

智慧自動化工程科
機器視覺與實習

期末專題報告
瞳孔控制鼠標

1092B0012 賴致帆

1092B0033 黃泓翔

民國 113 年 6 月 24 日

壹. 摘要

本專題目標為透過人類眼睛視線控制電腦鼠標移動，意在開發一種不使用傳統滑鼠或觸控板操控電腦的方法，利用眼球運動達成更便利、直覺的使用體驗。

在本專題中，使用了下列技術原理與方法：

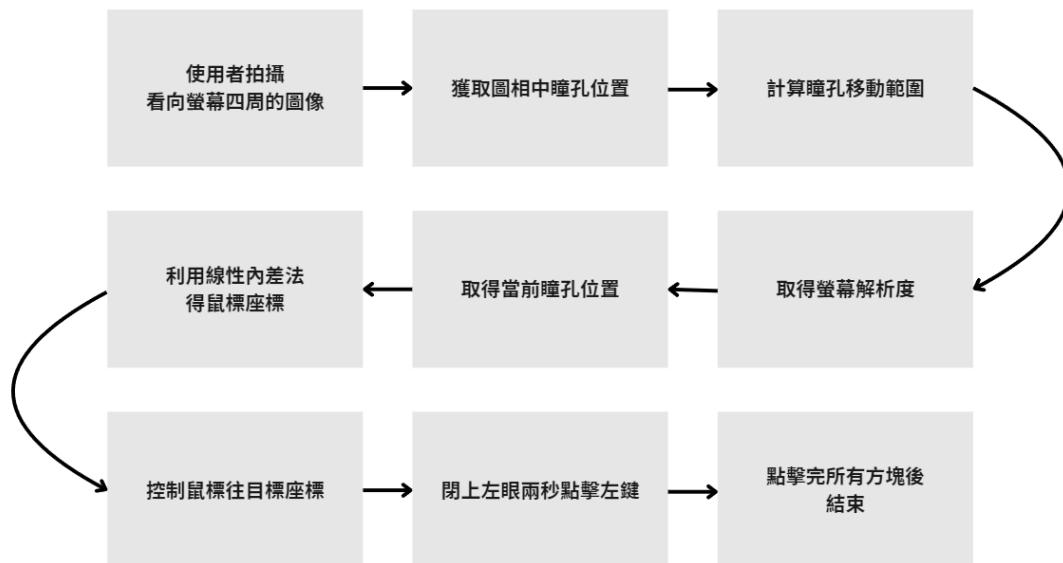
1. 使用機器視覺輔以深度學習方法辨識瞳孔。
2. 基於瞳孔與顯示器平面座標系間可互相映射的假設，由瞳孔位置推導出鼠標位置。
3. 使用 pyautogui 庫實現鼠標的移動以及眨眼點擊滑鼠左鍵之功能。

貳. 研究動機

在 VR、AR 等科技蓬勃發展的時代，隨著深度學習及影像處理技術的成熟，期望透過機器視覺來探索新的人機互動方式。

參. 程式架構

❖ 程式流程圖



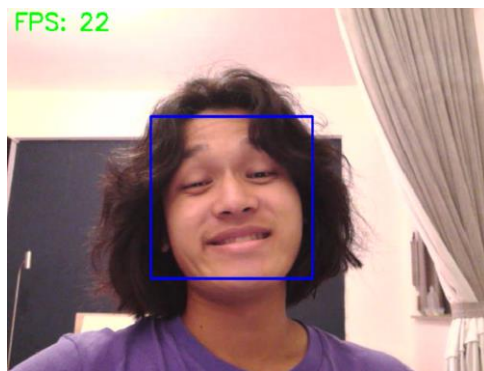
❖ eye.py:獲取瞳孔位置

➤ 預處理

- 轉成灰階
- 左右翻轉

➤ 臉部 ROI

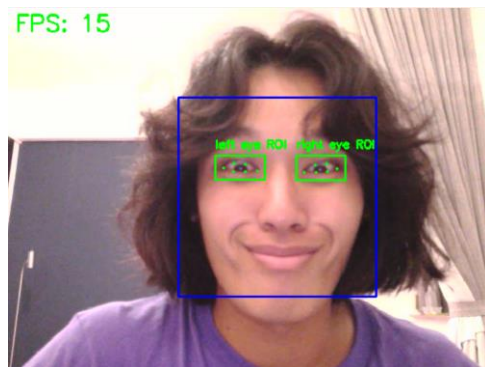
- OpenCV 預訓練分類器
 - 優點: 速度快
 - 缺點: 不準確
- 程式實作:
 - 返回偵測到的所有臉
 - 做面積大小排序: 假設面積最大的為使用者的臉
 - 紀錄 ROI
 - 示意圖:



➤ 眼睛 ROI

- OpenCV 預訓練分類器
 - 返回偵測到的所有臉
 - 優點: 速度快
 - 缺點: 不準確
- dlib 人臉關鍵點檢測
 - 優點: 準確、較多資訊
 - 缺點: 運算量較大
- 程式實作: 選擇使用 dlib 人臉關鍵點檢測
 - 實時檢測: 輸入臉 ROI, 返回 68 個關鍵點的座標

- 將眼睛周圍的關鍵點作為 ROI
- 示意圖



➤ 眨眼檢測

- 將關鍵點間的距離作為是否閉眼的判定

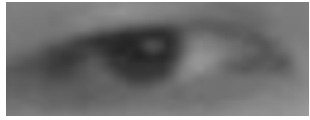
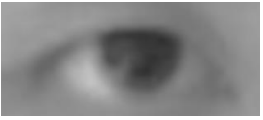
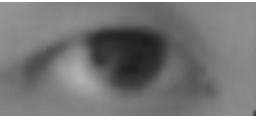
➤ 針對 ROI 做影像處理

- 直方圖匹配

因為影像處理的算法具有特定光照及對比度的侷限性，因此使用直方圖匹配來讓來源圖像的對比度和亮度趨近於模板圖像。


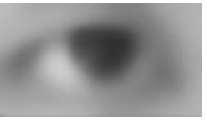
程式碼:

```
match = match_histograms(img, reference)
```

		
模板	直方圖匹配前	直方圖匹配後

- 雙邊濾波

使用雙邊濾波來消除雜訊，同時又能保留邊緣

	
雙邊濾波前	雙邊濾波後

- 二值化



- 反轉

將二值化的圖像進行反轉以利後續操作



- 開運算

可以看到在做完反轉後的圖像瞳孔的部分還連接了周圍的小輪廓，因此要使用開運算將中間斷開。



➤ 計算瞳孔位置

- 找輪廓

使用 OpenCV 的 `findContours()` 函式找到圖像中的所有輪廓。

- 找最大輪廓

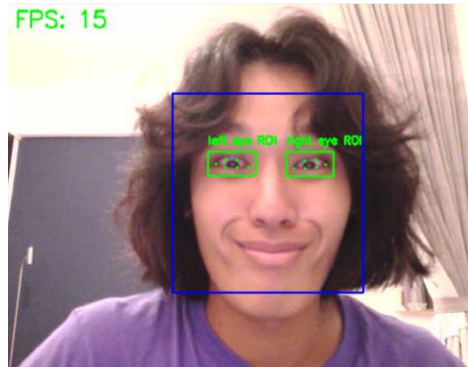
將上一個步驟找到的輪廓面積大小做排序，取出最大的輪廓。

- 計算質心當作瞳孔中心

計算最大輪廓的質心，作為瞳孔中心座標。

- 計算與參考點的距離

將瞳孔中心座標與參考點(人臉關鍵點檢測的眼角關鍵點)做相減，定義為瞳孔位置。



❖ calibration.py: 獲取瞳孔對應於螢幕的移動範圍

➤ 攝影機初始化

使用 OpenCV 庫 VideoCapture() 函式初始化攝影機，此步驟目標為確保攝影機工作正常準備拍攝影像。

➤ 影像捕捉

使用者根據螢幕上的提示注視螢幕的四邊，並按下空白鍵拍照，程式將根據照片中瞳孔的位置計算瞳孔運動範圍。此步驟執行時皆為全螢幕，為了提升鼠標控制的精準度，需要使用者注視顯示器最邊緣的部分。一共需要注視上、下、左、右四邊緣的圖像，故使用一紅點作為提示，紅點會依序顯示在螢幕的四邊，按下空白鍵成功拍攝後顯示下一點，四張拍完後視窗消失儲存影相並計算瞳孔移動範圍。



➤ 計算瞳孔移動範圍

由捕捉的影像計算瞳孔的移動範圍，eye.EyeTracker() 為獲取瞳孔位置之函數，輸入值為一圖像。使用 eye.EyeTracker() 得到瞳孔座標後，以存入一 list 代表相應的範圍。

