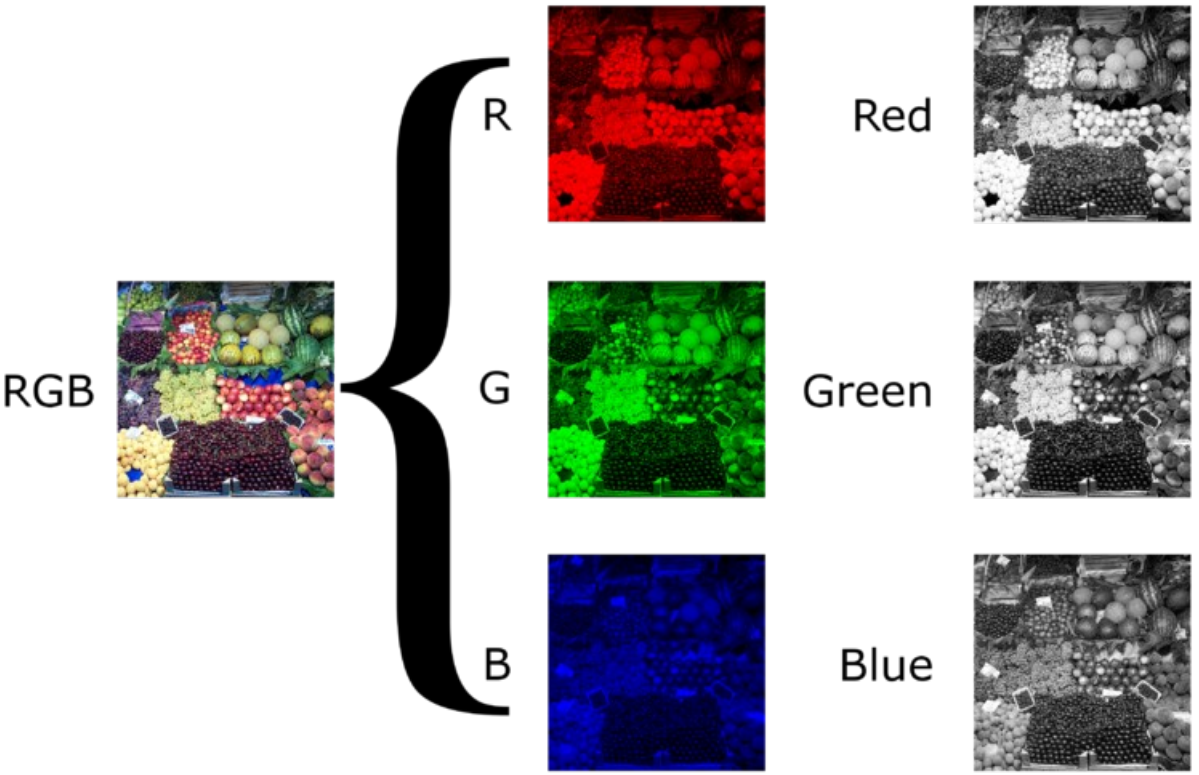
- R(Red), G(Green), B(Blue)

- R(紅),G(綠),B(藍)
- 光的三原色
- 疊加型混色的色彩模型
- 常用在電腦上表現色彩
  - 8-bit:(255,0,0),#FF0000





https://en.wikipedia.org/wiki/RGB_color_model



彩色，灰階，黑白

NUMBERS		
R 255 G 0 B 0	R 102 G 102 B 255	R 51 G 204 B 153
R 255 G 255 B 102	R 255 G 0 B 204	R 51 G 204 B 255
R 51 G 51 B 0	R 51 G 51 B 153	R 255 G 153 B 153

GRAY = 1 SET OF DIGITS		
11111111	11100110	11001101
10110100	10011011	01110011
01010000	00101000	00000000

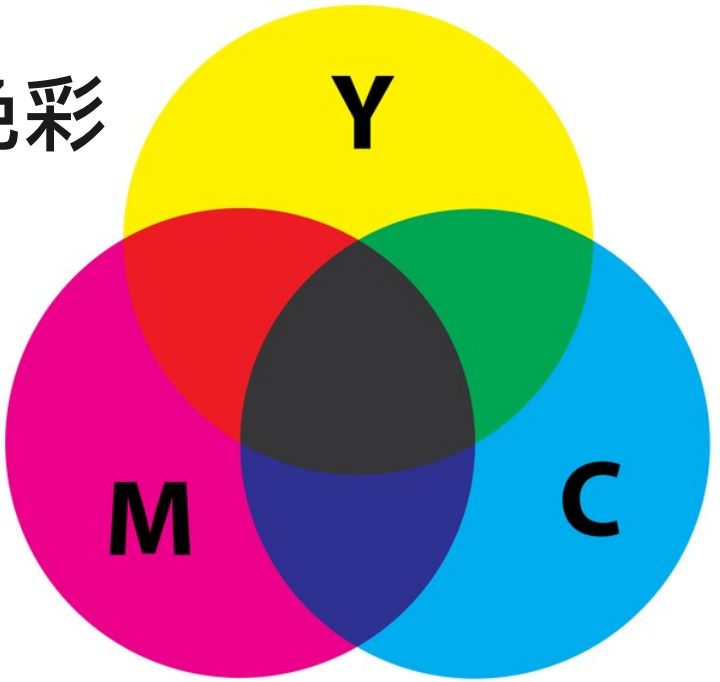
BLACK & WHITE		
0	1	0
1	0	1
0	1	0

https://en.wikipedia.org/wiki/Grayscale

CMY(K)

- C(Cyan), M(Magenta), Y(Yellow), K(Black)

- C(青), M(紫), Y(黃), K(黑)
- 彩色墨水三原色
- 削減型混色的色彩模型
- 常用在印表機 / 影印機表現色彩
- 黑色 , R=G=B=1

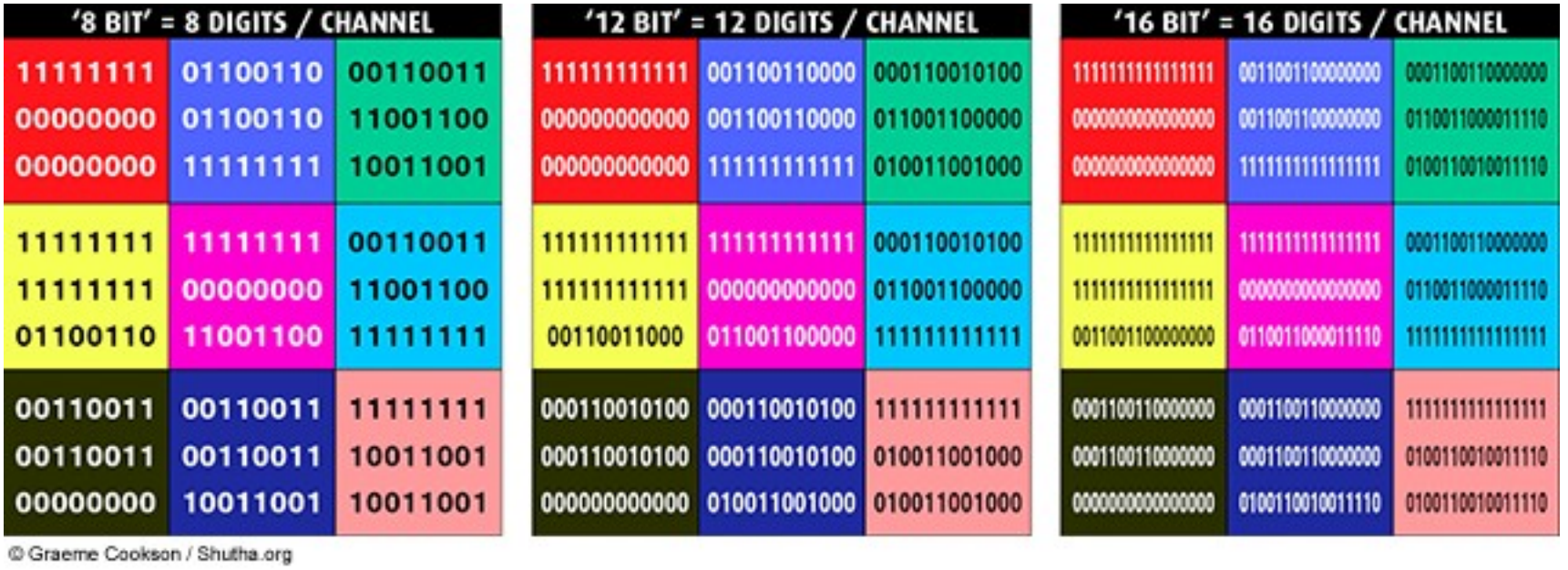


https://en.wikipedia.org/wiki/CMYK_color_model

不同色彩空間儲存的資料量不相同

http://www.shutha.org/node/789

同一色彩空間也有不同的色彩深度



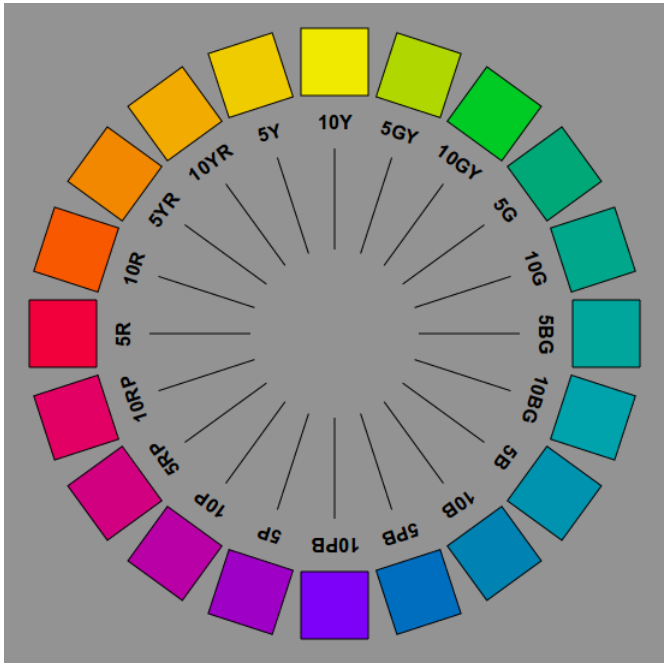
© Graeme Cookson / Shutha.org

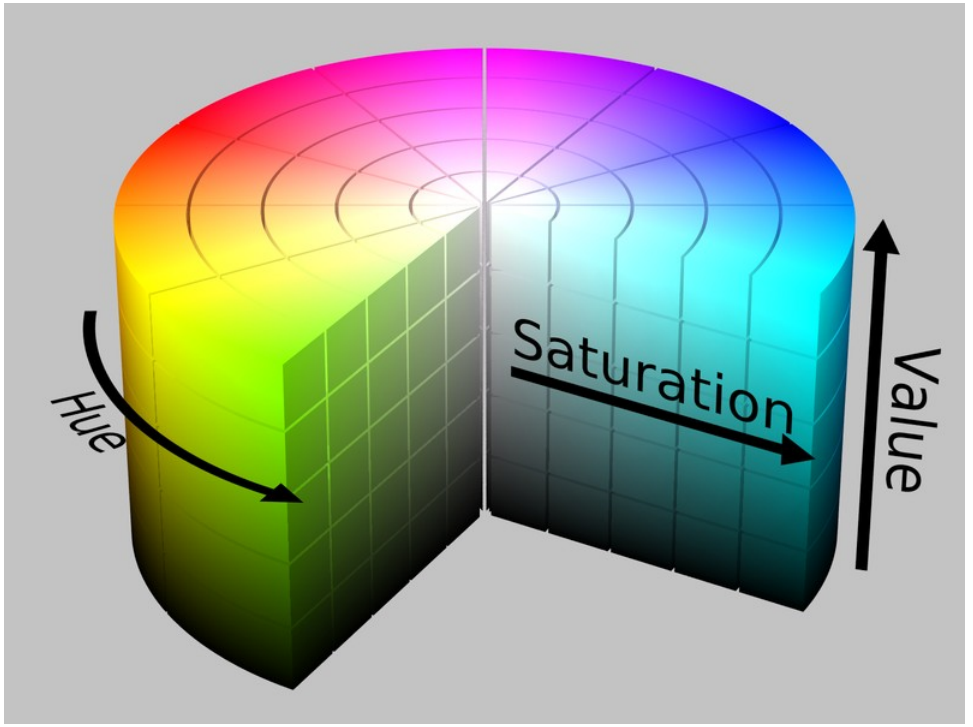
http://www.shutha.org/node/789

HSV

- H(Hue), S(Saturation), V(Value)

- H(彩度 ,0-179), S(飽和度 ,0-255), V(明度 ,0-255)
- 符合人對顏色的感知
- 常用在**數位影像處理**





https://en.wikipedia.org/wiki/HSL_and_HSV

色彩空間的轉換

- 以 RGB 為中心
  - RGB to CMY

$$\begin{pmatrix} C \\ M \\ Y \end{pmatrix} = \begin{pmatrix} 1-R \\ 1-G \\ 1-B \end{pmatrix}$$

  - CMY to RGB

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1-C \\ 1-M \\ 1-Y \end{pmatrix}$$

http://isis-data.science.uva.nl/koelma/horus/dox/horus2.0/refcpp/html/HxColConvert_8h.html

色彩空間的轉換

- 以 RGB 為中心
  - HSV to RGB

$$C = V \times S$$

$$X = C \times (1 - |(H / 60^\circ) \bmod 2 - 1|)$$

$$m = V - C$$

$$(R', G', B') = \begin{cases} (C, X, 0) & , 0^\circ \le H < 60^\circ \\ (X, C, 0) & , 60^\circ \le H < 120^\circ \\ (0, C, X) & , 120^\circ \le H < 180^\circ \\ (0, X, C) & , 180^\circ \le H < 240^\circ \\ (X, 0, C) & , 240^\circ \le H < 300^\circ \\ (C, 0, X) & , 300^\circ \le H < 360^\circ \end{cases}$$

$$(R,G,B) = ((R'+m)\times 255, (G'+m)\times 255, (B'+m)\times 255)$$

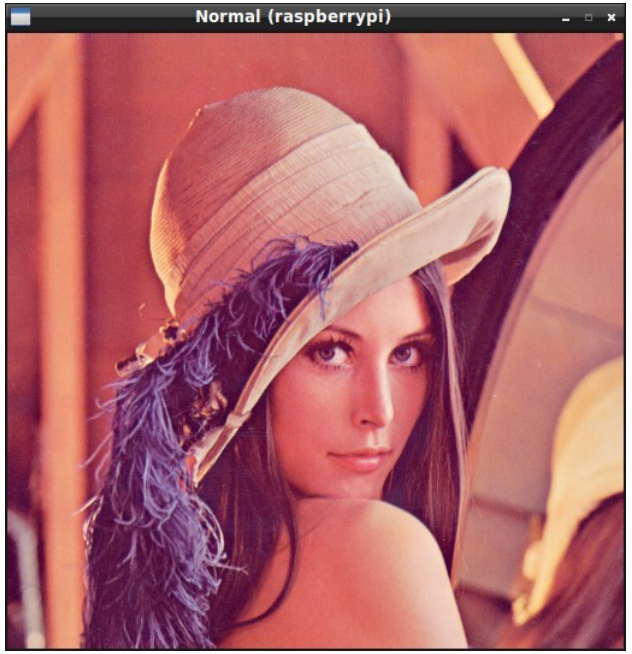
Color	Color name	(H,S,V)	Hex	(R,G,B)
	Black	(0°,0%,0%)	#000000	(0,0,0)
	White	(0°,0%,100%)	#FFFFFF	(255,255,255)
	Red	(0°,100%,100%)	#FF0000	(255,0,0)
	Lime	(120°,100%,100%)	#00FF00	(0,255,0)
	Blue	(240°,100%,100%)	#0000FF	(0,0,255)
	Yellow	(60°,100%,100%)	#FFFF00	(255,255,0)
	Cyan	(180°,100%,100%)	#00FFFF	(0,255,255)
	Magenta	(300°,100%,100%)	#FF00FF	(255,0,255)
	Silver	(0°,0%,75%)	#C0C0C0	(192,192,192)
	Gray	(0°,0%,50%)	#808080	(128,128,128)
	Maroon	(0°,100%,50%)	#800000	(128,0,0)
	Olive	(60°,100%,50%)	#808000	(128,128,0)
	Green	(120°,100%,50%)	#008000	(0,128,0)
	Purple	(300°,100%,50%)	#800080	(128,0,128)
	Teal	(180°,100%,50%)	#008080	(0,128,128)
	Navy	(240°,100%,50%)	#000080	(0,0,128)

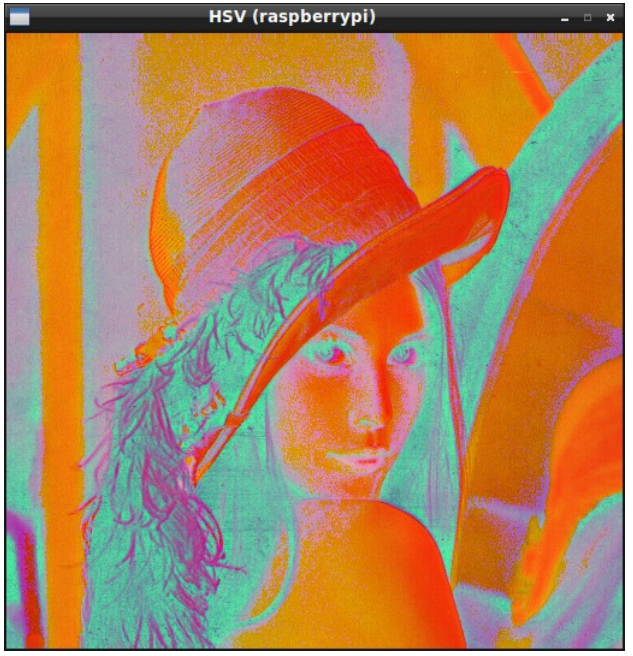
http://www.rapidtables.com/convert/color/hsv-to-rgb.htm

線上轉換工具
http://goo.gl/EV4l0z

色彩空間的轉換

- `cv2.cvtColor(img,flag)`
  - RGB 轉灰階 ,`flag = cv2.COLOR_BGR2GRAY`
  - RGB 轉 HSV,`flag = cv2.COLOR_BGR2HSV`
  - HSV 轉 RGB,`flag = cv2.COLOR_HSV2BGR`





http://docs.opencv.org/3.0-beta/modules/imgproc/doc/miscellaneous_transformations.html

DEMO
color_space.py

$ cd ~/camera-opencv/04-image_processing

$ python color_space.py lena512rgb.png

除了看圖也看值

- 如何看 RGB 的綠色轉成 HSV 是多少？
- $ python

```
>>> import cv2
>>> import numpy as np
>>> green = np.uint8([[[0,255,0 ]]])
>>> hsv_green = cv2.cvtColor(green,
cv2.COLOR_BGR2HSV)
>>> print hsv_green
[[[ 60 255 255]]]
```

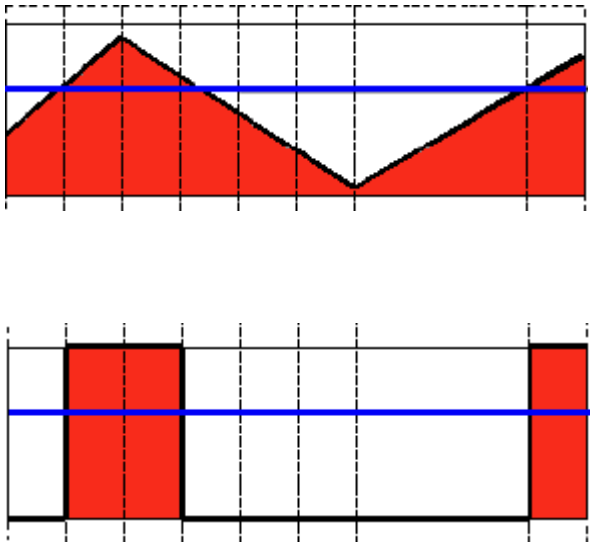
0-180

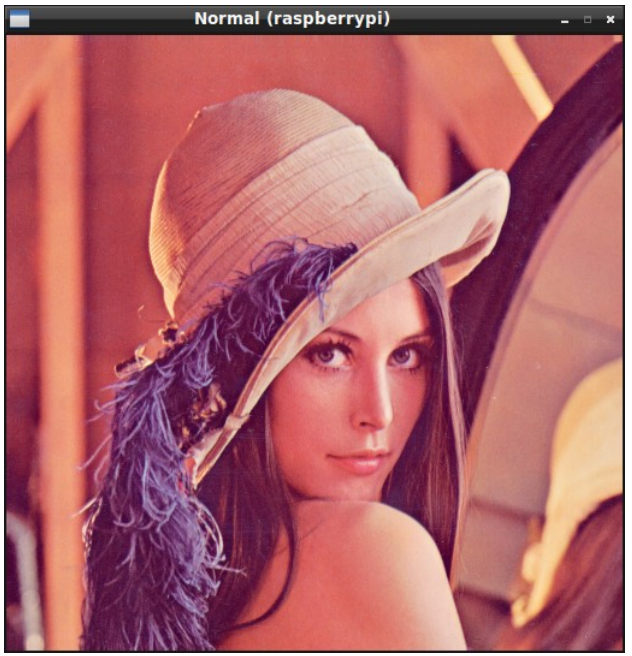
http://docs.opencv.org/3.2.0/df/d9d/tutorial_py_colorspaces.html

