手勢辨識

報告：鍾權彥、翁得恩、楊舒凱
學號：411500399、411500506、411506529
日期：2022/12/26

大綱

01 研究動機與目的

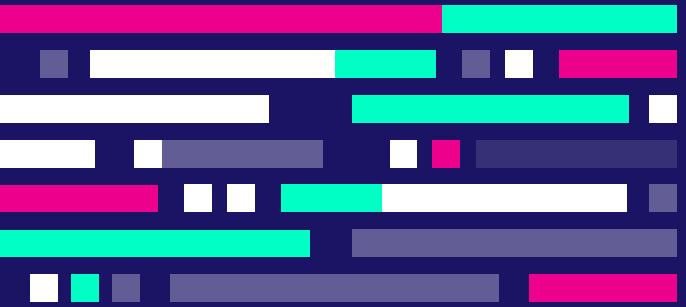
02 研究方法

03 研究成果

04 心得

01

研究動機與目的



研究動機與目的

在這個人工智慧的時代，視覺辨識是不可少的，除了從前的色塊追蹤，現在由於資料庫模型逐漸成熟，開始能判斷人臉、肢體動作甚至是手勢等，我之前曾經寫過色塊的物體追蹤，希望能藉此機會研究更高難度的手勢辨識並好好利用之前的經驗，在這次專題中設計一個能讓生活更方便的小工具。

02

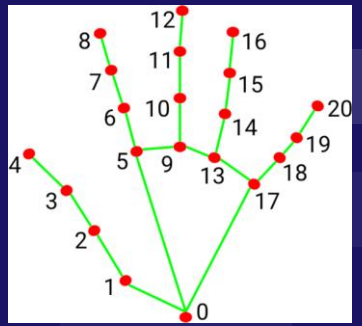
研究方法

研究方法

經過資料的搜尋後，決定使用opencv來進行影像的處理，再利用Google所開發的Mediapipe多媒體機器學習模型來進行手勢的辨識，最後設計一套能實際運用的辨識功能。

opencv：取用相機畫面並進行尺寸處理，給予RGB色彩，並跳出獨立視窗。

Mediapipe：使用其中的手掌偵測功能(mp)讀取畫面中出現的手掌，並判斷出手上的21個節點並給予在畫面上的座標。

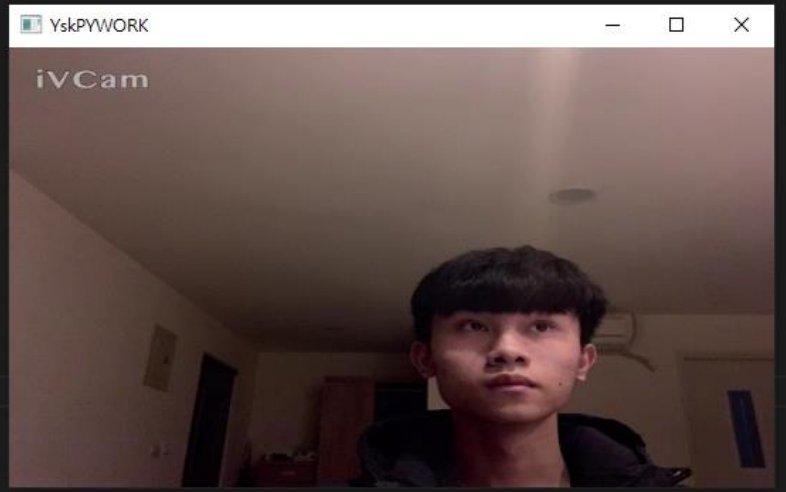


Opencv練習

透過網路上的教程，改寫程式碼，進行練習，取得鏡頭畫面

D: > python_work > opencvtest.py > ...

