MediaPipe Face Detection

Eugene Lin

下載3.7.4、3.9也可以

- Python 3.7.4 - July 8, 2019
  - Download macOS 64-bit/32-bit installer
  - Download macOS 64-bit installer

快速ㄉ虛擬環境

- https://pythonviz.com/basic/visual-studio-code-virtual-environment-setup/

Picture

前置設定

```
1   # 導入需要模組
2   import cv2
3   import mediapipe as mp
4
5   # 選用臉部辨識方法
6   mp_face_detection = mp.solutions.face_detection
7   # 選用繪圖方法
8   mp_drawing = mp.solutions.drawing_utils
9   # 檔案存放位置 (./ <== 自動鍵入根目錄位置)
10  IMAGE_FILES = ['./facedetection/image/people5.jpg']
11
```

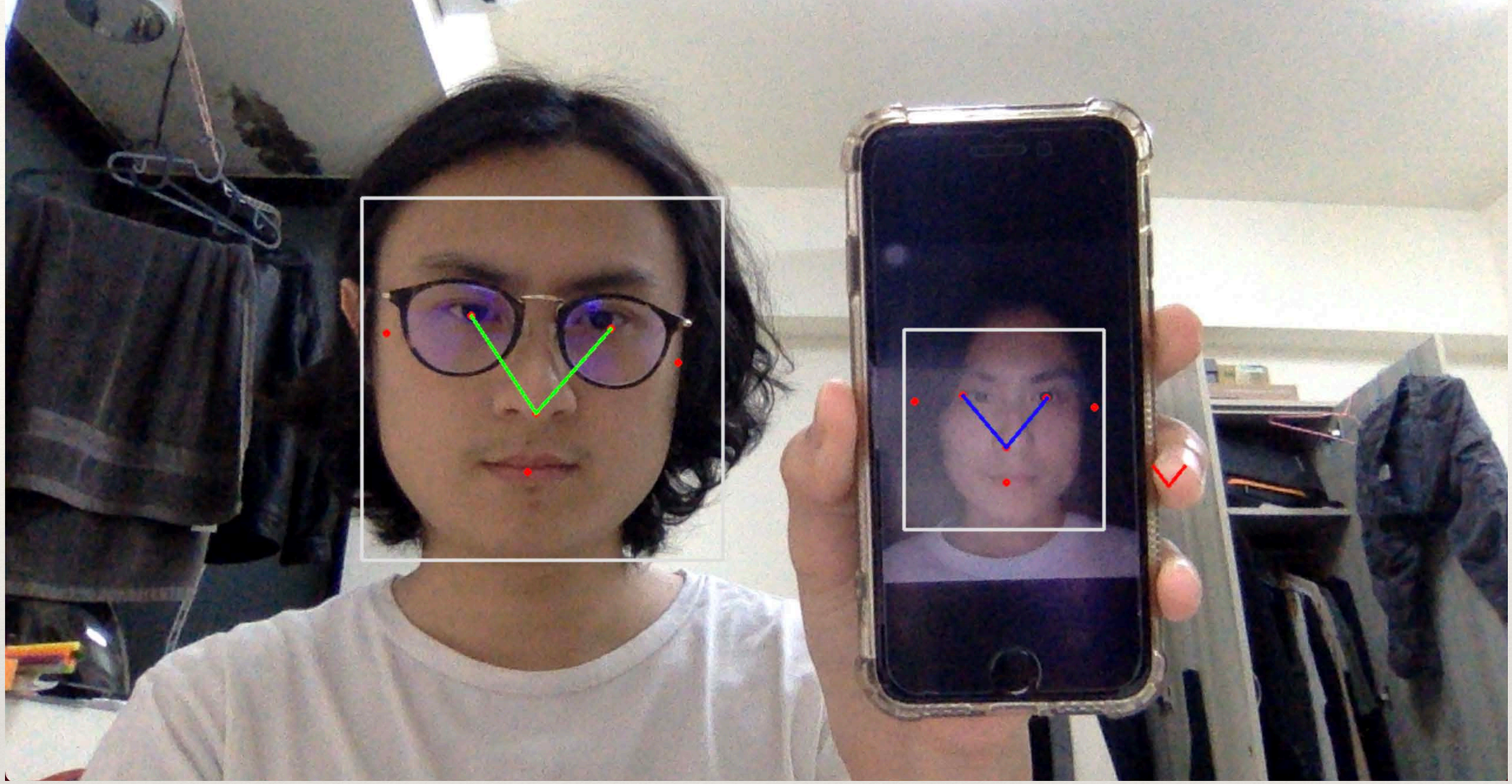
人臉辨識細節設定、影像辨識處理

```
12  # 設定臉部辨識細節
13  # model_selection = 0(2公尺內) 或 1(5公尺內) ; 偵測距離
14  # min_detection_confidence = float(0.0~1.0) ; 偵測結果信心度（如不需要很精準位置，可以調低）
15  with mp_face_detection.FaceDetection(
16      model_selection=1, min_detection_confidence=0.5) as face_detection:
17
18    # 遍歷資料夾內的每個檔案，並將資料夾內的資料轉換為具有 index 的 list 結構資料
19    for idx, file in enumerate(IMAGE_FILES):
20      image = cv2.imread(file)
21
22      # 先將資料轉換為 RGB 格式，使用 .process 處理欲辨識影像內容
23      results = face_detection.process(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
24
25      # 如果無偵測到任何臉部，忽略本次檔案，並繼續（避免程式崩潰）
26      if not results.detections:
27        continue
28
29      # 將原始檔案拷貝一份（避免更改原始檔案）
30      annotated_image = image.copy()
```

