

國立臺灣海洋大學

機電與機械工程學系

碩士學位論文

指導教授：劉倫偉 博士

應用 Raspberry Pi 實現

即時人臉偵測與辨識系統

Real-time Face Detection and Recognition

System Based On Raspberry Pi

研究生：黃明愷 撰

中華民國 105 年 6 月

應用 Raspberry Pi 實現

即時人臉偵測與辨識系統

Real-time Face Detection and Recognition
System Based On Raspberry Pi

研 究 生：黃明愷

Student：Ming-Kai Huang

指導教授：劉倫偉

Advisor：Luen-Woei Liou

國立臺灣海洋大學
機械與機電工程學系
碩 士 論 文

A Thesis
Submitted to Department of Mechanical and Mechatronic Engineering
College of Engineering
National Taiwan Ocean University
in Partial Fulfillment of the Requirements
for the Degree of
Master of Science
in
Department of Mechanical and Mechatronic Engineering
June 2016
Keelung, Taiwan, Republic of China

中華民國 105 年 6 月

摘要

Raspberry Pi 是一塊信用卡大小的單版機電腦，最初的設計是用於教育，它的目的是創造出一個低成本但高效能運算的計算機，其作業系統為使用 Linux 做設計。

本論文運用網路攝影機 webcam 結合 Raspberry Pi 單板機電腦，建構一個即時人臉偵測與辨識系統。人臉偵測辨識系統的流程為先在影像中搜尋膚色，再使用 Viola 人臉偵測方法判斷是否為人臉，並且辨識特徵點，加以擷取和分析。接著利用主成分分析（PCA）取得人臉影像的特徵向量，降低其資料的維度，再配合一套人臉資料庫做比對，找出最接近的人臉。

關鍵字：Raspberry Pi，OpenCV，Webcam，人臉偵測與辨識，Viola 人臉偵測，膚色偵測，PCA

Abstract

Raspberry Pi is a credit-card sized computer originally designed for educational use. The goal of Raspberry Pi was to create a low-cost, yet, high performance device that would facilitate programming. The Raspberry Pi was designed for the Linux operating system.

This paper combines webcam with Raspberry Pi to realize a real-time face detection and recognition system. In the proposed system, skin color is detected first, and then Viola method is used to determine whether if the image presented is a human face. Next, Principal Component Analysis (PCA) is used to extract face features to form facial feature vectors and to decrease the dimension of facial feature vectors thereafter. Finally, a facial database was built to exemplified the usage of this system.

Keywords: Raspberry Pi, OpenCV, Webcam, facial detection and recognition, Viola face detection, skin color detection, PCA

致謝

在此要先感謝我的指導教授 劉倫偉 老師，感謝您對我的細心教導與指引，在我迷失方向時給了我明確的目標，引導出正確的道路，同時也指引我如何找尋本論文中所需的重點與理論。

接著要感謝各位口試委員 李福星 老師與 傅群超 老師在百忙之中抽空，給予我的指導與建議，使我了解到影像辨識演算法的重要性，讓本論文更加完整。

再來要感謝我最好的同學國威，感謝你一起陪我走過這充實的三年，以及對我的關照；感謝學長沛儒、鈺菴、詠勝在我學習與研究過程中的引導，以及學弟照天、達倫、楚平、培修在生活中的幫忙與協助，使我在研究所的生活中，過得非常充實與快樂。

最後我要感謝我的父母，在我求學的過程中對我付出的一切，使我能全心投入於研究上，並無生活上的後顧之憂，感謝你們的陪伴才能讓我能順利完成碩士學業，邁向未來嶄新的人生。

目錄

摘要.....	I
Abstract.....	II
致謝.....	III
圖目錄.....	VI
表目錄.....	IX
第一章 緒論.....	1
1.1 研究動機.....	1
1.2 研究目的.....	1
1.3 文獻回顧.....	1
1.4 論文架構.....	3
第二章 數位影像理論背景.....	4
2.1 CCD 原理介紹	4
2.2 CMOS 原理介紹	5
2.3 影像偵測方法介紹.....	6
2.3.1 色彩分割法.....	7
2.3.2 灰階轉換.....	13
2.3.3 二值化處理.....	13
2.4 影像型態學.....	14
2.4.1 膨脹.....	14
2.4.2 侵蝕.....	15
2.4.3 斷開.....	16
2.4.4 閉合.....	17
2.5 Viola 人臉偵測方法	18
2.5.1 積分影像.....	18
2.5.2 Adaboost 演算法	20
2.5.3 層疊分類器.....	24
2.6 人臉偵測與辨識.....	26
2.6.1 人臉特徵點偵測.....	28
2.6.2 表情偵測.....	29
2.7 人臉識別與特徵臉介紹.....	30
2.7.1 特徵臉介紹.....	30
2.7.2 主成分分析(PCA)介紹	31
2.7.3 PCA 用於人臉識別原理.....	35
2.8 OpenCV 介紹	38
2.8.1 OpenCV 模組介紹	38
2.8.2 OpenCV 與 Matlab 的工具箱比較	39

2.8.3 OpenCV 使用介面介紹	40
2.8.4 OpenCV 之人臉識別應用	40
第三章 實驗設備與架構.....	44
3.1 Raspberry Pi 介紹.....	44
3.1.1 硬體介紹.....	46
3.1.2 作業系統.....	49
3.1.3 韌體介紹.....	49
3.1.4 遠端連線介紹.....	55
3.1.5 硬體架構.....	56
3.1.6 Linux 介面介紹	57
3.2 網路攝影機 webcam 介紹	61
第四章 實驗結果與分析.....	63
4.1 人臉特徵點偵測結果.....	63
4.1.1 膚色偵測結果.....	63
4.1.2 眼睛特徵點偵測結果.....	65
4.1.3 嘴唇特徵點偵測結果.....	66
4.1.4 鼻子特徵點偵測結果.....	67
4.1.5 眼鏡特徵點偵測結果.....	67
4.2 臉部出現干擾之偵測結果.....	68
4.3 人臉識別與檢測.....	69
4.3.1 人臉識別與檢測介紹.....	69
4.3.2 常用的人臉資料庫介紹.....	70
4.3.3 人臉資料庫收集與建立.....	71
4.3.4 人臉識別結果.....	74
第五章 結論與未來展望.....	80
5.1 結論.....	80
5.2 未來展望.....	80
參考文獻.....	82

圖目錄

圖 2.1 CCD 之結構圖	4
圖 2.2 CMOS 之結構圖	5
圖 2.3 濾光片矩陣之結構圖	6
圖 2.4 RGB 色彩模型	7
圖 2.5 RGB 混色圖示	8
圖 2.6 HSI 色彩模型	8
圖 2.7 膚色統計範圍	11
圖 2.8 YCbCr 色彩座標	12
圖 2.9 影像 A 與遮罩 B	14
圖 2.10 經膨脹後的影像 A 與遮罩 B	15
圖 2.11 影像 A 與遮罩 B	16
圖 2.12 經侵蝕後的影像 A 與遮罩 B	16
圖 2.13 經過斷開後的影像 A 與遮罩 B	17
圖 2.14 經過閉合後的影像 A 與遮罩 B	18
圖 2.15 點(x,y)處的積分影像值	19
圖 2.16 點(x,y)處的積分影像值示意圖	19
圖 2.17 計算矩形和示意圖	20
圖 2.18 矩形特徵示意圖	20
圖 2.19 輸入影像中像素灰階值之和的差值	21
圖 2.20 利用積分影像計算出矩形和	23
圖 2.21 利用矩形特徵分辨是否為人臉	23
圖 2.22 層疊分類器的示意圖	25
圖 2.23 人臉辨識流程圖	27
圖 2.24 人臉特徵比例關係示意圖示	28
圖 2.25 特徵臉示意圖	31
圖 2.26 PCA 示意圖	32
圖 2.27 人臉影像轉成向量 Γ 的過程	35
圖 2.28 OpenCV 檔案位置	40
圖 2.29 OpenCV 內建的基本函式.xml	40
圖 3.1 Raspberry Pi 外觀圖示	44
圖 3.2 A 型與 B 型之比較圖	45
圖 3.3 Raspberry Pi model B+ 外觀圖示	45
圖 3.4 Raspberry Pi 硬體配置	46
圖 3.5 Modle B GPIO 針腳編號	48
圖 3.6 Modle B+ GPIO 針腳編號	48
圖 3.7 API 連線的方框圖	50

圖 3.8 Raspberry Pi 開機畫面圖	51
圖 3.9 輸入使用者帳號.....	51
圖 3.10 輸入使用者密碼.....	52
圖 3.11 輸入指令的提示符號.....	52
圖 3.12 Raspberry Pi 圖形化使用者介面 X 視窗	53
圖 3.13 icon 圖示.....	54
圖 3.14 系統選單按鍵圖示.....	54
圖 3.15 檔案總管按鍵圖示.....	54
圖 3.16 切換虛擬畫面圖示.....	54
圖 3.17 CPU 執行圖示.....	55
圖 3.18 關機圖示.....	55
圖 3.19 PuTTY 登入介面.....	56
圖 3.20 Raspberry Pi 硬體架構圖	57
圖 3.21 Raspberry Pi 終端機介面	57
圖 3.22 Raspberry Pi 檔案資訊.....	58
圖 3.23 Raspberry Pi 新資料夾的建立	58
圖 3.24 Raspberry Pi 程式編譯介面	59
圖 3.25 使用 OpenCV 函式庫中的.xml 檔之介面	59
圖 3.26 進入 OpenCV 路徑之介面	60
圖 3.27 OpenCV 內建的.py 執行檔	60
圖 3.28 開啟執行檔之輸入指令.....	61
圖 3.29 Logitech Webcam C310 的外觀.....	62
圖 4.1 膚色的色塊圖示.....	63
圖 4.2 距離鏡頭 0.3 公尺的人臉膚色偵測.....	64
圖 4.3 距離鏡頭 3.3 公尺的人臉膚色偵測.....	64
圖 4.4 單眼偵測之左眼的辨識.....	65
圖 4.5 雙眼偵測之左眼的辨識.....	65
圖 4.6 嘴唇偵測的辨識.....	66
圖 4.7 雙眼及嘴唇偵測的辨識.....	66
圖 4.8 鼻子偵測的辨識.....	67
圖 4.9 眼鏡偵測的辨識.....	67
圖 4.10 口罩遮住部分臉部之偵測.....	68
圖 4.11 口罩遮住一半臉部之偵測.....	68
圖 4.12 人臉識別流程圖.....	69
圖 4.13 灰階人臉圖像.....	72
圖 4.14 不同角度的人臉圖像.....	72
圖 4.15 不同表情的人臉圖像.....	73
圖 4.16 人臉識別之人機介面.....	75

圖 4.17 實驗成員 no.1 之辨識結果.....	76
圖 4.18 實驗成員 no.2 之辨識結果.....	76
圖 4.19 實驗成員 no.3 之辨識結果.....	77
圖 4.20 實驗成員 no.4 之辨識結果.....	77
圖 4.21 實驗成員 no.5 之辨識結果.....	78
圖 4.22 實驗成員 no.6 之辨識結果.....	78

表目錄

表 2.1 影像尺寸與特徵數量之對應.....	21
表 3.1 Raspberry Pi 硬體規格.....	46
表 3.2 CMOS 與 CCD 之比較.....	62
表 4.1 自製人臉資料庫規格.....	73
表 4.2 實驗成員與其代碼.....	74
表 4.3 本實驗之辨識率.....	79

第一章 緒論

1.1 研究動機

日常生活中，與我們息息相關的辨識系統，例如：門禁系統、提款機或保全系統…等，大部分都是以鑰匙、密碼或晶片卡，作為開啟工具，但是這樣的防護措施，常常會被罪犯輕易的破解，以至於造成自身在生命財產上的慘重損失，如果能再加裝上一道保護措施，就可以讓我們的生命安全提高不少的保障，其中影像辨識也是辨識系統中極其重要的一環。

而電腦視覺(Computer Vision)在近幾年來應用於偵測、辨識、追蹤等相關的領域中有著顯著的進展，以電腦視覺幫助人類從事各種活動也越來越廣泛。因為對於人類的視覺而言，人臉特徵相較於其它特徵，人臉辨識是最直接的辨認方式，依照每個人在臉部影像所呈現不同的特徵做身份辨識。因此本論文以人臉影像偵測與辨識作為研究的方向。

1.2 研究目的

要如何從影像中找出人臉影像，並加以做特徵點偵測與辨識，是本論文所要探討的，Viola 與 Jones[1]提出了積分影像(Integral Image)概念和一個基於Adaboost 演算法的訓練人臉偵測分類器方法，此方法主要使用於搜尋人臉，並建立了一個即時的人臉偵測系統。

本論文運用網路攝影機 Webcam 並結合 Raspberry Pi 單板機電腦，建構一個即時人臉偵測與辨識系統。使用 Raspberry Pi 作為其影像處理運算之硬體的理由，主要是因為 Raspberry Pi 具有能夠支援各種程式設計、且體積小不占空間、價格低廉等特點。

1.3 文獻回顧

關於人臉偵測與辨識的方法，梁振升[2]提出以人臉五官為特徵的即時人臉辨識系統，並整理出人臉偵測的主要方法。而如何在複雜的背景中，擷取出正確

的人臉影像將是非常重要的。人臉偵測的研究已經發展多年，而目前國內外關於人臉辨識的研究也十分豐富，其人臉偵測的方法主要可分為以下類型。

- **Knowledge based method (基於知識之方式)**

此方法主要就是考慮到人臉上應有一定的規則，並以此規則在人臉候選區域進行搜尋，此方法的缺點是很難明確定義描述人臉特徵的規則，如果規則定義太過鬆散，則可能將非人臉的區域誤判為人臉，若定義太過嚴謹，又很容易偵測失敗。

- **Appearance based method (基於外觀之方式)**

此方法一般是透過統計分析及機器學習的方式找出人臉與非人臉間的特性。因此必須使用到分佈模型或鑑別函數，同時須使用降低維度來減少計算的複雜度。此方法的缺點是較複雜且需要大量的樣本學習，才可能達到較高的效能。

- **Template matching method (基於模板匹配之方式)**

此方法是先人工定義出一個標準樣本，計算出此標準樣本的相關數值，如人臉輪廓、眼睛、鼻子及嘴巴。然後將輸入影像基於此標準樣本的相關數值做計算及比較。雖然此方法較簡單且易實現，但缺點是只適用於，與標準影像相同大小且相同位置的人臉影像。

- **integral image method (基於積分影像之方式)**

此方法由 Viola[1]提出，為針對如何找出人臉，而提出的一種機器學系方法 (Machinelearning approach)，此方法能夠達到快速及準確的偵測率，也是本論文所使用的搜尋人臉方法。其主要分為三個特性：

1. 使用積分影像達到快速的特徵計算。
2. 以 AdaBoost 方法選取少量重要特徵來建構分類器。
3. 以串聯的方式結合許多複雜分類器，此方式是以重視有用的影像區域來提高其偵測速率。

1.4 論文架構

本論文架構分為五個章節：

第一章 緒論

第二章 數位影像理論背景

第三章 實驗設備與架構

第四章 實驗結果與分析

第五章 結論與未來展望

第二章 數位影像理論背景

2.1 CCD 原理介紹

CCD(Charged Coupled Devices)，是一種電荷耦合元件，能將感測到的光轉換成電荷訊號，再將訊號數位化並加以處理[3]，其基本構造如圖 2.1 所示：

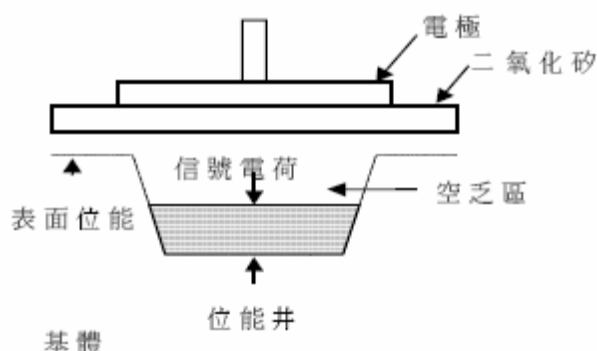


圖2.1 CCD 之結構圖

它是由一個 MOS 電容和一個附著在半導體表面上二氧化矽之頂端的電極所組成。當在基體和電極之間加上電壓，則在靠近二氧化矽的介面與半導體界面的附近區域形成一個空乏區。對少數載體而言，這個區域變成低能階的位能井，如果由光能量所產生的信號電荷被注入這位能井，則這些電信電荷是暫時被儲存且記憶成類比形態的量。CCD 是由許多非常微小的感應器組成的固態電子元件，每個微小感應器都能各自儲存電荷，儲存的電荷與接收到的光線成正比。再經由類比—數位轉換器把 CCD 感應光線而產生的類比電壓讀數轉換成代表各種不同的色彩或灰階數值，以將感應到的資料轉換成數位資料。

