期末成果演示-Python套件配合手部辨識、臉部辨識套上想要的濾鏡

組長：陳佑祥 109090035
組員：陳建新 108071052
鄭易元 108071050
陳其寧 109099003
林芝安 109091012

期中預期結果

使用者指出想配戴的部位

系統辨識手部位置

選擇配件樣式

套入個人專屬配件

期望使用者能跨越技術限制、並能實際玩轉AR技術，不論何時何地皆能以喜歡及合適的樣貌出現於線上平台，賦予線上會議 /社交網路更多個人化創意

實際演示流程

人物影像遷入

Hand.py針對

指尖末端position定位

透過指尖定位個人配件

以及臉部套入位置

套上濾鏡

實際演示流程

點擊臉部特徵(鼻子、嘴巴、眼睛)跳出狀態欄

實際演示流程

手指點擊狀態欄數字、套入所選濾鏡

指尖定位與手部辨識講解

- CV2嵌入影像
- drawlandmark()函式
- detect手指位置
- 座標位置規則化

Getting Started | hand.py

Users > cils51107 > Desktop > 程設 > hand.py

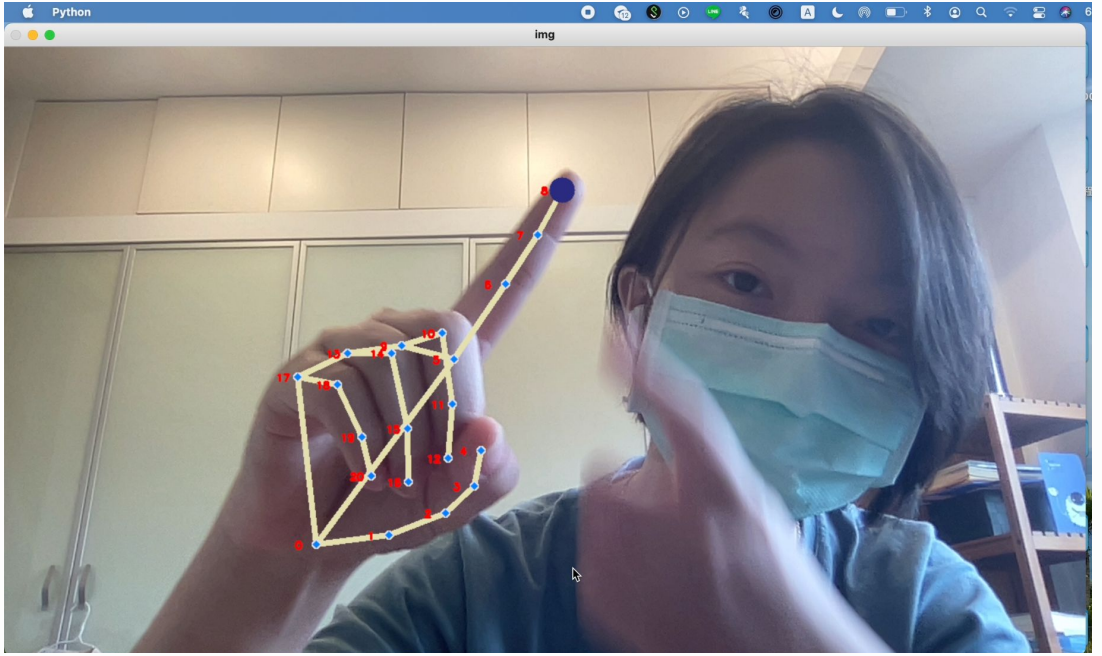
```python
import cv2
import mediapipe as mp


cap = cv2.VideoCapture(0)
mpHands = mp.solutions.hands
hands = mpHands.Hands(min_detection_confidence=0.5, min_tracking_confidence=0.5)
mpDraw = mp.solutions.drawing_utils
handLmsStyle = mpDraw.DrawingSpec(color=(255, 128, 0), thickness=6)
handConStyle = mpDraw.DrawingSpec(color=(176, 224, 230), thickness=6)

while True:
    ret, img = cap.read()
    if ret:
        imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        result = hands.process(imgRGB)
        imgHeight = img.shape[0]
        imgWidth = img.shape[1]

        if result.multi_hand_landmarks:
            for handLms in result.multi_hand_landmarks:
                mpDraw.draw_landmarks(img, handLms, mpHands.HAND_CONNECTIONS, handLmsStyle, handConStyle)
                for i, lm in enumerate(handLms.landmark):
                    xPos = int(lm.x * imgWidth)
                    yPos = int(lm.y * imgHeight)

                    cv2.putText(img, str(i), (xPos-25, yPos+5), cv2.FONT_HERSHEY_SIMPLEX, 0.4, (0, 0, 255), 2)
                    if i == 8:
                        cv2.circle(img, (xPos, yPos), 15, (128, 42, 42), cv2.FILLED)
                        print(xPos, yPos)

                    cv2.imshow('img', img)

    if cv2.waitKey(1) == ord('q'):
        break
```