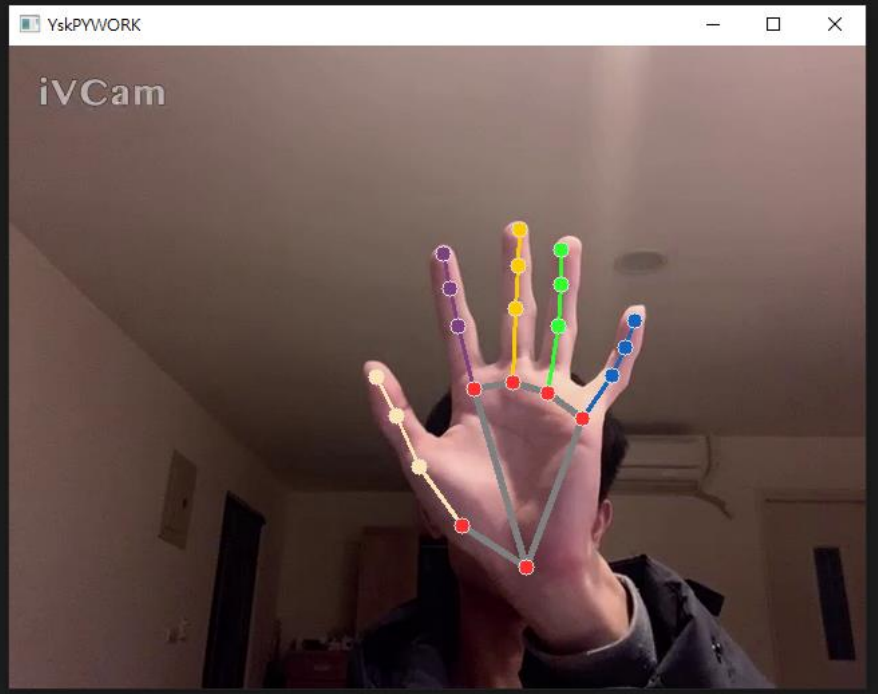
```python
import cv2
cap = cv2.VideoCapture(0)
if not cap.isOpened():
    print("Cannot open camera")
    exit()
w, h = 540, 310                         # 影像尺寸
while True:
    ret, frame= cap.read()              # 讀取影片的每一幀
    frame = cv2.resize(frame, (w,h))        # 縮小尺寸，加快處理效率
    if not ret:
        print("Cannot receive frame")   # 如果讀取錯誤，印出訊息
        break
    cv2.imshow('YskPYWORK', frame)      # 如果讀取成功，顯示該幀的畫面
    if cv2.waitKey(1) == ord('q'):      # 每一毫秒更新一次，直到按下 q 結束
        break
cap.release()                           # 所有作業都完成後，釋放資源
cv2.destroyAllWindows()                 # 結束所有視窗
```



Mediapipe練習-偵測手掌

D: > python_work > mediapipetest1.py > ...

```python
import cv2
import mediapipe as mp
mp_drawing = mp.solutions.drawing_utils          # mediapipe 繪圖方法
mp_drawing_styles = mp.solutions.drawing_styles  # mediapipe 繪圖樣式
mp_hands = mp.solutions.hands                    # mediapipe 偵測手掌方法
cap = cv2.VideoCapture(0)
with mp_hands.Hands(# mediapipe 啟用偵測手掌
    model_complexity=0,
    # max_num_hands=1,
    min_detection_confidence=0.5,
    min_tracking_confidence=0.5) as hands:
    if not cap.isOpened():
        print("Cannot open camera")
        exit()
    while True:
        ret, img = cap.read()
        if not ret:
            print("Cannot receive frame")
            break
        img2 = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)   # 將 BGR 轉換成 RGB
        results = hands.process(img2)                 # 偵測手掌
        if results.multi_hand_landmarks:
            for hand_landmarks in results.multi_hand_landmarks:
                # 將節點和骨架繪製到影像中
                mp_drawing.draw_landmarks(
                    img,hand_landmarks,mp_hands.HAND_CONNECTIONS,
                    mp_drawing_styles.get_default_hand_landmarks_style(),
                    mp_drawing_styles.get_default_hand_connections_style())
        cv2.imshow('YskPYWORK', img)
        if cv2.waitKey(5) == ord('q'):
            break    # 按下 q 鍵停止
cap.release()
cv2.destroyAllWindows()
```



Mediapipe練習-判斷手指是否伸直

根據得到的指節節點座標計算角度

```
def vector_2d_angle(v1, v2):
    v1_x = v1[0]
    v1_y = v1[1]
    v2_x = v2[0]
    v2_y = v2[1]
    try:
        angle_= math.degrees(math.acos((v1_x*v2_x+v1_y*v2_y)/(((v1_x**2+v1_y**2)**0.5)*((v2_x**2+v2_y**2)**0.5))))
    except:
        angle_ = 180
    return angle_
```