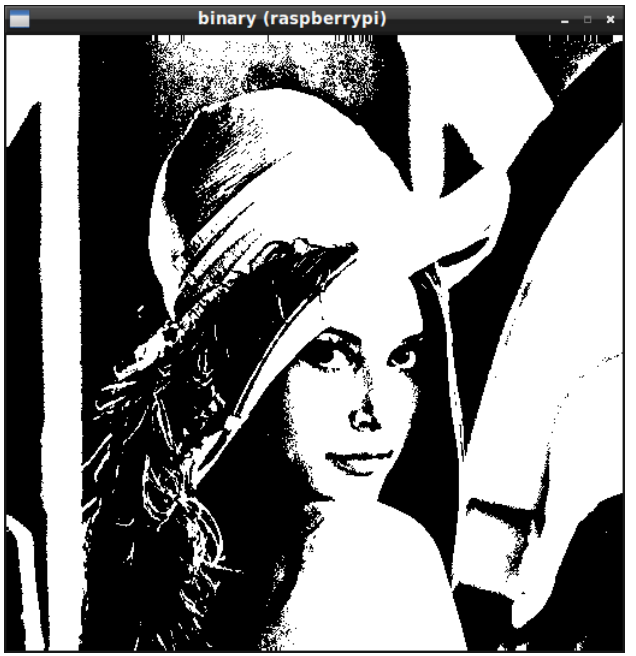
二值化

- 將影像以強度 (threshold) 區分

- `cv2.threshold(img,thresh,maxval,type)`
- 從灰階轉成黑白
  - `cv2.threshold(gray,127,255,cv2.THRESH_BINARY)`



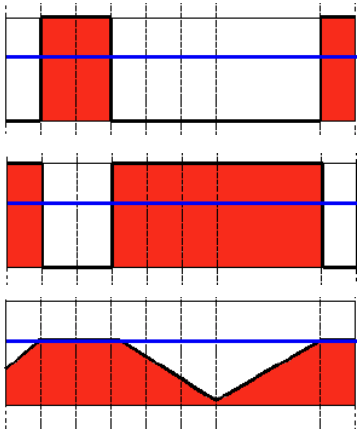


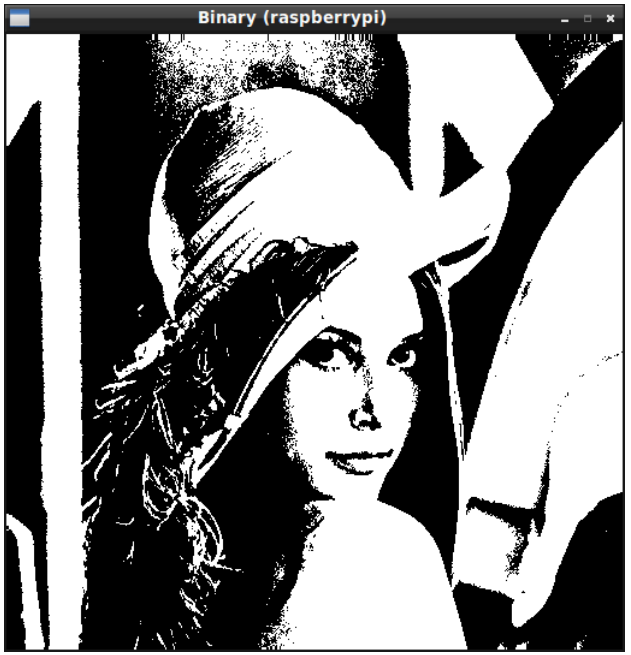


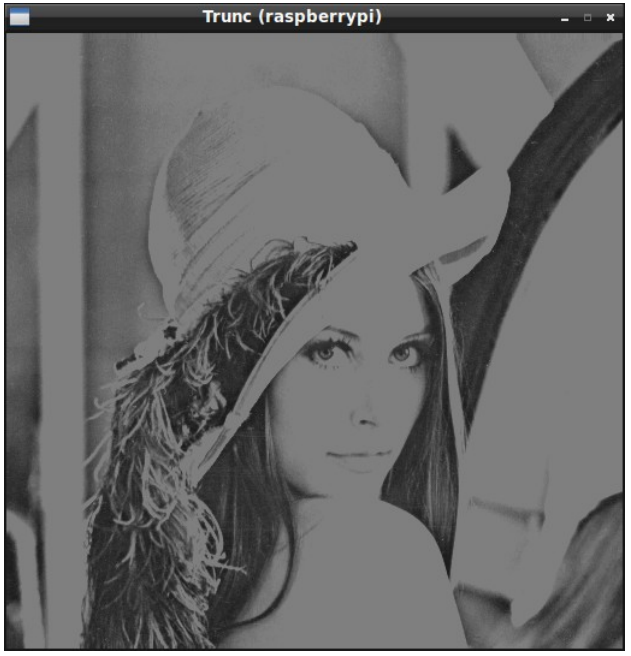
http://docs.opencv.org/3.0-beta/modules/imgproc/doc/miscellaneous_transformations.html

RGB 二值化的各種參數效果

- 轉成黑白，cv2.THRESH_BINARY
- 反向轉換，cv2.THRESH_BINARY_INV
- 超過刪除，cv2.THRESH_TRUNC





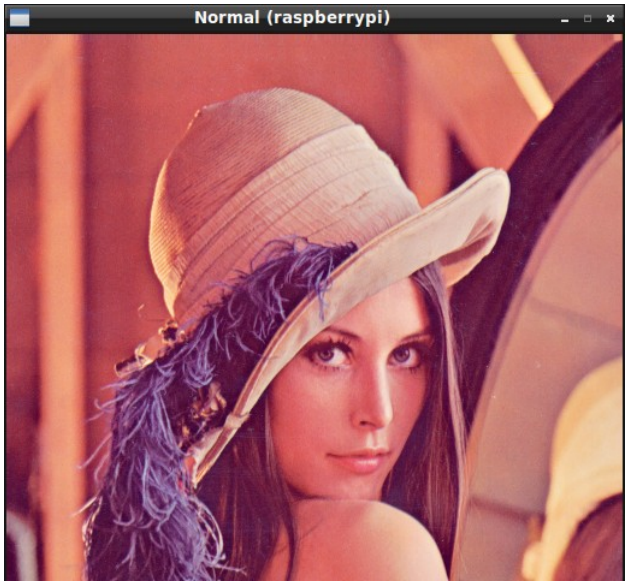


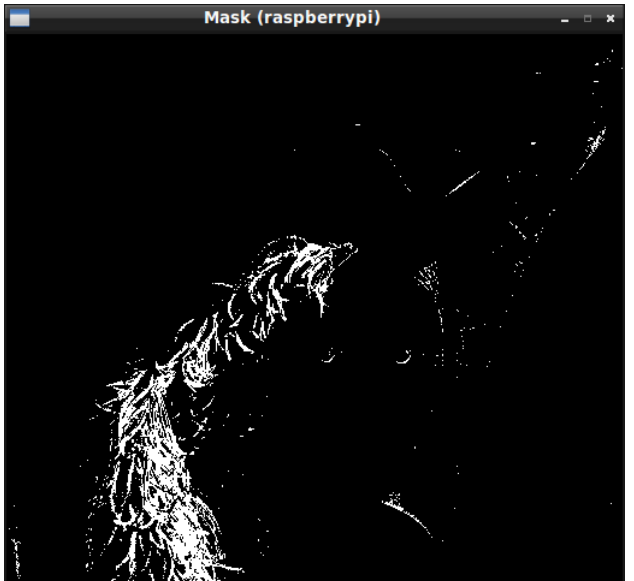
http://docs.opencv.org/trunk/d7/d4d/tutorial_py_thresholding.html

另一種二值化

- 在 HSV 空間裡，根據區間 (lower/upper) 分割

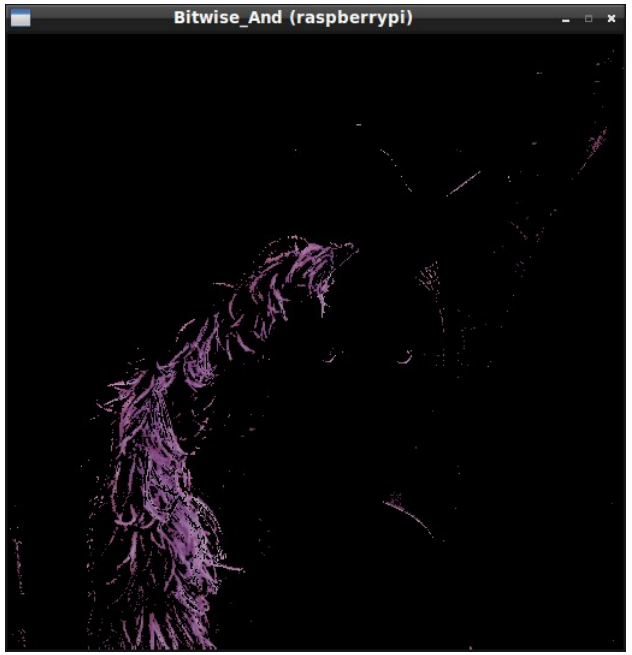
- `cv2.inRange(img,lowerval,upperval)`
- 範例：只取出紫色部份
  - `lower=np.array([141,0,0])`
  - `upper=np.array([164,145,197])`
  - `mask=cv2.inRange(hsv,lower,upper)`

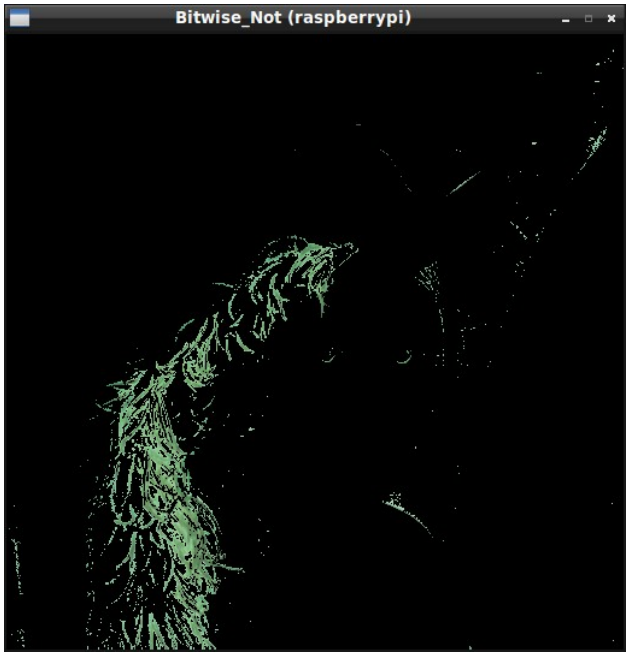




Bitwise 操作

- `cv2.bitwise_and(src1, src2, mask)`
- `cv2.bitwise_not(src, mask)`
- `cv2.bitwise_not(mask)  # 反向 mask 結果`





http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_core/py_image_arithmetics/py_image_arithmetics.html

色彩空間轉換與二值化處理

```
image = cv2.imread("lena512rgb.png")
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

(_, binary) =
cv2.threshold(gray,127,255,cv2.THRESH_BINARY)
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

lower = np.array( [141, 0, 0] )
upper = np.array( [164, 145, 197] )
mask = cv2.inRange(hsv, lower, upper)

bitwise = cv2.bitwise_and(image, image, mask=mask)
```

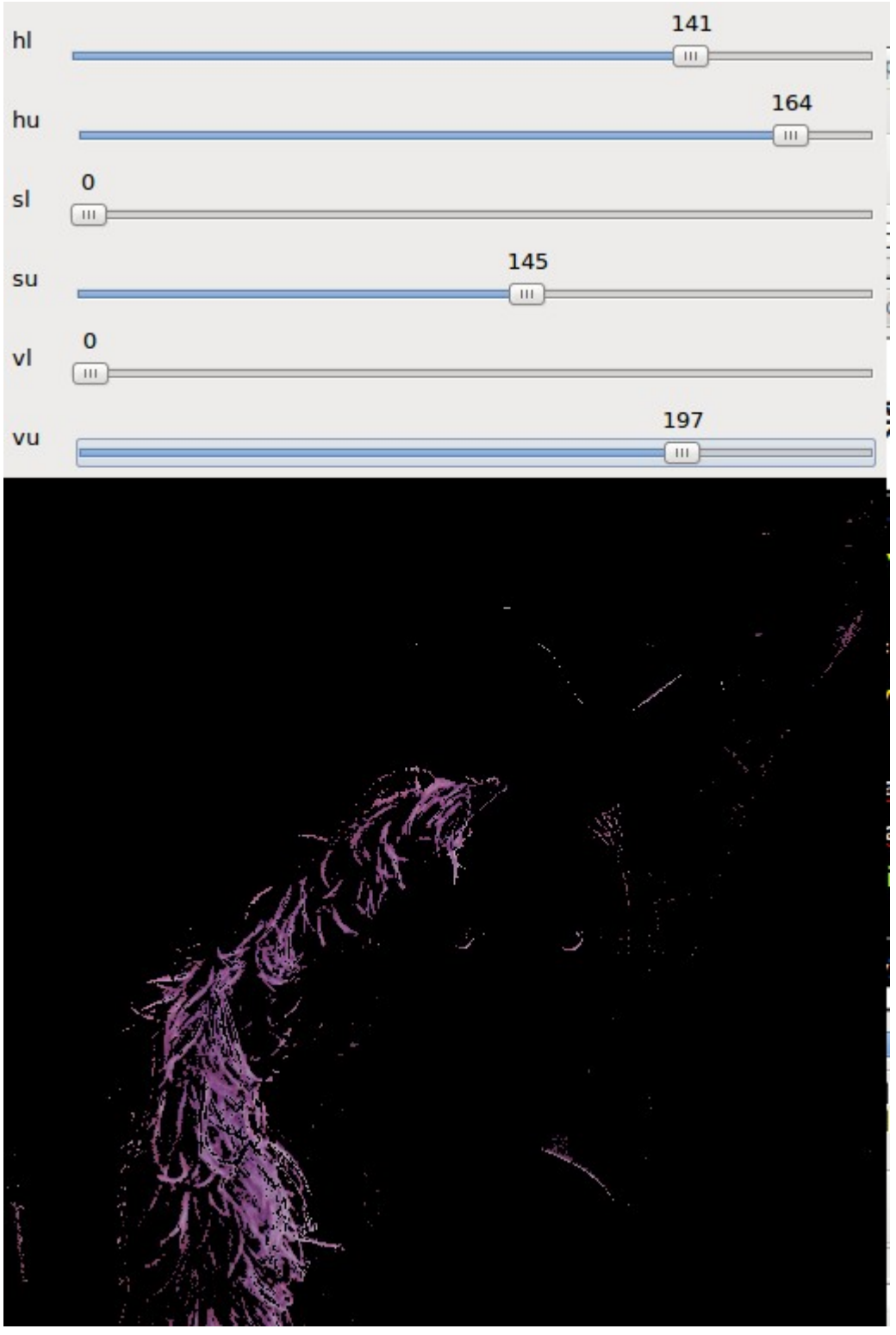
DEMO
bitwise_operate.py

$ cd ~/camera-opencv/04-image_processing

$ python bitwise_operate.py lena512rgb.png

HSV 的值？

- 即時調整



DEMO
hsv_value.py

$ cd ~/camera-opencv/04-image_processing

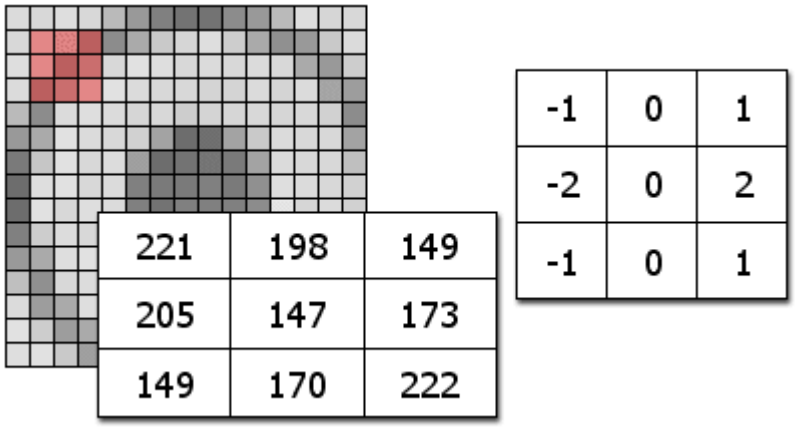
$ python hsv_value.py lena512rgb.png

影像的再處理

- 濾波器 (平滑化)
- 侵蝕 (Erode)
- 膨脹 (Dilate)

影像平滑

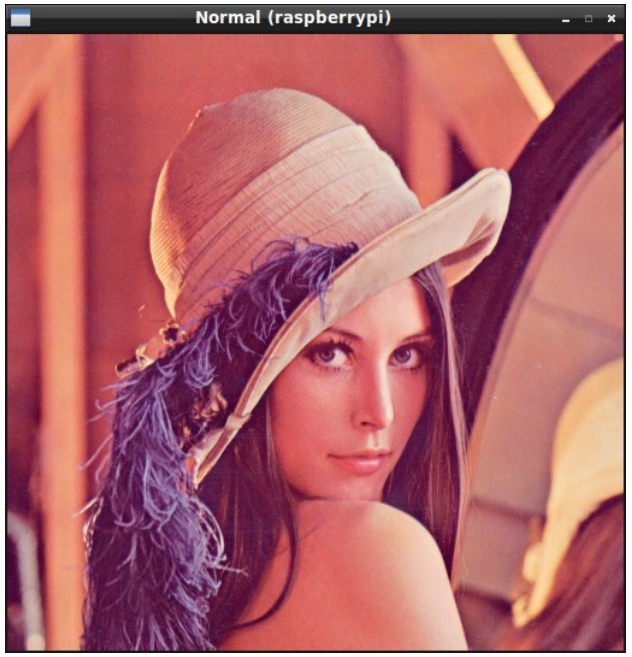
- 目的是為了去除雜訊，但對比度可能下降
- 每次要處理的像素區域稱為 Kernel



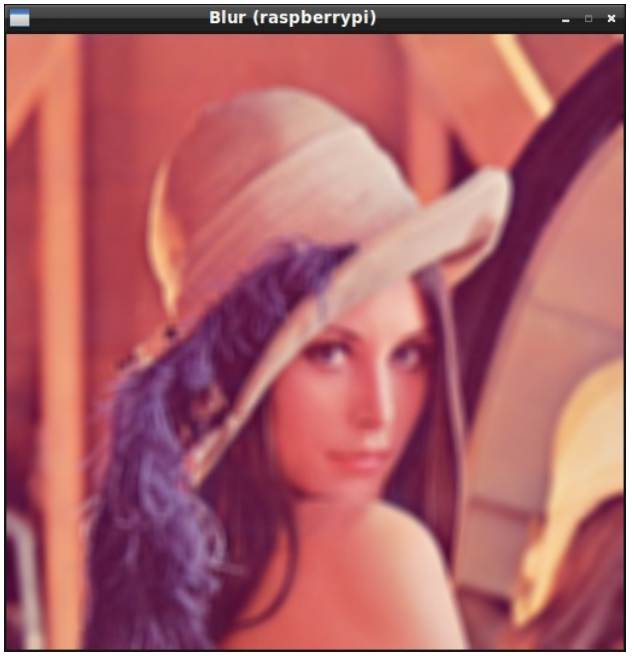
http://www.gamasutra.com/view/feature/131511/four_tricks_for_fast_blurring_in_.php

常見濾波器

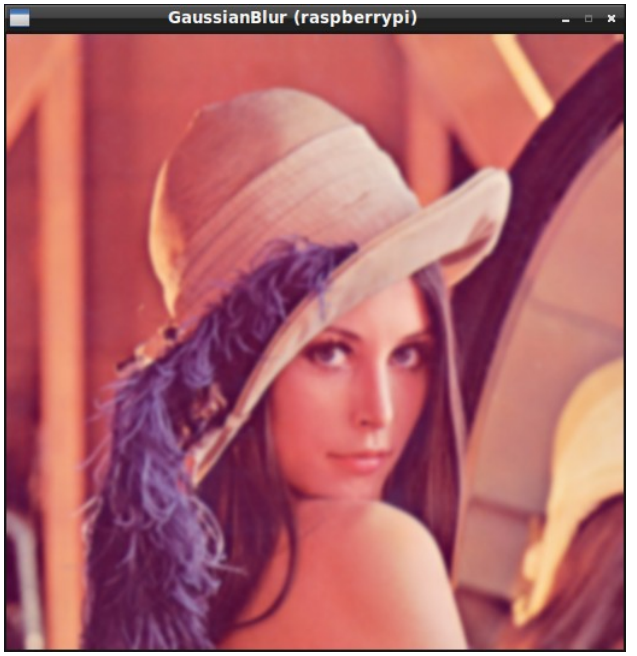
- 線性濾波
  - 平均平滑 (blur)
  - 高斯平滑 (GaussianBlur)
- 非線性濾波
  - 中值濾波 (medianBlur)
  - 雙邊濾波 (bilateralFilter)