HAND_CONNECTIONS: Any

Python 3.9.5 64-bit ⊗ 0 ⚠ 0 Ln 22, Col 77 (16 selected) Spaces: 4 UTF-8 LF Python

指尖定位與手部辨識講解

- 實際Demo
- 關節位置定位



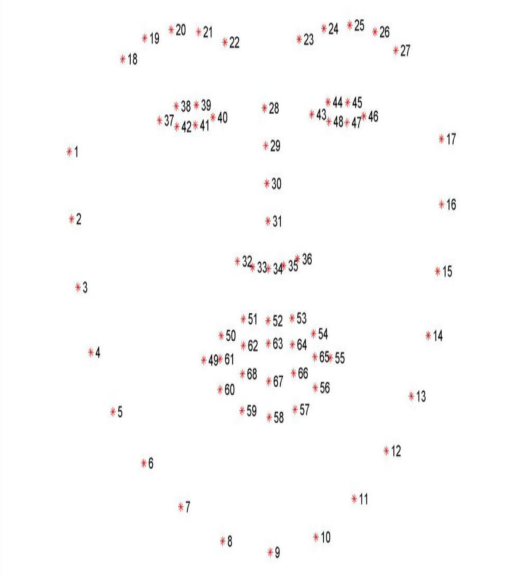
臉部辨識與臉部定位講解

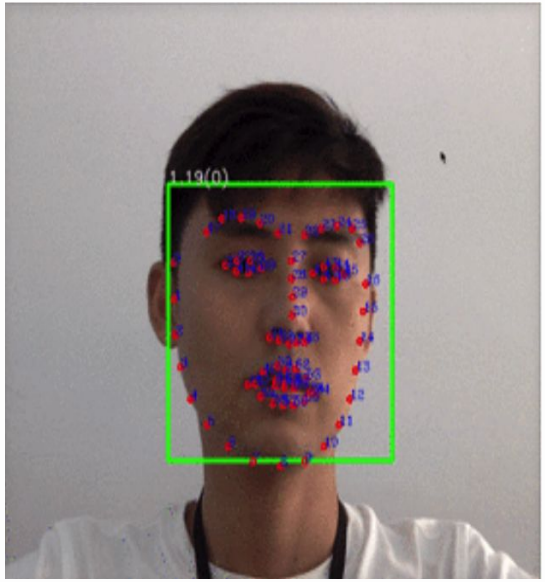
- VedioCapture(0)選擇攝影機
- 偵測人臉
- 調整人臉預設影像大小
- 載入人臉特徵模型
- 矩形範圍取得結果

```
cap = cv2.VideoCapture(0)
cap.set(cv2.CAP_PROP_FRAME_WIDTH,650)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT,500)

detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')

while(cap.isOpened()):
    ret, frame = cap.read()
    face_rects,scores,idx = detector.run(frame,0)

for i , d in enumerate(face_rects):
    x1 = d.left()
    y1 = d.top()
    x2 = d.right()
    y2 = d.bottom()
    text= "%2.2f(%d)"%(scores[i],idx[i])

cv2.rectangle(frame,(x1,y1),(0,255,0),4,cv2.LINE_AA)
```

