

# OPENCV in Python


## 電腦視覺與人臉辨識入門教學

---

吳智鴻 王啟勳

國立臺中教育大學 數位內容科技學系

2019/10/01

A solid blue horizontal bar spanning the width of the slide, located at the bottom.

# 大綱

---

- Opencv概說
- 在windows下安裝opencv
- 檢查是否安裝
- 程式一:讀檔秀檔
- 程式二:啟動鏡頭
- 程式三:按按鍵擷取圖檔
- 程式四:靜態人臉辨識
- 程式五:動態人臉辨識

# Opencv概說

---

OpenCV的全稱是Open Source Computer Vision Library，是一個跨平台的電腦視覺庫。OpenCV是由**英特爾**公司發起並參與開發，以BSD授權條款授權發行，可以在商業和研究領域中免費使用。OpenCV可用於開發即時的圖像處理、電腦視覺以及圖型識別程式。該程式庫也可以使用英特爾公司的IPP進行加速處理。

# Opencv概說

---

1. OpenCV可用於解決如下領域的問題：
2. 擴增實境、臉部辨識、手勢辨識、人機互動、動作辨識、運動跟蹤、物體辨識、圖像分割、機器人。
3. OpenCV用C++語言編寫，它的主要介面也是C++語言，但是依然保留了大量的C語言介面。也有大量的Python,Java and MATLAB/OCTAVE (版本2.5)的介面。
4. OpenCV可以在Windows、Linux、Android、Maemo、FreeBSD、OpenBSD、iOS、和Mac OS等平台上執行。

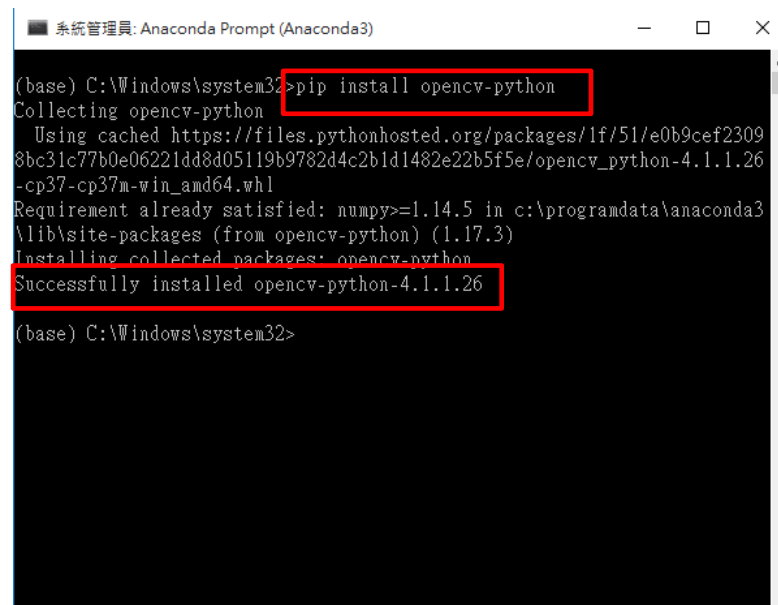
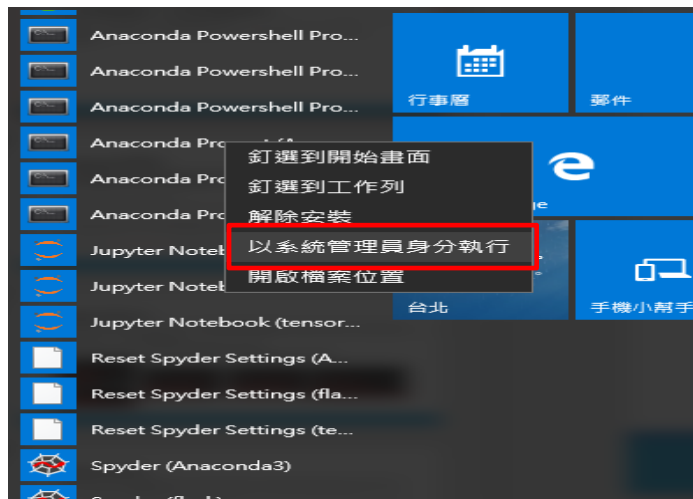
# 在windows下安裝opencv

Step1:左下角開始圖示---打開ananconda資料夾

Step2:在anaconda prompt上按滑鼠右鍵---點選” 以系統管理員身分執行” ---點選確定

Step3:輸入`pip install opencv-python`

Step4:安裝成功



# 檢查是否安裝

---

打開jupyter,輸入程式碼:

```
import cv2          #引入opencv函式庫  
cv2.__version__     #輸出目前安裝的版本
```

```
In [2]: 1 import cv2  
        2 cv2.__version__
```

```
Out[2]: '4.1.1'
```

```
In [ ]: 1
```