偵測結果存取、檔案寫入

```
31
32      # 將已處理完的影像資料，使用 .detection 存取點座標位置
33      # 因為無法預知會偵測到幾張臉，這裡使用迴圈遍歷所有偵測到的臉部結果
34      for detection in results.detections:
35
36          # 選取每個臉部的鼻子點座標，並列印出
37          print('Nose tip:')
38          print(mp_face_detection.get_key_point(
39              detection, mp_face_detection.FaceKeyPoint.NOSE_TIP))
40
41          # 將偵測到的所有臉座標存在剛剛拷貝的原始檔案的 annotated_image 變數裡
42          mp_drawing.draw_detection(annotated_image, detection)
43
44      # 新增一個名為 annotated_image 的 png 檔案，並將剛剛的結果(annotated_image)，寫入進去
45      # str(idx) 第1個檔案回傳 0，第2個檔案回傳 1，第375個檔案則回傳 374。 etc.
46      cv2.imwrite('./facedetection/image/annotated_image' + str(idx) + '.png', annotated_image)
```

結果



WebCam

前置設定

venv > lib > python3.9 > site-packages > mediapipe > python > solutions > __init__.py

```python
# Copyright 2020 The MediaPipe Authors.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#      http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

"""MediaPipe Solutions Python API."""

import mediapipe.python.solutions.drawing_styles
import mediapipe.python.solutions.drawing_utils
import mediapipe.python.solutions.face_detection
import mediapipe.python.solutions.face_mesh
import mediapipe.python.solutions.face_mesh_connections
import mediapipe.python.solutions.hands
import mediapipe.python.solutions.hands_connections
import mediapipe.python.solutions.holistic
import mediapipe.python.solutions.objectron
import mediapipe.python.solutions.pose
import mediapipe.python.solutions.selfie_segmentation
```

```python
import cv2
import mediapipe as mp
mp_face_detection = mp.solutions.face_detection # 選定要哪種解法並且存取進去這個大class裡面
mp_drawing = mp.solutions.drawing_utils # 標記結果點的函式

```

人臉辨識細節設定

venv > lib > python3.9 > site-packages > mediapipe > python > solutions > face_detection.py > ...

```
61
62  class FaceDetection(SolutionBase):
63    """MediaPipe Face Detection.
64
65    MediaPipe Face Detection processes an RGB image and returns a list of the
66    detected face location data.
67
68    Please refer to
69    https://solutions.mediapipe.dev/face_detection#python-solution-api
70    for usage examples.
71    """
72
73    def __init__(self, min_detection_confidence=0.5, model_selection=0):
74      """Initializes a MediaPipe Face Detection object.
75
76      Args:
77        min_detection_confidence: Minimum confidence value ([0.0, 1.0]) for face
78          detection to be considered successful. See details in
79          https://solutions.mediapipe.dev/face_detection#min_detection_confidence.
80        model_selection: 0 or 1. 0 to select a short-range model that works
81          best for faces within 2 meters from the camera, and 1 for a full-range
82          model best for faces within 5 meters. See details in
83          https://solutions.mediapipe.dev/face_detection#model_selection.
84      """
```

```
9   # with as
10  # 把執行的命令存放到一個叫做face_detection的變數裡
11  # 好處是不用擔心記憶體問題，因為with as 最後會自動完成file.close的命令
12  with mp_face_detection.FaceDetection(
13      model_selection=1, min_detection_confidence=0.5) as face_detection:
14
```

影像前置處理

```python

# 偵測是否成功打開，如果一直True就一直執行此迴圈
while cap.isOpened():
    success, image = cap.read() #cap.read()會回傳 True or False, image

    # 畫面長、寬
    height = image.shape[0]
    width = image.shape[1]

    # 雖然成功打開攝影機了，但多一層保護，避免空幀，如果有空幀，跳出cap.isOpened()迴圈
    # 在重複cap.read()指令，來再一次讀取影像內容
    if not success:
        print("Ignoring empty camera frame.")
        # If loading a video, use 'break' instead of 'continue'.
        continue

