生醫工程實驗

期末專題：使用者電腦輔助系統

第一組

電機四　許翔(B03901097)

電機四　陳緯哲(B03901109)

電機四　温明浩(B03901179)

Lab Section: 明達 304
Date: 2017/12/19

一、 簡介

現今的商業化產品，已經擁有使用聲控去對電腦（電子）系統下達指令，去進行特定行為的操作，像是Amazon Echo、Google Assistant、Apple Siri等等。然而，這些指令，往往是廠商在出貨前就已經設定好且寫死的既定模式，無法操作複雜或是不在設定中的行為。然而，透過我們設計的作品，除了事先設定好的語音及對應的行為外，更可以透過臉部的些許變化而達成即時的電腦操作，去完成更複雜的行為，像是遊戲操作與文本編輯，對於生理上或是當下不方便操控電腦的人而言，可謂一大福音。

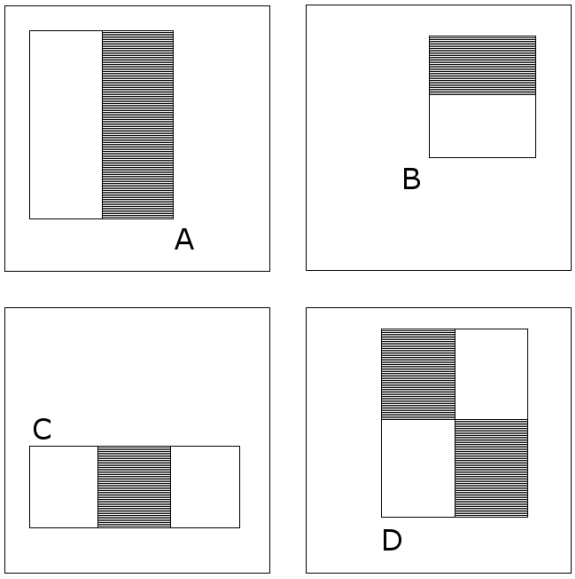
二、 原理

1. 臉部辨識:

Opencv唯一開源軟體，其中包含了許多臉部特徵的辨識模型，而其模型的訓練方式是經由以下三個步驟:

(1) Haar-like 特徵抽取

將一矩形放置於圖片上(如圖一)，把白色區塊的像素值總和減去黑色區塊的像素值總和，即得到了一特徵值。藉由移動矩形或使用不同形狀之矩形，便可得到多組特徵值。



圖一、特徵抽取之示意圖

(2) Adaboost

分類演算法，藉由多個弱分類器的疊代，去判斷特徵值是否為人臉的一部分。

(3) Cascade classifier

多階級分類器，每一Adaboost計算出的結果代表著是否與臉部特定部位吻合，再經由一層一層串聯，對每個結果過濾，得出判斷結果。

以上皆為簡單描述訓練架構，更多精彩細節請 閱讀參考資料(1)(2)。

2. 語音辨識

使用Google提供的語音辨識服務，將使用者輸入的語音，經由設定好的API通道，傳輸至Google系統，待辨識完成後把文字傳回本地端。

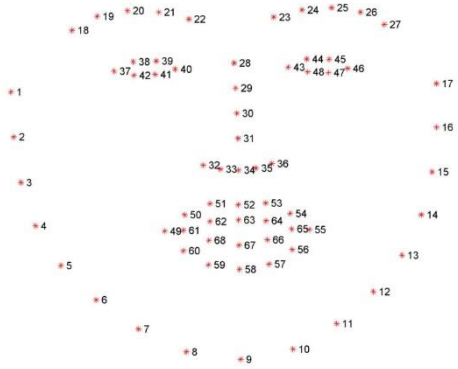
三、 系統規格與使用器材

電腦、耳機、麥克風、WebCam
電腦規格：Ubuntu 16.04.3 LTS、Python 3.5.2

四、 實驗過程

1. 安裝Ubuntu與Python3
2. 安裝Opencv
3. 建立人臉輪廓辨識系統

Opencv所偵測到的輪廓為我們提供了一個良好的基礎，能夠直接藉由輪廓的參數，來取得我們想要的應用，圖二中，便是Opencv預設的人臉輪廓，而我們將Opencv所偵測到的輪廓，對應到受測者上，如圖二所示。以下是我們在本次專題中運用輪廓偵測所做的各種辨識：



圖二、人臉輪廓偵測

圖三、偵測到使用者的輪廓

(1) 頭部方向之辨識

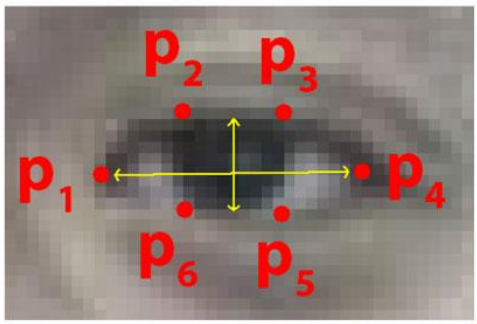
a. 上下：抬頭與低頭時，相機所看到的鼻梁長度會有明顯變化：抬頭時，鼻樑的輪廓會變短；而低頭時則會看到較長的鼻梁，從我們偵測到的鼻樑長度，便能判斷使用者目前為低頭或是抬頭，但因為每個使用者的鼻梁長度不盡相同，因此我們會針對每個使用者進行初始化的動作。

b. 左右：人臉在正對著螢幕時，相機所看到從鼻樑到輪廓的左側與鼻樑到輪廓的右側是相同的，但如果使用者向左轉或是向右轉時，其中一個的長度便會增加，而另一個則會減少，我們運用此點來進行使用者是往左偏或往右偏的偵測。