網路攝影機(Webcam)拍得實際影像後，將所得到的光源強度資料數位化，這些數位資料在電腦上排成數值的陣列，而陣列上的元素稱為圖像元素(image element)，或簡稱像素(pixel)，我們可以將視訊所取到的影像表示為 $N \times M$ 矩陣資料，如式(2.1)所示。

$$f(i, j) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,M) \\ f(1,0) & f(1,1) & \cdots & f(1,M) \\ \vdots & \vdots & \ddots & \vdots \\ f(N,0) & \cdots & \cdots & f(N,M) \end{bmatrix} \quad (2.1)$$

一幅影像矩陣不像物理定律或工程數學在演算法上有一般的規律，而要有效率的解決影像問題時，有時必須依特定狀況使用不同之影像處理方法。

2.2 CMOS 原理介紹

CMOS 影像感測器元件，由像素(Pixel)以矩陣排列組合而成，每個像素藉光電二極體(Photodiode)將光子轉換成電子，元件收集到的光線，轉換成電流後，再由類比轉換成數位訊號，組合構成整幅影像[4]。互補式金屬氧化物半導體影像感測器架構如圖 2.2 所示。

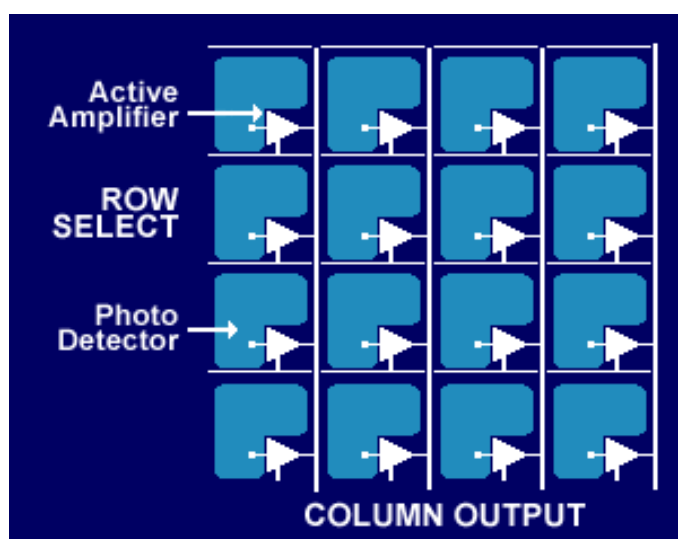


圖2.2 CMOS 之結構圖

CMOS 影像感測器元件由黑白邁向彩色時，須在光電二極體前方加上色彩濾光片 (Color Filter)，分離出 RGB 三原色。Kodak 首先發展出濾光片矩陣 (Color Filter Arrays) 架構，進行彩色影像感測工作。如圖 2.3 所示。

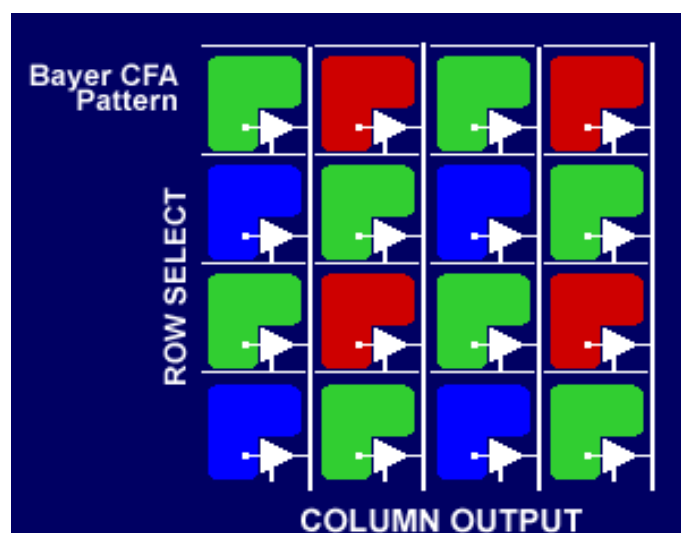


圖2.3 濾光片矩陣之結構圖

CMOS 影像感測器元件向來有感光度較差的缺陷，加上濾光片後，進光量銳減三分之一以上，更加提高影像感測困難度。當影像感測器邁向高畫素發展時，依然面臨進光量不足窘境，微鏡片（Microlens）應運而生。微鏡片通常加在濾光片前，聚集光線來增加進光量，可以有效增加感光品質，已普遍應用在各種產品上。本論文使用的 Webcam 為運用 CMOS 感測元件的 Logitech Webcam C310。

2.3 影像偵測方法介紹

在影像中去偵測一個特徵物體時，必須知道該物體的特徵資訊，如彩色影像的顏色、物體形狀、物體邊界、物體動態等資訊。目前許多系統所用到的物體偵測方法有：顏色分割法、動態偵測法、形狀分析法及物體景深法[5]。

一般來說，背景較複雜的靜態影像物體偵測是一件困難的事，在複雜背景與光源環境下會影響物體偵測的強健性和正確性。而動態偵測是指處理連續影像取出運動中的物體，由於連續影像的動態資訊豐富，所以偵測物體的難度較低。本論文主要以動態影像的人臉偵測為基礎。

本論文的影像處理過程中 RGB 格式並不適用，因此須先將 RGB 轉換成其他色彩空間，再做後續的影像處理。影像處理中常用的色彩空間有 RGB、HSI、YCbCr、NCC...等，其中 RGB 為彩色顯示器所使用之色彩空間，RGB 空間由於相當易於瞭解，因此在繪圖或呈現上多以此空間為基礎，以下將介紹最常使用的

RGB 色彩空間與 HIS 色彩空間，以及適合作膚色偵測的 YCbCr 色彩空間與 NCC 色彩空間。

2.3.1 色彩分割法

辨識系統常用的有顏色、形狀、運動等影像特徵，運用越多物體特徵，則系統越穩定，但是必須付出較多的時間計算。在此介紹最常使用的 RGB 色彩空間與 HSI 色彩空間。

一般的彩色數位影像處理器的色彩模型 RGB(R=Red, G=Green, B=Blue)三元色系統所組成如圖 2.4 與圖 2.5。彩色影像是由 3 個獨立的影像平面組成，每一個原色對應一個影像平面，也就是說每一個像素值分別有三個 RGB 分量，每個分量以 8 位元來儲存。但是 RGB 色彩極容易受光線影響，而且不是那麼適合人類描述色彩。例如，我們不會用組成其顏色之每一個主色的百分比來指稱一個車子的顏色，而會用色相(hue)、飽和度(Saturation)、亮度(Intensity) 來表示。色相是描述純色彩(純黃色、橙色或紅色)的色彩屬性，而飽和度則是一個純色彩添加白光之程度的量測，亮度是事實上不可能測量的一個主要觀點。因此在本文中使用較適合人類解讀色彩的 HSI(色相、飽和度、亮度)色彩模型如圖 2.6。HSI 模型是根據對人們自然和直覺的色彩描述發展影像處理演算法的一個理想工具。

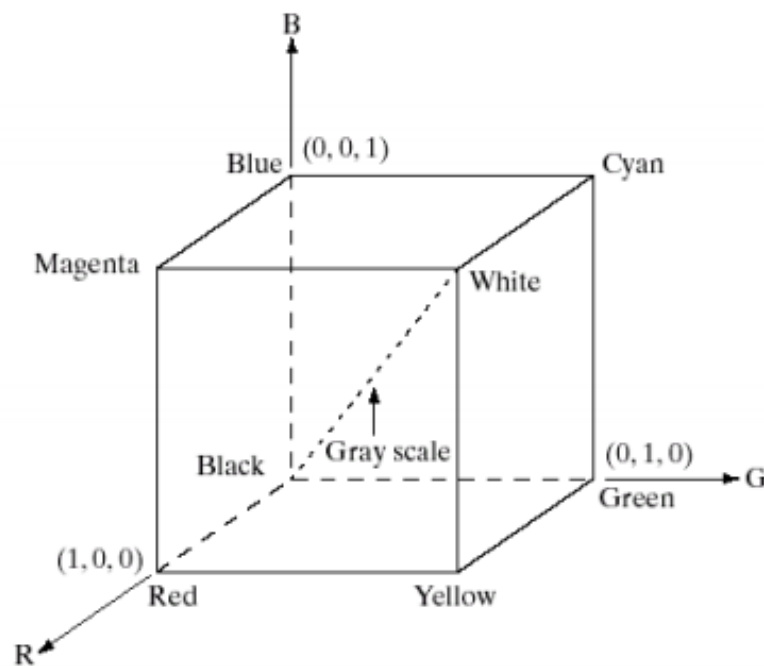


圖2.4 RGB 色彩模型

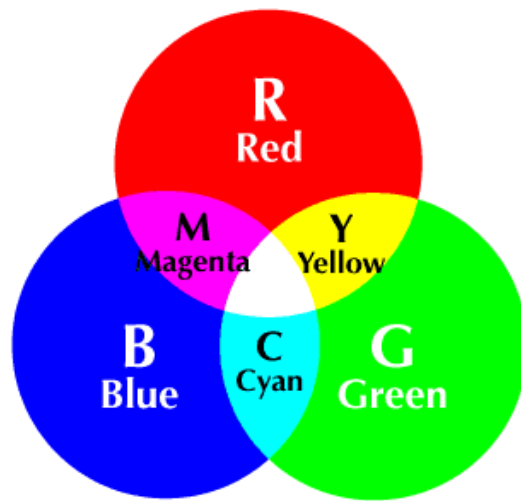


圖2.5 RGB 混色圖示

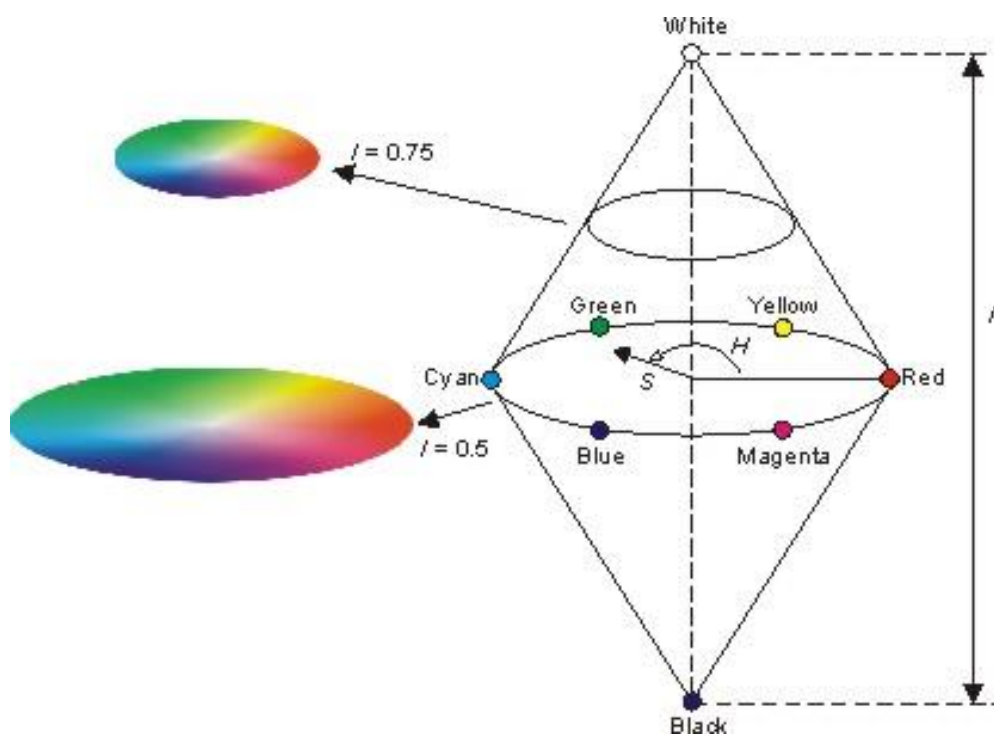


圖2.6 HSI 色彩模型

從 CMOS 影像中所得到的色彩是 RGB 色彩形式，必需轉換為 HSI 色彩作為判斷顏色的標準。其中 H 代表色相(Hue)，是平常用來區分顏色的主要成分，也就是人眼所能感覺出來的色彩，例如紅、橙、黃、綠、藍... 等。S 代表飽和度(Saturation)，表示顏色的飽和程度，也就是顏色中摻入白色的程度，彩度越高的色彩代表摻入白色越少，例如粉藍色就比藍色的彩度還低。I 代表亮度(Intensity)，指顏色的明暗程度，RGB 與 HSI 色彩空間的轉換公式可表示如式(2.2)～式(2.5)。

$$\theta = \cos^{-1} \left(\frac{\frac{1}{2}[(R-G) + (R-B)]}{\sqrt{(R-G)^2 + (R-B)(G-B)}} \right) \quad (2.2)$$

$$H = \begin{cases} \theta & B \leq G \\ 360 - \theta & B > G \end{cases} \quad (2.3)$$

$$S = 1 - \frac{3}{R+G+B} \times \min(R, G, B) \quad (2.4)$$

$$I = \frac{1}{3}(R+G+B) \quad (2.5)$$

在膚色偵測步驟中需先將影像由 RGB 色彩空間轉為其他的色彩空間，例如 YCbCr 或 NCC 等。而本論文所使用的色彩空間為 YCbCr，以下將介紹人臉偵測常使用的 YCbCr 與 NCC 色彩空間。

● NCC 色彩空間

當比較不同種族的人臉時，可以發現在彩色空間中，不同種族膚色會引起一種密集叢集(tight cluster)。因此如何在影像中找出膚色範圍，將是本節所要討論的方向。

NCC(Normalized Color Coordinates)色彩空間是解決RGB 色彩空間中，影像因光源亮度的強弱，造成物體在相同顏色的地方呈現出深淺不同的顏色。分別對 R與G做正規化，就可以使R與G對亮度的靈敏度降低。接下來將介紹在NCC色彩空間中，定義膚色範圍的方法。

而其膚色模組是Soriano為了使實驗在膚色判斷上有具體的依據，在暗房之中模擬出四種不同色溫的光源情況，將膚色在NCC色彩空間上的分佈統計出來。下列分別是四種不同的模擬情形：

1. 2300K 地平線光源(Horizon Daylight)。
2. 2856K 白灼光(Incandescent)，白灼光是指室內強光。
3. 4000K 螢光(Fluorescent)，螢光是指一般室內日光燈環境。
4. 6500K 日光(Daylight)，6500K的色溫可以用D65表示。

其中色溫是指光波在不同的能量下，對人眼所感受的顏色變化，以Kelvin為色溫計算單位。使用四種不同的口徑攝影機且做白平衡設定，在上述四種不同的光源環境(共十六種不同環境下)，拍攝白種人與黃種人取得彩色樣本影像，作為膚色取樣的依據。接下來將這些不同環境下拍攝的彩色影像，由RGB色彩空間轉換到NCC色彩空間。因為RGB色彩模型對光線變化有相當靈敏的反應，所以做NCC的轉換可以減少顏色對亮度的依賴，其轉換公式為式(2.6)與式(2.7)：

$$r = \frac{R}{R+G+B} \quad (2.6)$$

$$g = \frac{G}{R+G+B} \quad (2.7)$$

式(2.6)可以得到正規化後的紅色像素r，目的是在減少亮度對紅色系的響影，同理式(2.7)可以得到正規化後的綠色像素g，目的是在減少亮度對綠色系的影響。圖2.7是Soriano使用CCD與USB Webcam在他設定的十六種不同情況下，對白種人與黃種人所拍攝到膚色在NCC座標中的統計範圍分佈，X軸為正規化後紅色像素，其範圍約在0.2到0.6，Y軸為正規化後綠色像素，其範圍約在0.2到0.4，由圖2.7可以觀察出膚色分佈範圍相當集中。