```

```
print(type(image))

PROBLEMS    OUTPUT    DEBUG CO

<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
```

效能改進、人臉辨識處理

venv > lib > python3.9 > site-packages > mediapipe > python > solutions > face_detection.py > ...

```
96
97      def process(self, image: np.ndarray) -> NamedTuple:
98        """Processes an RGB image and returns a list of the detected face location data.
99
100       Args:
101         image: An RGB image represented as a numpy ndarray.
102
103       Raises:
104         RuntimeError: If the underlying graph throws any error.
105         ValueError: If the input image is not three channel RGB.
106
107       Returns:
108         A NamedTuple object with a "detections" field that contains a list of the
109         detected face location data.
110       """
111
112       return super().process(input_data={'image': image})
113
```

```
30        # 將影像標記為不可寫入
31        # C contiguous結構，增進效能
32        image.flags.writeable = False
33        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB) # OpenCV開啟順序BGR，但是mp需要RGB才算得準
34        results = face_detection.process(image) # 人臉辨識處理
35
36        # 將影像重新更改為可寫入
37        image.flags.writeable = True
38        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
39
```

numpy.ndarray.flags

numpy.ndarray.flags

attribute

ndarray.flags

Information about the memory layout of the array.

Notes

The **flags** object can be accessed dictionary-like (as in `a.flags['WRITEABLE']`), or by using lowercased attribute names (as in `a.flags.writeable`). Short flag names are only supported in dictionary access.

Attributes: (屬性)

C_CONTIGUOUS (C)

The data is in a single, C-style contiguous segment.

F_CONTIGUOUS (F)

The data is in a single, Fortran-style contiguous segment.

OWNDATA (O)

The array owns the memory it uses or borrows it from another object.

WRITEABLE (W)

The data area can be written to. Setting this to False locks the data, making it read-only. A view (slice, etc.) inherits WRITEABLE from its base array at creation time, but a view of a writeable array may be subsequently locked while the base array remains writeable. (The opposite is not true, in that a view of a locked array may not be made writeable. However, currently, locking a base object does not lock any views that already reference it, so under that circumstance it is possible to alter the contents of a locked array via a previously created writeable view onto it.) Attempting to change a non-writeable array raises a RuntimeError exception.

```
print(image.flags)
```

PROBLEMS OUTPUT DEBUG CO

```
  C_CONTIGUOUS : True
  F_CONTIGUOUS : False
  OWNDATA : True
  WRITEABLE : True
  ALIGNED : True
  WRITEBACKIFCOPY : False
  UPDATEIFCOPY : False
```

C_Contiguous

考慮二維數組`arr = np.arange(12).reshape(3,4)`。它看起來像這樣：

0	1	2	3
4	5	6	7
8	9	10	11

在計算機的內存中， 的值`arr`是這樣存儲的：

0	1	2	3	4	5	6	7	8	9	10	11

這意味著`arr`是一個**C 連續**數組，因為*行*存儲為連續的內存塊。下一個內存地址保存該行的下一行值。如果我們想向下移動一列，我們只需要跳過三個塊（例如，從 0 跳到 4 意味著我們跳過了 1,2 和 3）。

F_Contiguous

0	4	8
1	5	9
2	6	10
3	7	11

0	4	8	1	5	9	2	6	10	3	7	11

lib > python3.9 > site-packages > mediapipe > python > solutions > drawing_utils.py > ...