臉部辨識與臉部定位講解

- 實際Demo
- 臉部位置定位





選擇濾鏡Bar狀態欄講解

- 濾鏡Image套入縮放
- Image Resize
- Filter去白底
- 按鈕製作
- Menu製作

```python
def addfilter(image, x_offset, y_offset, imgurl, resize_x, resize_y):

    filter_image = cv2.imread(
        imgurl, cv2.IMREAD_COLOR)
    if imgurl == "/Users/EvanChen/project/facial/facial/nose/nose1.jpg":
        resize_y = 300
    filter_image_resized = cv2.resize(filter_image, (resize_x, resize_y))
    x_end = x_offset + filter_image_resized.shape[1]
    y_end = y_offset + filter_image_resized.shape[0]
    roi = image[y_offset:y_end, x_offset:x_end]

    filter_image_gray = cv2.cvtColor(
        filter_image_resized, cv2.COLOR_RGB2GRAY)
    ret, mask = cv2.threshold(     Hint: "ret" is not accessed
        filter_image_gray, 120, 255, cv2.THRESH_BINARY)
    bg = cv2.bitwise_or(roi, roi, mask=mask)
    mask_inv = cv2.bitwise_not(filter_image_gray)
    fg = cv2.bitwise_and(
        filter_image_resized, filter_image_resized, mask=mask_inv)
    final_roi = cv2.add(bg, fg)
    image[y_offset:y_end, x_offset:x_end] = final_roi
    return image
```

選擇濾鏡Bar狀態欄講解

- Detect臉部特徵區域
- 根據模型做Resize
- 實際執行

```python
def computePosAndResize():
    global current_menu
    if LANDMARKS != {}:
        if current_menu == 'EYES':
            pos_x = LANDMARKS[1][0]
            pos_y = LANDMARKS[18][1] - 30
            resize_x = LANDMARKS[17][0] - LANDMARKS[1][0]
            resize_y = LANDMARKS[34][1] - LANDMARKS[18][1]
            return [pos_x, pos_y, resize_x, resize_y]
        if current_menu == 'LIPS':
            pos_x = LANDMARKS[40][0]
            pos_y = LANDMARKS[51][1]
            resize_x = LANDMARKS[55][0] - LANDMARKS[49][0]
            resize_y = LANDMARKS[58][1] - LANDMARKS[51][1]
            return [pos_x, pos_y, resize_x, resize_y]
        if current_menu == 'NOSE':
            pos_x = LANDMARKS[32][0] - 15
            pos_y = LANDMARKS[28][1] + 20
            resize_x = LANDMARKS[36][0] - LANDMARKS[32][0] + 40
            resize_y = LANDMARKS[34][1] - LANDMARKS[28][1]
            return [pos_x, pos_y, resize_x, resize_y]
        return [0, 0, 0, 0]
    return [0, 0, 0, 0]
```

結論

Filter

Augmented Reality (AR)

- Python、OpenCV
- 初探AR
- 團隊合作

參考資料

Resources

- https://towardsdatascience.com/creating-a-snapchat-style-filter-with-python-b42ecfd2ff54
- https://docs.opencv.org/4.x/index.html
- https://pyimagesearch.com/2018/01/22/install-dlib-easy-complete-guide/
- https://google.github.io/mediapipe/
- https://github.com/hibyby/GrandmaCan_python_hand_tracking/blob/main/handTracking.py
- https://ithelp.ithome.com.tw/m/articles/10263258
- https://pythonmana.com/2021/10/20211029010331626T.html