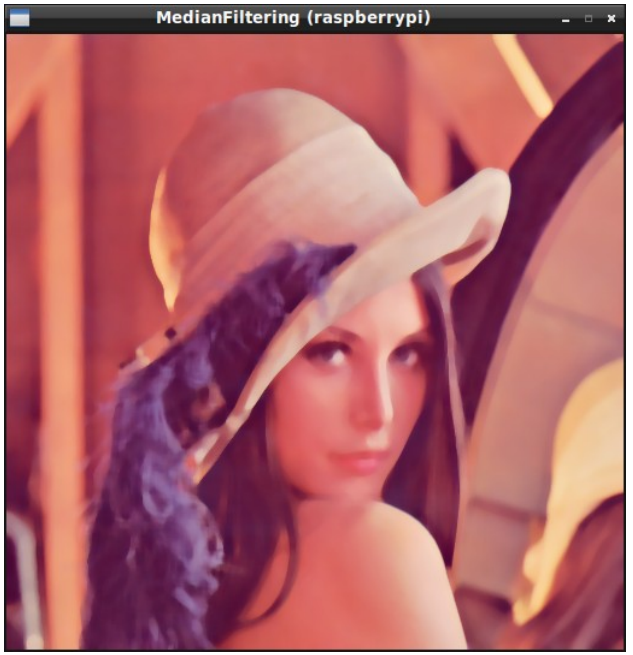
Normal



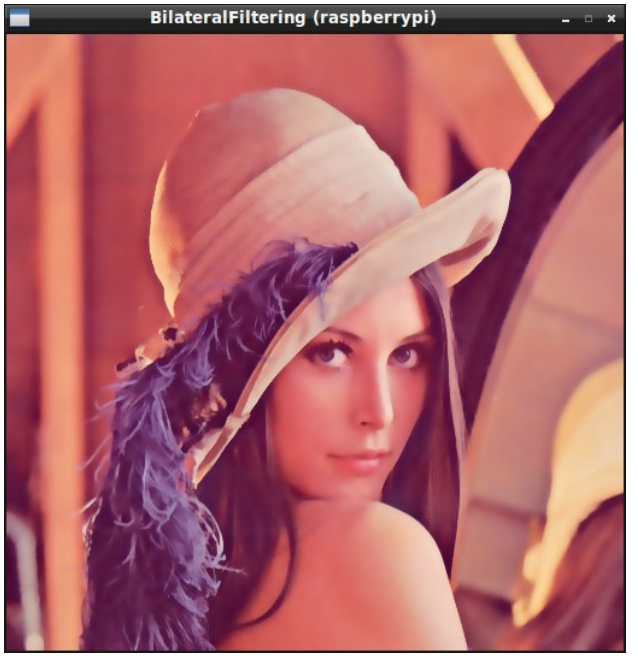
Blur



Gaussian Blur



Median Blur



Bilateral Blur

Original Image

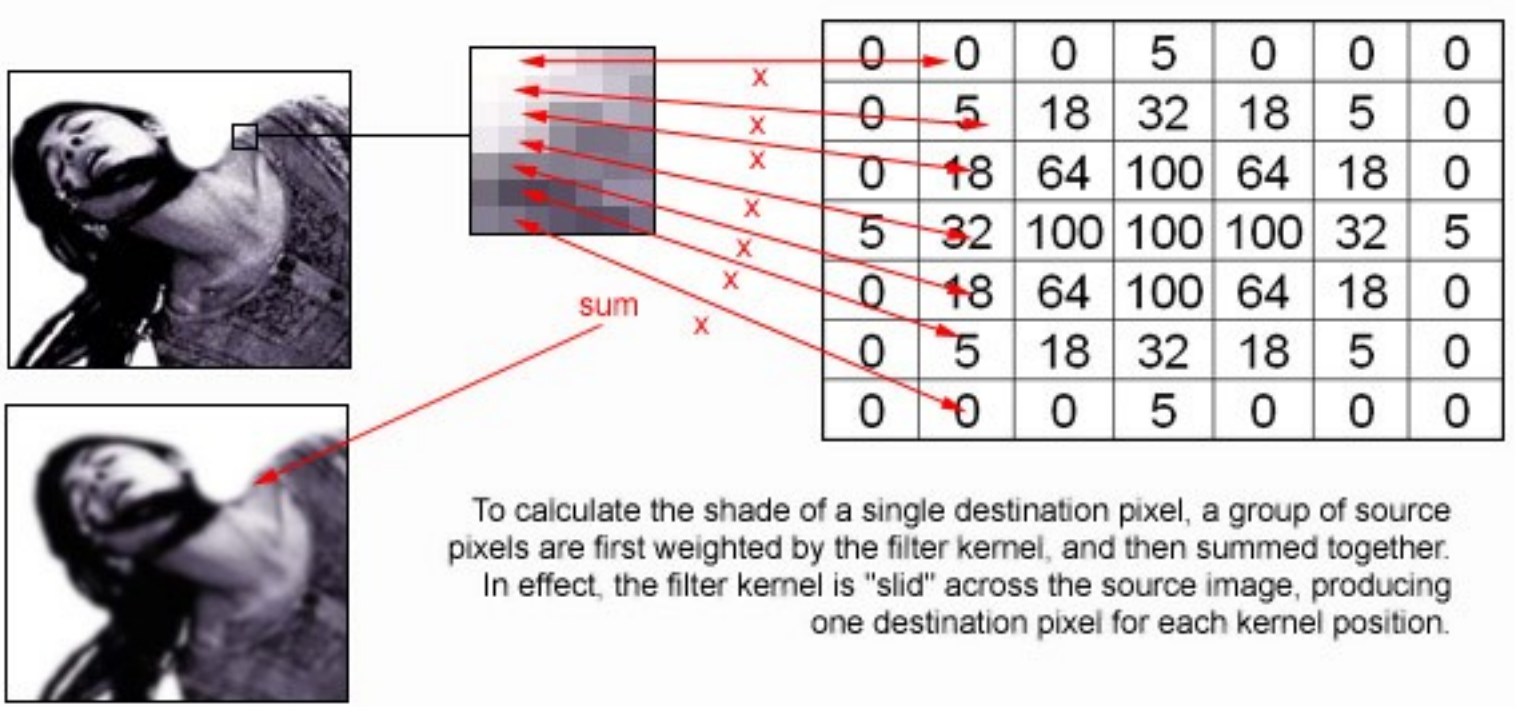
Box-filtered image

Gaussian-filtered image

Kernels used for blurring:
(note that the values shown have been scaled up to integers for clarity)

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

0	0	0	5	0	0	0
0	5	18	32	18	5	0
0	18	64	100	64	18	0
5	32	100	100	100	32	5
0	18	64	100	64	18	0
0	5	18	32	18	5	0
0	0	0	5	0	0	0



0	0	0	5	0	0	0
0	5	18	32	18	5	0
0	18	64	100	64	18	0
5	32	100	100	100	32	5
0	18	64	100	64	18	0
0	5	18	32	18	5	0
0	0	0	5	0	0	0

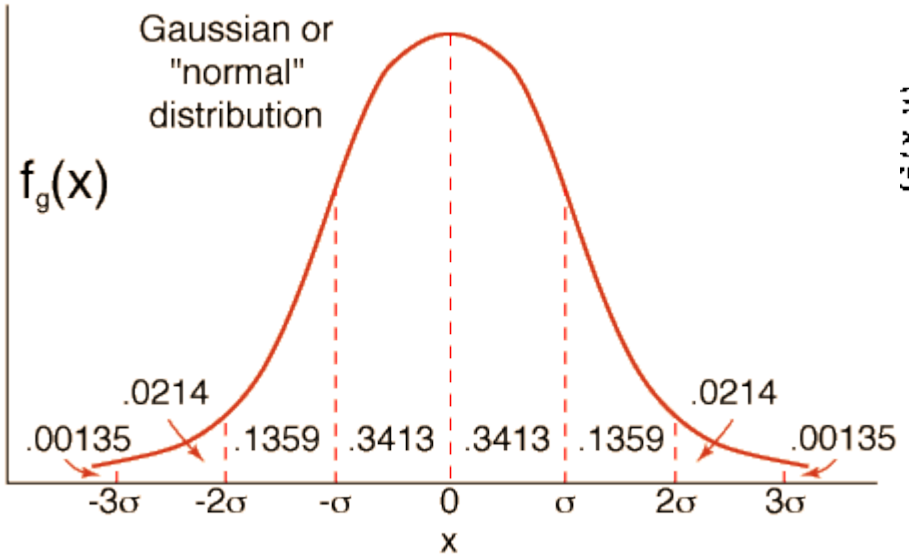
To calculate the shade of a single destination pixel, a group of source pixels are first weighted by the filter kernel, and then summed together. In effect, the filter kernel is "slid" across the source image, producing one destination pixel for each kernel position.

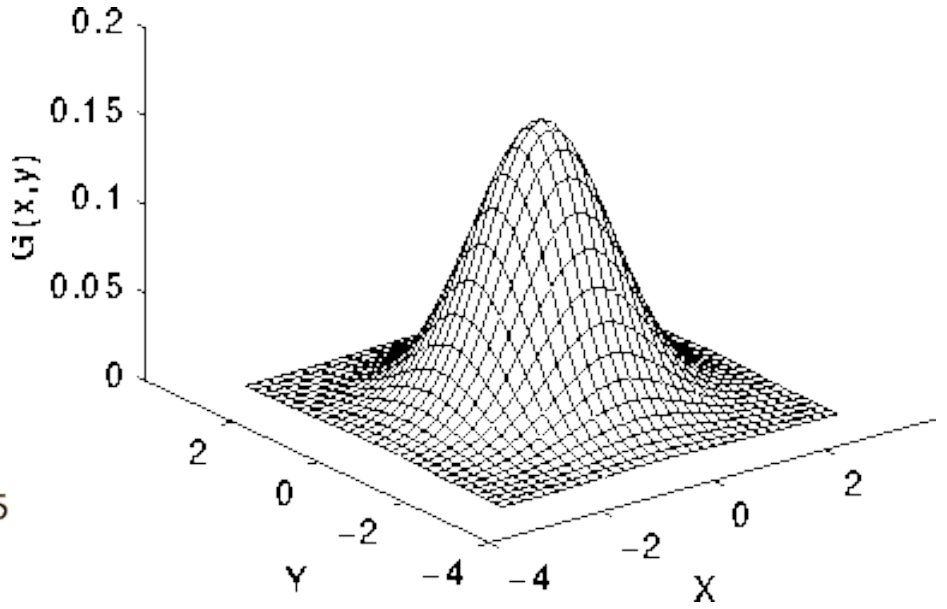
http://www.gamasutra.com/view/feature/131511/four_tricks_for_fast_blurring_in_.php

高斯平滑 (GaussianBlur)

- `cv2.GaussianBlur(src, ksize, sigmaX)`

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$





http://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm

高斯平滑效果

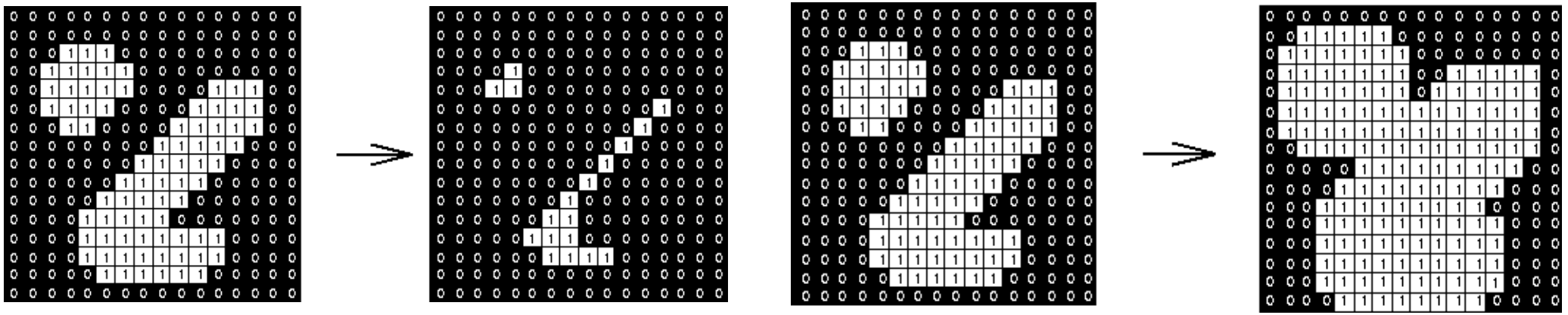
```
cv2.namedWindow('Gaussian_Blur')
cv2.createTrackbar('ksize', 'Gaussian_Blur', 0, 10, nothing)

imagePath = sys.argv[1]

while True:
    ksize  = cv2.getTrackbarPos('ksize', 'Gaussian_Blur')
    image = cv2.imread(imagePath)
    blur = cv2.GaussianBlur(image, (2*ksize+1, 2*ksize+1), 0)
    cv2.imshow('Gaussian_Blur', blur)
```

DEMO
gaussian_blur.py

```
$ cd ~/camera-opencv/04-image_processing
$ python gaussian_blur.py lena512rgb.png
```

從形態學的觀點

- 侵蝕 (Erode)
  - 消融物體的邊界
- 膨脹 (Dilate)
  - 擴大物體的邊界

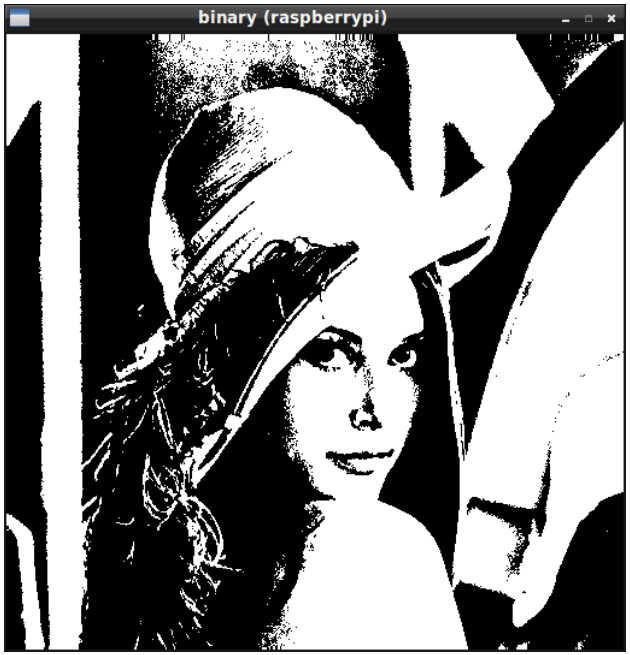


通常以奇數矩形為結構元素 (3x3, 5x5, 7x7...), 預設為 3x3

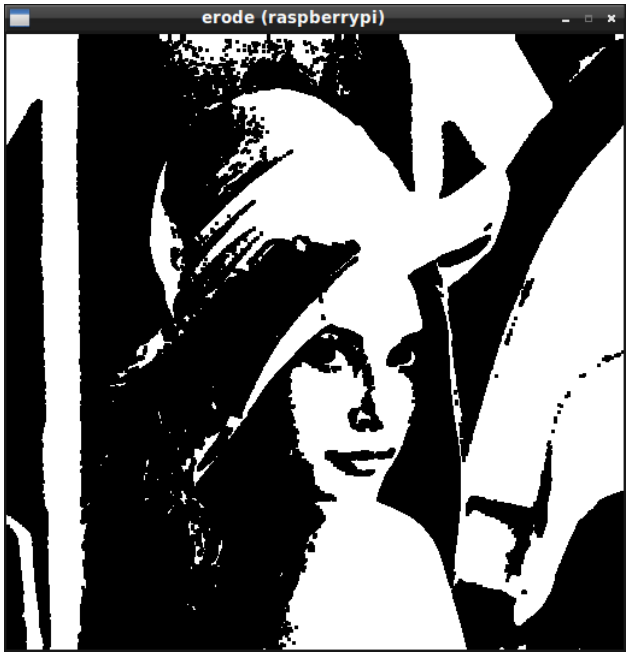
https://www.cs.auckland.ac.nz/courses/compsci773s1c/lectures/ImageProcessing-html/topic4.htm

侵蝕 (Erode)

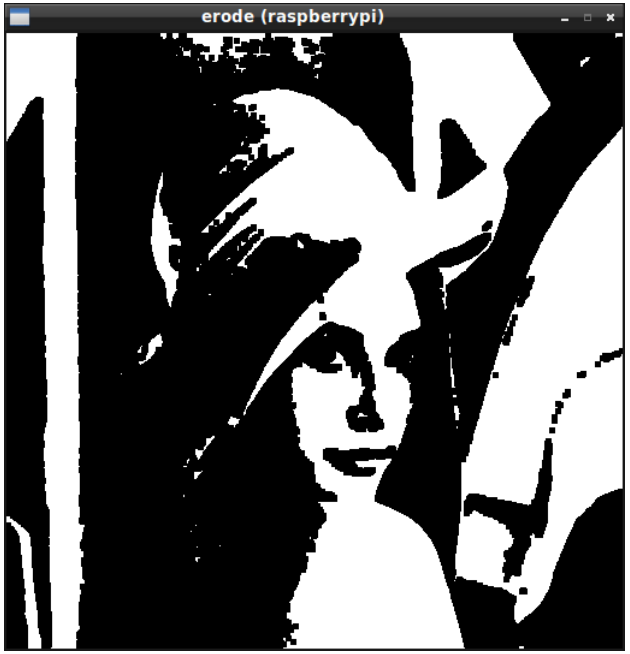
- `cv2.erode(src, kernel[, iterations])`
  - `iterations - number of times dilation is applied.`



黑白



iteration=1

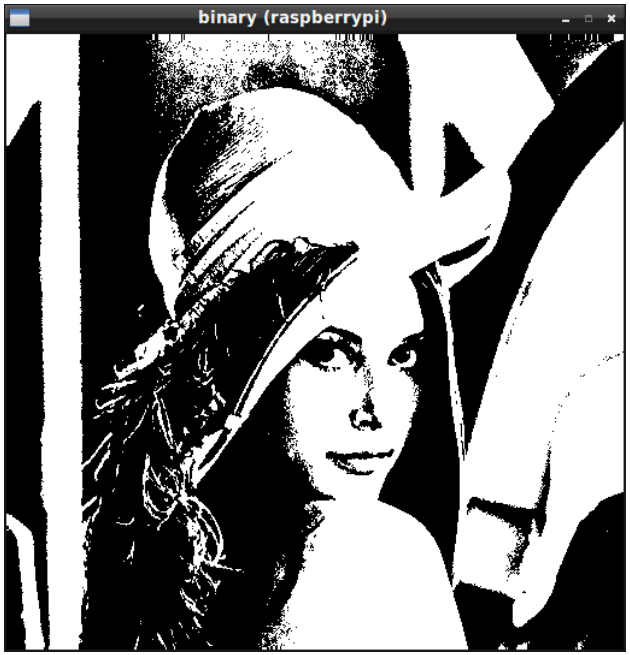


iteration=2

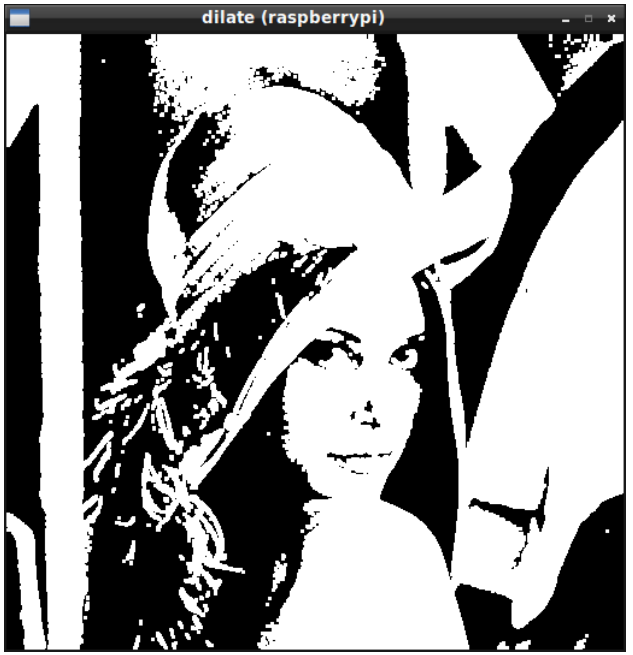
http://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html

膨脹 (Dilate)

- `cv2.dilate(src, kernel[, iterations])`
  - `iterations - number of times dilation is applied.`



黑白



iteration=1

iteration=2

http://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html

侵蝕效果

```
image = cv2.imread("lena512rgb.png")
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
(_, binary) = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)

kernel = np.ones((3,3),np.uint8)
erode  = cv2.erode(binary, kernel, iterations=1)
cv2.imshow("erode", erode)

dilate = cv2.dilate(binary, kernel, iterations=1)
cv2.imshow("dilate", dilate)
```

DEMO
erode_dilate.py

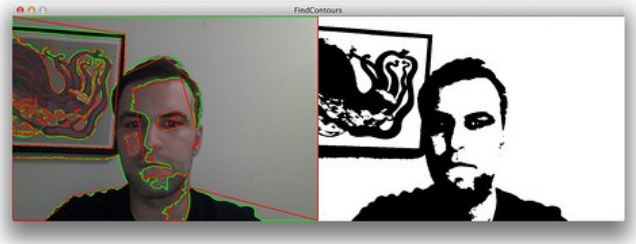
$ cd ~/camera-opencv/04-image_processing

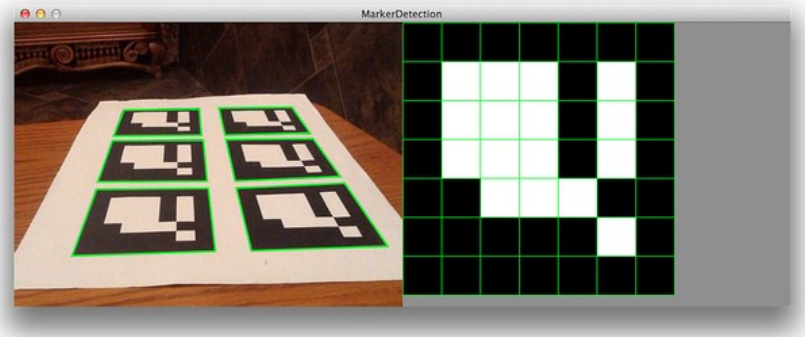
$ python erode_dilate.py lena512rgb.png

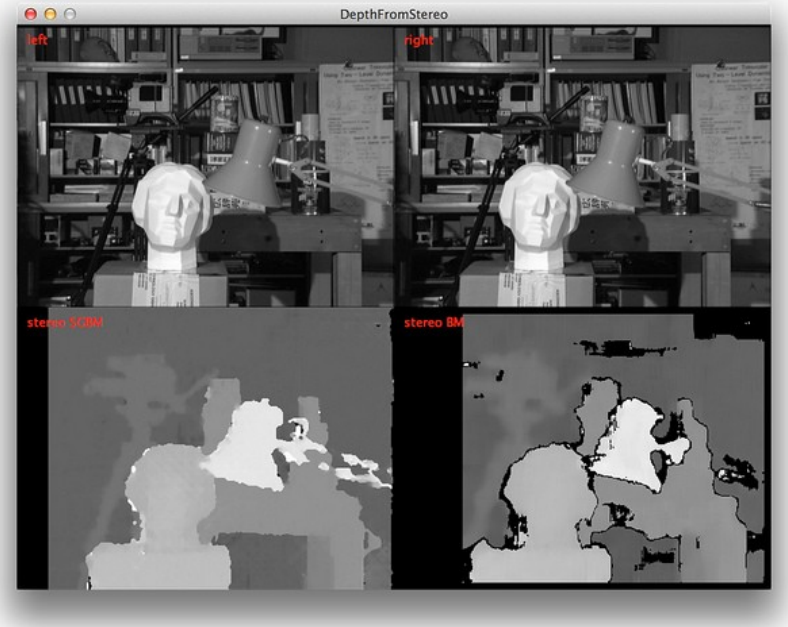
影像處理後的應用

輪廓的概念

- 輪廓就是一堆連續的點，可以將兩個不同顏色或是不同密度的點分離
- 是物體辨識或是形狀分析應用的前處理
- 使用二值化（黑白）的圖形可更精準的找出輪廓







http://urbanhonking.com/ideasfordozens/2013/07/10/announcing-opencv-for-processing/