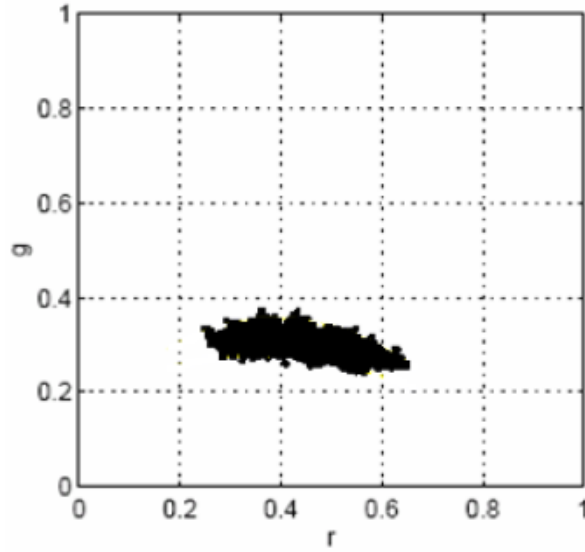


圖2.7 膚色統計範圍

● YCbCr色彩空間

此色彩空間為本論文所使用。由於RGB色彩空間的會受到亮度極大的影響，因此有許多的色彩空間，如 YCbCr、NCC、YES、HSI、及 HSV 等被提出，這些色彩空間被用來改善色彩的一致性或分割。然而經由許多的測試之後，證明 YCbCr 是最適用於膚色的分割。YCbCr 色彩空間的 Y 代表亮度分亮，Cb 及 Cr 代表藍色色度分量及紅色色度分量，YCbCr 與 RGB 之間的轉換為式(2.8)。

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \frac{1}{256} \begin{bmatrix} 657.38 & 129.057 & 25.064 \\ -37.945 & -74.494 & 112.439 \\ 112.439 & -94.154 & -18.285 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (2.8)$$

圖 2.8 為 YCbCr 空間的色彩座標，膚色集中區塊為以下所示：

Cr : 0.55~0.6

Cb : -0.4~-0.5

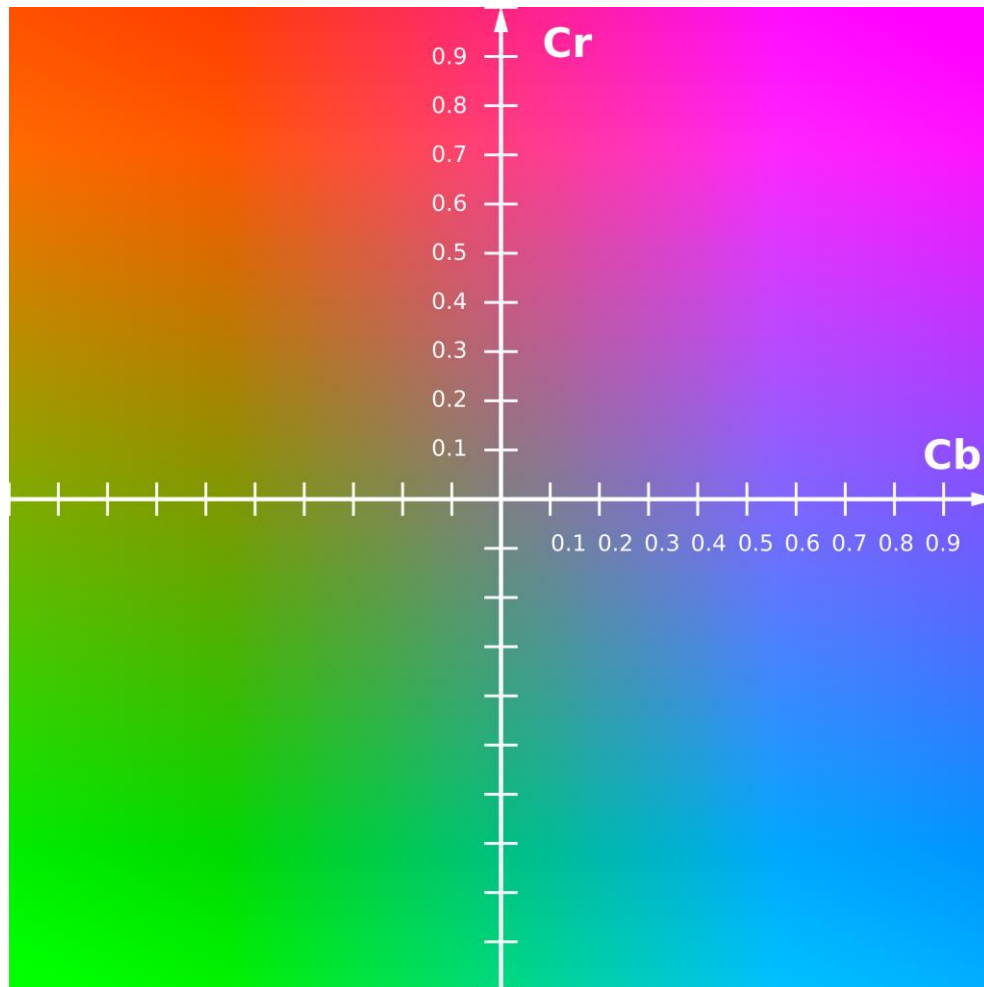


圖2.8 YCbCr 色彩座標

YCbCr 色彩空間具有下列特點:

- YCbCr 中亮度元素 Y 是獨立的色彩，因此可以用來解決亮度變化的問題且容易使用。
- 膚色在 YCbCr 中的群聚性，較其他色彩空間來得緊密。
- 在多變的亮度條件下，膚色及非膚色資料在 YCbCr 色彩空間中有較小的重疊性。
- YCbCr 色彩空間被廣泛地應用在影像壓縮標準中，例如 MPEG 與 JPEG 圖檔。

- YCbCr 是使用於家庭影像系統中的色彩空間。
- YCbCr 是兩個主要用來表示數位影像的色彩空間的其中一個(另一個是 RGB)。YCbCr 與 RGB 不同的地方在於，YCbCr 以亮度和兩個不同的色彩訊號來表示色彩，而 RGB 是以紅、綠、藍表示色彩。

將輸入影像由RGB色彩空間轉換成YCbCr色彩空間之後，接著再以下列三條規則來判斷某一偵測到的像素是否為膚色：

- $Y > \alpha$: Y (亮度)必須大於值，其中 $\alpha = 120$ 。
- $Cb > \beta$: Cb 必須小於 β 值，其中 $\beta = 95$ 。
- $Cr > \gamma$: Cr 必須小於 γ 值，其中 $\gamma = 110$ 。

2.3.2 灰階轉換

灰階(Grayscale)影像，是以亮度為基礎，將彩色影像轉換到黑色至白色之間，依深淺不同共分為 256 個值，其中 0 代表黑色，255 代表白色，在 0~255 之間尚有 254 種由深至淺的灰色[6]。經過灰階轉換後，可減少三分之二的影像運算量。其計算公式如式(2.9)。

$$Grayscale = 0.299R + 0.587G + 0.114B \quad (2.9)$$

其中因為 R、G、B 三種色彩波長不同，所以各有不同的係數。

2.3.3 二值化處理

在影像處理與辨別上，有時只需要若干灰階值，例如黑與白，通常我們會依直方圖(Histogram)決定一個臨界值，將灰階影像轉換為黑白影像，這種方法就是臨界值法(Thresholding)。一個二值化的影像 $g(x, y)$ 定義如下式(2.10)：

$$g(x, y) = \begin{cases} 0 & f(x, y) > T \\ 1 & otherwise \end{cases} \quad (2.10)$$

其中 $f(x, y)$ 為原影像、 T 為臨界值， $g(x, y)$ 為 1 是黑色，反之則是白色。

2.4 影像型態學

影像形態學[7]，是影像處理中的其中一環，專門用來處理、分析影像中的形狀。一般形態學可分幾個基本運算元，例如膨脹(dilation)、侵蝕(erosion)、斷開(open)、閉合(close)，以下將運算元分別的說明。

2.4.1 膨脹

膨脹(dilation)是將二值圖形，等量向外擴張的演算法，假設有兩個二值化影像集合 A 、 B ，使用 B 來膨脹(dilation) A ，記為：

$$A \oplus B \quad (2.11)$$

B 為一個遮罩，當影像遮罩 B 沿著影像 A 由左上到右下掃描時，若遮罩 B 鄰域範圍內有任何一點影像 A 的二值化的值為 1 時，則將遮罩 B 中心位置給新的影像 A' 設為 1，反之為 0，如圖 2.9 與圖 2.10 所示。

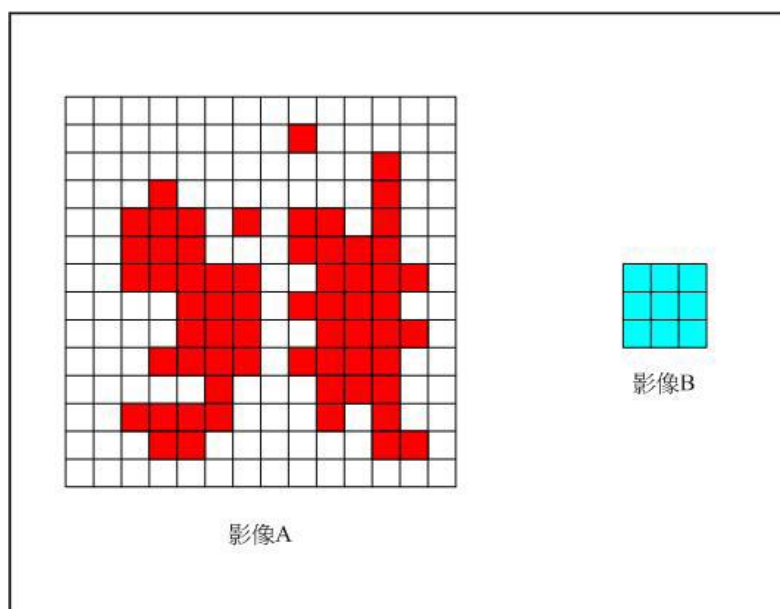


圖2.9 影像 A 與遮罩 B

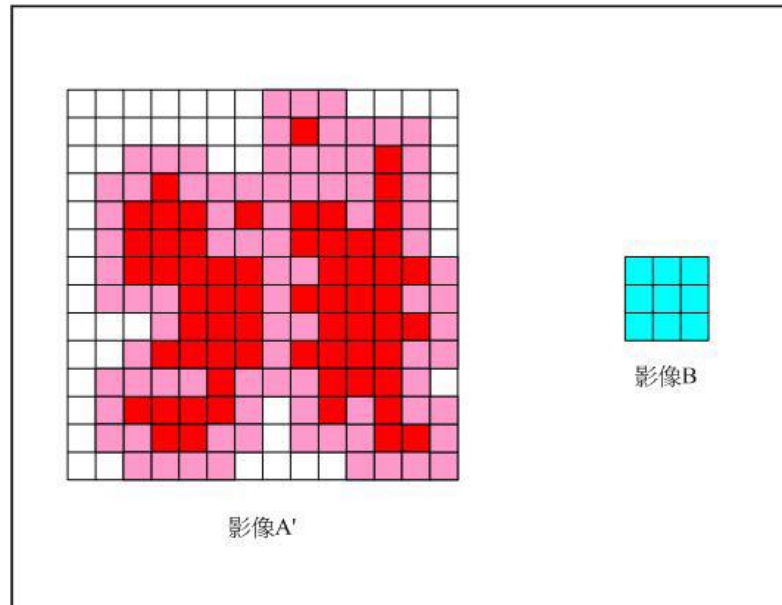


圖2.10 經膨脹後的影像 A 與遮罩 B

2.4.2 侵蝕

侵蝕(erosion)是等量向內縮減圖形的演算法侵蝕(erosion)，假設有兩個二值化影像集合A、B，使用B來侵蝕(erosion)A，記為：

$$A \ominus B \quad (2.12)$$

B為一個遮罩，當影像遮罩B沿著影像A由左上到右下掃描時，若遮罩B鄰域範圍內有任何一點影像A的二值化的值為0時，則將遮罩B中心位置給新的影像A' 設為0，反之為1，如圖2.11與圖2.12所示。

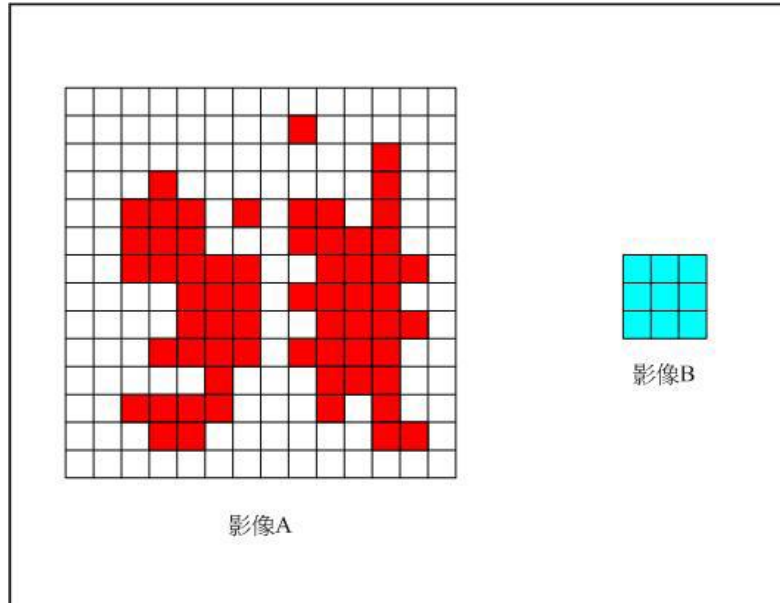


圖2.11 影像 A 與遮罩 B

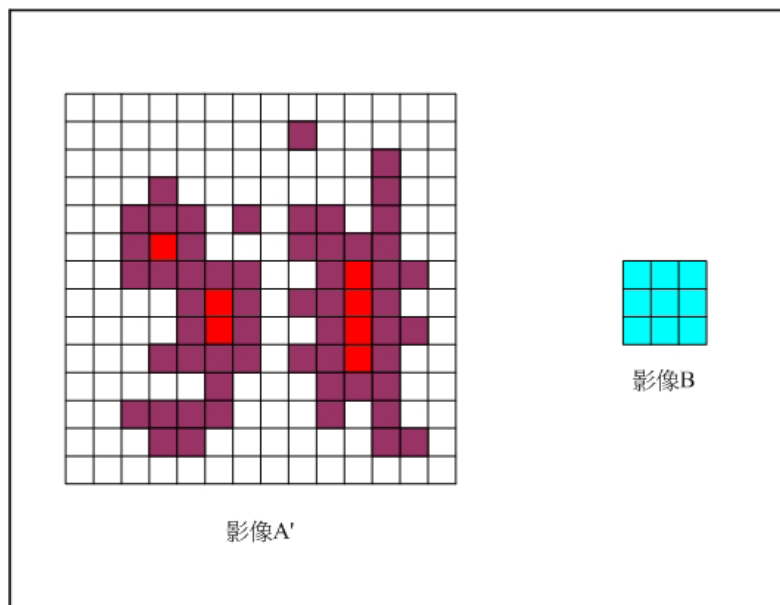


圖2.12 經侵蝕後的影像 A 與遮罩 B

2.4.3 斷開

斷開(open)是平滑影像輪廓，截斷窄的相連，消除細的突出，也就是先侵蝕後膨脹，假設有兩個二值化影像集合A、B，並使用B來斷開(open) A，記為：

$$AOB = (A \ominus B) \oplus B \quad (2.13)$$

先將遮罩B侵蝕影像A後，再膨脹影像A，結果如圖2.13。通常用在平滑影像，截斷細小連接點，去除細微突出部份。

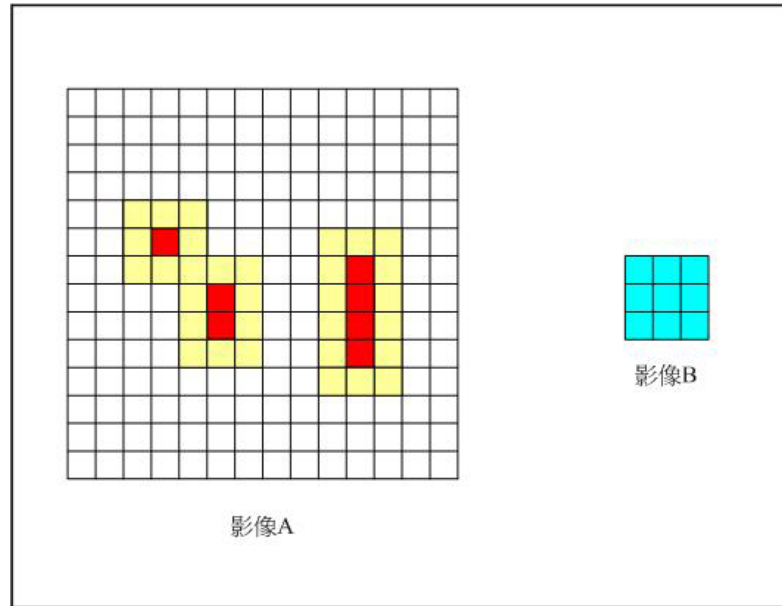


圖2.13 經過斷開後的影像 A 與遮罩 B

2.4.4 閉合

閉合(close)傾向於平滑輪廓，不過與斷開的作用是相反的。它會把窄的中斷部份和長細缺口補起來，以消除小洞、填補輪廓上的缺口，也就是先膨脹後侵蝕，假設有兩個二值化影像集合A、B，並使用B來閉合(close)A，記為：