➤ 保存結果並回傳瞳孔移動範圍

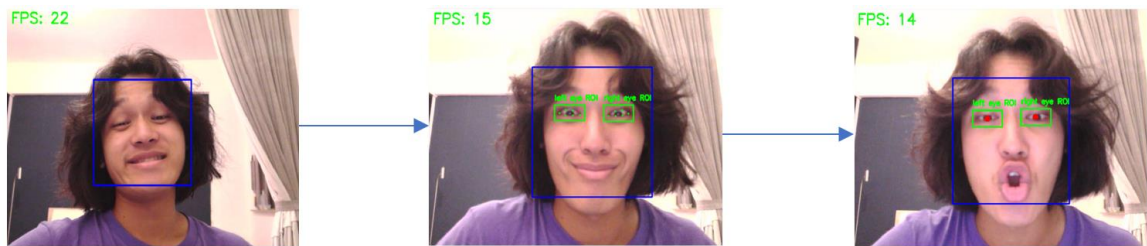
❖ game.py: 透過簡單的小遊戲達成瞳孔對於鼠標可控性的驗證

利用 tkinter 製作一將全螢幕分為四等分按鈕的小遊戲。按鈕初始皆為紅色，按鈕會隨機亮起，亮起時為黃色。成功透過閉上左眼按下滑鼠左鍵點擊按鈕後變更為綠色。全部按鈕變成綠色後顯示”Good Job”字樣並結束遊戲。

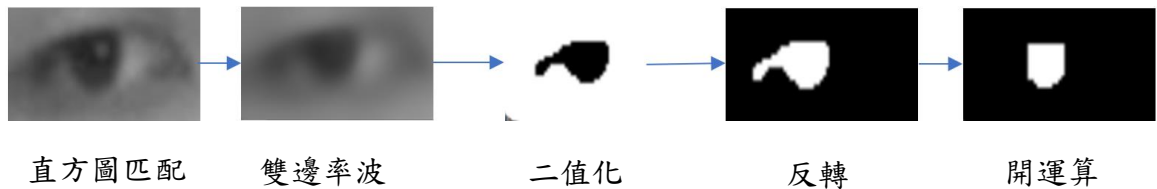
肆. 實驗結果

➤ 辨識瞳孔—由臉的 ROI 與眼睛 ROI 限制範圍後進行影像處理

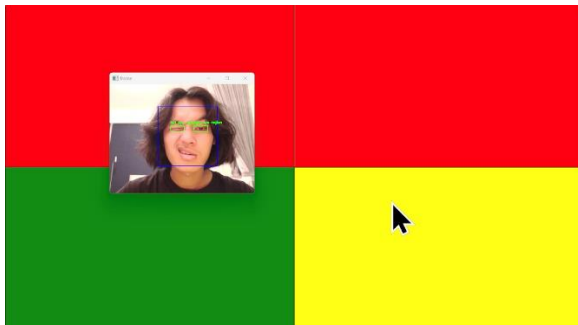
並獲取位置：



➤ 瞳孔影像處理:藉由直方圖匹配、濾波等方法得出瞳孔：



➤ 結合上述結果，達成用瞳孔控制鼠標的目標。



伍. 問題與討論

1. 影像處理 vs AI Model：

影像處理侷限性較高，亮度和對比度改變就有可能得到完全不同的結果，因此若是將整套算法以深度學習模型做實踐，可以在不同環境下得到較為一致的效果，但在標註資料集時需要非常準確。

2. 瞳孔辨識精準度：

受限於影像處理對於不同情況的穩定性不足，稍微改變臉部角度都有可能影響到辨識精度，也因此直接影響到鼠標操控的準確性和穩定性。

3. 瞳孔操控精準度：

本專題因為鏡頭距離眼睛較遠，因此在測量瞳孔移動範圍時上下或左右差異不大，導致沒辦法做細微的控制。

陸. 總結

此專題是基於一種眼動追蹤控制鼠標的控制系統。隨著科技的進步，應該發展出更多元的人機互動方式，人機互動方式的創新，可以帶給使用者更便捷、自然的使用體驗。就成果來說，這次專題的核心目標達成，可以利用眼神控制鼠標按下按鈕。實現眼動控制滑鼠的功能，證明了更多元人機交互的可能性。但在製作過程中我們也發現，因為瞳孔運動範圍窄小與攝影機距離過遠等限制所造成解析度的瓶頸，以及對於環境光源的適應性與人眼樣態的泛用性還有進步的空間。總結來說，眼控鼠標技術是可行的，且可以結合應用在日益發展的AR\VR技術上。儘管還存在一些進步的空間，但這項技術在未來有機會得到更廣泛的應用與發展，創造更良好與直覺的人機互動體驗。

柒. 參考資料

- Github 視線偵測:
<https://github.com/mohenghui/eyeTrail/blob/main/eye.py>
- Github 眼球追蹤:
https://github.com/antoinelame/GazeTracking/blob/master/gaze_tracking/pupil.py
- OpenCV findContours 教學:
<https://www.youtube.com/watch?v=JfaZNiEbreE&t=13s>
- 鼠標控制:
https://blog.csdn.net/m0_69013551/article/details/135897465