Edge Detection

- A edge pixel is a pixel at a “boundary”
- Edge detection 是為了找出灰階有劇烈變化的邊界

http://docs.geoserver.org/stable/en/user/extensions/css/cookbook/raster.html

灰階變化 = 梯度 (gradient)

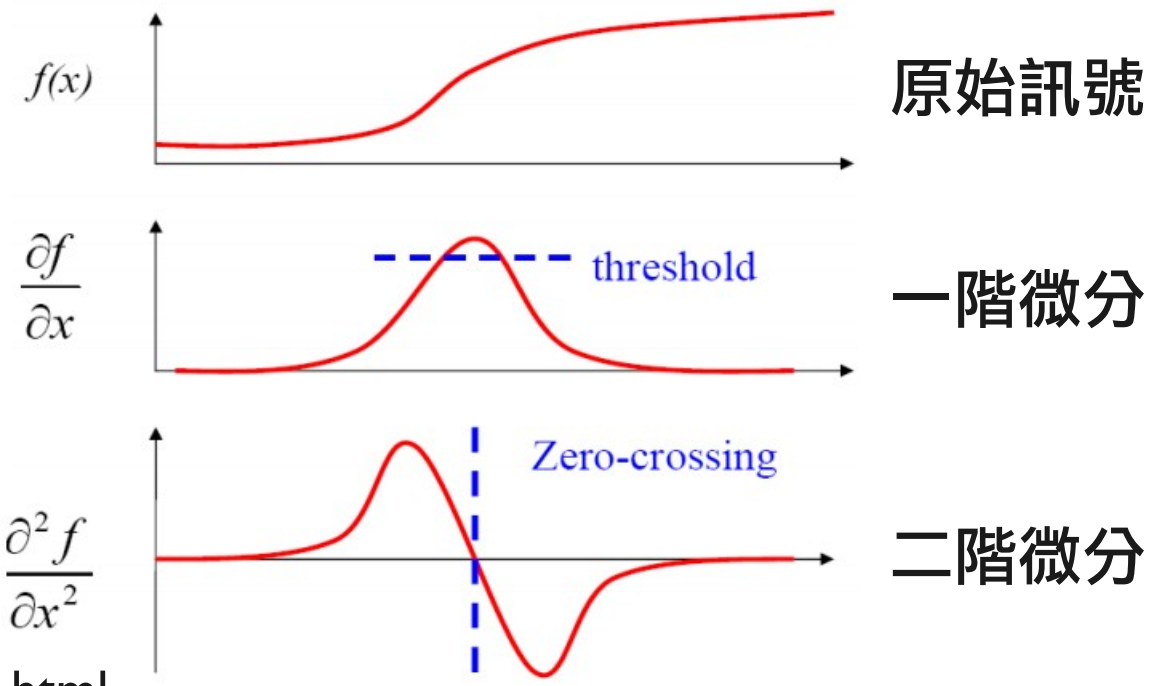
- 梯度可從一階微分 (derivative) 和二階微分求得

(a) Step edge

(b) Ramp

(c) Roof edge

(d) Real edge



https://miac.unibas.ch/SIP/07-Segmentation.html

常見邊緣檢測法

- 一階微分 (梯度法)
  - Roberts operator
  - Sobel operator
  - Prewitt operator
- 二階微分
  - Laplacian
  - Laplacian of Gaussian
  - Difference of Gaussian(DOG)
- 多級邊緣檢測
  - Canny edge detection

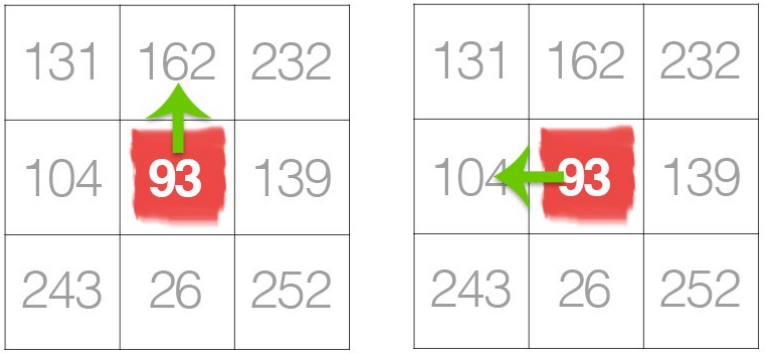
Canny 邊緣檢測 (Edge Detection)

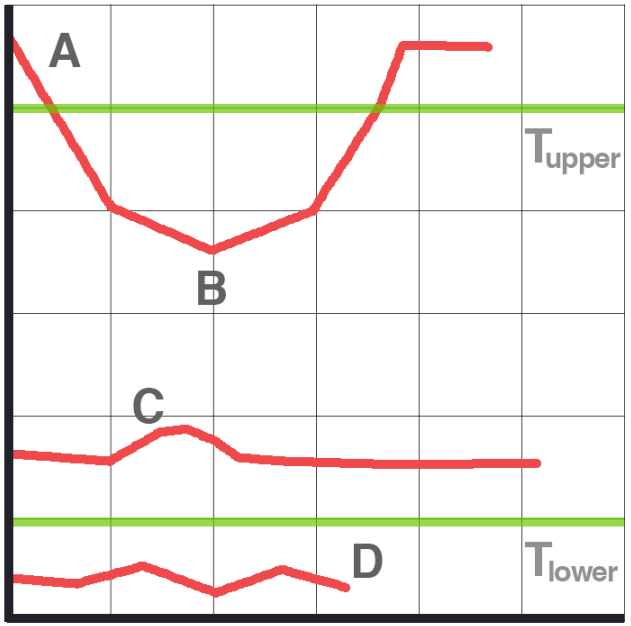
- John F. Canny 所開發出的多級邊緣檢測演算法
- 優點：低錯誤率，高定位性，最小回應

https://en.wikipedia.org/wiki/Canny_edge_detector
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.420.3300&rep=rep1&type=pdf

Canny Edge Detection 步驟

1. 去除雜訊 (常用高斯平滑濾波)
2. 計算梯度方向和強度 (Sobel)
3. 非最大抑制





4. 判斷邊界

- `if pixel gradient > upper threshold, pixel = edge`
- `if pixel gradient < lower threshold, pixel != edge`
- `if lower < pixel gradient < upper &&`
  `neighbor > upper threshold, pixel = edge`

http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html

Normal

Gaussian Blur

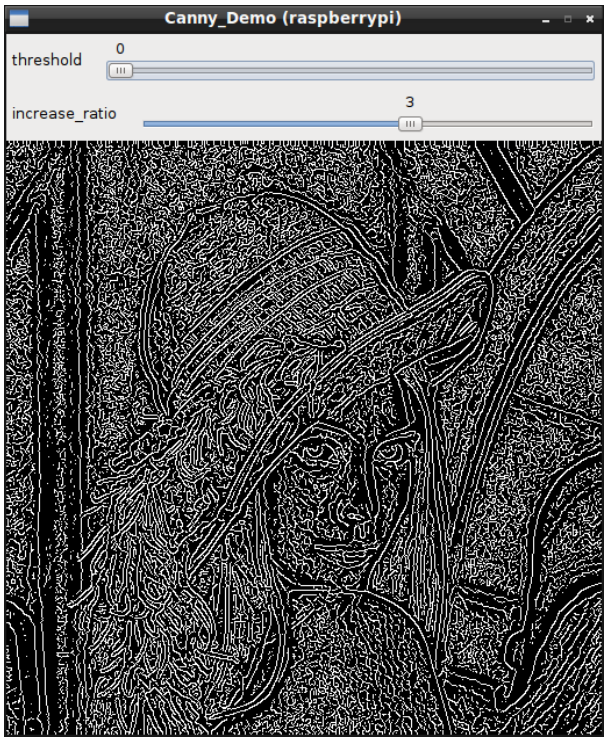
Sobel gradient

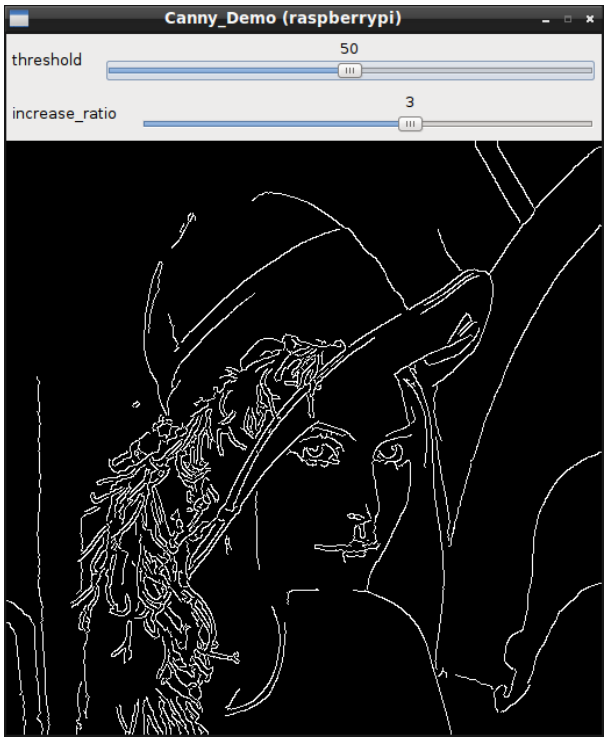
Non-maximum suppreson

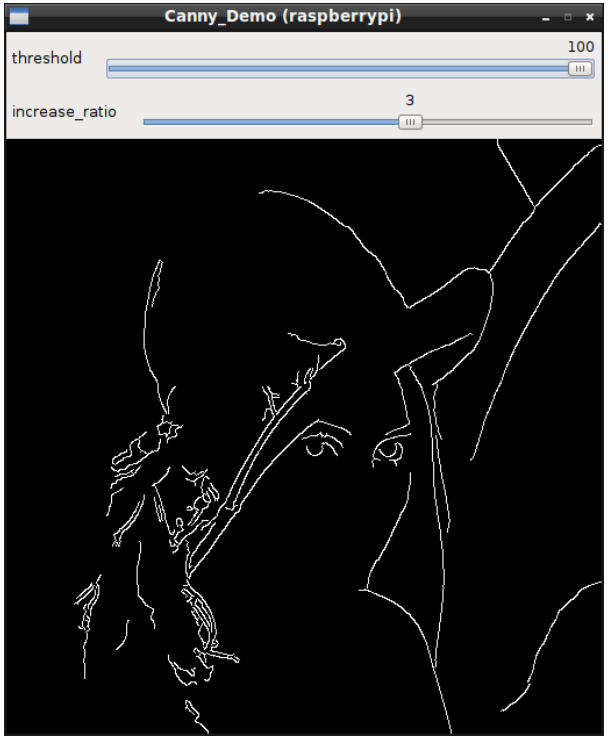
http://slideplayer.com/slide/8629030/

使用 Canny

- `cv2.Canny(img,lowerT,upperT,[apertureSize])`
  - 通常上下門檻值的比例在 2:1 到 3:1 之間
  - `apertureSize(Sobel kernel)` 預設是 3







http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html

即時更新門檻值

```
cv2.createTrackbar('threshold', 'canny_demo', 0, 100, nothing)
cv2.createTrackbar('ratio',     'canny_demo', 0, 5,   nothing)

while True:
    threshold = cv2.getTrackbarPos('threshold', 'canny_demo')
    ratio     = cv2.getTrackbarPos('ratio',     'canny_demo')
    image = cv2.imread(imagePath)
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    edges = cv2.GaussianBlur(gray, (5, 5), 0)
    edges = cv2.Canny(edges, threshold, threshold * ratio)
    cv2.bitwise_and(image, image, mask=edges)
```

DEMO
canny_edge_detect.py

$ cd ~/camera-opencv/04-image_processing

$ python canny_edge_detect.py lena512rgb.png

找邊緣的目的是為了找線

Hough Transform

- P.Hough 所提出，用在二值化影像的形狀偵測
- 實做步驟為
  1. 空間映射（影像空間映射到參數空間，二維到一維）
  2. 座標轉換（笛卡兒座標轉換到極座標）
  3. 使用累加器(Accumulator)

(a) Input image (480×270)

(b) Edge of input image

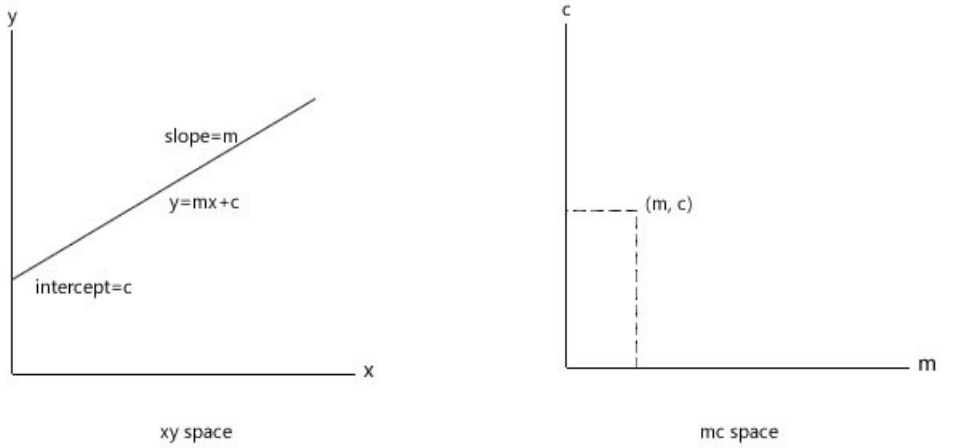
(c) Line detection (480×480 HS)

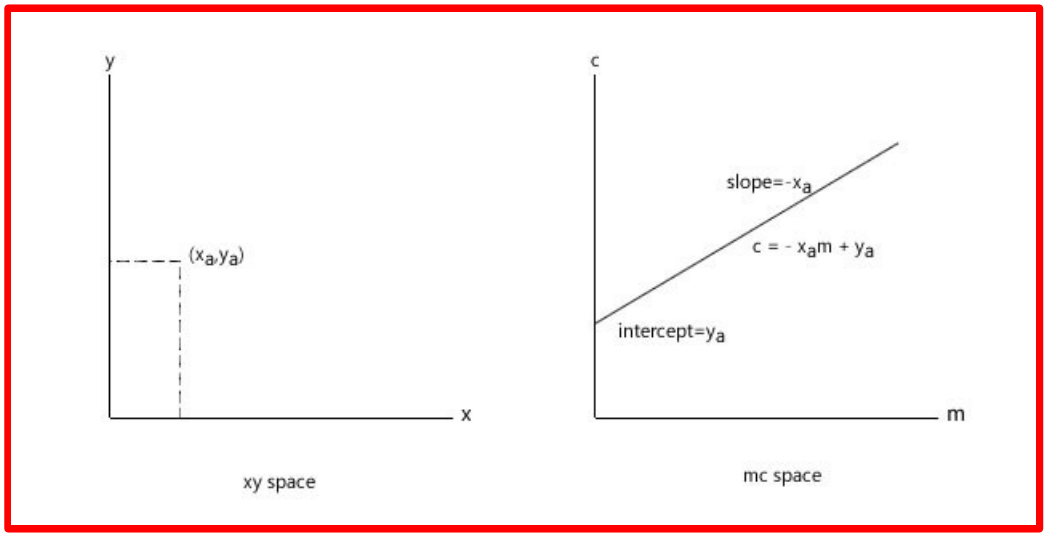
(d) Line detection (120×120 HS)

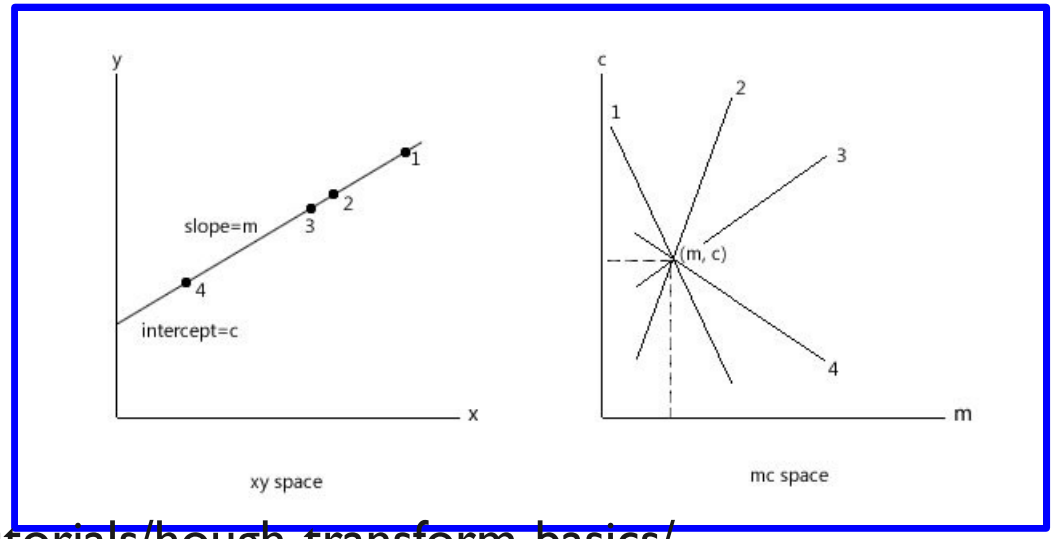
http://goo.gl/3O1tcq

空間映射

- 影像空間映射到參數空間





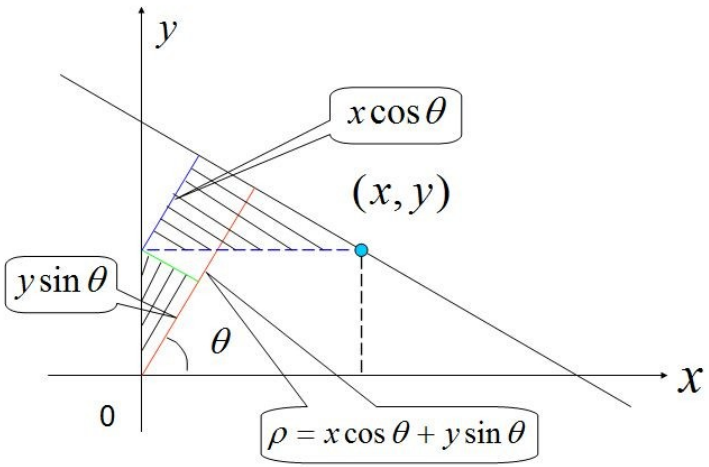


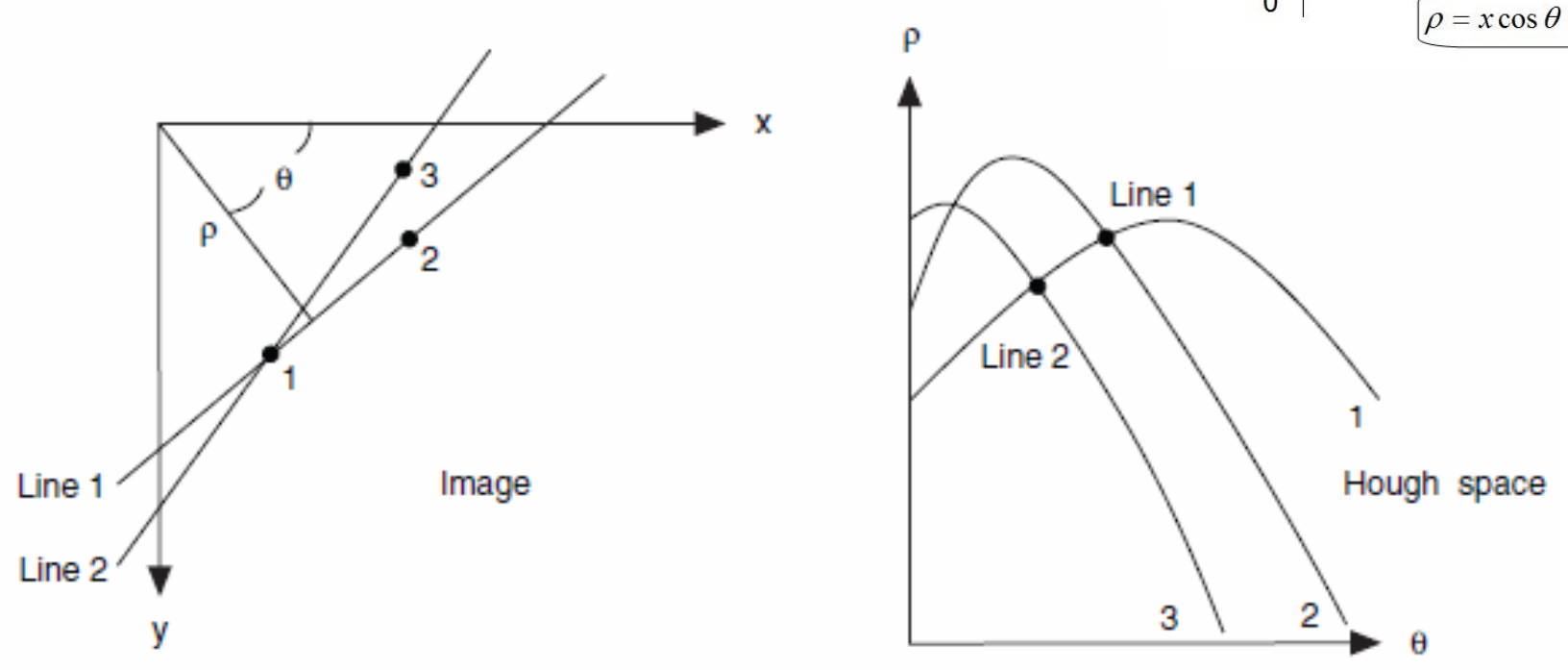
http://www.aishack.in/tutorials/hough-transform-basics/