$$A \bullet B = (A \oplus B) \ominus B \quad (2.14)$$

先將遮罩B膨脹影像A後，再侵蝕影像A結果如圖2.14。通常也用在平滑影像，連接細微斷裂和紋路，並去除細小空洞。

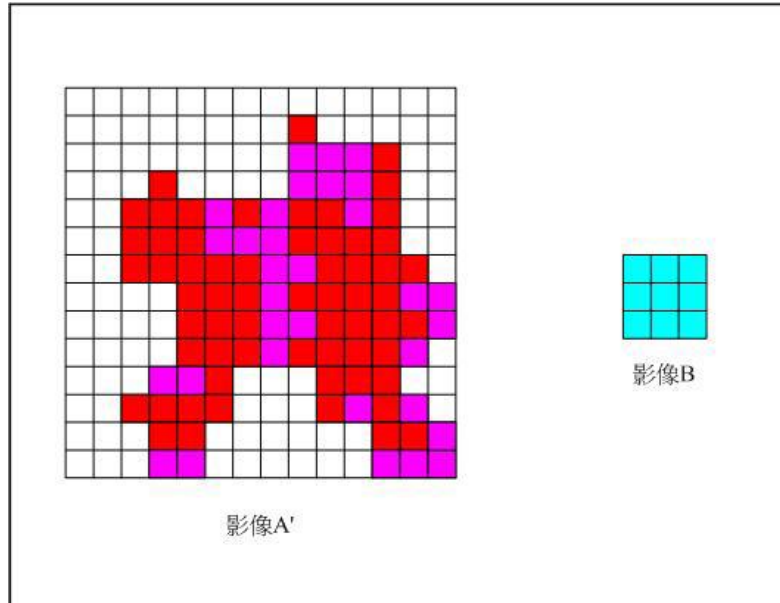


圖2.14 經過閉合後的影像 A 與遮罩 B

2.5 Viola 人臉偵測方法

Viola 與 Jones 提出了積分影像(Integral Image)概念和一個基於 Adaboost 的演算法[8]，此方法主要使用於搜尋人臉，並建立了一個即時的人臉偵測系統。此人臉偵測的方法主要有三個特性，以下將個別介紹。

- 使用積分影像達到快速的特徵計算。
- 使用 AdaBoost 演算法選取少量重要特徵來建構分類器。
- 使用層疊分類器類似串聯的方式，結合許多複雜分類器，此方式是以重視有用的影像區域來明顯地提高偵測速率。

2.5.1 積分影像

設定一個輸入影像 I，像素 (x, y) 處的積分影像值定義為式(2.15)。

$$ii(x, y) = \sum_{x \leq x', y \leq y'} i(x', y') \quad (2.15)$$

積分影像值 $ii(x, y)$ 等於陰影部分所有灰階值的和，如圖 2.15。

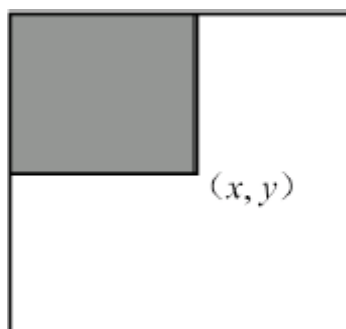


圖2.15 點 (x,y) 處的積分影像值

如果要得到一個輸入影像 I 的積分影像時，只需對原圖所有像素掃描一次，即可求得。計算公式如式(2.16)與式(2.17)

$$s(x, y) = s(x, y - 1) + i(x, y) \quad (2.16)$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \quad (2.17)$$

其中 $s(x, y) = 0$, $i(0, y) = 0$ 如圖 2.16。

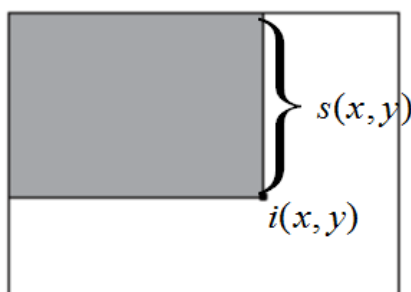


圖2.16 點 (x,y) 處的積分影像值示意圖

在使用積分影像時，任何影像內部的矩形和(Rectangular Sum)可以僅使用四個參考值計算即可取得，如下圖 2.17 所示。在 Viola 的系統中，每個矩形特徵值的計算，最多只需要從積分影像中取 9 個元素做加減運算。

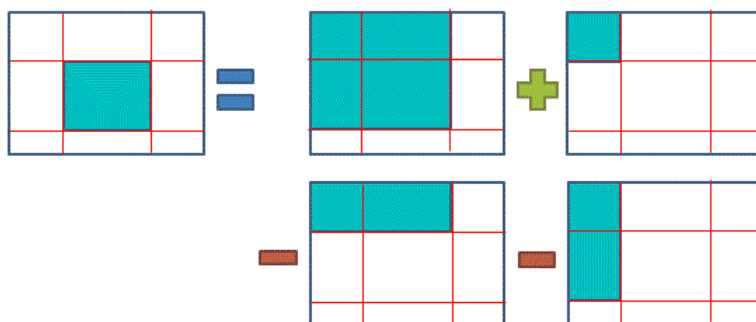


圖2.17 計算矩形和示意圖

2.5.2 Adaboost 演算法

Adaboost演算法是一種疊代方法，主要用途是透過從大量的弱分類器中，選取最具有分類意義的那些組合，成為一個強分類器，其中弱分類器定義為比隨機分類略好一點的分類器，亦即分類正確率略大於50%。在Viola的方法中，使用矩形特徵作為分類的依據。矩形特徵的值是指影像上兩個或者多個形狀大小相同的矩形內部所有像素灰階值之和的差值，如圖2.18與圖2.19。

不同大小與位置的矩型特徵，都可構成一個弱分類器。如表2.1所示，以給定為大小為24×24的影像為例，共會得到高達162,336個特徵值，亦即將建構出162,336個弱分類器，藉由AdaBoost每次疊代的訓練過程中，可從如此龐大數量的弱分器中，選取一個最具有分類意義的弱分類器，亦即分類誤差最小，透過多次的疊代，最後整合選出的數個弱分類器建構成一個強分類器。

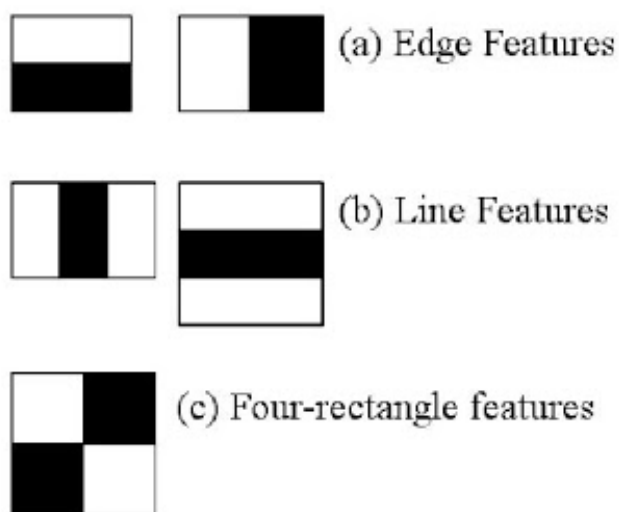


圖2.18 矩形特徵示意圖

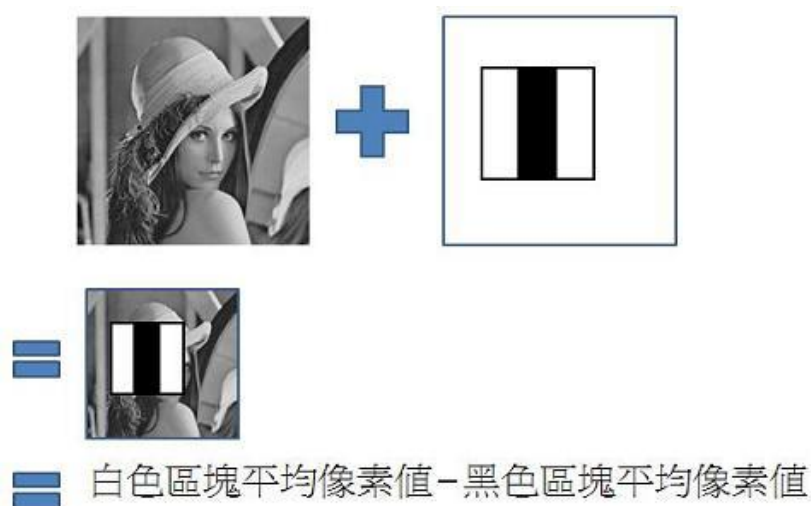


圖2.19 輸入影像中像素灰階值之和的差值

表2.1 影像尺寸與特徵數量之對應

影像尺寸	18×18	24×24	30×30	36×36
特徵數量	51705	162336	394725	816264

設定訓練用影像樣本集合 X ， $X = \{x_1, x_2 \cdots x_n\}$ ，對應之類別標籤 Y ， $Y = \{y_1, y_2 \cdots y_n\}$ ，而 $y_i = \{0, 1, i = 1, \cdots, n\}$ ，當 $y_i = 0$ 時代表非人臉，反之即為人臉。對 X 中所有影像於相同位置套用其中一類矩形特徵 f ，再求其特徵值 $f(x)$ ， $x \in X$ 。一個弱分類器構造如下。一個矩形特徵 j 對應著一個弱分類器 h_j ，對於一個候選輸入區塊 x ，設該矩形特徵 x 上的值為 $f_j(x)$ ，則弱分類器分類函數表達如式(2.18)

$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) < p_j \theta_j \\ 0 & \text{otherwise} \end{cases} \quad (2.18)$$

其中 p_j 為 ± 1 ， θ_j 為一個閾值， x 為一個影像中 24×24 像素的子區塊，對於這樣大小的區塊，共會建構出 162,336 個弱分類器。在實際計算中，必須找到那些對於分類比較重要的矩形特徵，Adaboost 演算法是選取這些特徵的有效手段。在給定的訓練樣本上，Adaboost 演算法從所有可能的弱分類器中挑選錯誤最低的

那一個，然後改變樣本的權值，使得那些被錯分的樣本得到進一步重視，重複上述操作。這樣，每一步操作都得到一個弱分類器，最後的強分類器由這些弱分類器的線性組合構成。

而弱分類器的結構代碼如下所示：

```
typedef struct CvCARTHaarClassifier
{
    CV_INT_HAAR_CLASSIFIER_FIELDS()

    int count;

    int* compidx;

    CvTHaarFeature* feature;

    CvFastHaarFeature* fastfeature;

    float* threshold;

    int* left;

    int* right;

    float* val;
}
CvCARTHaarClassifier;
```

其中 threshold 表示為演算法中的閾值，再將這些弱分類器的線性組合，去構成一個強分類器，而強分類器的結構代碼如下：

```
typedef struct CvStageHaarClassifier
{
    CV_INT_HAAR_CLASSIFIER_FIELDS()

    int count;
```

```

float threshold;

CvIntHaarClassifier** classifier;
}

CvStageHaarClassifier;

typedef struct CvIntHaarClassifier
{
    CV_INT_HAAR_CLASSIFIER_FIELDS()
}

CvIntHaarClassifier;

```

矩形特徵的主要目的，就是為了將人臉特徵量化，以區分偵測到的影像是否為人臉，使用積分影像可求得此矩形和，如圖 2.20，設定範例圖像 $(x_1, y_1) \cdots (x_n, y_n)$ ，使用 $y_i = 0, 1$ 做為區別，若 $y_i = 0$ 表示非人臉，而 $y_i = 1$ 則為人臉，如果把這樣的矩形放在一個非人臉區域，那麼計算出的特徵值將會和人臉特徵值不一樣，藉由此分辨出人臉，如圖 2.21。

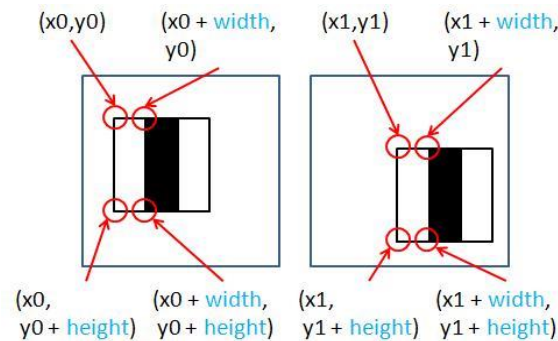


圖2.20 利用積分影像計算出矩形和



圖2.21 利用矩形特徵分辨是否為人臉

Adaboost演算法的完整流程為以下:

首先定義 x_i 為第 i 筆輸入資料, $i=1,2,\dots,N$, 其中 $x_i \in X$ 為維度空間 X 之中的特徵向量, y_i 為 x_i 的標籤資訊, 其中 $y_i \in Y = \{-1,+1\}$ 。

接著定義輸入 $(x_1, y_1) \dots (x_N, y_N)$ 與指定弱分類器的數量 T 。再做初始化, 設定初始樣本權重 d_n , 而 d_n 代表第 n 個樣本權重, 又 $d_n^{(t)} = 1/N$ for all $n=1,2,\dots,N$ 。其中權重是一個相對的概念, 是針對某一指標, 在該指標的整體評價中, 其相對重要的程度。

再定義輸出之強分類器 $H(x) = \text{sign}(\sum_{t=1}^T \alpha_t h_t(x))$, 從 $t=1, \dots, T$, 重複執行以下的步驟:

1. 先使用樣本機率分佈 $d^{(t)}$ 訓練弱分類器 $h_t: X \rightarrow \{-1,+1\}$ 。

2. 計算弱分類器 h_t 加權誤差 $\varepsilon_t = \sum_{n=1}^N d_n^{(t)} I(y_n \neq h_t(x_n))$ 。

3. 計算弱分類器之權重 $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$ 。

4. 最後更新樣本機率之分佈 $d_n^{(t+1)} = \frac{d_n^{(t)} \exp(-\alpha_t y_n h_t(x_n))}{Z_t}$ 。

2.5.3 層疊分類器

層疊分類器, 是由多個強分類器的組合, 如圖 2.22 所示。每一層都是 Adaboost 演算法訓練得到的一個強分類器, 都經過閾值調整, 使得每一層都能讓幾乎全部人臉樣本通過, 而排除了大部分非人臉樣本。在通過第一層分類器時, 進行決策排除非人臉樣本而保留下大多數人臉樣本, 接著再送到第二層分類器不斷地重複篩選過程, 直到最後一層所輸出的結果皆為人臉樣本。透過此方法可快速的排除非人臉樣本, 達到快速的人臉偵測。