分別計算各手指的角度加入陣列

```
angle_ = vector_2d_angle(
    ((int(hand_[0][0])- int(hand_[2][0])),(int(hand_[0][1])-int(hand_[2][1]))),
    ((int(hand_[3][0])- int(hand_[4][0])),(int(hand_[3][1])- int(hand_[4][1])))
    )
angle_list.append(angle_)
```

Mediapipe練習-判斷手指是否伸直

```python
def hand_pos(finger_angle):
    f1 = finger_angle[0]   # 大拇指角度
    f2 = finger_angle[1]   # 食指角度
    f3 = finger_angle[2]   # 中指角度
    f4 = finger_angle[3]   # 無名指角度
    f5 = finger_angle[4]   # 小拇指角度
    # 小於 50 表示手指伸直，大於等於 50 表示手指捲縮
    if f1>=50 and f2>=50 and f3>=50 and f4>=50 and f5>=50:
        return '0'
    elif f1>=50 and f2<50 and f3>=50 and f4>=50 and f5>=50:
        return '1'
    elif f1>=50 and f2<50 and f3<50 and f4>=50 and f5>=50:
        return '2'
    elif f1>=50 and f2<50 and f3<50 and f4<50 and f5>50:
        return '3'
    elif f1>=50 and f2<50 and f3<50 and f4<50 and f5<50:
        return '4'
    elif f1<50 and f2<50 and f3<50 and f4<50 and f5<50:
        return '5'
    elif f1<50 and f2>=50 and f3>=50 and f4>=50 and f5<50:
        return '6'
    elif f1<50 and f2<50 and f3>=50 and f4>=50 and f5>=50:
        return '7'
    elif f1<50 and f2<50 and f3<50 and f4>=50 and f5>=50:
        return '8'
    elif f1<50 and f2<50 and f3<50 and f4<50 and f5>=50:
        return '9'
    else:
        return ''
```

再判斷各手指角度，確認所指的手勢

03

研究成果

研究成果1–猜拳

剛開始決定設計一款小遊戲，既然點擊的猜拳不好玩，那不如就製作一款真實的猜拳，透過手勢辨識判斷出的拳，並和電腦PK

```
# if f1<50 and f2>=50 and f3>=50 and f4>=50 and f5>=50:
#     return 'stone'
# elif f1>=50 and f2>=50 and f3>=50 and f4>=50 and f5>=50:
#     return 'stone'
# elif f1>=50 and f2<50 and f3<50 and f4>=50 and f5>=50:
#     return 'scissors'
# elif f1<50 and f2<50 and f3<50 and f4<50 and f5<50:
#     return 'paper'
# elif f1<50 and f2<50 and f3<50 and f4>=50 and f5>=50:
#     return 'scissors'
```

經過測試，能夠成功判斷手勢，但也發現了問題，就是猜拳是很講究出拳時機的遊戲，要如何判斷時機並且能跟電腦公平對戰成了一大問題。

研究成果1–猜拳

為了抓準時機，於是決定使用由玩家決定開始時機再進行倒數，經過資料的搜尋決定導入time模組進行測試。

測試影片

先使用openCV繪製出一個長方框，再讀取食指第二節點的座標，在放入方框內時開始計時，離開後會重製，在放入3秒後會將視窗關閉。

研究成果2–YT影片控制

經過各種測試，還是無法將猜拳做到公平且好玩，於是我決定改做別的東西，在一次吃飯看影片時發現，如果在吃油膩的東西時要切影片按暫停調聲音非常麻煩，於是決定設計一個能透過手勢控制Youtube的程式。

首先要想如何控制YT，在經過研究後發現，YT有很多鍵盤的快捷鍵可以使用，於是就上網尋找能控制鍵盤的python模組，最後使用keyboard模組經過測試能夠使用。

D: > python_work > keyboardtest.py

```
1  import keyboard
2
3  keyboard.press_and_release("h")
4  keyboard.press_and_release("e")
5  keyboard.press_and_release("l")
6  keyboard.press_and_release("l")
7  keyboard.press_and_release("o")
8  hello
```

研究成果2–YT影片控制

在能夠透過程式控制鍵盤後，開始設計控制的手勢，原先要使用手掌向左向右揮來控制，但在設計上需用到軌跡紀錄，且不太準確，最後決定使用單純的手勢指向控制。

```
if f1>50 and f2<50 and f3>50 and f4>50 and f5>50:
    if yf2-y0 <-20 and abs(xf2-x0)<40 :
        return 'up'
    elif yf2-y0 >20 and abs(xf2-x0)<60:
        return 'down'
elif f1<50 and f2>50 and f3>50 and f4>50 and f5>50:
    if xf1-x0 <-20 and abs(yf1-y0)<60 :
        return 'left'
    elif xf1-x0 >20 and abs(yf1-y0)<60:
        return 'right'
elif f1<50 and f2<50 and f3<50 and f4<50 and f5<50:
    return 'stop'
else:
    return ''
```