座標轉換

$$\rho = x\cos\theta + y\sin\theta$$

- 笛卡兒座標轉換到極座標

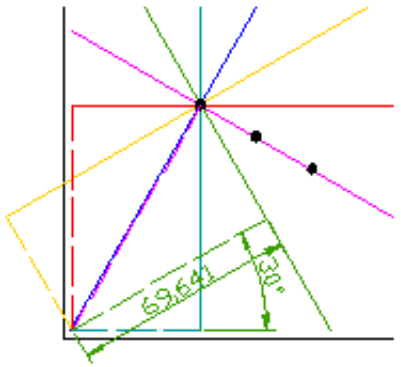




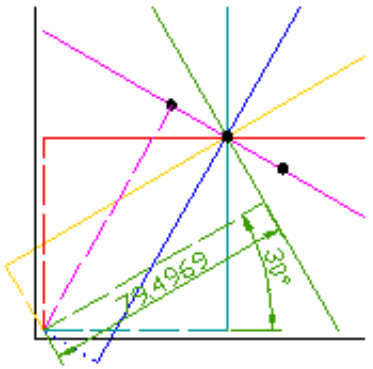
http://zone.ni.com/reference/en-XX/help/372916L-01/nivisionconcepts/edge_detection_concepts/

使用累加器

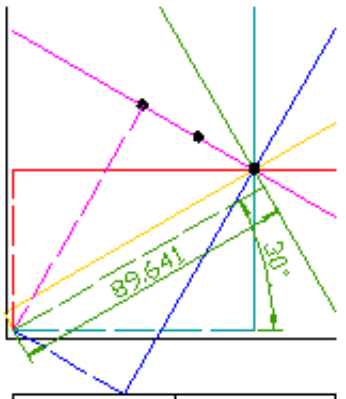
- 找出最大值（或超過門檻值）



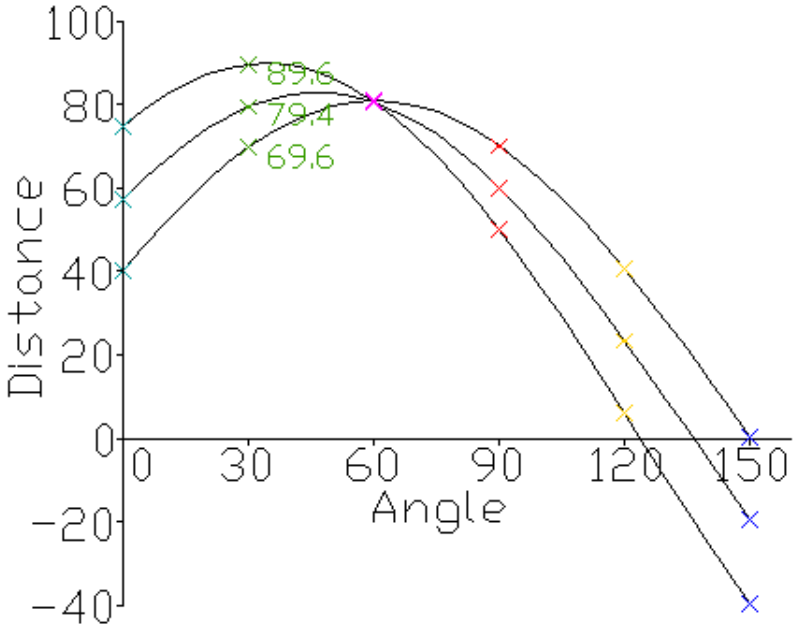
Angle	Dist.
0	40
30	69.6
60	81.2
90	70
120	40.6
150	0.4



Angle	Dist.
0	57.1
30	79.5
60	80.5
90	60
120	23.4
150	-19.5



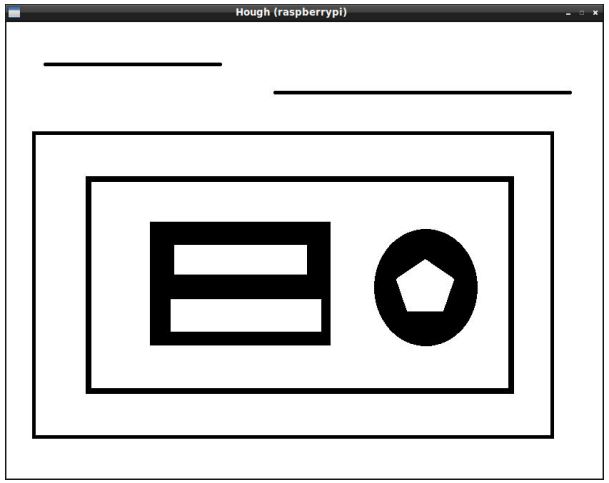
Angle	Dist.
0	74.6
30	89.6
60	80.6
90	50
120	6.0
150	-39.6



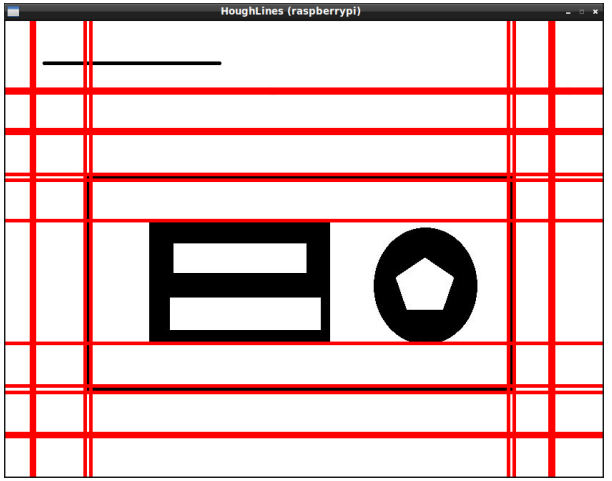
https://en.wikipedia.org/wiki/Hough_transform

HoughLines & HoughLinesP

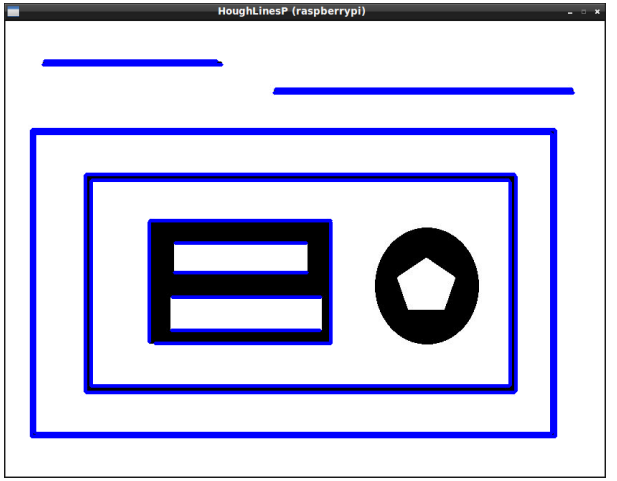
- HoughLines()，找出直線（無窮長）
- HoughLinesP()，找出線段



原始圖



HoughLines



HoughLinesP

http://ccw1986.blogspot.tw/2016/04/hough-transform-using-opencv.html

HoughLines

- 原型：`cv2.HoughLines(image, rho, theta, threshold[, lines[, srn[, stn]]])`
- 範例：`cv2.HoughLines(edges, 1, np.pi/180, 250)`
  - `image`：輸入影像（8 位元單通道二值化圖）
  - `rho`：距離解析度（極坐標中極徑 r 的最小單位）
  - `theta`：角度解析度（極坐標中極角 θ 的最小單位）
  - `threshold`：門檻值（超過此值的線才會存在 `lines` 裡）
  - `lines`：輸出結果（每條線都包含 r 和 θ）
  - `srn`：可有可無的距離除數
  - `stn`：可有可無的角度除數

http://docs.opencv.org/2.4/modules/imgproc/doc/feature_detection.html

找出直線

```
gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
h, w = gray.shape

edges = cv2.Canny(gray, 50, 150, 3)
lines = cv2.HoughLines(edges, 1, np.pi/180, 250)[0]

for (rho, theta) in lines:
    x0 = np.cos(theta)*rho
    y0 = np.sin(theta)*rho
    pt1 = ( int(x0 + (h+w)*(-np.sin(theta))), int(y0 +
(h+w)*np.cos(theta)) )
    pt2 = ( int(x0 - (h+w)*(-np.sin(theta))), int(y0 -
(h+w)*np.cos(theta)) )
    cv2.line(image, pt1, pt2, (0, 0, 255), 3)
```

DEMO
hough_find_lines.py

$ cd ~/camera-opencv/04-image_processing

$ python hough_find_lines.py lena512rgb.png

HoughLinesP

- 原型：`cv2.HoughLinesP(image, rho, theta, threshold[, lines[, minLineLength[, maxLineGap]]])`
- 範例：`cv2.HoughLinesP(edges, 1, np.pi/180, 50, None, 100, 5)`
  - `image`：輸入影像（8 位元單通道二值化圖）
  - `rho`：距離解析度（極坐標中極徑 r 的最小單位）
  - `theta`：角度解析度（極坐標中極角 θ 的最小單位）
  - `threshold`：門檻值（超過此值的線才會存在 `lines` 裡）
  - `lines`：輸出結果（每條線都包含 `x1, y1, x2, y2` 線段頂點）
  - `minLineLength`：線段最短距離（超過此值的線才會存在 `lines` 裡）
  - `maxLineGap`：最大間隔

http://docs.opencv.org/2.4/modules/imgproc/doc/feature_detection.html

找出線段

```
gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)

edges = cv2.Canny(gray, 50, 150, 3)
plines = cv2.HoughLinesP(edges, 1, np.pi/180, 50, None,
100, 5)[0]

for pl in plines:
    cv2.line(image, (pl[0], pl[1]), (pl[2], pl[3]),
(255, 0, 0), 3)
```

DEMO
hough_find_linesp.py

$ cd ~/camera-opencv/04-image_processing

$ python hough_find_linesp.py lena512rgb.png

DEMO
鄉民查水表

實驗 5: 人臉偵測

目的 : 從 OpenCV 瞭解 Camera 和 Webcam

載入圖檔並顯示

```
import cv2
import sys

imagePath = sys.argv[1]
image = cv2.imread(imagePath)

cv2.imshow("preview", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

除了拍照存檔後開圖以外，
可以即時預覽 Camera 的結果嗎？

讀取 Camera 並顯示

```
import cv2

cap = cv2.VideoCapture(0)
cap.set(cv2.cv.CV_CAP_PROP_FRAME_WIDTH,  320)
cap.set(cv2.cv.CV_CAP_PROP_FRAME_HEIGHT, 240)

while True:
    ret, frame = cap.read()
    cv2.imshow(“preview”, frame)
    if cv2.waitKey(1) & 0xFF == ord(“q”):
        break