# 程式一:讀檔秀檔 (prg1)

Step1:網路上找一張圖,將檔名改成英文

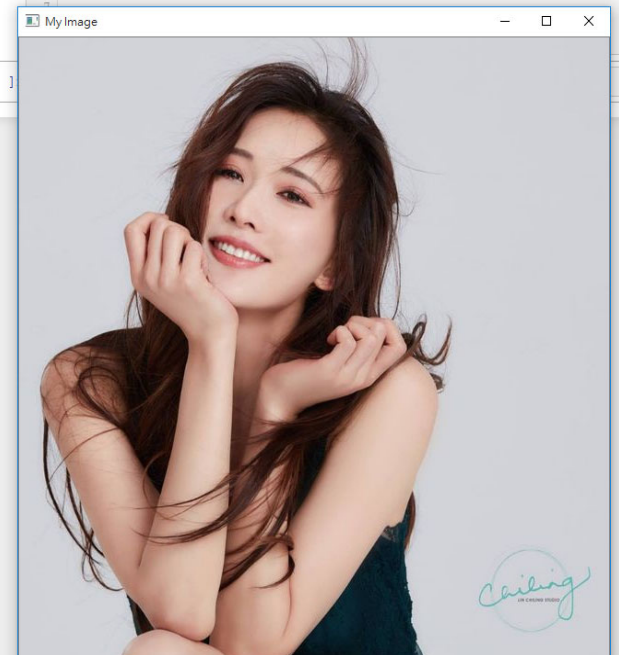
Stes2:打開Jupyter,輸入程式

```
import cv2  
cv2.__version__
```

```
'3.4.2'
```

```
img = cv2.imread('star1.png') # 讀取圖檔  
cv2.imshow('My Image',img) # 顯示圖片,第一個參數是視窗的名字,第二個參數是圖檔的變數名稱  
cv2.waitKey(0) # 參數是等待時間(單位為毫秒),若設定為 0 就表示持續等待至使用者按下任按鍵  
cv2.destroyAllWindows() # 按下任意鍵後關閉所有 OpenCV 的視窗。
```

```
In [*]: 1 import cv2 #引入opencv函式庫  
2  
3 img = cv2.imread('lin.jpg') # 讀取圖檔  
4 cv2.imshow('My Image',img) # 顯示圖片,第一個參數是視窗的名字,第二個參數是圖檔的變數名稱  
5 cv2.waitKey(0) # 參數是等待時間(單位為毫秒),若設定為 0 就表示持續等待至使用者按下任按鍵  
6 cv2.destroyAllWindows() # 按下任意鍵後關閉所有 OpenCV 的視窗。
```



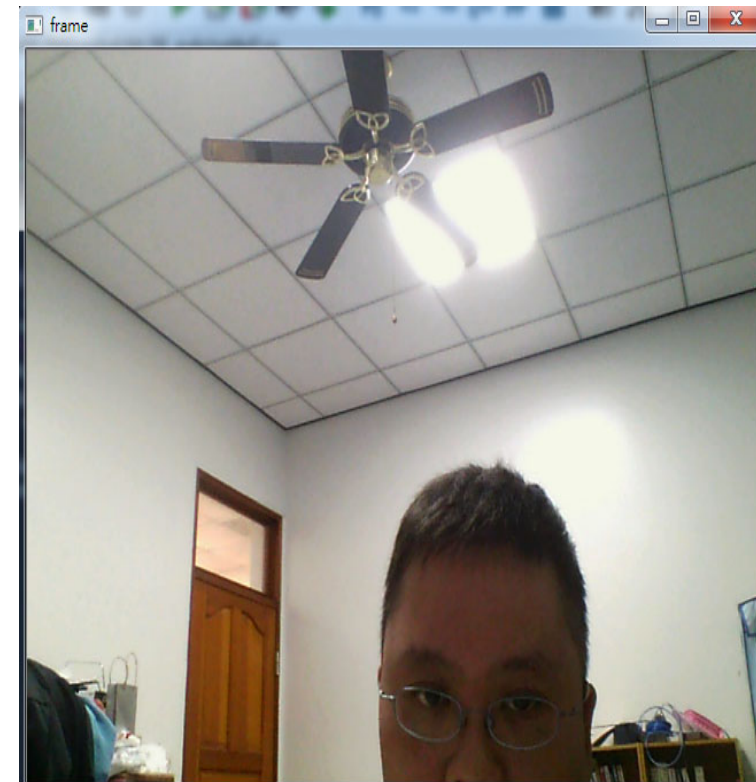
# 程式二:啟動鏡頭 (prg2)