只指食指：
朝上 up
朝下 down
只指拇指：
朝右 right
朝左 left
五指張開：stop

透過取得的手指座標跟手掌中心座標來比較判斷指向方向，並且設定指向方向角度的限制。

研究成果2–YT影片控制

```
if text=='up':
    end = time.time()
    cv2.putText(img, '{:.2f}'.format(end-start), (400,300), cv2.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 5, cv2.LINE_AA)
    if (end-start)>=.5:
        keyboard.press_and_release("up")
        start = time.time()
        end =0
elif text=='down':
    end = time.time()
    cv2.putText(img, '{:.2f}'.format(end-start), (400,300), cv2.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 5, cv2.LINE_AA)
    if (end-start)>=.5:
        keyboard.press_and_release("down")
        start = time.time()
        end =0
elif text =='right':
    end = time.time()
    cv2.putText(img, '{:.2f}'.format(end-start), (400,300), cv2.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 5, cv2.LINE_AA)
    if (end-start)>=1:
        keyboard.press_and_release("shift+n")
        time.sleep(.5)
        start = time.time()
        end =0
elif text =='left':
    end = time.time()
    cv2.putText(img, '{:.2f}'.format(end-start), (400,300), cv2.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 5, cv2.LINE_AA)
    if (end-start)>=1:
        keyboard.press_and_release("shift+p")
        time.sleep(.5)
        start = time.time()
        end =0
elif text=='stop':
    end = time.time()
    cv2.putText(img, '{:.2f}'.format(end-start), (400,300), cv2.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 5, cv2.LINE_AA)
    if (end-start)>=2:
        keyboard.press_and_release("space")
        time.sleep(.5)
        start = time.time()
        end =0
```