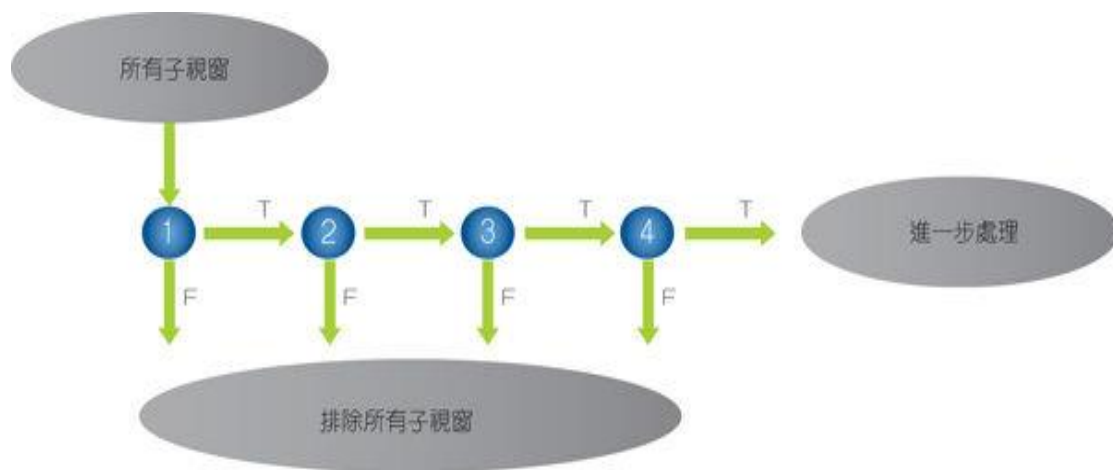


圖2.22 層疊分類器的示意圖

在分類器的訓練過程當中，每一層的Adaboost分類器，將會去指定成功率與錯誤率要達到定值，才算是完成訓練，而其成功率與錯誤率可表示為式(2.19)與(2.20)， D 為總正確偵測率， F 為總錯誤偵測率。

$$D = \prod_{i=1}^k d_i \quad (2.19)$$

$$F = \prod_{i=1}^k f_i \quad (2.20)$$

其中 d_i 表示第 i 層之偵測成功率， f_i 表示第 i 層之偵測錯誤率， k 為分類器之數量。

層疊分類器的結構代碼如下所示：

```

typedef struct CvTreeCascadeNode
{
    CvStageHaarClassifier* stage;

    struct CvTreeCascadeNode* next;

    struct CvTreeCascadeNode* child;

    struct CvTreeCascadeNode* parent;
}
  
```

```

    struct CvTreeCascadeNode* next_same_level;

    struct CvTreeCascadeNode* child_eval;

    int idx;

    int leaf;
}
CvTreeCascadeNode;

typedef struct CvTreeCascadeClassifier
{
    CV_INT_HAAR_CLASSIFIER_FIELDS()

    CvTreeCascadeNode* root;

    CvTreeCascadeNode* root_eval;

    int next_idx;
}
CvTreeCascadeClassifier;

```

2.6 人臉偵測與辨識

人臉偵測與辨識系統的首要步驟，就是對輸入的影像做人臉偵測，若在人臉偵測的步驟中發生錯誤，那勢必會影響後續的人臉辨識流程。因此如何在複雜的背景中，正確地擷取出人臉影像就成為最重要的部分。

人臉偵測大致的流程為，首先在偵測到的影像中搜尋膚色，由於 RGB 色彩空間會受到亮度極大的影響，因此需將影像由 RGB 色彩空間轉換成 YCbCr 色彩空間進行膚色偵測，順利偵測到膚色後使用 2.5 節所介紹的 Viola 人臉偵測方法判斷是否為人臉，將搜尋到的人臉做特徵點辨識，並加以擷取和分析，以達到人臉偵測與辨識的目的，圖 2.23 為人臉辨識的流程圖。

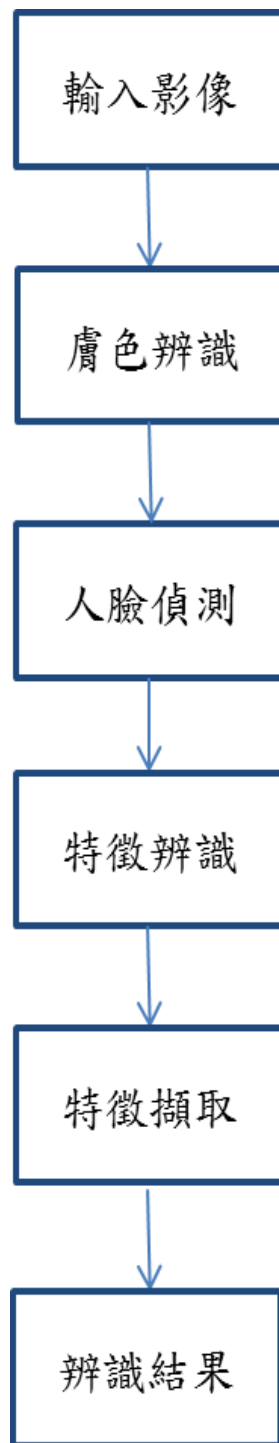


圖2.23 人臉辨識流程圖

首要方式為先偵測出影像中所有膚色區塊，一般臉部膚色區塊會比其他部位膚色區塊面積較大，用此特徵搜尋出影像中人臉可能的位置，並取出人臉的區塊影像。因為所取出的臉部影像將用於人臉辨識系統，所以必須有準確的臉部影像定位。當取得人臉之後，再做更進一步的特徵點分析。人臉辨識在取得膚色之後，可以分為兩種主要辨識方法，一是整體特徵方法，一是局部特徵方法。整體特徵

方法(Eigenface & Fisherface)直接將整張人臉當作單一特徵來做辨識;局部特徵方法先找出臉上的局部特徵，通常是眼睛、鼻子和嘴巴，然後分別根據這些局部特徵做辨識，最後將個別局部特徵的結果統合，得到最後的結果。近來的研究發現，局部特徵方法比整體特徵方法有更高的準確率，但局部特徵方法存在局部特徵的對位問題，在實做上有較高的困難度。

而使用局部特徵方法來尋找人臉的方法，有學者認為人臉中最顯而易見的是人臉中的眼睛，所以先利用影像處理正確的判斷出眼睛的位置後，再從眼睛的相對位置來畫出人臉位置。也有學者認為，從人臉的正面看到的眼睛到嘴巴可以連成一個三角形，若能找到該三角形的重心，則從此重心的周圍畫出一個圓，即偵測為人臉。若單純只利用人臉特徵的方法，容易因為背景環境受到干擾而產生誤判，因為不僅眼睛鼻子嘴巴有相對位置，連背景都有可能產生類似眼睛鼻子嘴巴的相對位置。而且我們有時取得的圖片，不一定是人臉的正面，使用人臉的五官特徵來偵測判斷是否為人臉，誤判的機率就會提高。

2.6.1 人臉特徵點偵測

人臉的特徵點可利用 OpenCV 所提供的五官分類器，來對人臉區域做五官偵測[9]，接著利用正常人臉五官的比例位置與大小關係，來定位出眼睛、嘴唇與鼻子或眼鏡等區域，接著辨識其特徵點。圖 2.24 為臉部上的特徵點比例圖，以下介紹各種特徵點的辨識方法。

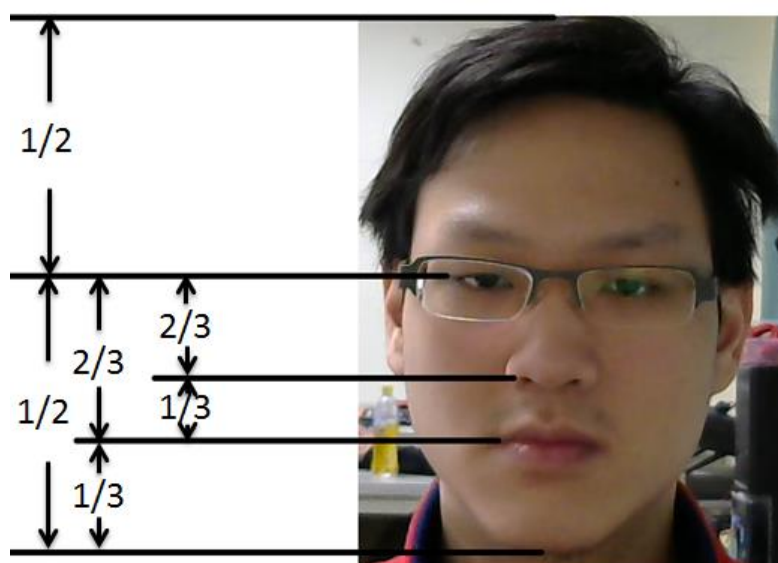


圖2.24 人臉特徵比例關係示意圖示

● 眼睛與嘴唇的特徵點偵測

因為 OpenCV 所提供的眼睛及嘴唇分類器會受到不同環境光線的影響，而為了減少眼睛及嘴唇的錯誤偵測率，先將眼睛及嘴唇的搜尋區域限制於人臉偵測步驟中所得到的臉區域當中，接著為了減少眼睛及嘴唇的錯誤偵測率以及提高偵測的速率，也將眼睛及嘴唇的搜尋區域侷限在特定區域。例如左右眼搜尋區域為人臉上半部，其高度則為人臉區域高度的 1/2，而嘴唇搜尋區域則限定在人臉區域高度 2/3 以下的區域。

● 鼻子的特徵點偵測

在找出眼睛及嘴唇的區域之後，可由觀察而得知一般正面人臉中鼻子與眼睛及嘴唇的比利位置及大小關係，並得到鼻子的高度等於嘴唇區塊寬度的 0.6 倍，鼻子的寬等於嘴唇區域寬度的 0.8 倍。由下列式(2.21)～式(2.23)可以得到鼻子區域左上角 (x, y) ：

$$X_{nose\ center} = X_{mouth} + Width_{mouth} \times 0.5 \quad (2.21)$$

$$X_{nose} = X_{nose\ center} - Width_{eye} \times 0.5 \quad (2.22)$$

$$Y_{nose} = Y_{eye} + Width_{eye} \quad (2.23)$$

最後再將鼻子的 X 及 Y 座標，分別加上鼻子的寬度及高度，即可定位出鼻子的區域。

● 眼鏡的特徵點偵測

取得眼睛的區塊中心會位於整體的人臉的高度的 2/7 部份，而眼鏡的大小大約位於眼睛與眉毛之間的高度，因此取眼鏡的範圍為影像高度的 1/10 (範圍越大修復時的參考來源影像會越多)，藉由如此的比例擷取，大部份的人臉中都可以正確的取得眼睛與眼鏡位置，擷取後的影像 FG (Face Glasses Image) 就是一張具眼鏡與人臉範圍的影像。

2.6.2 表情偵測

一般人類的臉部表情大致上可分為快樂 (Happiness)、驚訝 (Surprise)、害怕 (Fear)、生氣 (Anger)、傷心 (Sadness)、嫌惡 (Disgust) 等六類。透過定位出的人臉

座標找出空間中五官的特徵位置，甚至使用人類靠著臉部的肌肉的收縮與鬆弛，形成臉部表情的表情肌(Muscles of Facial Expression)，進一步建構出臉部的肌肉模型，來當作進行臉部表情辨識的特徵，最後根據特徵點的位置，計算出其表情的變化量加以實現表情辨識的應用。

2.7 人臉識別與特徵臉介紹

本論文的最終目的為實現人臉識別，其目的為找出影像中人臉並判斷出是否為資料庫中的人，其判別方法可使用特徵臉做辨識，利用每個人不同的特徵臉識別出不同的人，而特徵臉的獲得方法為主成分分析（PCA），以下將對特徵臉與主成分分析（PCA）作介紹。

2.7.1 特徵臉介紹

特徵臉（Eigenface）[10]，是指用於機器視覺領域中的人臉識別問題的一組特徵向量，這些特徵向量是從高維度向量空間的人臉圖像的協方差矩陣計算而來。

一組特徵臉可以通過在許多組描述不同人臉的圖像上，進行主成分分析（PCA）加以獲得。任意一張人臉圖像都可以被認為是這些標準臉的組合。例如，一張人臉圖像可能是特徵臉 1 的 10%，加上特徵臉 2 的 55%，在減去特徵臉 3 的 3%。值得注意的是，它不需要太多的特徵臉來獲得大多數臉的近似組合。另外，由於人臉是通過一系列向量（每個特徵臉一個比例值）而不是數字圖像進行保存，可以節省很多存儲空間。

特徵臉的最直接的應用就是人臉識別。在這個需求下，特徵臉相比其他手段在效率方面比較有優勢，因為特徵臉的計算速度非常快，短時間就可以處理大量人臉。但是，特徵臉在實際使用時有個問題，就是在不同的光照條件和成像角度時，會導致識別率大幅下降。因此，使用特徵臉需限制使用者在統一的光照條件下使用正面圖像進行識別。圖 2.25 為特徵臉示意圖。

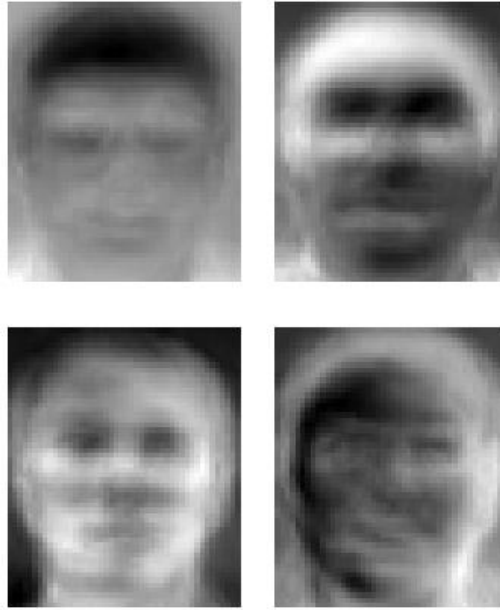


圖2.25 特徵臉示意圖

2.7.2 主成分分析(PCA)介紹

在多元統計分析中，主成分分析（Principal components analysis）[11]是一種分析及簡化數據集的技術。主成分分析經常用於減少數據集的維數，同時保持數據集中的對變異數貢獻最大的特徵。這是通過保留低階主成分，忽略高階主成分做到的。這樣低階成分往往能夠保留住數據的最重要方面。但是，這也不是一定的，要視具體應用而定。由於主成分分析依賴所給數據，所以數據的準確性對分析結果影響很大。圖 2.26 為 PCA 示意圖。

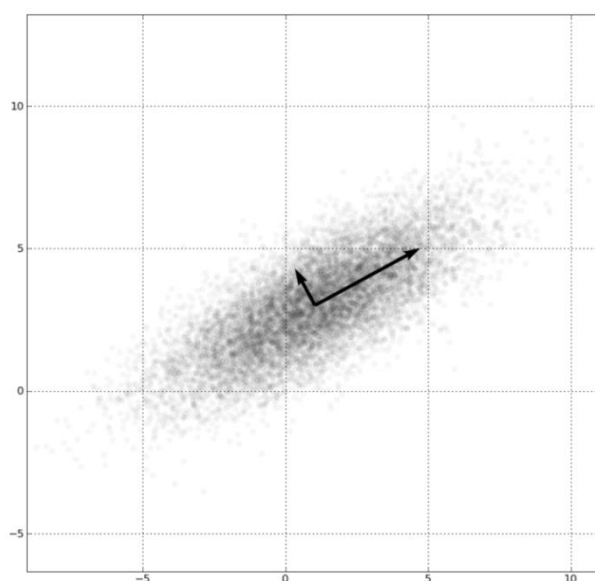


圖2.26 PCA 示意圖

主成分分析用於分析數據及建立數理模型。其方法主要是通過對共變異數矩陣進行特徵分解，以得出數據的主成分（即特徵向量）與它們的特徵值。PCA 是最簡單的以特徵量分析多元統計分佈的方法。其結果可以理解為對原數據中的變異數做出解釋，例如哪一個方向上的數據值對變異數的影響最大，換言之，PCA 提供了一種降低數據維度的有效辦法；如果分析者在原數據中除掉最小的特徵值所對應的成分，那麼所得的低維度數據必定是最優化的。主成分分析在分析複雜數據時尤為有用，比如人臉識別。

PCA 是最簡單的以特徵量分析多元統計分佈的方法。通常情況下，這種運算可以被看作是揭露數據的內部結構，從而更好的解釋數據的變量的方法。如果一個多元數據集能夠在一個高維數據空間座標系中被顯現出來，那麼 PCA 就能夠提供一幅比較低維度的圖像，這幅圖像即為在訊息最多的點上原對象的一個『投影』。這樣就可以利用少量的主成分使得數據的維度降低了。PCA 跟因子分析密切相關，並且已經有很多混合這兩種分析的統計包。而真實要素分析則是假定底層結構，求得微小差異矩陣的特徵向量。而主成份分析具有下列三點特性：