cap.release()
cv2.destroyAllWindows()
```

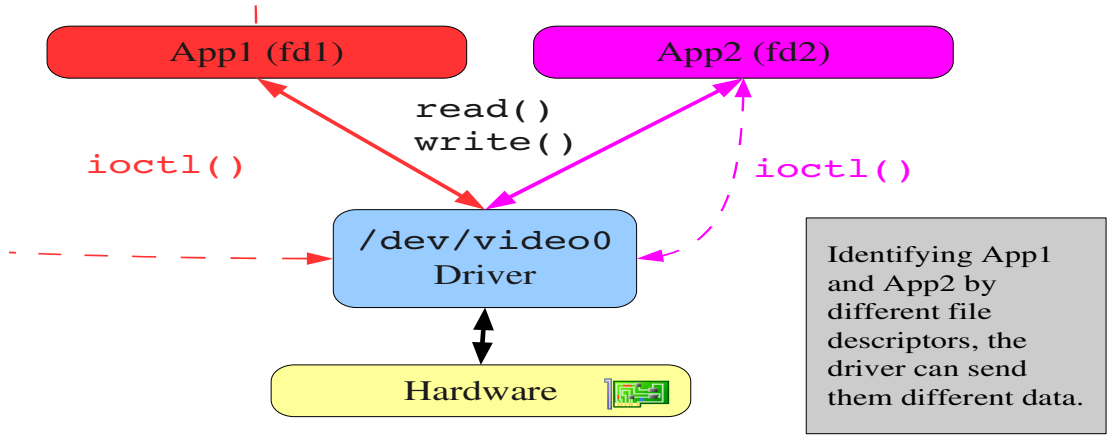
回來看 camera
你有用過 webcam 嗎？

Webcam

http://www.slideshare.net/gxben/abs-2014-android-kit-kat-internals

Video4Linux 2nd(V4L2)

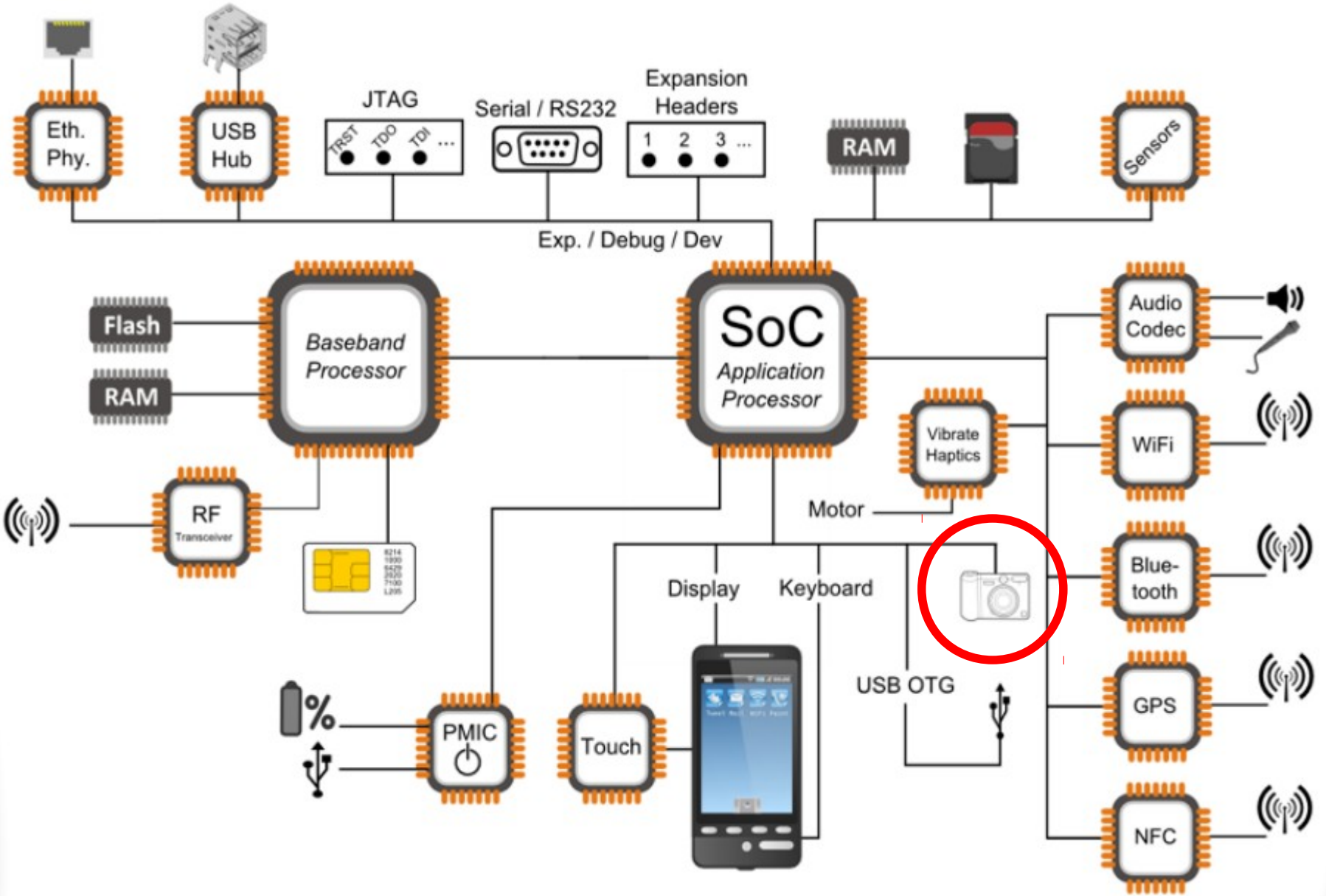
- 預定義好的Userspace API
- Video and radio streaming devices
  - video cameras
  - analog and digital TV receiver cards
  - AM/FM receiver cards



http://free-electrons.com/doc/embedded_linux_multimedia.pdf

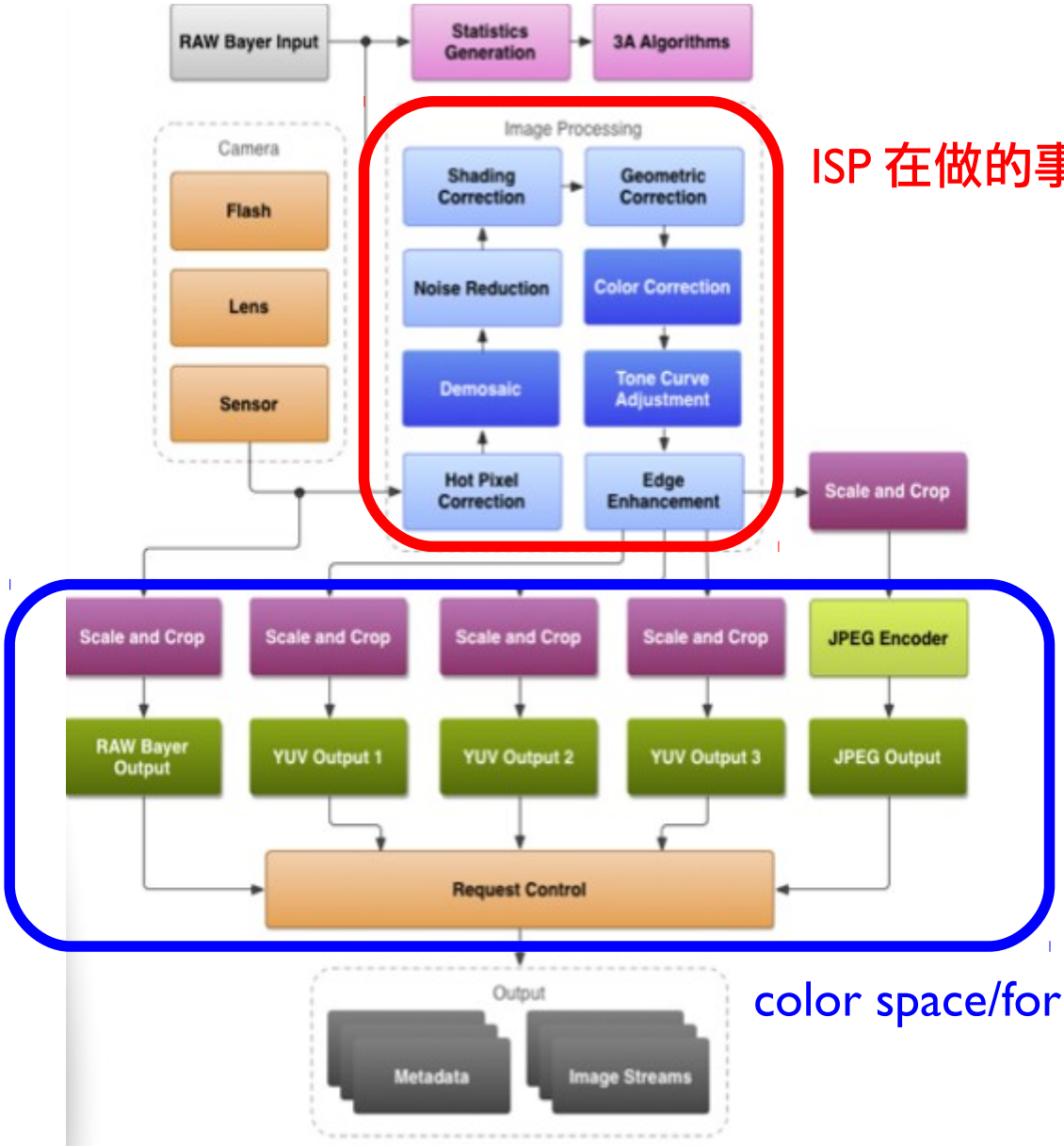
Camera ≠ Webcam

Modern Device Internal



http://www.slideshare.net/gxben/abs-2014-android-kit-kat-internals

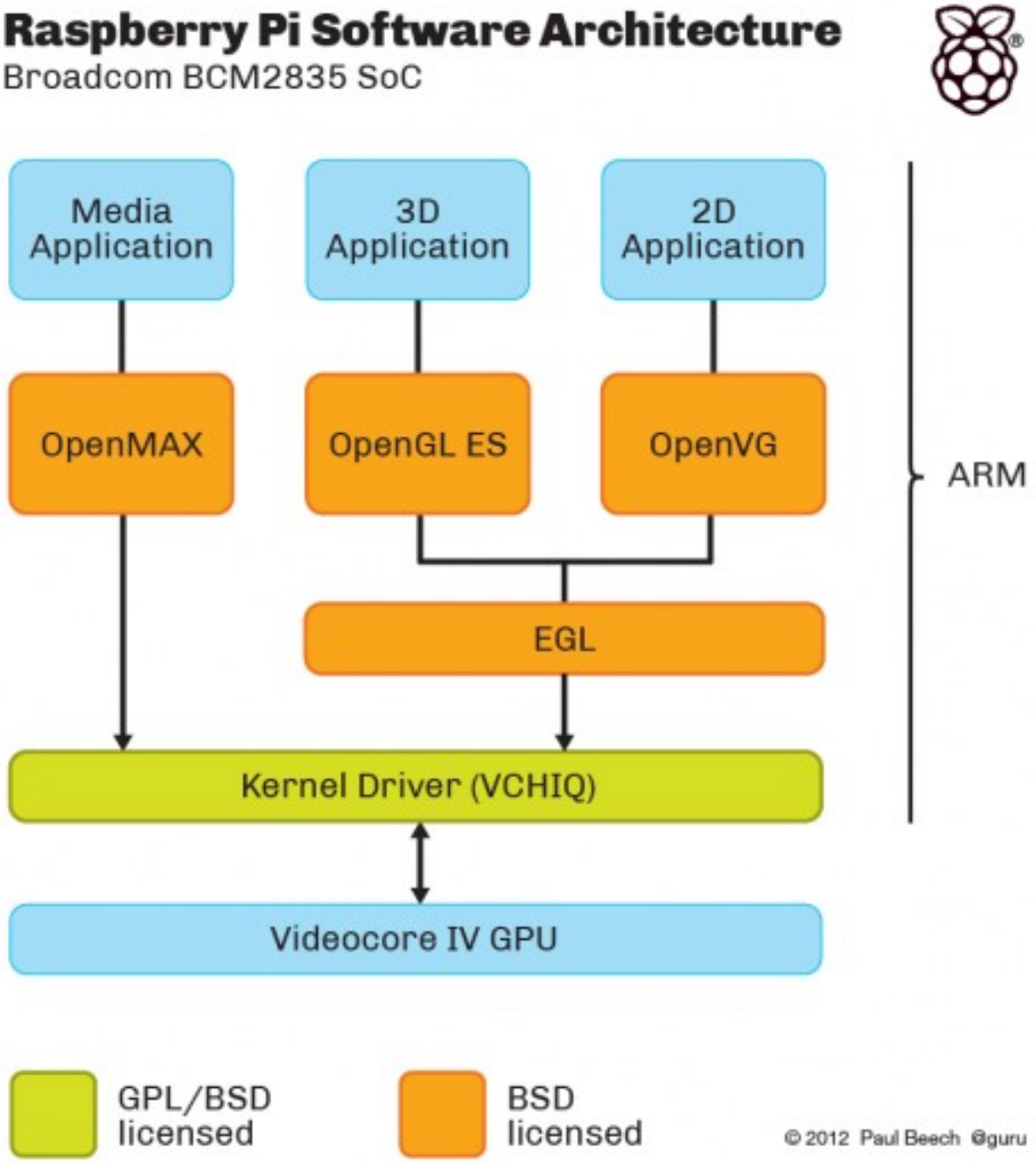
Image Processing Pipeline



ISP 在做的事 (HW)

color space/format 轉換 (SW)

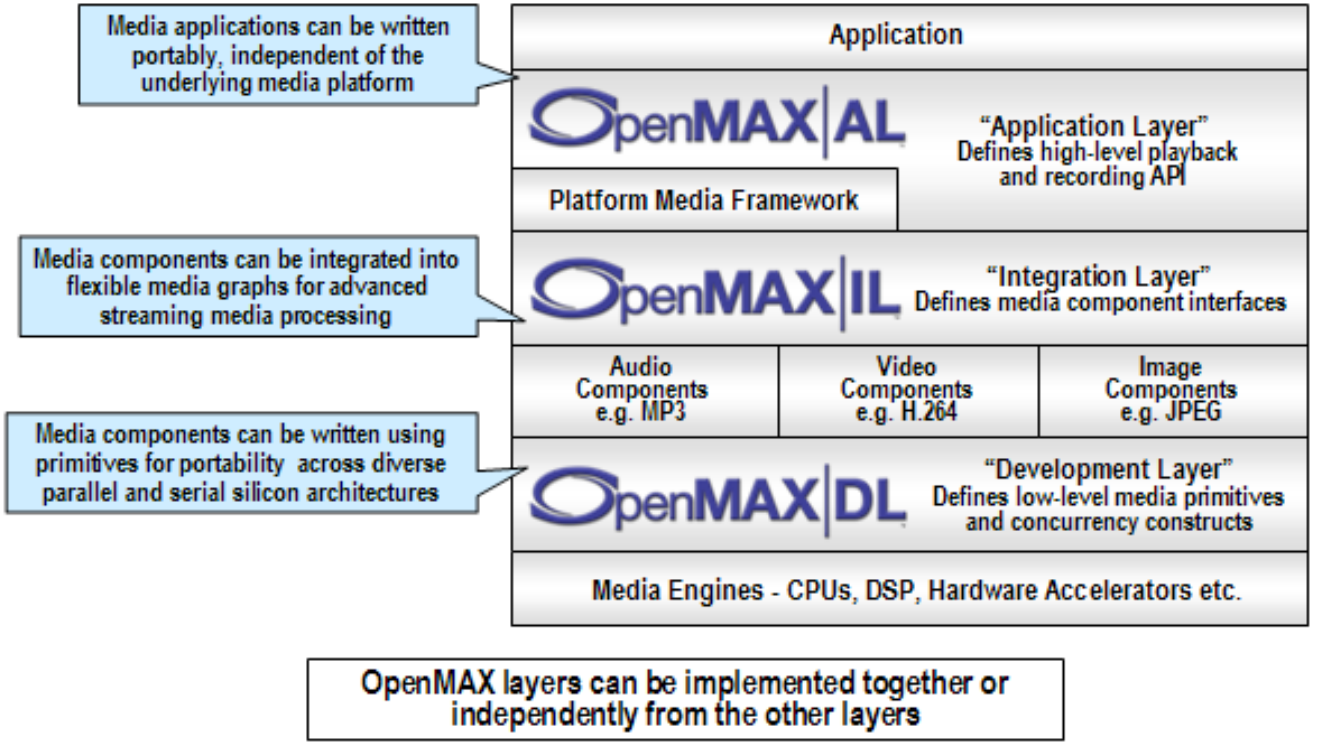
http://www.slideshare.net/gxben/abs-2014-android-kit-kat-internals



http://elinux.org/Raspberry_Pi_VideoCore_APIs

OpenMAX

- 開放多媒體加速層 (Open Media Acceleration)
  - 由 Khronos Group 提出的標準
  - 統一的介面，加速大量多媒體資料的處理



https://www.khronos.org/openmax/

Official V4L2 Driver

- Kernel driver
- 使用 camera 像是 webcam 一樣
  - $ sudo modprobe bcm2835-v4l2
- 可直接存取 /dev/videoX
  - $ v4l2-ctl --list-devices
  - $ v4l2-ctl --list-formats
  - $ v4l2-ctl -L

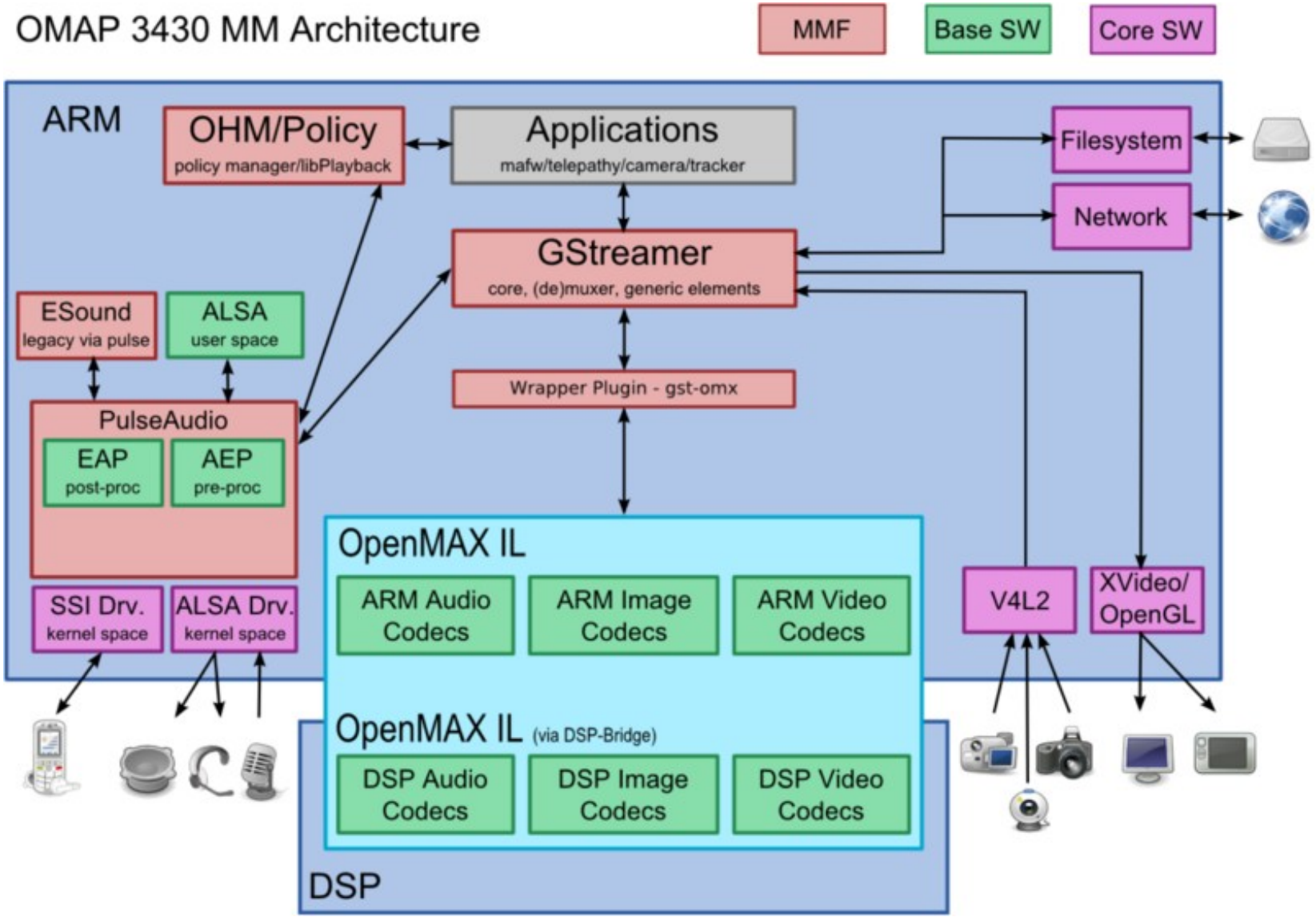
https://github.com/raspberrypi/linux/blob/rpi-3.10.y/Documentation/video4linux/bcm2835-v4l2.txt

User Space V4L2 Driver(UV4L)

- 安裝
  - `$ sudo apt-get install v4l-utils`
- 缺點
  - closed sourced
  - slow because it runs as a user program
- 相關應用
  - Real-time HTTP Streaming Server
  - Object detection and object tracking
  - Compute module: stereoscopic vision

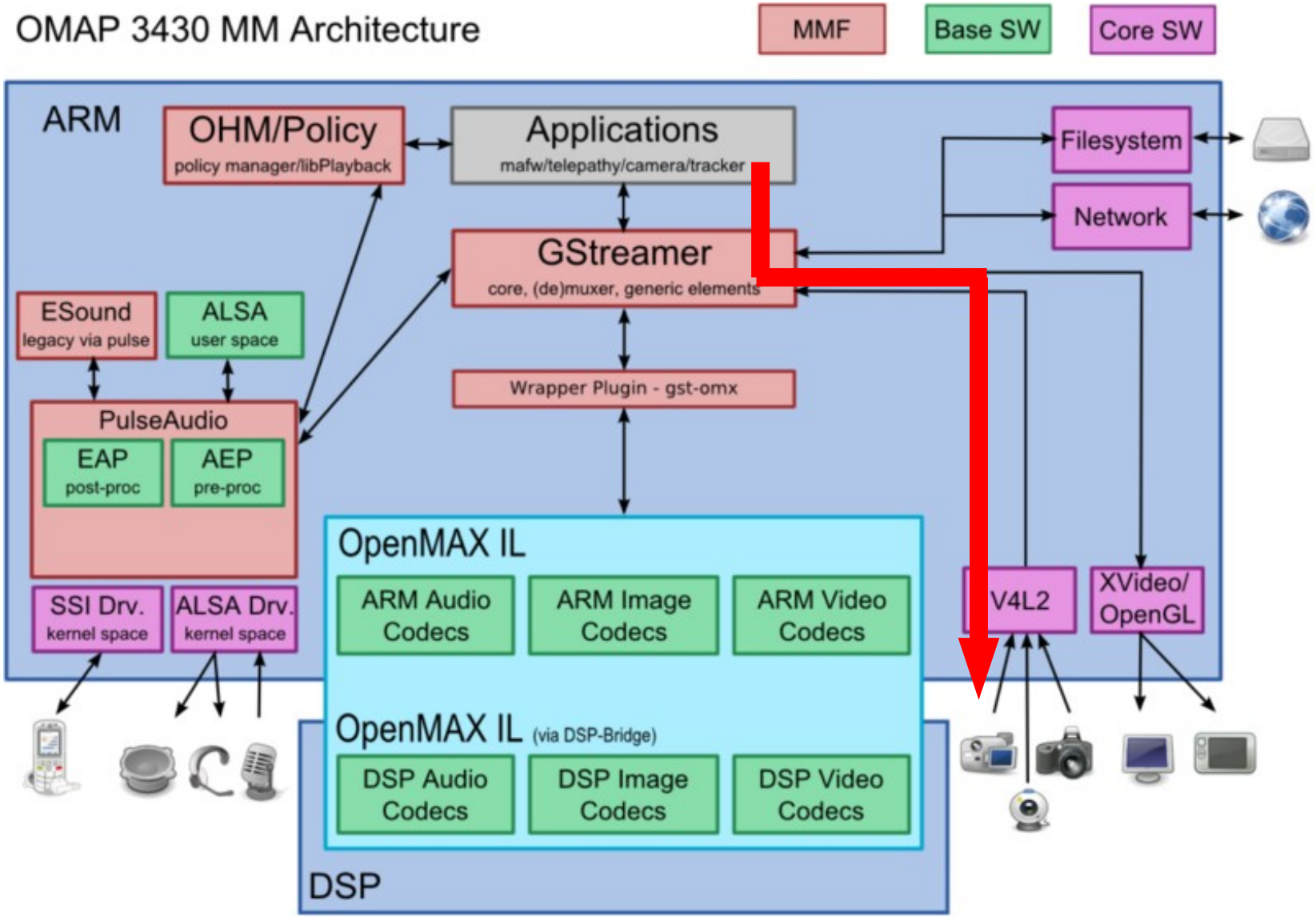
http://www.linux-projects.org/modules/sections/index.php?op=viewarticle&artid=16

Multimedia Stack Example
- TI OMAP 3430



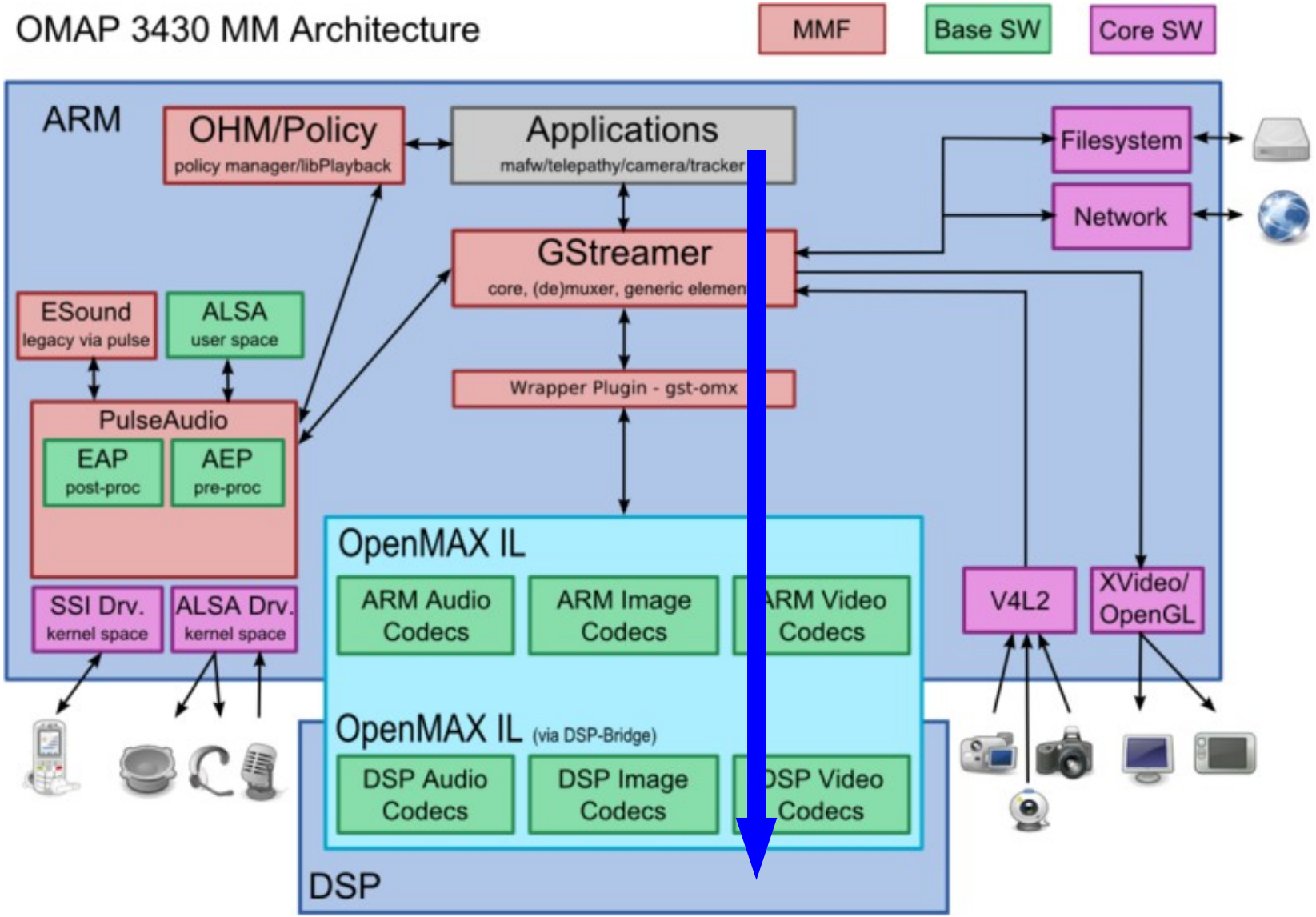
http://wiki.maemo.org/Documentation/Maemo_5_Developer_Guide/Architecture/Multimedia_Domain

走 V4L2



http://wiki.maemo.org/Documentation/Maemo_5_Developer_Guide/Architecture/Multimedia_Domain

走 OpenMAX



http://wiki.maemo.org/Documentation/Maemo_5_Developer_Guide/Architecture/Multimedia_Domain

比較兩種路徑的分別

- OpenMAX

  優點：

  - 接到 GPU，可硬解

  缺點：

  - 範例程式少
  - MMAL 只有標頭檔
  - 不易接 OpenCV

- V4L2

  優點：

  - 標準的 Linux API
  - 多數應用程式可支援
  - 可直接接到 OpenCV

  缺點：

  - CPU 運算，軟解，慢

但我想使用 C 語言接到 OpenMAX

- 參考這幾隻程式吧
- raspicam
  - https://github.com/raspberrypi/userland/tree/master/host_applications/linux/apps/raspicam
- rpi-omx-tutorial
  - https://github.com/SonienTaegi/rpi-omx-tutorial
- omxcam
  - https://github.com/gagle/raspberrypi-omxcam
- rpi-mmal-demo
  - https://github.com/tasanakorn/rpi-mmal-demo/tree/develop

使用前記得要先載入模組
$ sudo modprobe bcm2835-v4l2

DEMO

camera_preview.py

$ cd ~/camera-opencv/05-face_detection

$ python camera_preview.py

人臉偵測 (Face Detection)

人臉偵測與人臉識別

- Facial Detection:
  Where is the face?

- Facial Recognition:
  Who is this?

https://www.youtube.com/watch?v=sWTvK72-SPU

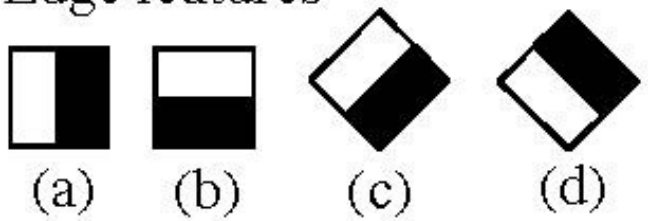
Haar Feature Cascade

- 由 Viola & Jones 提出，並由 Lienhart & Maydt 改善
- 採監督式學習的類神經網路演算法，並有以下特色
  - 特徵比對 (Haar features)
  - 積分影像計算 (Integral Image)
  - 串接分類器 (Cascade)
  - 學習機制 (AdaBoost)

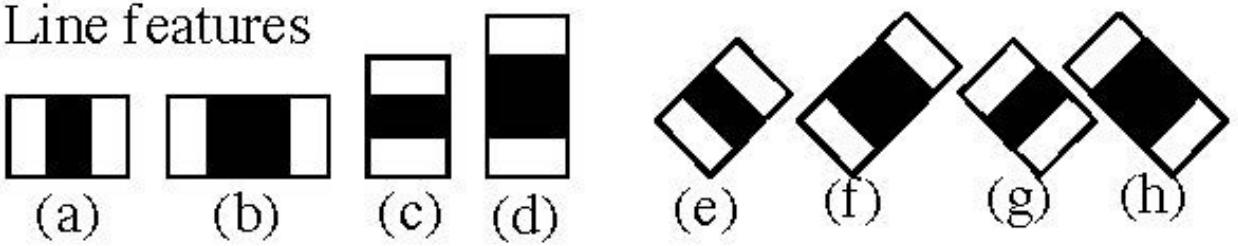
http://www.open-electronics.org/raspberry-pi-and-the-camera-pi-module-face-recognition-tutorial/

Haar-Like Features

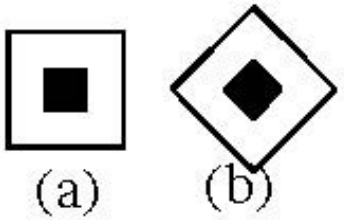
1. Edge features



2. Line features



3. Center-surround features

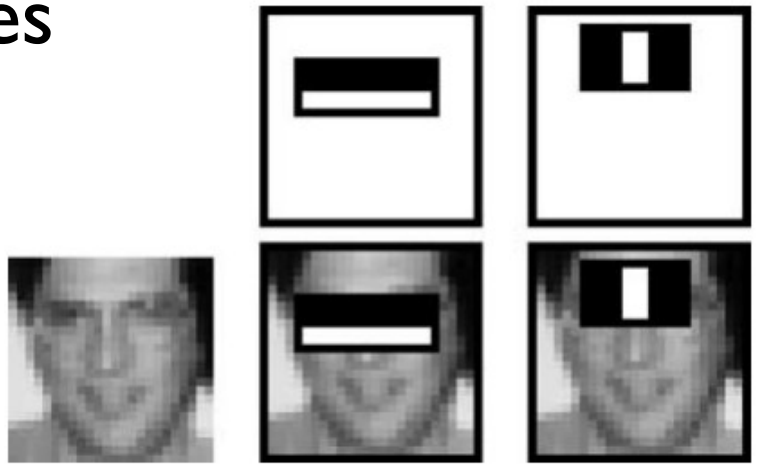


4. Special diagonal line feature used in [3,4,5]

http://archive.cnx.org/contents/d13cd00c-cabf-402b-844a-0cbe92aaef3f@2/algorithms

特徵比對

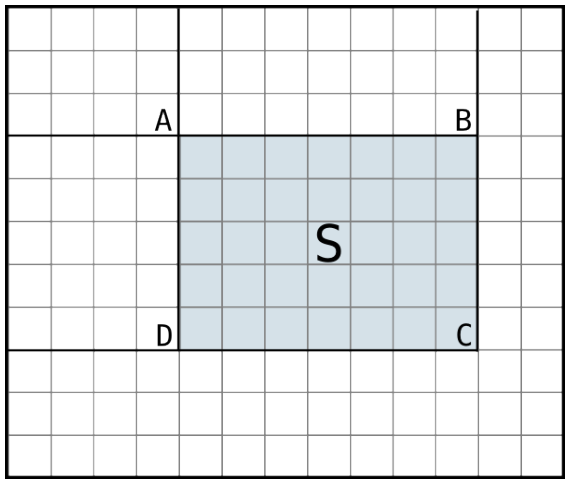
- Pick a scale (ex: 24x24 pixels) for the feature
- Slide it across the image
- Compute the average pixel values under the white area and the black area
- If the difference between the areas is above some threshold, the feature matches



https://www.youtube.com/watch?v=sWTvK72-SPU

積分影像計算

- A "summed area table" is created in a single pass across the image
- The sum of any region in the image can be computed by a single formula



Original				
5	**2**	**3**	4	1
1	**5**	**4**	2	3
2	2	1	3	4
3	5	6	4	5
4	1	3	2	6

Integral				
5	7	10	14	15
6	13	**20**	26	30
8	17	25	34	42
11	25	39	52	65
15	30	47	62	81

$5 + 2 + 3 + 1 + 5 + 4 = 20$

Original				
5	2	3	4	1
1	**5**	**4**	**2**	3
2	**2**	**1**	**3**	4
3	5	6	4	5
4	1	3	2	6

$5 + 4 + 2 + 2 + 1 + 3 = 17$

Integral				
5	7	10	**14**	15
6	13	20	26	30
8	17	25	**34**	42
11	25	39	52	65
15	30	47	62	81

$34 - 14 - 8 + 5 = 17$

http://computervisionwithvaibhav.blogspot.tw/2015/08/viola-jones-in-nut-shell.html

Haar Cascade

- A "cascade" is a series of "Haar-like features" that are combined to form a classifier
- Haar Cascade = Classifier

https://www.youtube.com/watch?v=sWTvK72-SPU

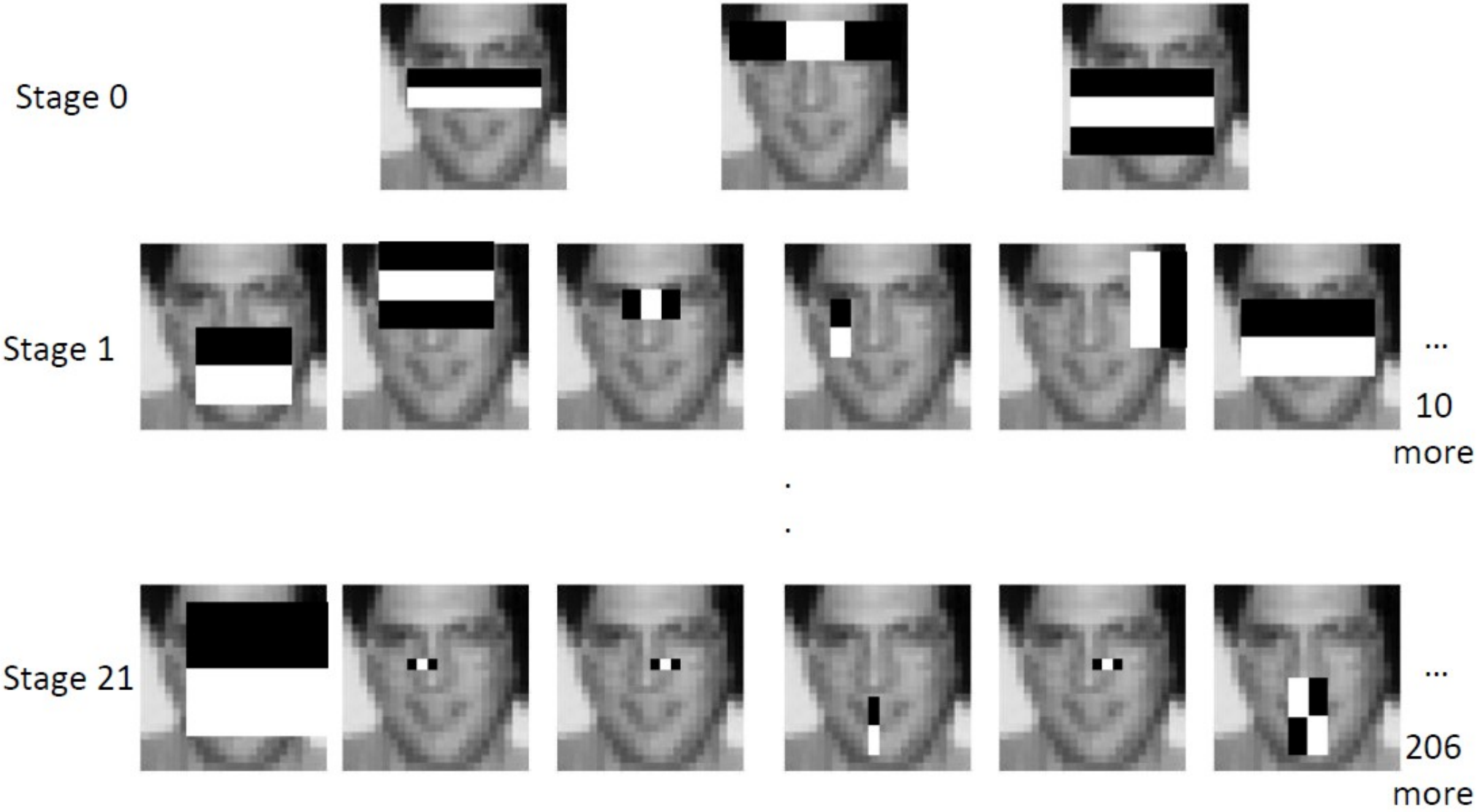
弱分類器

- Feature:  and 
- Classifier:  

- A single classifier isn't accurate enough
  - It's called a "weak classifier"

https://www.youtube.com/watch?v=sWTvK72-SPU

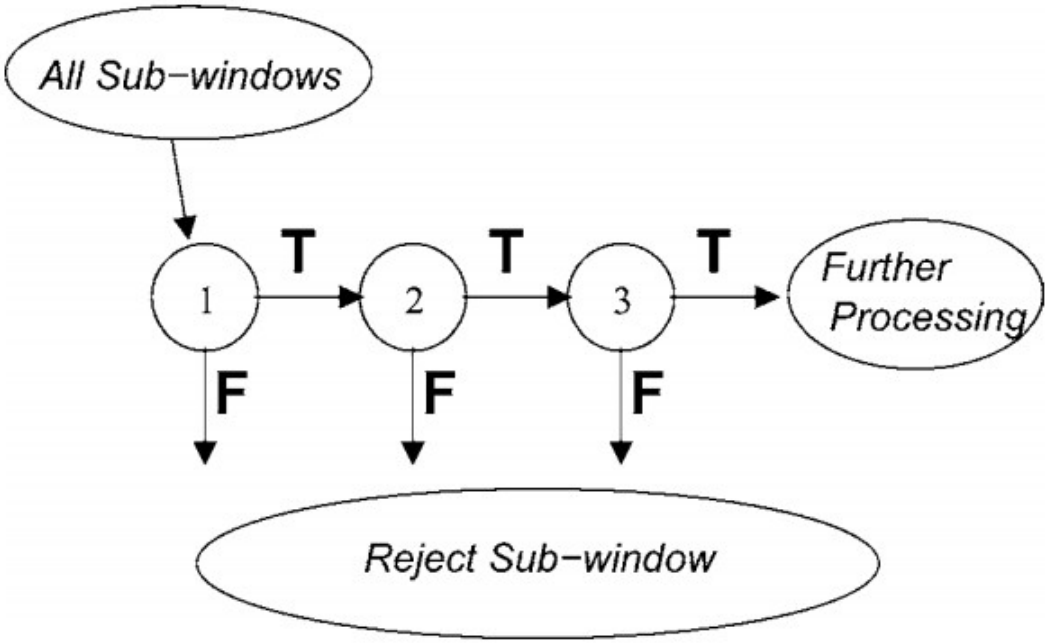
串接分類器



https://www.researchgate.net/post/Can_someone_answer_a_few_questions_on_Haar_Features_and_Adaboost

串接分類器

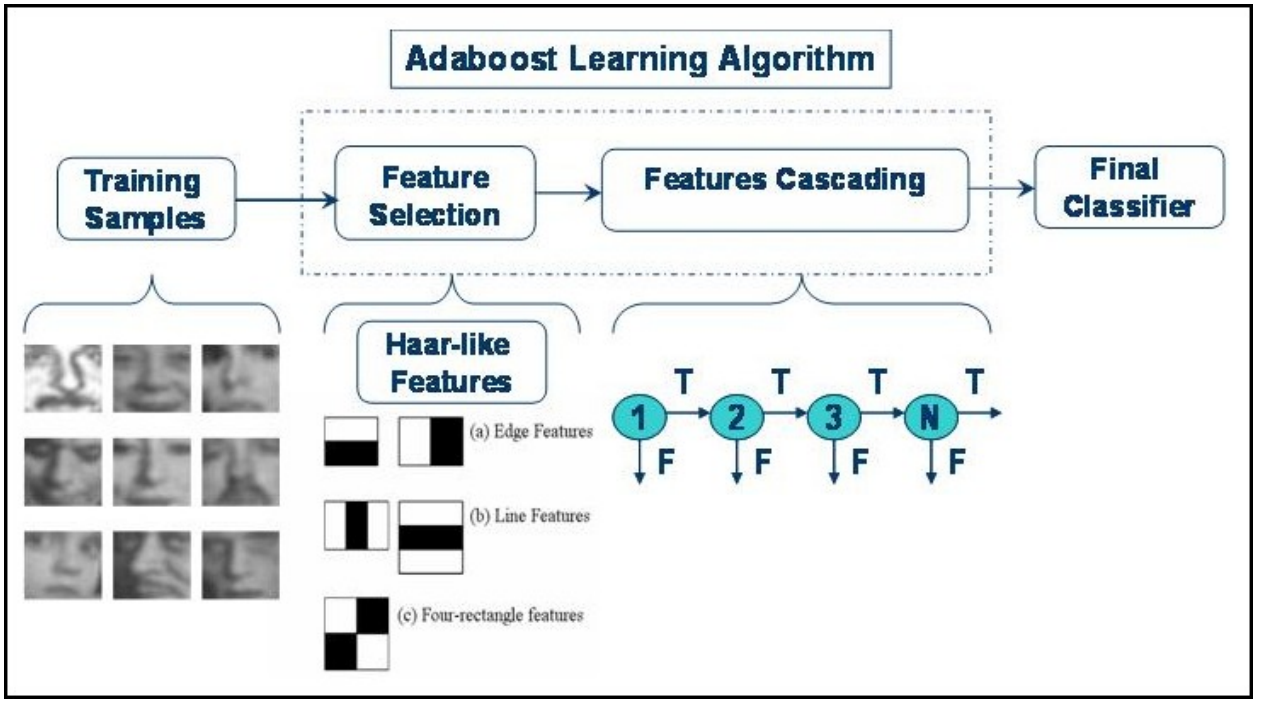
- Haar cascades consists of a series of weak classifiers - those barely better than 50% correct
- If an area passes a single classifier, go to the next classifier; otherwise, area doesn't match



https://www.youtube.com/watch?v=sWTvK72-SPU

AdaBoost(Adaptive Boosting)

- Adaboost tries out multiple weak classifiers over several rounds, selecting the best weak classifier in each round and combining the best weak classifiers to create a strong classifier



https://www.youtube.com/watch?v=sWTvK72-SPU

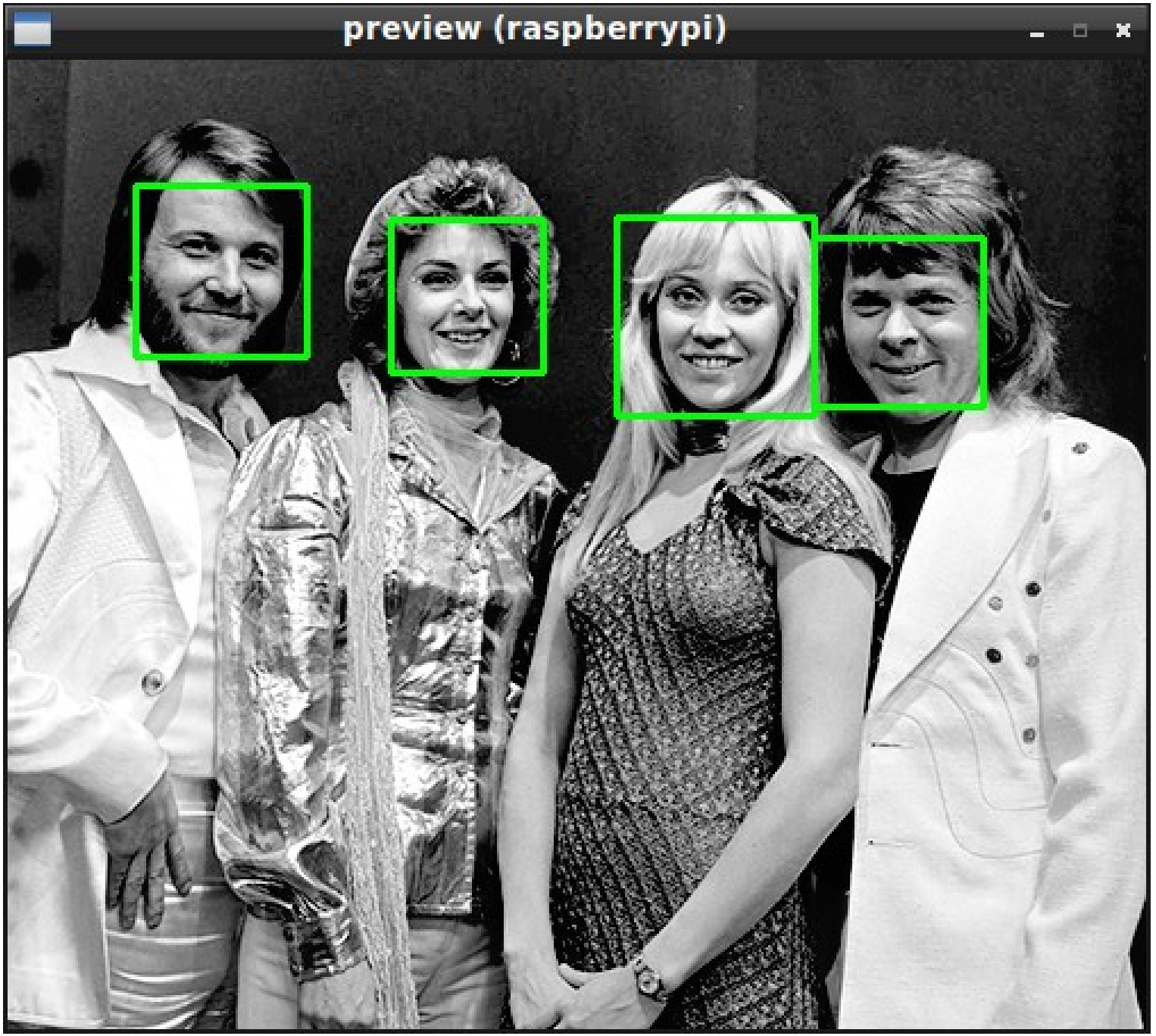
載入圖檔並辨識