判斷回傳值進行音量調整、切換影片與暫停。
使用time模組中的sleep功能設定需持續指相同動作一段時間才會執行，避免誤判。

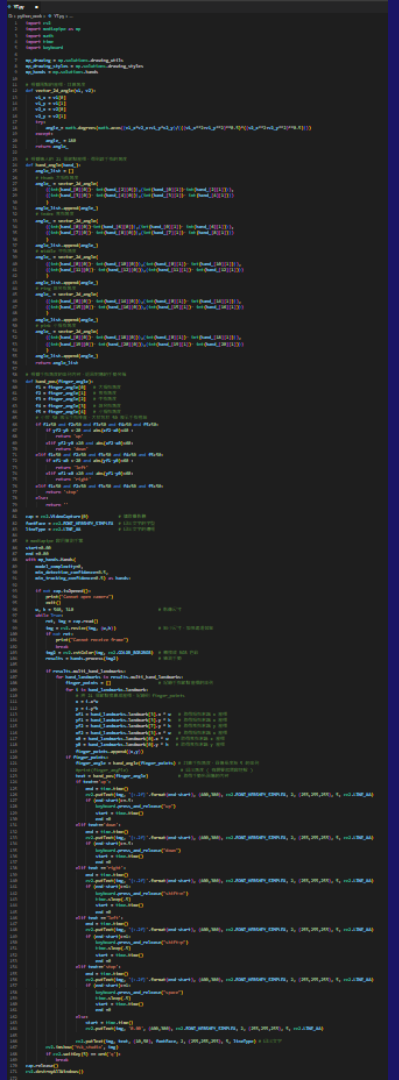
研究成果2-YT影片控制

←皆無移動滑鼠
與觸碰鍵盤

測試結果

研究成果2–YT影片控制

此程式設計完成，後續還可添加其他功能，如快進/退10秒、5秒，也可再提升手勢方向判斷的精準度，甚至能透過判斷移動軌跡設計新功能。



04

心得

心得–鍾權彥411500399

這次的AI程式語言學到的程式是Python，是我從來沒有看過的，以前只有學過C和C++，不過一直以來都有聽過這個叫Python的程式語言，大家好像給的回答都是它比起C語言來說更加的簡單好上手，但其實也沒有想像中的那麼容易，還是要花時間去理解它，融會貫通後才能再去做之後的運用，這次我們做的報告是有關於手掌辨識，隨著科技不斷的進步，關於辨識這個功能也在不斷的進化，從手部辨識到甚至臉部辨識，我們做的報告主要分成兩項，這次很幸運的是有一個以前就有做過類似程式的組員，他對我們組的幫助很大，而且他對程式語言很感興趣很有一定的了解，所以主要都是他帶著我們做，第一項就是猜拳，跟電腦猜拳利用手部的節點座標形成的角度來去判斷，但很看重出拳的時機，第二項就是在不碰除到任何物品下去控制電腦的YouTube，同樣的也是用手掌的節電座標來去告訴程式該做出什麼動作，這次的課程很好的讓我們運用到日常生活上，也很期待能做出什麼不一樣的變化。

心得-翁得恩411500506

在這堂課前從來沒有學過Python，但是多多少少都有聽過這個程式語言，助教在上課我常常遲到所以每次幾乎都只有聽到半節課，大概只有第一次的時候準時到過。為了補足那些沒有到上課的時間，特地到YT上找到4小時初學者入門來看費了不少功夫。在之後做期末專題報告，在當下我其實沒有什麼想法也不知道有哪些題目可以拿來報告。但是我們這組友人之前就有做過類似的專題報告，讓我們有一個方向可以去想，這也讓我鬆了一口氣，對我們這組也帶來很大的幫助。最終這個專題我們決定要做手部辨識，這個手部辨識在是未完成品，就有做過猜拳、辨別手勢等。最後決定把它弄成可以控制影片像是停止、下一首、上一首、調高音量、調低音量等。藉由這次的期末專題報告除了更熟悉Python之外，對於OpenCV、Mediapipe、Time、Keyboard等模組有了些許的瞭解。在之後我也會多寫程式多練習，增進自己的實力。

心得–楊舒凱411506529

我以前就有自學過Python、Java、C++等程式語言，所以有一定的基礎，在高中時也寫過Python的爬蟲專題，曾經在機器人比賽中研究過視覺辨識色塊的物體追蹤，所以才會決定這次挑戰研究手勢辨識。在這次的專題中我重新複習了Python基礎，觀看了4個小時的影片，又上網查詢了許多資料，開始研究如何辨識手掌，我先是找到一篇完整教學的文章，照著文章中的步驟學習，但改用習慣用的VSCode作為編寫工具而沒有使用Jupyter，看完文章後我練習了opencv的鏡頭取用又成功使用Mediapipe辨識手部節點，在練習完後又看了其他教學影片，將文章中的程式完全搞懂。在經過這些學習後開始製作屬於我的程式，原先設想製作一款猜拳遊戲，但在經過測試後發現了實作的難度以及可行性，決定更改主題，雖然沒有完成猜拳遊戲，但在開發的過程中學會了Time模組的運用，在一次吃飯時有了靈感，決定製作影片切換器，找到了Keyboard模組來控制影片，在思索要使用何種手勢控制影片時找到了軌跡紀錄，但後來礙於精準度以及時間關係，還是使用了簡單的手勢配上方向判斷，再搭配上之前學過Time模組，完成了影片切換器。這次的程式雖然已經完成，但還有很多地方可以改進與修正，有兩隻手同時出現無法使用的Bug，還可以增加更多影片的功能，還能增加軌跡判斷，在經過這次的研究後，我也更加會使用Python撰寫程式，也對OpenCV、Mediapipe、Time、Keyboard等模組更加熟悉。

THANKS !

CREDITS: This presentation template was created by Slidesgo, incluiding icons by Flaticon, and infographics & images by Freepik.

參考資料

https://www.youtube.com/watch?v=Ob_LKCLxg2o
https://steam.oxxostudio.tw/category/python/ai/opencv.html
https://steam.oxxostudio.tw/category/python/ai/opencv-read-video.html
https://www.youtube.com/watch?v=xjrykYpaBBM
https://steam.oxxostudio.tw/category/python/ai/ai-mediapipe-hand.html
https://steam.oxxostudio.tw/category/python/ai/ai-mediapipe-gesture.html
https://www.youtube.com/watch?v=x4eeX7WJIuA
https://officeguide.cc/python-measure-execution-time-tutorial-examples/
https://www.delftstack.com/zh-tw/howto/python/python-simulate-keyboard-input/