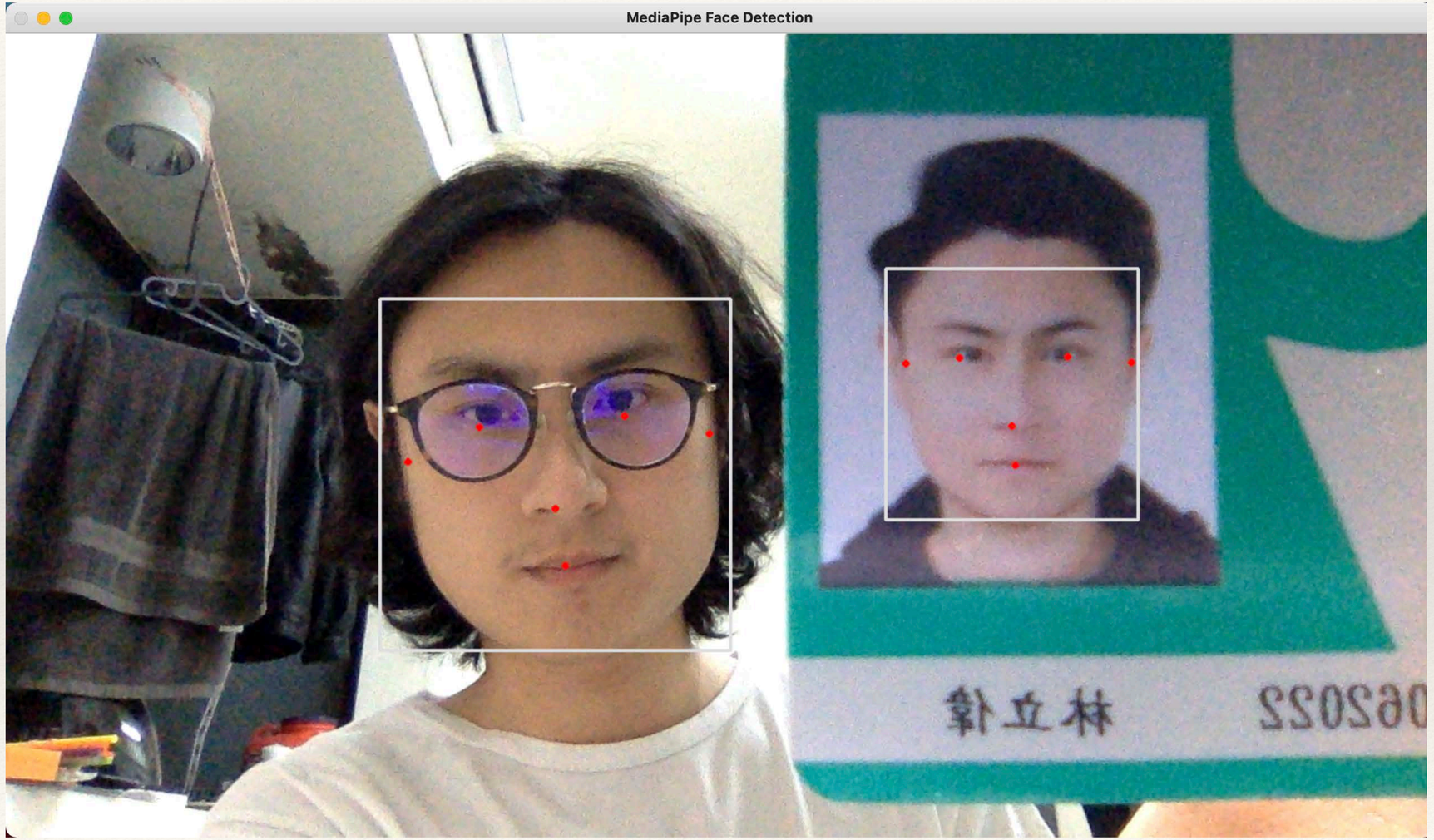
```python
def draw_detection(
    image: np.ndarray,
    detection: detection_pb2.Detection,
    keypoint_drawing_spec: DrawingSpec = DrawingSpec(color=RED_COLOR),
    bbox_drawing_spec: DrawingSpec = DrawingSpec()):
  """Draws the detction bounding box and keypoints on the image.

  Args:
    image: A three channel RGB image represented as numpy ndarray.
    detection: A detection proto message to be annotated on the image.
    keypoint_drawing_spec: A DrawingSpec object that specifies the keypoints'
      drawing settings such as color, line thickness, and circle radius.
    bbox_drawing_spec: A DrawingSpec object that specifies the bounding box's
      drawing settings such as color and line thickness.

  Raises:
    ValueError: If one of the followings:
      a) If the input image is not three channel RGB.
      b) If the location data is not relative data.
  """
```

畫出來！

```python
# .detections 存取處理完的結果，會回傳 True or False 以及 list結構資料
if results.detections:

  # 從results.detections回傳的list抓出第一個值，放進變數 detection 跑一圈迴圈,etc...
  # 因為不知道有幾個臉，把偵測到的第一個臉的第1個點放到變數 detection 裏面(這樣下面迴圈就可以直接用了)
  # 接著跑一次迴圈的內容，這樣子就可以做到每一次的臉部位置都不會漏畫！！！
  for detection in results.detections:
      mp_drawing.draw_detection(image, detection) #畫哪張圖，然後把要畫的東西畫在此圖上

```

```
print(results.detections)

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

[ 這是一個 list !!!!
label_id: 0
score: 0.9343113303184509
location_data {
  format: RELATIVE_BOUNDING_BOX
  relative_bounding_box {
    xmin: 0.3759737014770508
    ymin: 0.45092010498046875
    width: 0.26785415410995483
    height: 0.4762759804725647
  }
  relative_keypoints {
    x: 0.4616844356060028
    y: 0.6166045069694519
  }
  relative_keypoints {
    x: 0.5580832362174988
    y: 0.5959683656692505
  }
  relative_keypoints {
    x: 0.513852059841156
    y: 0.7208510041236877
  }
  relative_keypoints {
    x: 0.5233160853385925
    y: 0.8073493838310242
  }
  relative_keypoints {
    x: 0.41933193802833557
    y: 0.6774657368659973
  }
  relative_keypoints {
    x: 0.6255189180374146
    y: 0.6291239857673645
  }
}
]
```

Hi!