1. 各主要成份向量方向互相垂直。
2. 各主要成份向量為獨立向量。
3. 各主要成份向量為互相正交，彼此的相關係數為 0。

假設向量 $b = \{V_p \cdots, V_n\}$ 為向量空間 V 的一組基底。每一個屬於 V 的向量 u 能夠以 b 的線性組合表示，如(2.24)式所示：

$$u = \sum_{i=1}^n \lambda_i v_i \quad (2.24)$$

假設 b 為線性獨立，則(2.25)式成立。

$$\sum_{i=1}^n \alpha_i v_i = 0 \quad (2.25)$$

當 $\alpha_i = 0$ ， $i = 1 \cdots n$ ，則(2.24)式的 λ_i 係數是唯一。

假設一組向量 x^n ， $n = 1 \cdots N$ ，從 $V = R^d$ 投影到 V 在 $u = R^M$ 的子空間向量 Z^n 。

選擇 u_p, \dots, u_M 作為 u 的基底，其中所有向量的基準為 1，且互為正交，如(2.26)式所示：

$$u_i^T u_j = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \quad \delta_{ij} : \text{Kronecker delta} \quad (2.26)$$

以一組向量 $\{x_p \cdots, x_d\}$ 的形式代表向量 x ，或是以(2.27)式表示：

$$x = \sum_{i=1}^M x_i u_i + \sum_{i=M+1}^d x_i u_i \quad (2.27)$$

將主要部份 M 向量投影到 M 維空間得(2.28)式，其中 b_i 為常數。

$$z = \sum_{i=1}^M x_i u_i + \sum_{i=M+1}^d b_i u_i \quad (2.28)$$

如何選擇係數 b_i 和向量 u_i 使得被投影向量 z^n 最接近原始向量 x^n 。經(2.29)式

計算兩向量間平方誤差的總和，可以得知 z^n 與 x^n 近似的程度。

$$E = \frac{1}{2} \sum_{n=1}^N \|x^n - z^n\|^2 = \frac{1}{2} \sum_{n=1}^N \sum_{i=M+1}^d \sum_{j=M+1}^d (x_i^n - b_i)(x_j^n - b_j) u_i^T u_j \quad (2.29)$$

將(2.26)式代入(2.29)式得到(2.30)式的結果。

$$E = \frac{1}{2} \sum_{n=1}^N \sum_{i=M+1}^d (x_i^n - b_i)^2 \quad (2.30)$$

在將 b_i 映射到 0，得到(2.31)式。

$$b_i = \frac{1}{N} \sum_{n=1}^N x_i^n \quad (2.31)$$

(2.31)式由 i 個向量 x 得到平均向量 \bar{x} ，將此座標系統映射到座標 u_p, \dots, u_d ，

則將(2.29)式改寫為(2.32)式。其中 C 為資料的協方差矩陣(auto-covariance matrix)。

$$E = \frac{1}{2} \sum_{i=M+1}^d \sum_{n=1}^N \{u_i^T (x^n - \bar{x})\} \{(x^n - \bar{x}) u_i\} = \frac{1}{2} \sum_{i=M+1}^d u_i^T C u_i \quad (2.32)$$

使用 Lagrange 乘法器得到，當 E 映射到向量 u_i 的不變點發生在 C 的特徵向量，所以將 $C u_i = \lambda_i u_i$ 代入(2.32)式得(2.33)式的結果。

$$E = \frac{1}{2} \sum_{i=M+1}^d \lambda_i \quad (2.33)$$

C 為具有對稱性的矩陣，當矩陣的每一個特徵向量 $\lambda_i \geq 0$ 。假設有一組 d 特徵向量，可以經由選擇 $d-M$ 的最小特徵值，使得誤差 E 能夠達到最小。再取投影資料與最大的 M 特徵值相對應的特徵向量空間，稱為主成份向量。

通常將主要成份以遞減的特徵值表示，所以第一個主要成份與最大的特徵值一致，並且能呈現原始變化量的最大變異的線性組合，所以能選擇一小部份的變化值來決定原始資料的變化量，如(2.34)式所示：

$$\sum_{i=1}^M \lambda_i = \sum_{i=1}^d \lambda_i \quad (2.34)$$

2.7.3 PCA 用於人臉識別原理

PCA 會被用來作為人臉特徵擷取的理由有下面三個要點：

1. PCA 方法能夠快速容易的計算出結果。
2. 在線性投影中 PCA 方法能夠保留下投影資料中最大的資訊。
3. PCA 是用整個臉部取特徵，可以克服人臉有戴眼鏡與表情的變化。

假設有 M 張大小為 $N \times N$ 像素之人臉影像，經訓練建立成人臉資料庫，將每一張影像以列(row)串接的方式表示成 M 個向量，表示為 $\Gamma_1, \Gamma_2, \dots, \Gamma_M$ 其中每個 Γ 以向量維度 $N^2 \times 1$ 來描述 $N \times N$ 影像，如圖 2.27 所示。

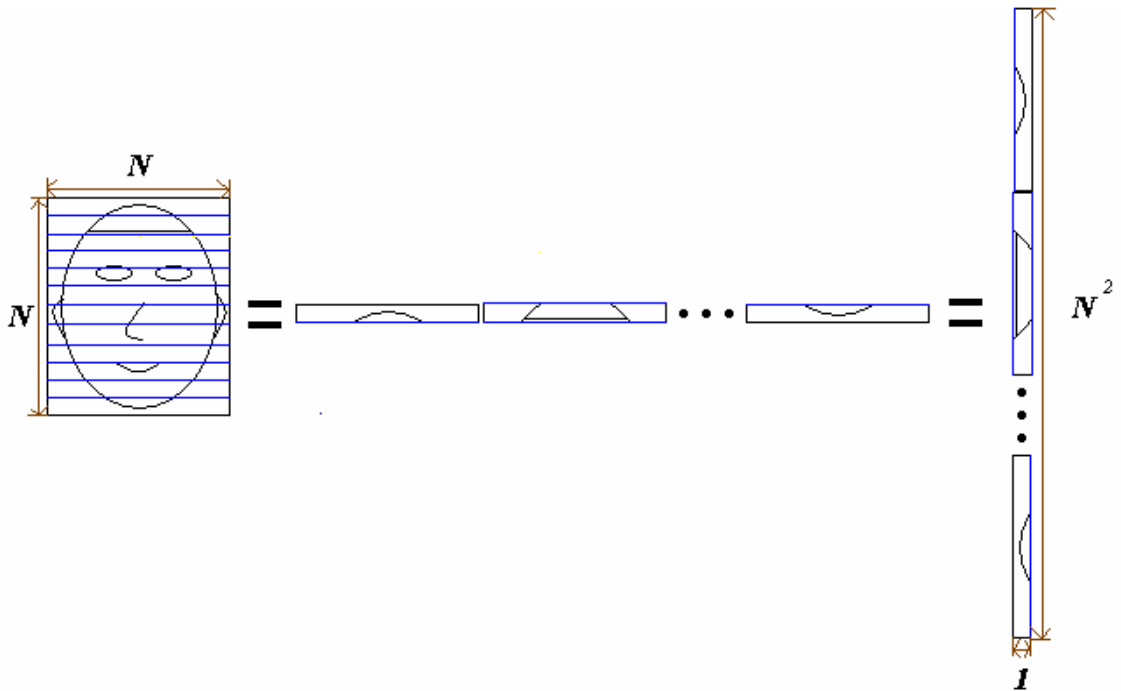


圖2.27 人臉影像轉成向量 Γ 的過程

利用 M 張影像所個別產生的 Γ ，找出平均向量(mean vector) Ψ 以(2.35)式表示：

$$\Psi = \frac{1}{M} \sum_{j=1}^M \Gamma_j \quad (2.35)$$

平均值 Ψ 代表 M 張影像所共有的特徵成份，也就是所謂的平均臉(mean face)，為了突顯這些影像彼此不同的部份，必須將 M 張影像中共有的成份移除，得到每一張影像的差像(difference image)向量，為(2.36)式所示：

$$\varphi_j = \Gamma_j - \Psi \quad j=1, \dots, M \quad (2.36)$$

使用 PCA 的投影來辨識人臉的想法，是想找到一向量 u_1 ，使得所有差像在 u_1 上的投影平方和為最大，如(2.37)式所示：

$$\max_{u_1} S = \sum_{j=1}^M \left(\frac{\varphi_j^T u_1}{\|u_1\|} \right)^2 = \frac{u_1^T \left(\sum_{j=1}^M \varphi_j \varphi_j^T \right) u_1}{u_1^T u_1} = \frac{u_1^T C u_1}{u_1^T u_1} \quad (2.37)$$

其中 C 為所有人臉影像的協方差矩陣(covariance matrix)，定義如(2.38)式所示：

$$C = \frac{1}{M} \sum_{j=1}^M \varphi_j \varphi_j^T = A A^T \quad (2.38)$$

其中， $A = [\varphi_1 \varphi_2 \cdots \varphi_M]$ ，根據 Rayleigh Quotient 定理，若 u_1 為矩陣 C 中最大特徵值所對應的最大特徵向量，則此時 S 為極大值，其餘之特徵向量將使 S 有局部極大值(local maximum)。為了得到這些特徵向量，在滿足(2.39)與(2.40)式條件下，求矩陣 C 的特徵值(eigenvalues) λ_k 與特徵向量(eigenvectors) u_k 。

$$C u_k = \lambda_k u_k \quad k=1, \dots, N^2 \quad (2.39)$$

$$\lambda_k = \frac{1}{M} \sum_{j=1}^M (u_k^T \varphi_j)^2 \quad (2.40)$$

其中 $u_l^T u_k = \begin{cases} 1 & \text{if } l=k \\ 0 & \text{otherwise} \end{cases}$ δ_{lk} : Kronecker delta，因為矩陣 A 的維度為

$N^2 \times M$ ，所以 C 為一個 $N^2 \times N^2$ 的大矩陣，對於這麼大的矩陣要求解所有的特徵

值與特徵向量是一件龐大計算的工作。如果不能有效的降低計算量，此處龐大的計算量將會使得人臉特徵資料庫在訓練工作上花費相當長的時間。解決的方法是先計算矩陣 $A^T A$ ，將矩陣維度降至 $M \times M$ 求其特徵向量 v_i ，如(2.41)式所示：

$$A^T A v_i = \mu_i v_i \quad i=1, \dots, M \quad (2.41)$$

將(2.41)式等號兩邊同乘矩陣 A 得到

$$A A^T A v_i = \mu_i A v_i \quad (2.42)$$

因為 $C = A A^T$ ，所以(2.42)式與(2.39)式比較得到 $u_k = A v_i$ 和 $\lambda_k = \mu_i$ ，所以利用(2.41)式的矩陣 $A^T A$ 求出特徵向量 v_i 再換算成特徵向量 u_k ，如(2.43)式所示：

$$u_k = \sum_{j=1}^M \varphi_j v_{ij} \quad i, k=1, \dots, M \quad (2.43)$$

所求得特徵向量 u_k 即為特徵臉(eigenface)。因為 $A A^T$ 為 $N^2 \times N^2$ 矩陣要接直求得特徵向量 u_k 所花費的計算時間很久，反觀由維度為 $M \times M$ 矩陣 $A^T A$ 所求得特徵向量 v_i ，再作線性轉換成 u_k 所花費的計算量會降低很多。

將各別的人臉影像向量 $\Gamma_1, \Gamma_2, \dots, \Gamma_M$ ，投影在由特徵臉所組成的特徵空間，求其在特徵空間的權重向量 f ，如(2.44)式所示：

$$\Omega_k = u_k^T (\Gamma - \Psi) = u_k^T \varphi \quad k=1, \dots, M \quad (2.44)$$

由於所有的差像 φ_j 在 u_1 軸上的投影分量為最大，在 u_2, u_3, \dots, u_M 上的投影為降冪排列的局部極大值。所以在實際應用上，可以省略後面幾個特徵軸的投影，也就是只留下主要成份(principal components)部份。將各別人臉訓練影像 Γ_j ， $j=1, \dots, M$ ，分別代入(2.44)式中的 Γ ，求出各人臉影像在特徵空間 $[u_1 u_2 \dots u_M]$ 上的權重向量 f ，所求得的 f_j ， $j=1, \dots, M$ ，即為人臉影像的資料庫。

2.8 OpenCV 介紹

OpenCV(Open Source Computer Vision Library)[12]，為本論文所使用的電腦視覺程式開發軟體，它是 Intel 開發的電腦視覺函式庫，裡面提供了許多影像處理有關的函式。它由一系列 C 函數和少量 C++類別構成，實現了影像處理和電腦視覺方面許多通用演算法。

OpenCV 擁有包括 300 多個 C 函數的跨平台的中、高層 API。它不依賴於其它的外部庫，但它也可以使用某些外部庫。OpenCV 對於非商業應用和商業應用都是免費的。而且 OpenCV 的 Code 有最佳化過，處理速度非常的快。目前 OpenCV 包含下面幾個部分：

- **CXCORE**

基本函數（各種數據類型的基本運算等）、支援 XML 與繪製函式。

- **CV**

圖像處理和電腦視覺功能（圖像處理、結構分析、運動分析、物體跟蹤、模式識別和攝影機定位目標）。

- **MLL**

Machine Learning Library（機器學習程式庫），包含統計分類器與分群工具。

- **CvAux**

背景分割、特徵物體與立體視覺。

- **HighGUI**

用戶互動部分（GUI、圖像視頻 I/O、系統調用函數）。

2.8.1 OpenCV 模組介紹

OpenCV 採用 BSD 授權條款，雖然提供完整的原始程式，但仍然受到專利保護，以下介紹廣泛被使用的 OpenCV 模組。

- **Core**

此模組定義了基本資料結構，包含緊密型多為陣列、矩陣與基本函式，供其他模組使用。

- **Imgproc**

此模組為影像處理模組，包含線性與非線性影像濾波器、幾何影像轉換、色彩空間轉換、直方圖等。

- **Video**

此模組為視訊分析模組，包含動作估計、背景相減法與物體追蹤演算法。

- **Calib3d**

此模組包含基本多重視圖幾何學演算法、單一與 3D 攝像頭校正、物體姿態估測、立體對應演算法、3D 重構元素。

- **Features2d**

此模組包含顯著特徵偵測器、描述子、以及描述子匹配器。

- **Objdetect**

此模組的功能包含物體偵測與預先定義類別的實體，例如臉部、眼睛、人群與汽車等。

- **Highgui**

此模組易於使用於錄影界面、影像與視訊編解碼器、簡單的 UI 功能。

- **GPU**

此模組包含從其他 OpenCV 模組而來、利用 GPU 加速執行的演算法。

2.8.2 OpenCV 與 Matlab 的工具箱比較

以下將 OpenCV 與 Matlab 中的圖像處理工具箱作比較功能：

- OpenCV 寫完後需要編譯，所以效率高；Matlab 使用腳本語言，直觀較方便。

- OpenCV 適合開發實際系統；Matlab 適合測試模擬時使用。
- OpenCV 開放 source code；Matlab 不提供。

2.8.3 OpenCV 使用介面介紹

先在 Raspberry Pi 中安裝好 OpenCV，之後可在 Raspberry Pi 的路徑/home/pi 找到其檔案，本論文所安裝的 OpenCV 版本為 2.4.9，在 Raspberry Pi 的螢幕上所顯示的結果如下圖 2.28。



圖2.28 OpenCV 檔案位置

進入 OpenCV 的路徑/home/pi/OpenCV-2.4.9/data/haarcascades 中，有許多內建的基本函式.xml 檔，如下圖 2.29，可使用這些檔案進行人臉特徵點辨識。



圖2.29 OpenCV 內建的基本函式.xml

2.8.4 OpenCV 之人臉識別應用

在此章節介紹 OpenCV 於人臉識別之應用：

- **PCA 分析**

使用 OpenCV 來創建用於標注訓練圖像的 PCA 分析，先將加載數據至一個矩陣 Mat 之中，通過 loadPCA 函數實現，以下為 loadPCA 函數的程式代碼。

```
PCA loadPCA(char* fileName, int& rows, int& cols, Mat& pcaset)
{
    FILE* in = fopen(filename, "r");
    Int a;
    fscanf(in, "%d%d", &rows, &cols);

    pcaset = Mat::eye(rows, cols, CV_64F);
    int i, j;

    for(i=0; i<rows; i++){
        for(j=0; j<cols; j++){
            fscanf(in, "%d", &a);
            pcaset.at<double>(i, j) = a;
        }
    }

    PCA pca(pcaset,
    Mat(),
    CV_PCA_DATA_AS_ROW,
    Pcaset.cols
    );
    Return pca;
}
```