Step2:輸入以下程式

```
import numpy as np
import cv2

cap = cv2.VideoCapture(0)

while(True):
    ret, frame = cap.read()#傳回值有兩個,第一個是否有讀到圖片, 值為True或False, 第二個是擷取當前的一幀圖片
    cv2.imshow('frame',frame)
    if cv2.waitKey(5) == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```



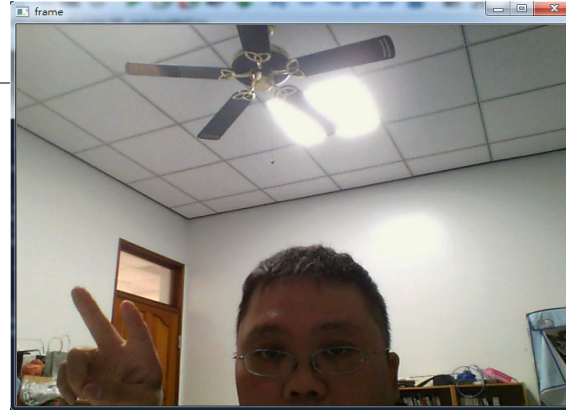


# 程式三:按按鍵擷取圖檔

## Step:輸入程式

```
import cv2
cap = cv2.VideoCapture(0)
while(True):
    ret, frame = cap.read()
    cv2.imshow("frame", frame)
    if cv2.waitKey(1) == ord('q'):
        cv2.imwrite("test.png", frame) #儲存成test.png
        break

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



Test.png  
儲存後結果

# Exercise#1

---

Requirement:

將檔案另存成另外名稱。Ex. Adt106001.png

可以按 s 鍵儲存檔案。

# 程式四:靜態人臉辨識

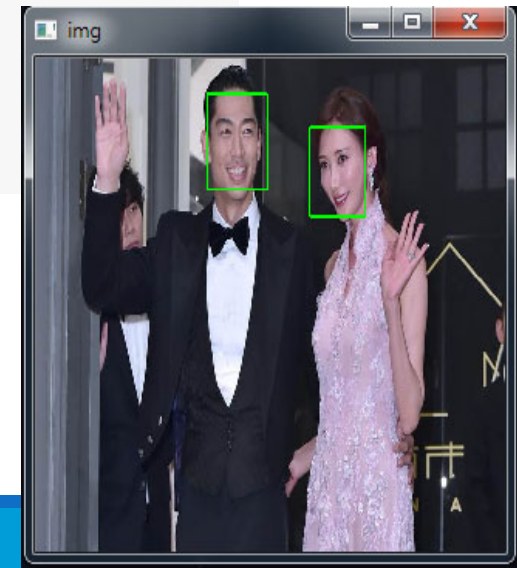
需確認位置

```
import cv2
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml') # 載入分類器

img = cv2.imread('star1.png') # 轉成灰階圖片
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # 需轉換成灰階

# 偵測臉部
faces = face_cascade.detectMultiScale(gray, scaleFactor=1.08, minNeighbors=5, minSize=(32, 32)) # 繪製人臉部份的方框
for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2) # (0, 255, 0) 欄位可以變更方框顏色(Blue, Green, Red) # 顯示成果

cv2.namedWindow('img', cv2.WINDOW_NORMAL) # 正常視窗大小
cv2.imshow('img', img) # 秀出圖片
cv2.imwrite("result.jpg", img) # 保存圖片
cv2.waitKey(0) # 等待按下任一按鍵
cv2.destroyAllWindows() # 關閉視窗
```



# Exercise#2

---

## Requirement

把人臉方框印成紅色

找一個兩個人的照片，把faces印出來看看

# 把faces印出來看看

```
import cv2
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml') # 載入分類器

img = cv2.imread('star1.png') # 轉成灰階圖片
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# 偵測臉部
faces = face_cascade.detectMultiScale(gray, scaleFactor=1.08, minNeighbors=5, minSize=(32, 32)) # 繪製人臉部份的方框
for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2) # (0, 255, 0) 欄位可以變更方框顏色(Blue, Green, Red) # 顯示成果

cv2.namedWindow('img', cv2.WINDOW_NORMAL) # 正常視窗大小
cv2.imshow('img', img) # 秀出圖片
cv2.imwrite("result.jpg", img) # 保存圖片
cv2.waitKey(0) # 等待按下任一按鍵
cv2.destroyAllWindows() # 關閉視窗
```

faces

```
array([[185, 69, 140, 140]], dtype=int32)
```

(x,y), (w, h)