取值座標

```python
# 自動化宣告每個人臉其臉部的各個點座標為全域變數
# 這樣就可以直接使用這些點座標
# 左邊的臉會先被畫，最多5個臉
for idx, multiface in enumerate(results.detections):
    print("========開始========")
    print("Face ID:",idx+1,"=========")

    print("座標================")
    # 把所有點座標自動宣告成全域變數，方便外部直接取用
    # 例：第一張臉的鼻子X座標，命名範例 ==> NOSE_TIP_X_1
    #     第二張臉的嘴巴Y座標，命名範例 ==> MOUTH_CENTER_Y_2
    #========================
    # mp_face_detection.FaceKeyPoint(i).name <=== i 傳入 0 則會回傳 RIGHT_EYE
    # mp_face_detection.FaceKeyPoint(i).name <=== i 傳入 1 則會回傳 LEFT_EYE
    # multiface.location_data.relative_keypoints[i].x <=== i 傳入 0 則會回傳 RIGHT_EYE的 x 座標的值
    # multiface.location_data.relative_keypoints[i].y <=== i 傳入 1 則會回傳 LEFT_EYE的 y 座標的值
    # 輸入 i 回傳值請參考下面
    #========================
    # RIGHT_EYE = 0
    # LEFT_EYE = 1
    # NOSE_TIP = 2
    # MOUTH_CENTER = 3
    # RIGHT_EAR_TRAGION = 4
    # LEFT_EAR_TRAGION = 5
    #========================
    for i in range(6):
        globals()[str(mp_face_detection.FaceKeyPoint(i).name)+"_X_"+str(idx+1)] \
        = int((round(multiface.location_data.relative_keypoints[i].x,4)) * width)

        globals()[str(mp_face_detection.FaceKeyPoint(i).name)+"_Y_"+str(idx+1)] \
        = int((round(multiface.location_data.relative_keypoints[i].y,4)) * height)

        print(str(mp_face_detection.FaceKeyPoint(i).name)+"_X_"+str(idx+1)+" = "
        ,globals()[str(mp_face_detection.FaceKeyPoint(i).name)+"_X_"+str(idx+1)])

        print(str(mp_face_detection.FaceKeyPoint(i).name)+"_Y_"+str(idx+1)+" = "
        ,globals()[str(mp_face_detection.FaceKeyPoint(i).name)+"_Y_"+str(idx+1)])
```

list 的 index

此 index 裡的值

0 Ln 67, Col 26 Spaces: 2

for 循环使用 enumerate

```
>>> seq = ['one', 'two', 'three']
>>> for i, element in enumerate(seq):
...     print i, element
...
0 one
1 two
2 three
```

```
========開始========
Face ID: 1 =========
座標================
RIGHT_EYE_X_1 =  665
RIGHT_EYE_Y_1 =  381
LEFT_EYE_X_1 =  782
LEFT_EYE_Y_1 =  381
NOSE_TIP_X_1 =  727
NOSE_TIP_Y_1 =  445
MOUTH_CENTER_X_1 =  726
MOUTH_CENTER_Y_1 =  504
RIGHT_EAR_TRAGION_X_1 =  598
RIGHT_EAR_TRAGION_Y_1 =  428
LEFT_EAR_TRAGION_X_1 =  842
LEFT_EAR_TRAGION_Y_1 =  427

偵測到第1張臉
========結束========
```