- **人臉檢測**

在 WebcamFaceRec 項目之中包含 OpenCV 的 Haar 矩形特徵，以便於搜尋人臉，以下為偵測人臉的代碼。

```
Rect faceRect;
Int scaledWidth = 320;
detectLargestObject(cameraImg, faceDetector, faceRect, scaledWidth);
```

```

if (faceRect.width > 0)
cout << "We detected a face" << end;

```

而在搜尋到人臉後，需要有一個矩形框出人臉，擷取出人臉圖像，可使用以下代碼。

```

Mat faceImg = cameraImg(faceRect);

```

● 人臉預處理

此功能為調整人臉圖像，目的是為了讓圖像更為清晰，其代碼為以下。

```

Mat mask = Mat(warped.size(), CV_8UC1, Scalar(255));
double dw = DESIRED_FACE_WIDTH;
double dh = DESIRED_FACE_HEIGHT;
Point faceCenter = Point( cvRound(dw * 0.5),
                           cvRound(dh * 0.4) );
Size size = Size( cvRound(dw * 0.5), cvRound(dh * 0.8) );
ellipse(mask, faceCenter, size, 0, 0, 360, Scalar(0),
        CV_FILLED);

```

```

Filtered.setTo(Scalar(128), mask);

```

● 收集人臉

此功能為將收集到的人臉圖像，轉換為平均人臉後放入建立人臉資料庫中，之後要讓系統去辨識與判別，其程式代碼為以下。

```

Mat eigenvectors = model->get<Mat>("eigenvectors");
printMatInfo(eigenvectors, "eigenvectors");

for (int i = 0; i < min(20, eigenvectors.cols); i++) {
Mat eigenvector = eigenvectors.col(i).clone();
Mat eigenface = getImageFromIDFloatMat(eigenvector, faceHeight);

imshow(format("Eigenface%d", i), eigenface);
}

```

- 人臉識別

此功能會在建立的人臉資料庫中，利用 Mat 矩陣的特徵值作比對，找出與 Webcam 偵測的人臉最為相似的，如此就能判斷出 Webcam 偵測到的人是否為資料庫中的人，其程式代碼為以下。

```
Mat eigenvectors = model->get<Mat>("eigenvectors");
```

```
Mat averageFaceRow = model->get<Mat>("mean");
```

```
Mat projection = subspaceProject(eigenvectors, averageFaceRow,  
                                preprocessedFace.reshape(1,1);
```

```
Mat reconstructionRow = subspaceReconstruct(eigenvectors,  
                                             averageFaceRow, projection);
```

```
Mat reconstructionMat = reconstructionRow.reshape(1, faceHeight);
```

```
Mat reconstructedFace = Mat(reconstructionMat.size(), CV_8U);
```

```
reconstructionMat.convertTo(reconstructedFace, CV_8U, 1, 0);
```

第三章 實驗設備與架構

3.1 Raspberry Pi 介紹

樹莓派 (Raspberry Pi) [13]，是一款基於 Linux 系統的單板機電腦。它由英國的樹莓派基金會所開發，目的是以低價硬體及自由軟體刺激在學校的基本的電腦科學教育。樹莓派的生產是透過有生產許可的兩家公司：Element 14/Premier Farnell 和 RS Components。這兩家公司都在網上出售樹莓派。樹莓派配備一枚 700MHz 博通出產的 ARM 架構 BCM2835 處理器，256MB 內存（B 型已升級到 512MB 記憶體），使用 SD 卡當作儲存媒體，且擁有一個 Ethernet，兩個 USB 介面，以及 HDMI（支援聲音輸出）和 RCA 端子輸出支援。Raspberry Pi 只有一張信用卡大小，體積大概是一個火柴盒大小，可以執行像雷神之鎚 III 競技場的遊戲和進行 1080p 視訊的播放。操作系統採用開源的 Linux 系統，比如 Debian、ArchLinux，自帶的 Iceweasel、KOffice 等軟體能夠滿足基本的網路瀏覽，文字處理以及電腦學習的需要，圖 3.1 為 Raspberry Pi 的外觀。



圖3.1 Raspberry Pi 外觀圖示

最初銷售的樹莓派是 B 型，計劃 2013 年早些時候推出 A 型。A 型有一個 USB 介面，並沒有乙太網路控制器，成本將小於 B 型（有兩個 USB 埠和一個 10/100 以太網控制器）。

雖然 A 型沒有 RJ45 乙太網介面，但依然可以連接到網路（使用用戶提供的 USB 乙太網或 Wi-Fi 適配器）。現實中外部的乙太網適配器 A 型和 B 型一個內建之間沒有差異，因為 B 型的乙太網埠實際上是一個內建的 USB 乙太網路轉卡。

一如典型的現代電腦，樹莓派支援 USB 鍵盤和滑鼠。圖 3.2 為 A 型與 B 型外觀和功能之差異。

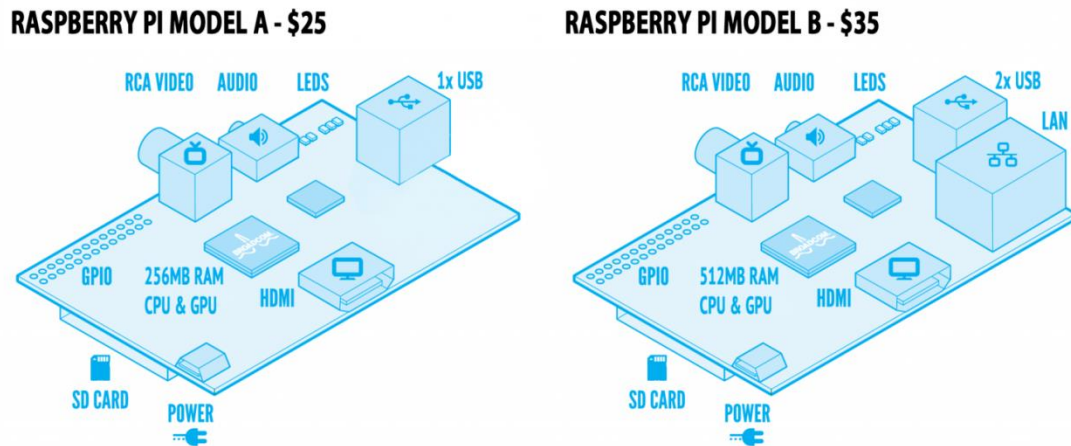


圖3.2 A 型與 B 型之比較圖

2012 年 2 月，樹莓派基金會發表一款 B 型的改良版，名稱為「Raspberry Pi model B+」，其主要差異為將輸入輸出的 GPIO 接腳從原本的 26 個提升至 40 個，以及 USB Ports 接頭從原本的 2 個提升至 4 個，其外觀如圖 3.3。

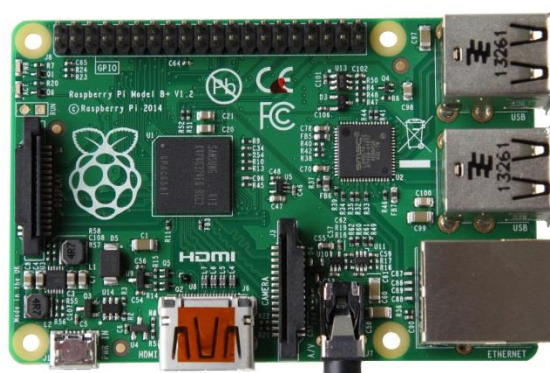


圖3.3 Raspberry Pi model B+ 外觀圖示

3.1.1 硬體介紹

樹莓派基金會提供了基於 ARM 架構的 Debian、Arch Linux 和 Fedora.....等等的發行版供大眾下載。還計劃提供支援 Python 作為主要程式語言,支援 BBC BASIC、C，和 Perl 等程式語言。其硬體規格與硬體配置如表 3.1 和圖 3.4。

表3.1 Raspberry Pi 硬體規格

型號	B+
SoC	BroadcomBCM2835
CPU	ARM1176JZF-S 核心 700MHz
記憶體	512 MByte
USB2.0 數量	4
音源輸出	3.5mm 插孔
板載儲存	SD/MMC/SDIO 卡插槽
網路介面	10/100 乙太網介面
工作電流	600mA
工作電壓	5V
總體尺寸	85.6mm x 53.98mm
重量	45g

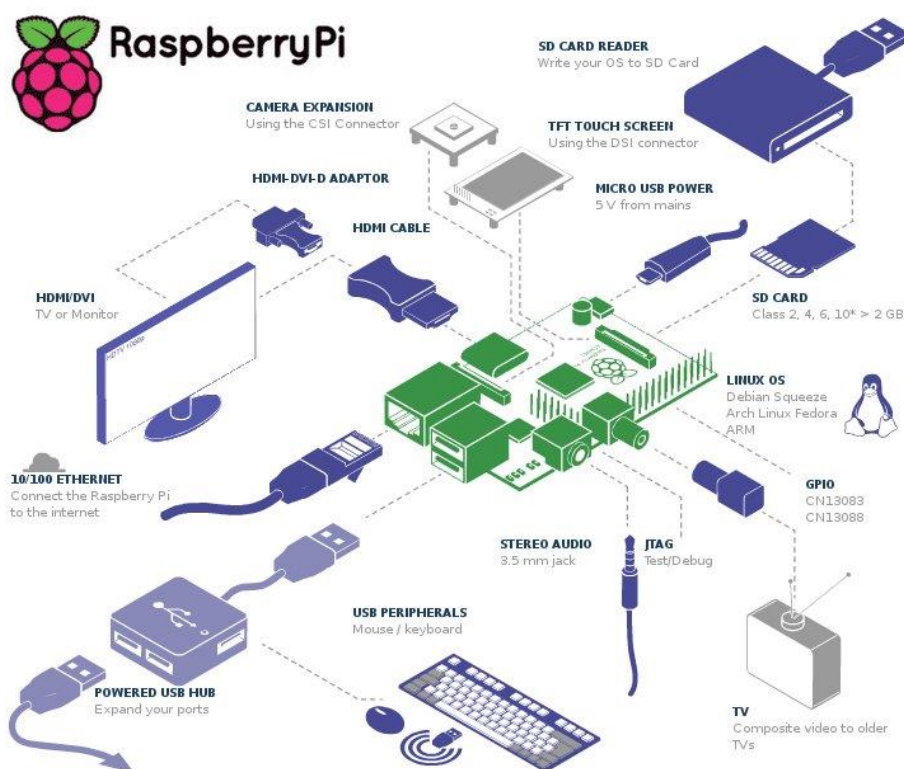


圖3.4 Raspberry Pi 硬體配置

Raspberry Pi 上面擁有 26 pin 的 GPIO 接腳，我們可以使用這些接腳來進行硬體控制。目前用來控制 Raspberry Pi 上的 GPIO 常見使用 Python、Java 以及 C 語言，其他如 Ruby、Perl、PHP 等在網路上也有人寫出函式庫提供使用。另外也有人直接使用 Shell script 來寫 GPIO 控制程式。其 GPIO 的功能可分為四類：

- **電源**

Model B 的針腳編號 1 與 17 的 3.3V，針腳編號 2 與 4 的 5V，針腳編號 6、9、14、20 與 25 的接地（GND）。Model B+與 Pi 2 Model B 多了針腳編號 30、34 與 39 的接地（GND）。這些針腳可以提供連接 Raspberry Pi 零件的電源，例如 LED 與感應器。不過要特別注意零件需要的電源是 3.3V 或 5V，如果接錯了，很有可能燒壞零件。

- **一般用途**

Model B 的針腳編號 7、11、12、13、15、16、18 與 22，Model B+與 Pi 2 Model B 多了針腳編號 29、31、32、33、35、36、37、38 與 40。這些針腳可以設定為輸入或輸出，設定為輸出的時候，可以使用程式控制輸出的電壓，例如沒有輸出的 0V 或 3.3V；設定為輸入的時候，可以透過程式偵測輸入的電壓。

- **I²C**

Model B 的針腳編號為 8 與 9。Model B+與 Pi 2 Model B 則多了針腳編號 27（I²C IDSD）與 28（I²C IDSC），保留給 PiPlate ID EEPROM 使用。I²C（Inter-Integrated Circuit）是一種串列通訊匯流排，由飛利浦公司在 1980 年代研發，讓主機板、嵌入式系統或行動電話可以連接低速週邊裝置而發展，是一種很常見的架構，有很多設備都採用 I²C 設計，例如行動電話的接近與光線感應器。

- **SPI**

針腳編號 19、21、23、24 與 26。SPI（Serial Peripheral Interface Bus）是一種類似 I²C 的序列資料傳輸協定，也有很多採用這種架構的設備。

圖 3.5 和圖 3.6 分別為 Model B 與 Model B+的 GPIO 針腳編號及功能，主要差異為 Model B+的 GPIO 接腳從原本的 26 個提升至 40 個。