```
faceCascade = cv2.CascadeClassifier(sys.argv[2])
image = cv2.imread(sys.argv[1])
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

faces = faceCascade.detectMultiScale(
    gray,
    scaleFactor=1.1,
    minNeighbors=5,
    minSize=(30, 30),
    flags = cv2.cv.CV_HAAR_SCALE_IMAGE
)

for (x, y, w, h) in faces:
    cv2.rectangle(image, (x, y), (x+w, y+h), (0, 255, 0), 2)
```

https://sites.google.com/site/5kk73gpu2012/assignment/viola-jones-face-detection

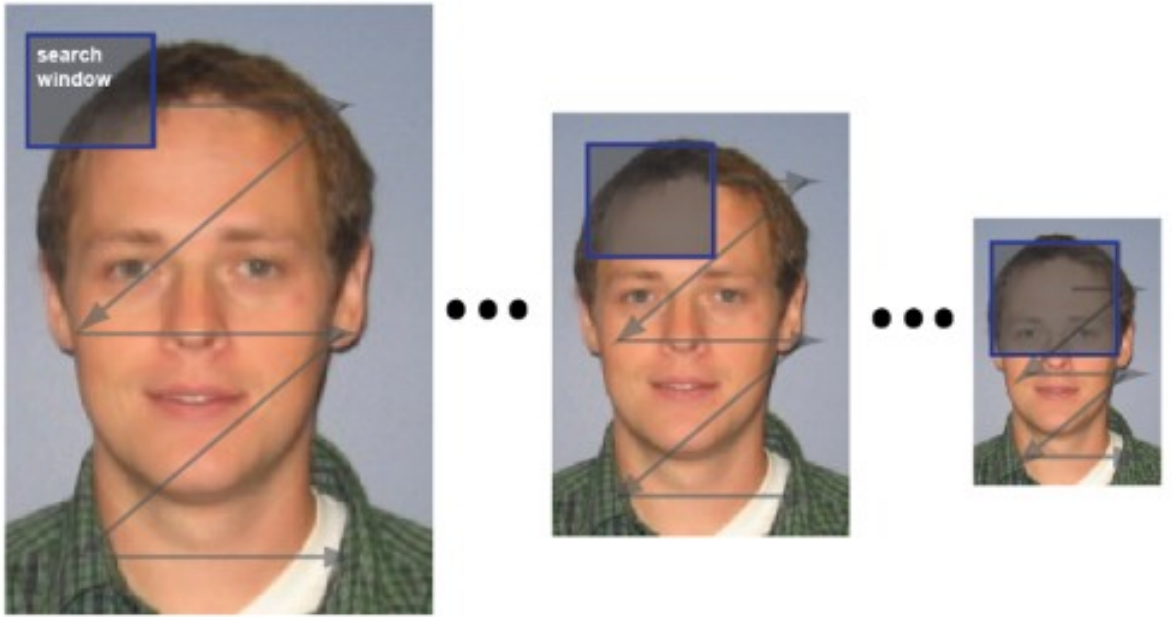
DEMO
image_face_detect.py