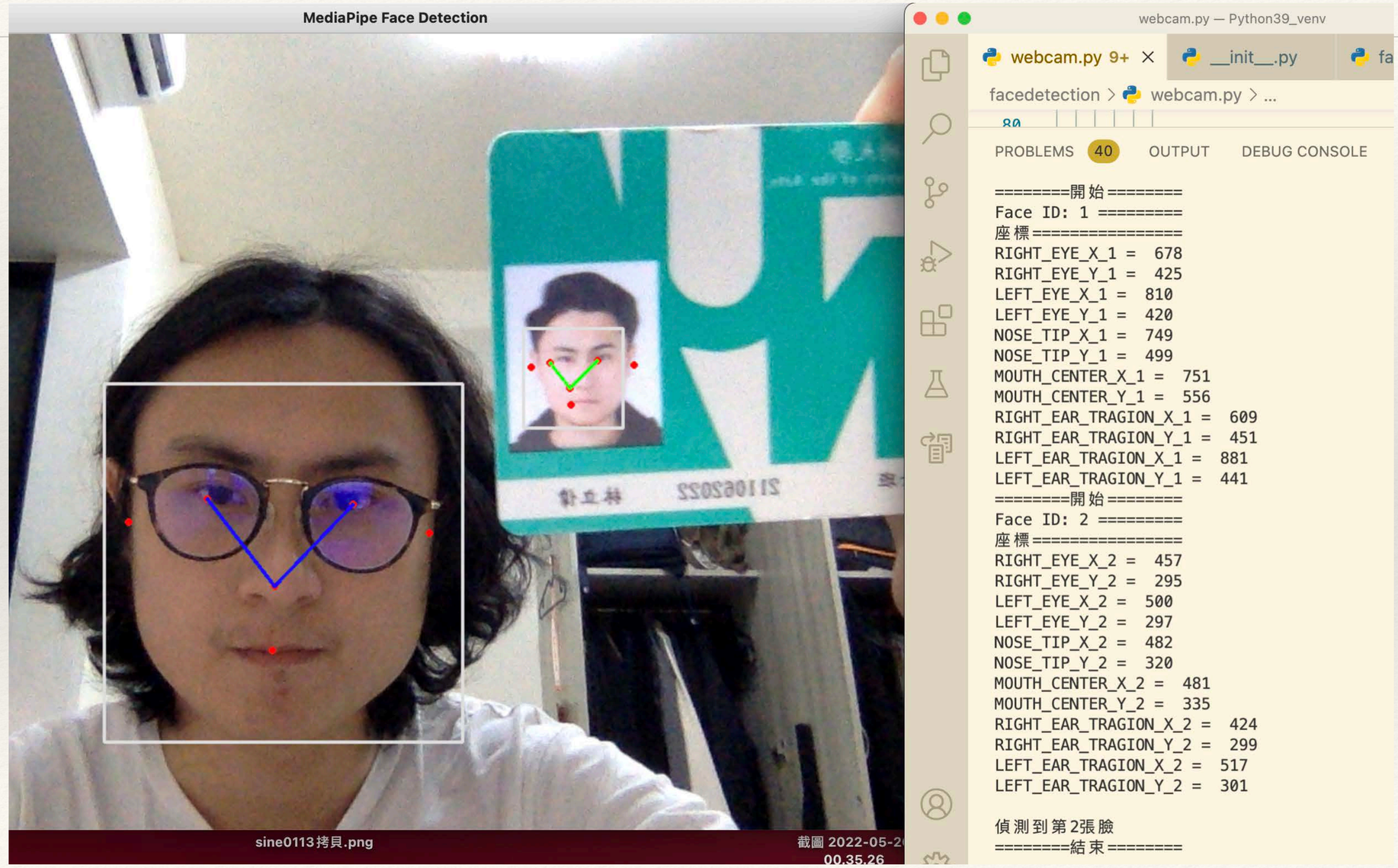
使用痊癒變數、畫線

```
86
87      # 這邊放全域變數要做的事情
88      # 這邊可以使用剛剛宣告好並且取值成功的全域變數了！！！ ：ＤＤＤ
89      try:
90        cv2.line(image,(RIGHT_EYE_X_1 ,RIGHT_EYE_Y_1),(NOSE_TIP_X_1 ,NOSE_TIP_Y_1),(255,0,0),2)
91        cv2.line(image,(LEFT_EYE_X_1 ,LEFT_EYE_Y_1),(NOSE_TIP_X_1 ,NOSE_TIP_Y_1),(255,0,0),2)
92
93        cv2.line(image,(RIGHT_EYE_X_2 ,RIGHT_EYE_Y_2),(NOSE_TIP_X_2 ,NOSE_TIP_Y_2),(0,255,0),2)
94        cv2.line(image,(LEFT_EYE_X_2 ,LEFT_EYE_Y_2),(NOSE_TIP_X_2 ,NOSE_TIP_Y_2),(0,255,0),2)
95
96        cv2.line(image,(RIGHT_EYE_X_3 ,RIGHT_EYE_Y_3),(NOSE_TIP_X_3 ,NOSE_TIP_Y_3),(0,0,255),2)
97        cv2.line(image,(LEFT_EYE_X_3 ,LEFT_EYE_Y_3),(NOSE_TIP_X_3 ,NOSE_TIP_Y_3),(0,0,255),2)
98
99        cv2.line(image,(RIGHT_EYE_X_4 ,RIGHT_EYE_Y_4),(NOSE_TIP_X_4 ,NOSE_TIP_Y_4),(255,0,255),2)
100       cv2.line(image,(LEFT_EYE_X_4 ,LEFT_EYE_Y_4),(NOSE_TIP_X_4 ,NOSE_TIP_Y_4),(255,0,255),2)
101
102       cv2.line(image,(RIGHT_EYE_X_5 ,RIGHT_EYE_Y_5),(NOSE_TIP_X_5 ,NOSE_TIP_Y_5),(255,255,255),2)
103       cv2.line(image,(LEFT_EYE_X_5 ,LEFT_EYE_Y_5),(NOSE_TIP_X_5 ,NOSE_TIP_Y_5),(255,255,255),2)
104
105       cv2.line(image,(RIGHT_EYE_X_6 ,RIGHT_EYE_Y_6),(NOSE_TIP_X_6 ,NOSE_TIP_Y_6),(100,55,255),2)
106       cv2.line(image,(LEFT_EYE_X_6 ,LEFT_EYE_Y_6),(NOSE_TIP_X_6 ,NOSE_TIP_Y_6),(100,55,255),2)
107
108     # 雖然宣告了2,3,4,5張臉的數值為全域變數，但似乎在臉還沒出現之前，這些全域變數會沒辦法取用
109     # 這邊使用例外處理來讓程式忽略 NameError 避免程式崩潰
110     # 程式本身沒有問題，但因 except 內部必須告知例外處理資訊，這邊單純pass
111     except:
112       pass
113
114     # 永遠一直 print 偵測到第幾張臉
115     finally:
116       if (idx+1) <= 6 :
117         print("偵測到第"+str(idx+1)+"張臉")
118       else :
119         print("超過6張臉")
120   print("========結束========")
121   print(" ")
```