# 誤判的情況(star2.png)

```
import cv2
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml') # 載入分類器

img = cv2.imread('star2.png') # 轉成灰階圖片
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# 偵測臉部
faces = face_cascade.detectMultiScale(gray, scaleFactor=1.08, minNeighbors=5, minSize=(32, 32)) # 繪製人臉部份的方
for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2) # (0, 255, 0) 欄位可以變更方框顏色(Blue, Green, Red)

cv2.namedWindow('img', cv2.WINDOW_NORMAL) # 正常視窗大小
cv2.imshow('img', img) # 秀出圖片
cv2.imwrite("result.jpg", img) # 保存圖片
cv2.waitKey(0) # 等待按下任一按鍵
cv2.destroyAllWindows() # 關閉視窗
```

faces

```
array([[425, 90, 100, 100],
       [730, 85, 106, 106],
       [735, 203, 121, 121],
       [ 89, 95, 117, 117]], dtype=int32)
```

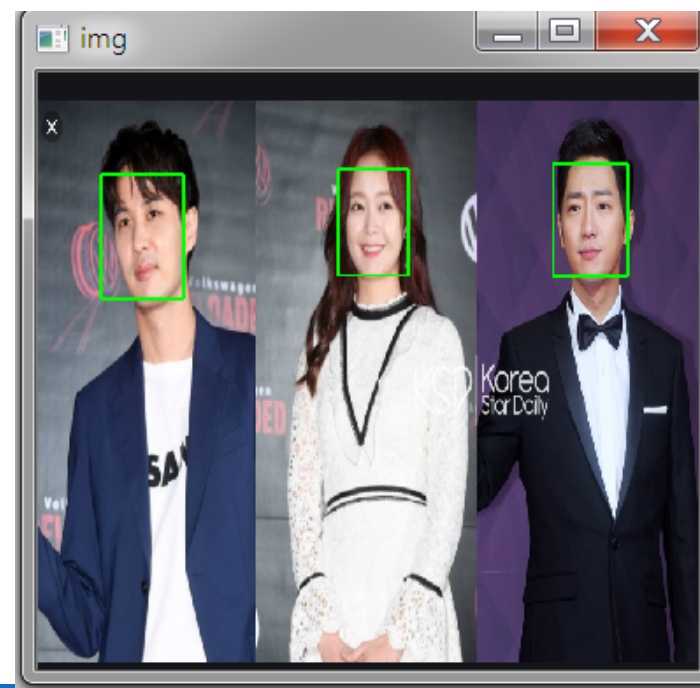


# Exercise#3

修正誤判的狀況

上網搜尋一下參數的意義

試著調整參數，以獲得正確的結果



# Face\_cascade.detectMultiScale ()

## 參數介紹

---

上網搜尋一下參數的意義



# 透過修正參數，得到較正確的結果

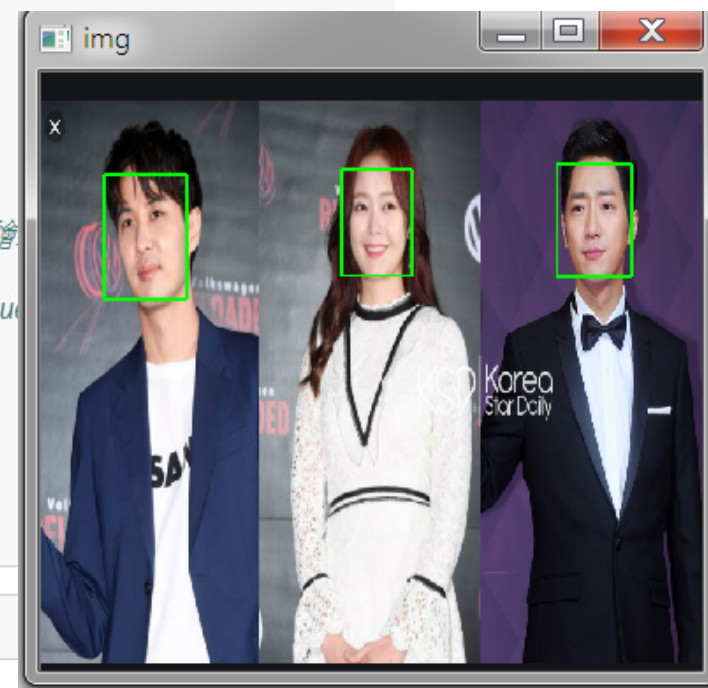
```
import cv2
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml') # 載入分類器

img = cv2.imread('star2.png') # 轉成灰階圖片
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# 偵測臉部
faces = face_cascade.detectMultiScale(gray, (1.1, 1.1), 3) # 繪出方框
for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2) # (0, 255, 0) 欄位可以變更方框顏色(Blue)

cv2.namedWindow('img', cv2.WINDOW_NORMAL) # 正常視窗大小
cv2.imshow('img', img) # 秀出圖片
cv2.imwrite("result.jpg", img) # 保存圖片
cv2.waitKey(0) # 等待按下任一按鍵
cv2.destroyAllWindows() # 關閉視窗
```