```
$ cd ~/camera-opencv/05-face_detection
$ python image_face_detect.py abba.png haarcascade_frontalface_default.xml
```


preview (raspberrypi)

可調整的參數

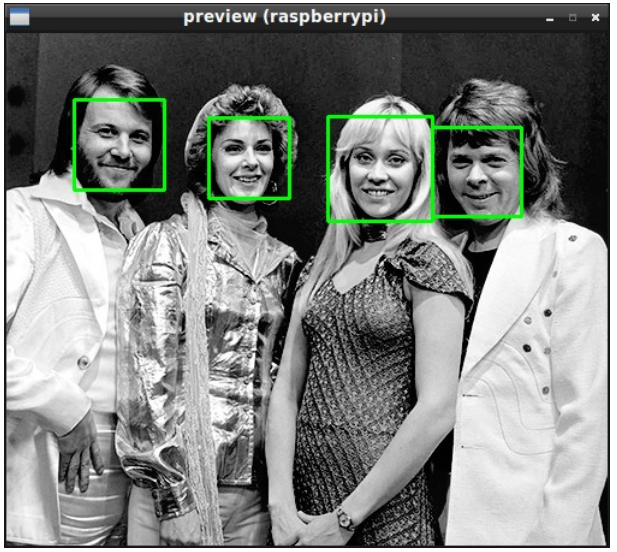
- `scaleFactor`：檢測視窗縮放比率
- `minNeighbors`：檢測區域鄰域內最少包含的檢測出的備選人臉區域 (次數)
- `minSize`：被檢測物體的最小尺寸



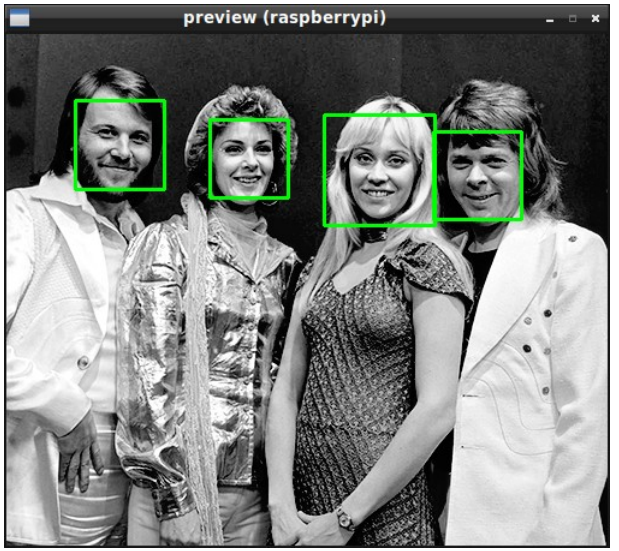
https://www.mathworks.com/help/vision/ref/vision.cascadeobjectdetector-class.html

scaleFactor

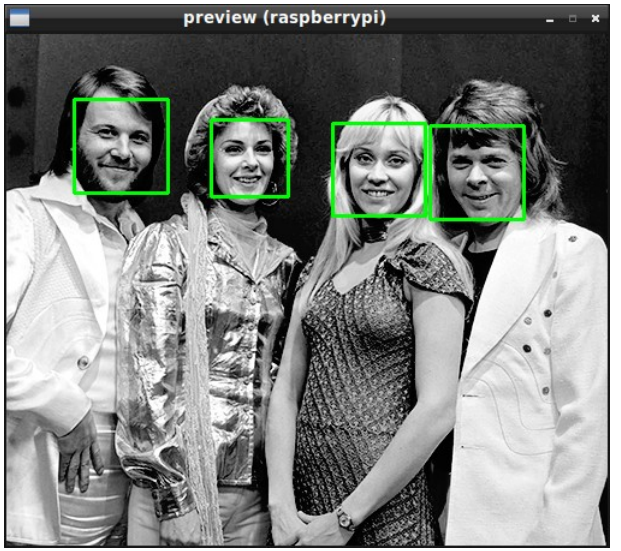
- minNeighbors=5, minSize=(30, 30)



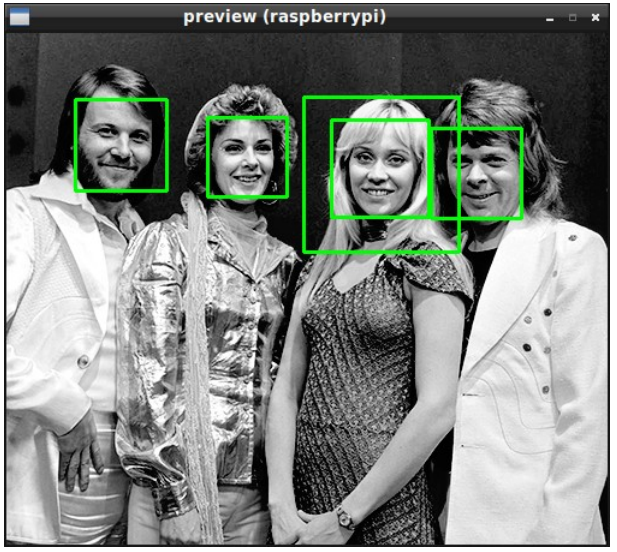
scaleFactor=1.1



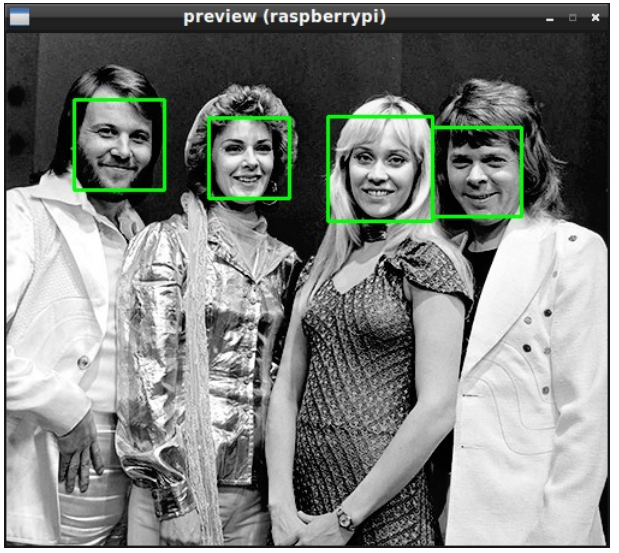
1.2



1.3



1.4

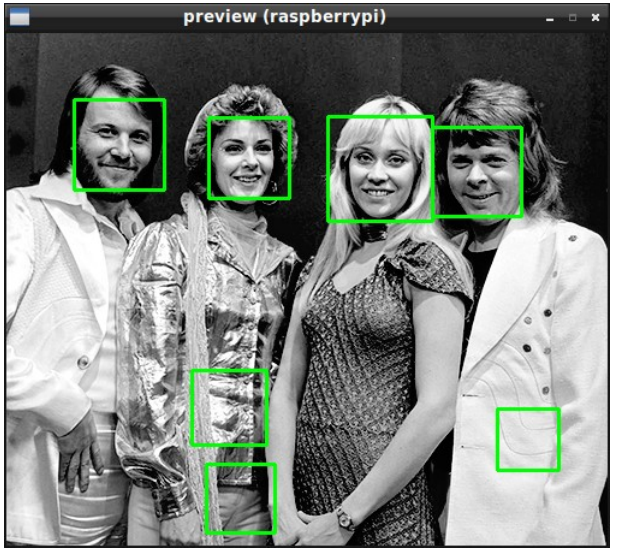


1.5

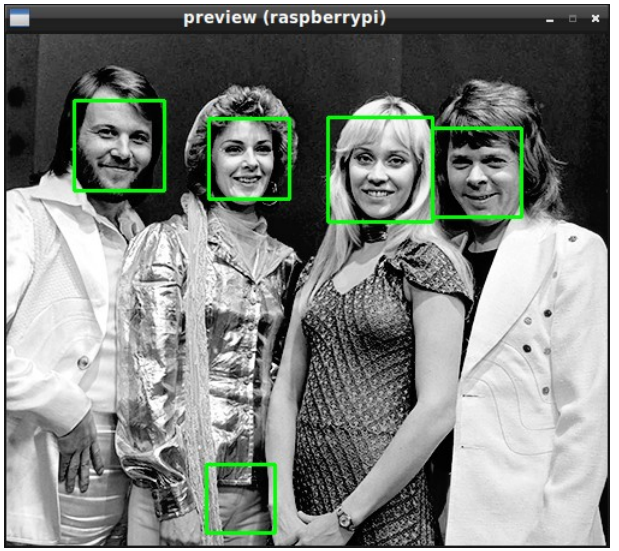
1.6

minNeighbors

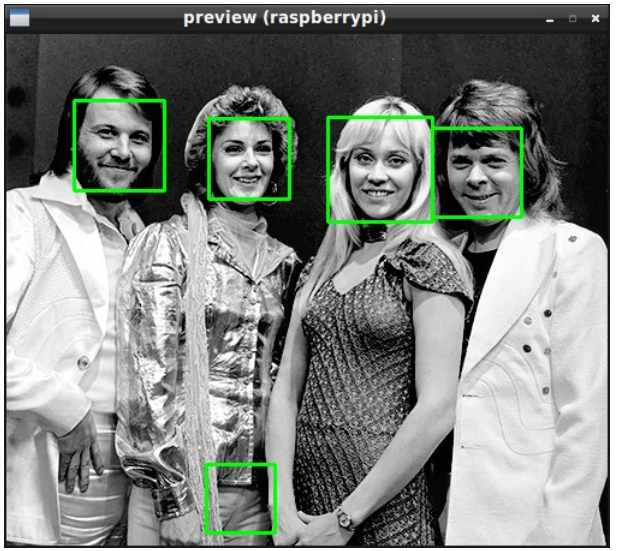
- scaleFactor=1.1, minSize=(30, 30)



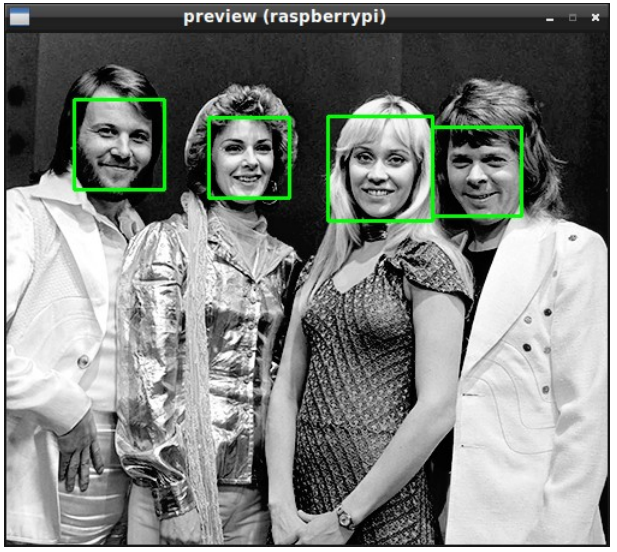
minNeighbors=1



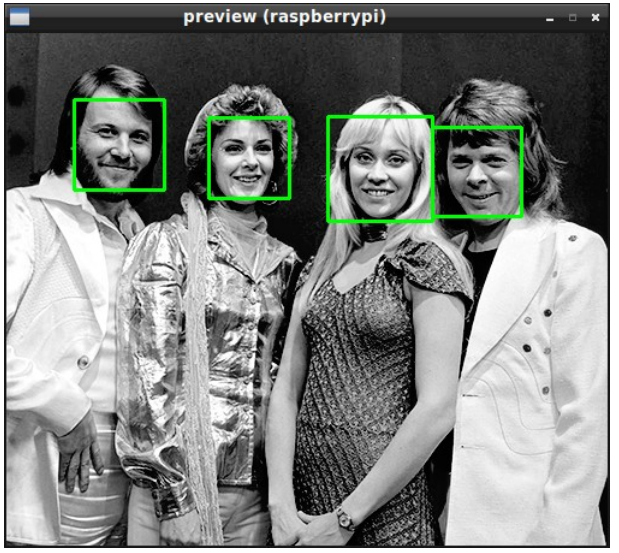
2



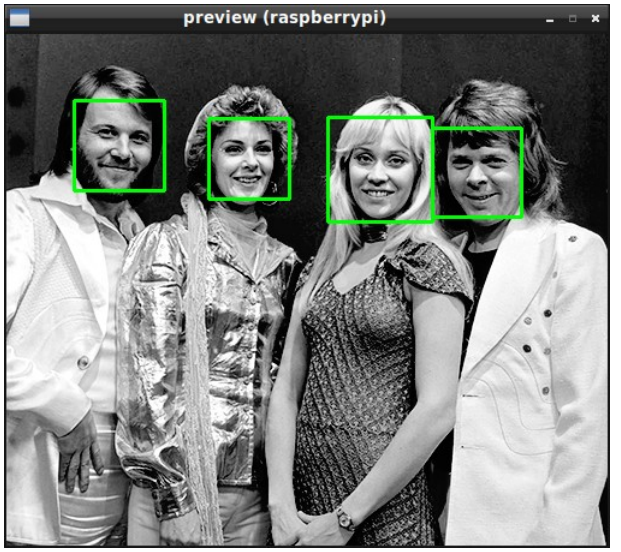
3



5



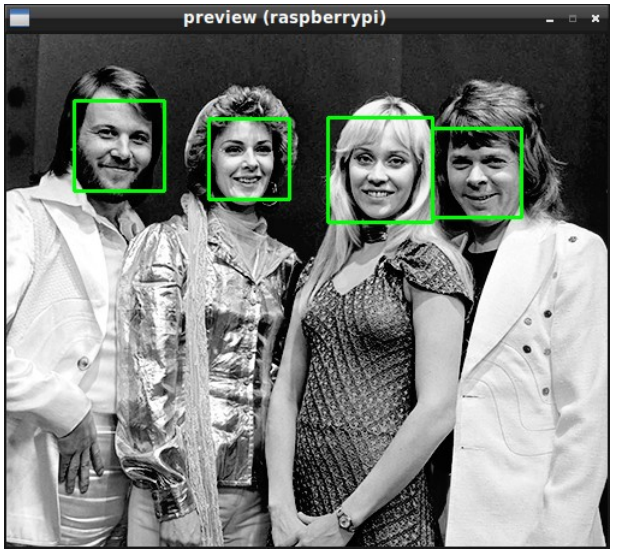
10



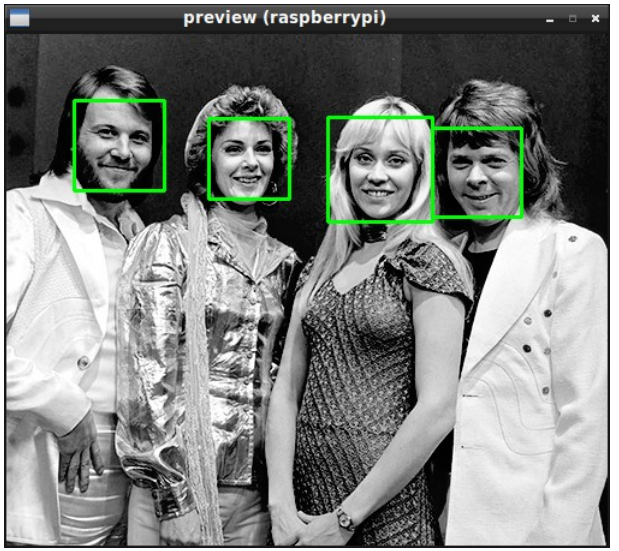
20

minSize(x, y)

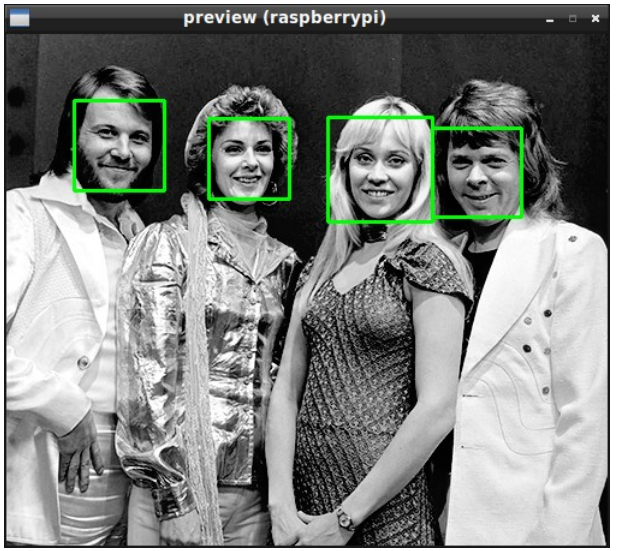
- scaleFactor=1.1, minNeighbosr=5



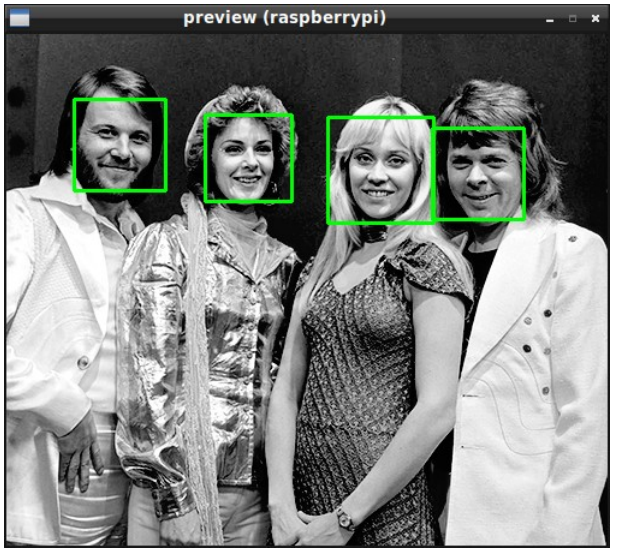
minSize=(15, 15)



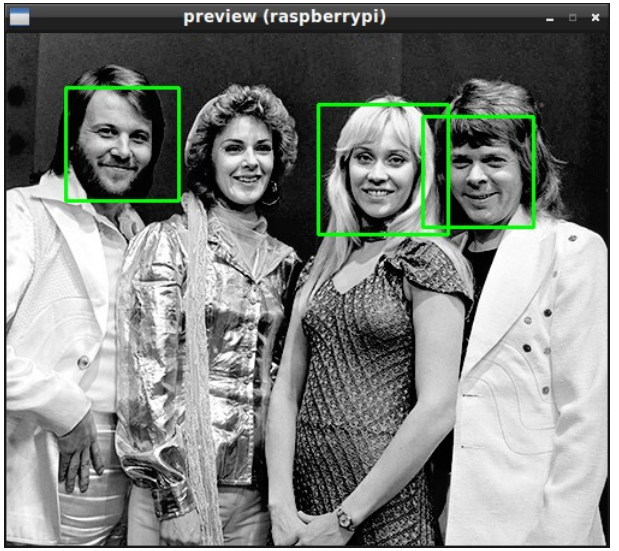
(30, 30)



(60, 60)



(90, 90)

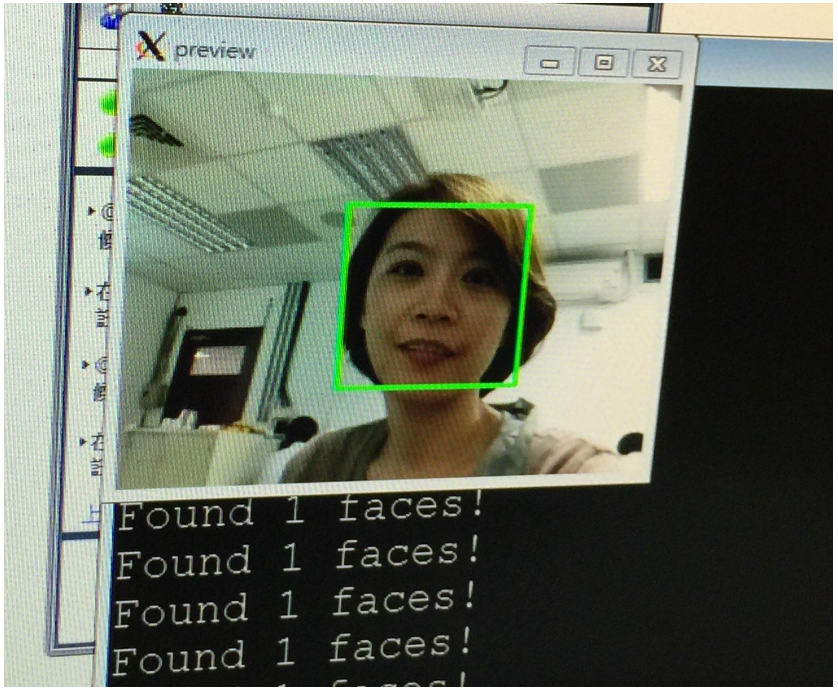


(120 ,120)

(150, 150)

讀取 Camera 並辨識

```
cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.1,
        minNeighbors=5,
        minSize=(30, 30),
        flags=cv2.cv.CV_HAAR_SCALE_IMAGE
    )

    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
```

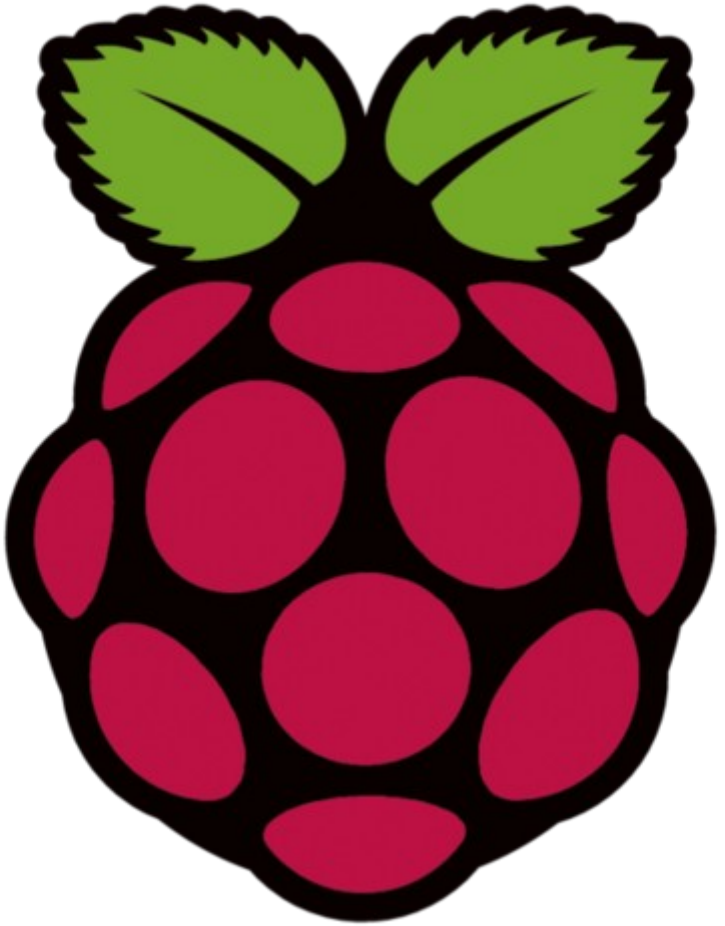


DEMO
camera_face_detect.py

$ cd ~/camera-opencv/05-face_detection

$ python camera_face_detect.py abba.png haarcascade_frontalface_default.xml

Raspberry Pi Rocks the World



Thanks