畫面輸出

```
120             # 把畫面翻轉成鏡像並輸出
121             cv2.imshow('MediaPipe Face Detection', cv2.flip(image, 1))
122             if cv2.waitKey(5) & 0xFF == 27: # 每5ms輸出一次畫面 且 檢查按鍵有沒有 = Esc
123                 break
124
125     cap.release()
```

Hi


MediaPipe Face Detection
sine0113拷貝.png
webcam.py — Python39_venv
webcam.py 9+
__init__.py
facedetection > webcam.py > ...
PROBLEMS 40
OUTPUT
DEBUG CONSOLE
========開始========
Face ID: 1 =========
座標================
RIGHT_EYE_X_1 = 678
RIGHT_EYE_Y_1 = 425
LEFT_EYE_X_1 = 810
LEFT_EYE_Y_1 = 420
NOSE_TIP_X_1 = 749
NOSE_TIP_Y_1 = 499
MOUTH_CENTER_X_1 = 751
MOUTH_CENTER_Y_1 = 556
RIGHT_EAR_TRAGION_X_1 = 609
RIGHT_EAR_TRAGION_Y_1 = 451
LEFT_EAR_TRAGION_X_1 = 881
LEFT_EAR_TRAGION_Y_1 = 441
========開始========
Face ID: 2 =========
座標================
RIGHT_EYE_X_2 = 457
RIGHT_EYE_Y_2 = 295
LEFT_EYE_X_2 = 500
LEFT_EYE_Y_2 = 297
NOSE_TIP_X_2 = 482
NOSE_TIP_Y_2 = 320
MOUTH_CENTER_X_2 = 481
MOUTH_CENTER_Y_2 = 335
RIGHT_EAR_TRAGION_X_2 = 424
RIGHT_EAR_TRAGION_Y_2 = 299
LEFT_EAR_TRAGION_X_2 = 517
LEFT_EAR_TRAGION_Y_2 = 301
偵測到第2張臉
========結束========