(2) 眨眼辨識

輪廓偵測同樣可以抓到眼睛周圍的輪廓，如圖四所示，我們利用偵測到的6個點，進行以下的運算：

$$\text{EAR} = \frac{||P2 - P6|| + ||P3 - P5||}{2 * ||P1 - P4||}$$



圖四、眼睛周圍的輪廓

來進行使用者眨眼的偵測，當使用者閉上眼睛時，EAR便會變得極小，而我們就能藉此偵測到眨眼的動作；而為了避免使用者只是一般性的眨眼，而非想進行操作，我們對使用者眼睛閉上的時間。

4. 安裝Google Cloud Speech API
5. 建立各種語音指令，指令的詳細內容請見 六、4 語音指令
6. 安裝PyautoGUI，以系統化的方式控制鍵盤與滑鼠，並將人臉輪廓辨識系統和語音辨識系統的偵測到的結果輸入此系統，達到實驗目的

五、 使用方法

以下影片有完整示範，先在此以條列式的方式呈現：

1. 初始化：

   系統會發出指令”Please look up for 3 seconds”與”Please look down for 3 seconds”，要求使用者向上看3秒，再向下看3秒，使用者只需要跟著指令便能完成初始化的動作，於影片中00:10~00:24。

2. 移動滑鼠：

   完成初始化後，使用者只需要藉由抬頭、低頭、頭往左轉、頭往右轉，便能完成相對應的操作，於影片中00:27~00:45。

3. 按下滑鼠左鍵：

   當使用者用力眨左眼時，就等同於按下滑鼠左鍵，使用者可以自如利用眨眼來操控滑鼠左鍵，於影片中00:50~01:00。

4. 語音指令

如果想要輸入語音指令，只要對著麥克風喊一聲”Google”，系統便會回覆”Say Something”，接著會依照使用者所說的詞彙，以下分成兩種操作模式：

(1)一般語音：當使用者說的並非系統預定指令，則系統會直接將使用者所說的話打出來，並在螢幕上顯示，於影片中01:00~02:00。

(2)控制模式：

以下字詞皆為系統預設指令，當使用者講出這些字詞時，系統會執行特殊動作：

a. Ok： 相當於按下鍵盤上的ENTER，於影片中01:24-01:30。

b. 刪除：系統會紀錄上次輸入的字串長度，進行”刪除”時會刪除同等長度的字串，於影片中03:10~03:22。

c. 滾動：相當於滑鼠的滾輪，進入滾動模式後，即可用低頭與抬頭來控制瀏覽頁面的滾動，若想回到控制游標的模式，只要在喊一次”Google! 滾動”即可，於影片中02:10~02:40。

d. 空格：相當於按下鍵盤上的空白鍵，於影片中02:50~03:03。

e. 控制：由於鍵盤上的按鍵族繁不及備載，因此我們可以藉由控制模式，來按下不在系統預設內的按鍵，例如F1-F12、TAB、SHIFT等等，於影片中01:45-02:04

f. 遊戲：除了一般的操作外，我們也能針對各種遊戲進行操控，讓遊戲的遊玩過程更加順暢，在本次專題中，我們以 slither.io 這個遊戲作為範例進行遊玩，儘管操作尚不及實際操縱滑鼠流暢，但對遊玩過程並沒有產生太大的影響。

六、 示範影片(DEMO)

https://www.dropbox.com/s/5hsd7u5sjzgdexl/demo.mp4?dl=0

影片大小：331MB

七、 後記

於學期初時，我們想要做出能夠解決生活周遭常出現的問題，且希望對社會有重大貢獻，在反覆思考下，我們想到了利用機器學習的方式，透過受測者的EEG、ECG訊號，來判斷是否有酒駕，並且對酒駕者作出相應的動作以防範危險發生。儘管我們對整個系統架構都設計好一套

完整的流程圖，但是最關鍵的資料卻始終無法取得，最後只好改以臉部辨識去判斷駕駛是否疲憊，進一步防範危險駕駛。之後，於進度報告中，藉由同學們的貼心提醒以及助教的回饋，我們發現到這個做法的可行性不高，最後在時間所剩無幾之下，我們決定用現在這個題目去幫助社會。

起初，我們想以眼球與眼眶的相對位置，去控制滑鼠在螢幕上的位置，但受限於眼眶大小，其換算後無法在解析度高的螢幕下精準地控制。因此，在不讓使用者做出太多違和舉動下，我們選擇讓使用者輕微移動頭部去控制滑鼠移動。另外，我們也實現了多執行緒系統，讓臉部操縱與聲控滑鼠得以同時進行，卻又互不衝突，增加系統流暢度。

現行的系統已足以應付一般使用需求，但事實上我們還有兩個功能尚未實現:「將滑鼠控制新增為獨立執行緒」及「使用者自訂義模組」。前者可以大大避免操控時偶發的卡頓，使控制滑鼠如以手控制般流暢；後者則是讓使用者在操控中自行增加指令，增加系統靈活度與個人化。

儘管過程匆忙，但我們對於程式的使用性還是頗有信心。之後我們會繼續優化及新增上述功能，並開源至網路上，讓更多人看到且喜歡我們的作品。

八、 參考資料

(1) cmlab.csie.ntu.edu.tw/~cyy/learning/tutorials/AdaBoostApp.pdf

(2) alex-phd.blogspot.tw/2014/03/haarhaar-adaboost.html