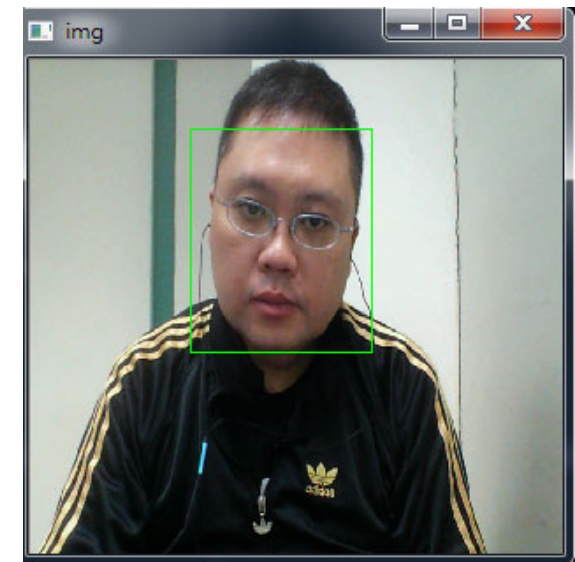
faces



# 程式五:動態人臉辨識

```
import cv2# 載入分類器

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
# 從視訊鏡頭擷取影片
cap = cv2.VideoCapture(0)
while True:
    # Read the frame
    _, img = cap.read()
    # 轉成灰階
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # 偵測臉部
    faces = face_cascade.detectMultiScale(gray, 1.1, 4)
    # 繪製人臉部份的方框
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)
    # 顯示成果
    cv2.namedWindow('img', cv2.WINDOW_NORMAL) #正常視窗大小
    cv2.imshow('img', img) #秀出圖片
    if cv2.waitKey(30) == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```



# Exercise#4

---

1. 按下s可以儲存照片 `save.png`
2. 偵測到人臉的話就自動儲存照片 `people.png`（安全監控）
3. 列印出幾個人臉 利用 `len()`函數
4. 以及臉部區域

# Exercise#5

---

1. 偵測到人臉的話就自動儲存照片 people.png（安全監控）
2. 依照有幾個人就存幾張照片  
people1.png, people2.png, …..

提示：利用str(i) 與字串相加函數  
利用cv2.imwrite

3. 裁切臉部區域存檔即可

# 裁切影像

首先按照普通的方式，將圖片用 OpenCV 的 `imread` 函數讀取進來：

```
import cv2

# 讀取圖檔
img = cv2.imread("lena.jpg")
```

這裡用 OpenCV 讀取進來的圖片 `img` 其實就是 NumPy 的陣列，如果想要對圖片進行裁切，就用索引的方式，將指定的區域取出即可：

```
# 裁切區域的 x 與 y 座標 (左上角)
x = 100
y = 100

# 裁切區域的長度與寬度
w = 250
h = 150

# 裁切圖片
crop_img = img[y:y+h, x:x+w]
```

接著使用 OpenCV 的 `imshow` 顯示裁切的結果：

```
# 顯示圖片
cv2.imshow("cropped", crop_img)
cv2.waitKey(0)
```

# Exercise#4, 5解答

```
import cv2# 載入分類器

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
# 從視訊鏡頭擷取影片
cap = cv2.VideoCapture(0)
while True:
    # Read the frame
    _, img = cap.read()
    # 轉成灰階
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # 偵測臉部
    faces = face_cascade.detectMultiScale(gray, 1.1, 4)

    i=1
    # 繪製人臉部份的方框
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)
        
    # 顯示成果

    cv2.namedWindow('Human Face Found!', cv2.WINDOW_NORMAL) #正常視窗大小
    cv2.imshow('Human Face Found!', img)#秀出圖片

    if cv2.waitKey(30)== ord('s'):
        cv2.imwrite("save.png",img)
    if cv2.waitKey(30)== ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

```
print(faces)
print('偵測到 {0}人臉 !'.format(len(faces)))
```