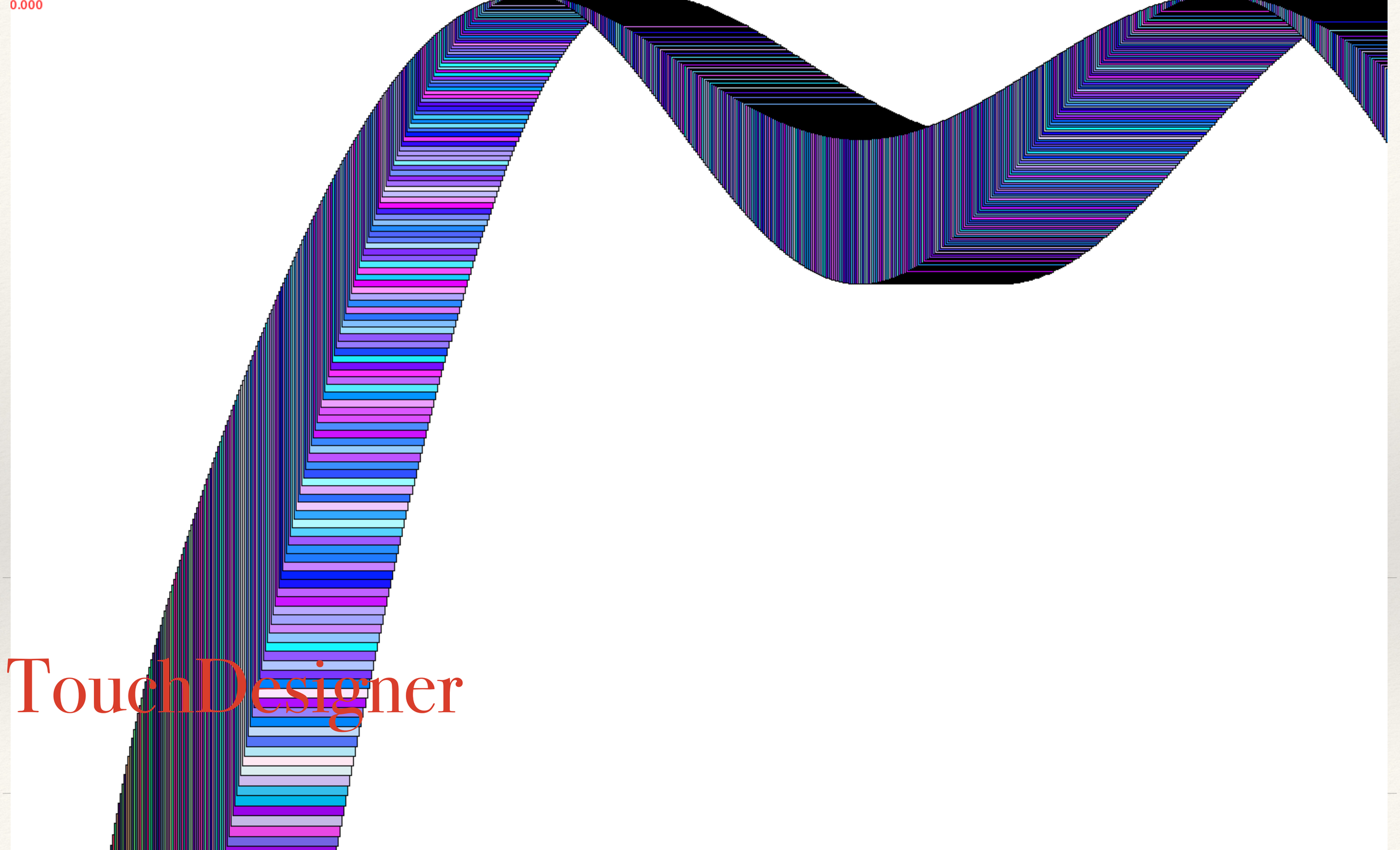
Video

唯一差別

```
6    # 影像輸入，可以自己測試看看
7    # cap = cv2.VideoCapture('./facedetection/image/4facevideo.mp4')
8    cap = cv2.VideoCapture('./facedetection/image/deepdream2.mp4')
9    # cap = cv2.VideoCapture('./facedetection/image/Bruno Mars.mp4')
10   # cap = cv2.VideoCapture('./facedetection/image/Daft Punk.mp4')
11   # cap = cv2.VideoCapture('./facedetection/image/Pentatonix.mp4')
```

最高紀錄

```
========開始========
Face ID: 35 =========
座標================
RIGHT_EYE_X_35 =  697
RIGHT_EYE_Y_35 =  498
LEFT_EYE_X_35 =  708
LEFT_EYE_Y_35 =  494
NOSE_TIP_X_35 =  718
NOSE_TIP_Y_35 =  507
MOUTH_CENTER_X_35 =  724
MOUTH_CENTER_Y_35 =  516
RIGHT_EAR_TRAGION_X_35 =  686
RIGHT_EAR_TRAGION_Y_35 =  500
LEFT_EAR_TRAGION_X_35 =  710
LEFT_EAR_TRAGION_Y_35 =  495
超過6張臉
========結束========
```


0.000
TouchDesigner

Reference

- https://google.github.io/mediapipe/solutions/face_detection.html
- https://www.analyticsvidhya.com/blog/2022/03/detect-the-faces-in-the-image-using-the-mediapipe-library/
- https://numpy.org/doc/stable/reference/generated/numpy.ndarray.flags.html
- https://stackoverflow.com/questions/26998223/what-is-the-difference-between-contiguous-and-non-contiguous-arrays
- https://blog.csdn.net/itnerd/article/details/105127984
- https://reurl.cc/8oDpEd