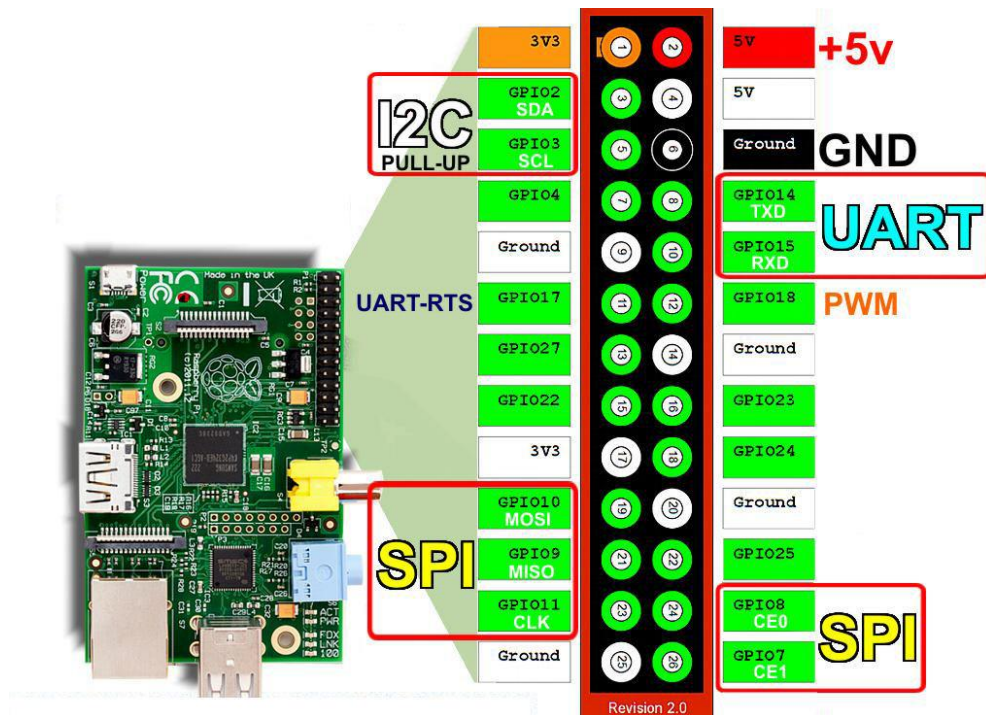


圖3.5 Modle B GPIO 針腳編號

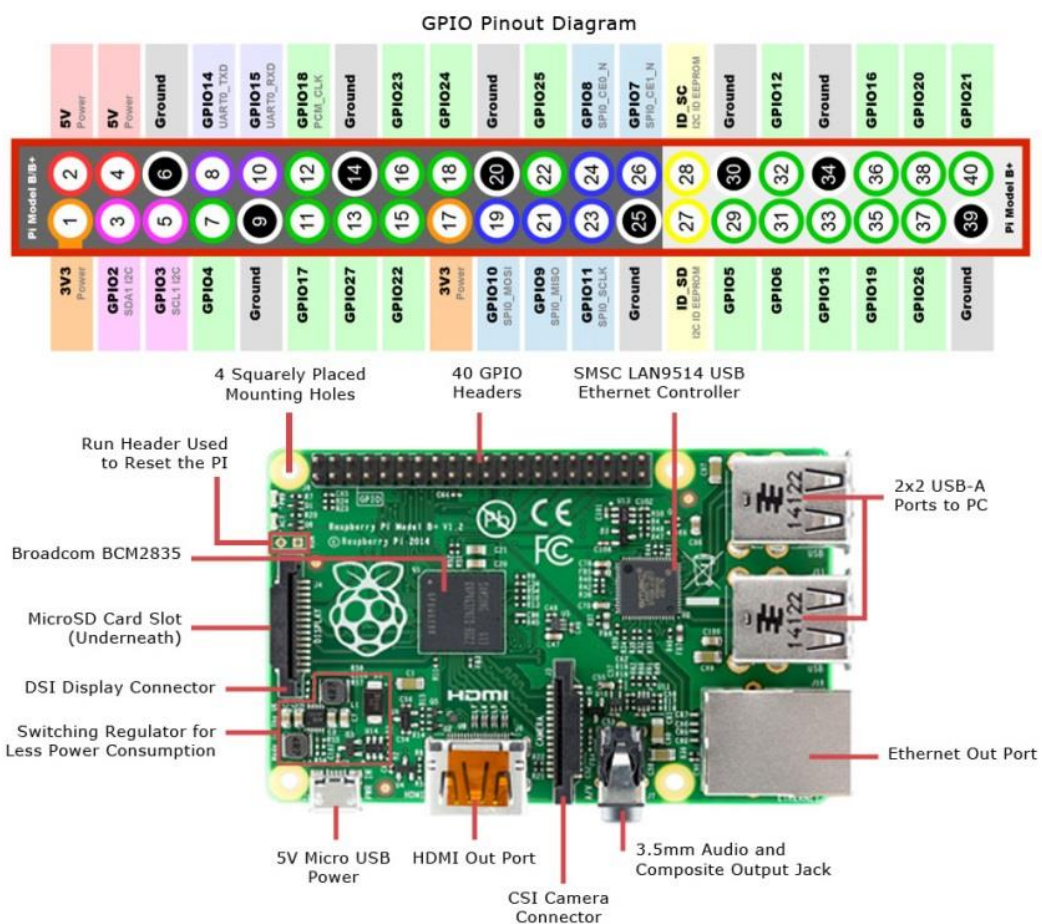


圖3.6 Modle B+ GPIO 針腳編號

3.1.2 作業系統

以下為能在樹莓派上執行的全部作業系統之清單：

- AROS
- Linux
- Android
- Android 4.0 (Ice Cream Sandwich)
- Arch Linux ARM
- Debian Squeeze
- Firefox OS
- Gentoo Linux
- Chromium OS
- Raspberry Pi Fedora Remix
- Raspbian (Debian Wheezy port with faster floating point support)
- Slackware ARM (formerly ARMedslack)
- WebOS
- Open webOS
- Kano
- Plan 9 from Bell Labs
- RISC OS
- Unix
- FreeBSD
- NetBSD

3.1.3 韌體介紹

樹莓派基金會使用基於 Linux 核心的作業系統。第一代樹莓派是基於 ARMv6 架構的 ARM11 晶片。目前的幾個流行的 Linux 版本，包括 Ubuntu 在內，將不能在 ARM11 上執行。在原生的樹莓派上是沒有可能執行 Windows，不過新的樹莓派 2 已經可以執行"Windows 10 物聯網核心版"。樹莓派 2 當前只支援 Ubuntu Snappy Core，Raspbian，OpenELEC and RISC OS。

在 2012 年 7 月，基金會發行了目前它推薦使用的作業系統，Raspbian，這是基於 Debian 的 Linux 發行版，並對於樹莓派的硬體的做了系統最佳化。Linux 不像大多數的作業系統只支援 Intel 架構的處理器，他可以支援樹莓派所使用的

ARM 處理器，但並不是每一種 Linux 都可以順利在 Pi 上執行，因為 Pi 上沒有 BIOS 與內建儲存裝置，因此必須透過 SD 卡來開機。圖 3.7 為 Raspberry Pi 的 API 連線的方框圖。

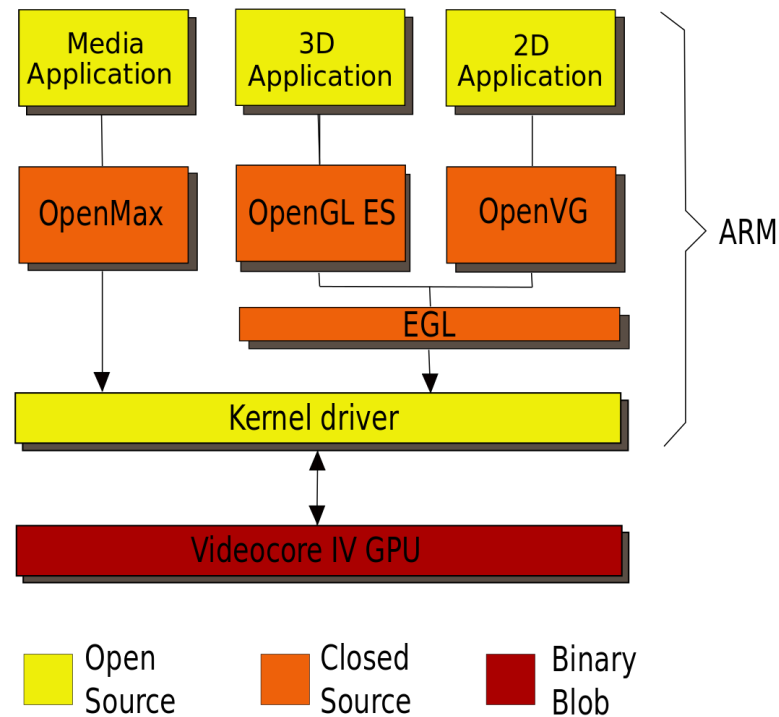


圖3.7 API 連線的方框圖

使用樹莓派之前置作業，需要在電腦上以讀卡機讀取 SD 卡，並使用 Win23DiskImager 把映像檔寫入 SD 卡內，接著把 SD 卡插入樹莓派中接上電源就可以開機並且等待使用者帳號登入，如圖 3.8。

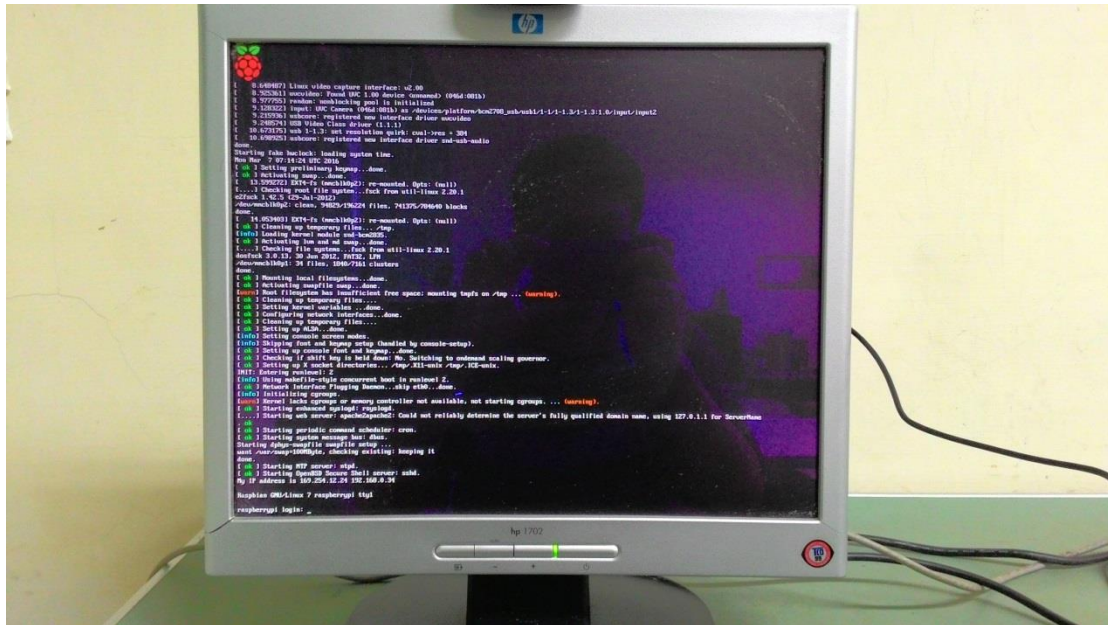


圖3.8 Raspberry Pi 開機畫面圖

當順利開機之後，會出現輸入帳號密碼的登入提示訊息，要求使用者做輸入動作，如圖 3.9 以及圖 3.10。

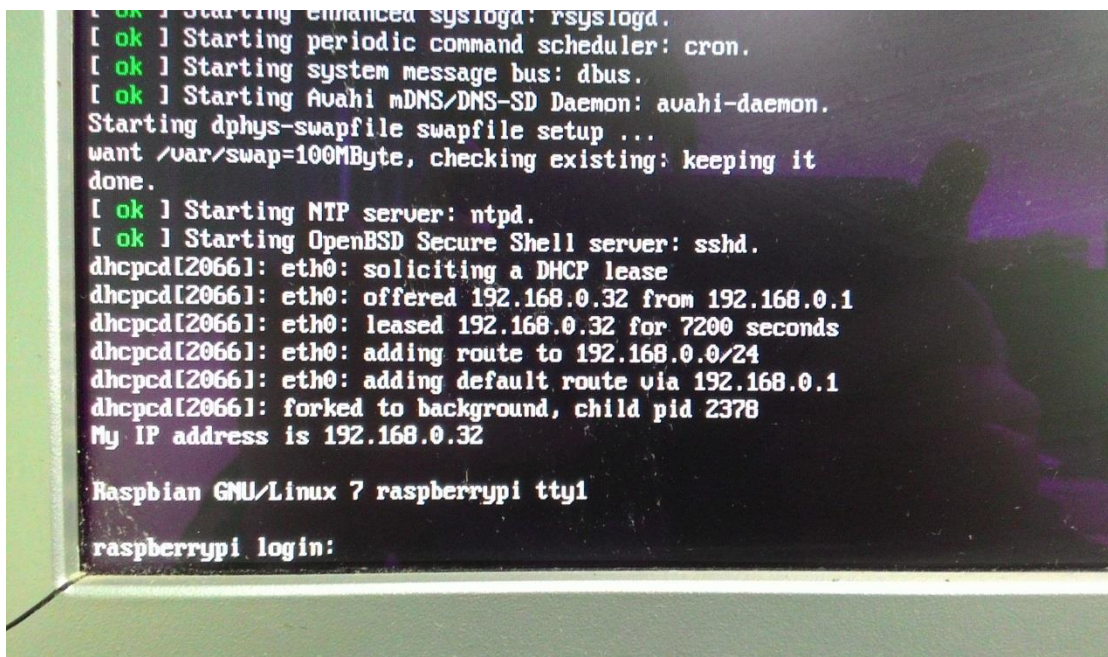


圖3.9 輸入使用者帳號

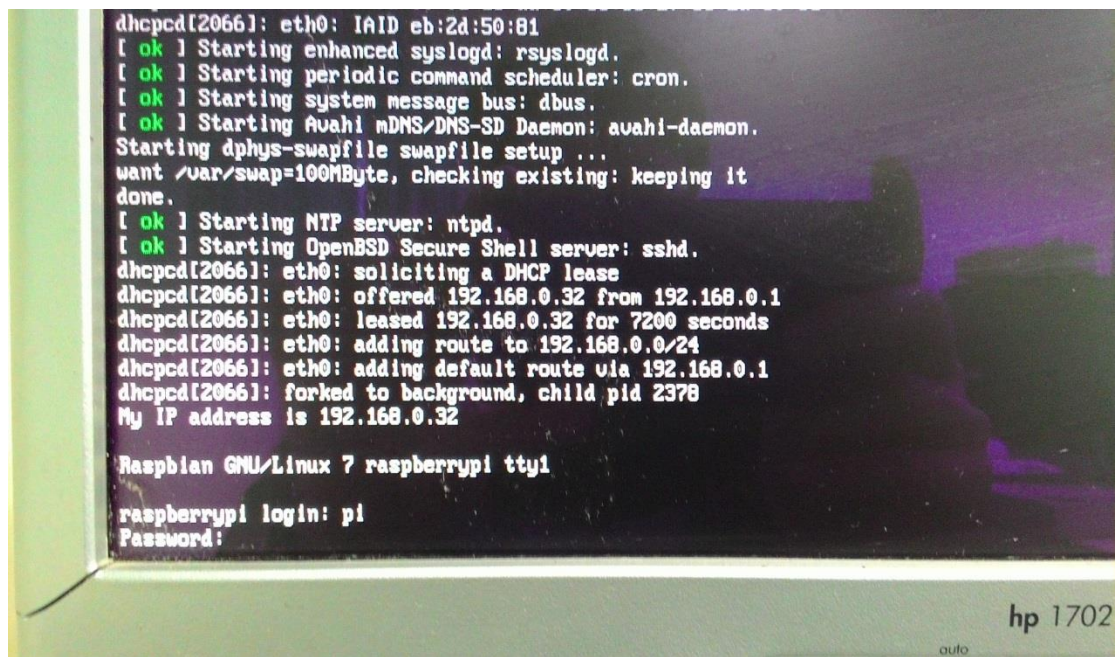


圖3.10 輸入使用者密碼

成功登入系統之後，就會出現如此的提示符號「pi@raspberrypi ~ \$」，如圖3.11。

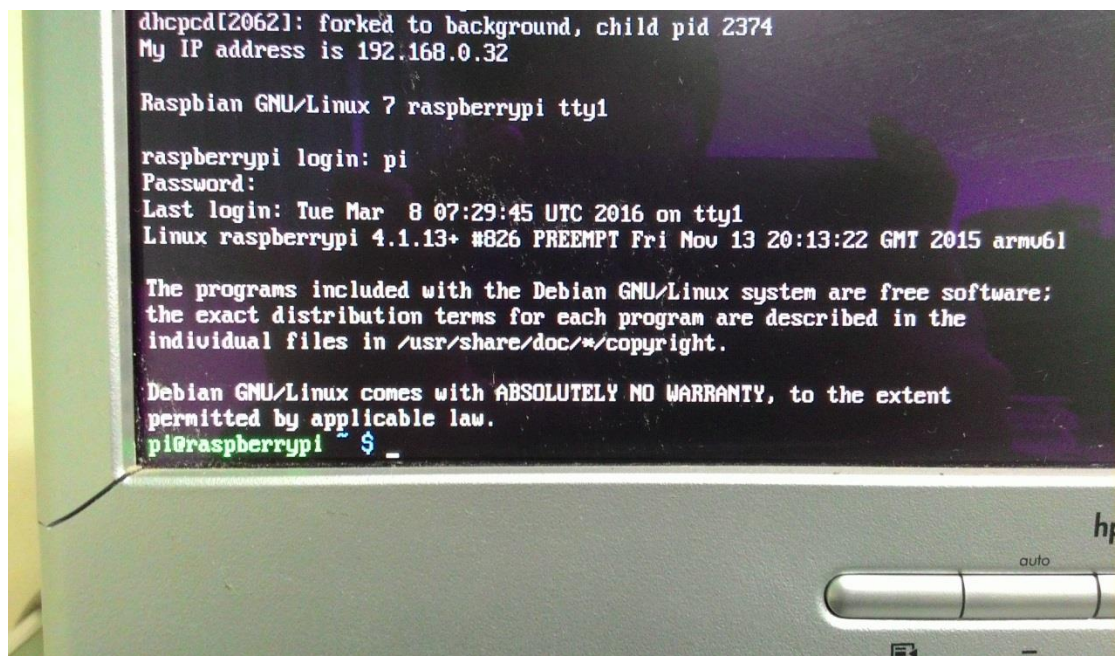


圖3.11 輸入指令的提示符號

在輸入 startx 後就可進入圖形化使用者介面 X 視窗的執行畫面，X 視窗是 Linux 系統所使用的圖形介面系統，使用此介面的原因為，由於以終端機介面操作系統時，需記住一些複雜的指令與參數，在使用上相當麻煩且不便，如果能使用視窗及一些圖示所建構出來的圖形化介面，便能簡化許多工作項目與雜物。此介面類似一般電腦 Windows 的桌面介面，圖型化的介面在使用上會更為方便。圖 3.12 為 Raspberry Pi 圖形化使用者介面的桌面。接著將介紹其各種圖型化介面的功能。

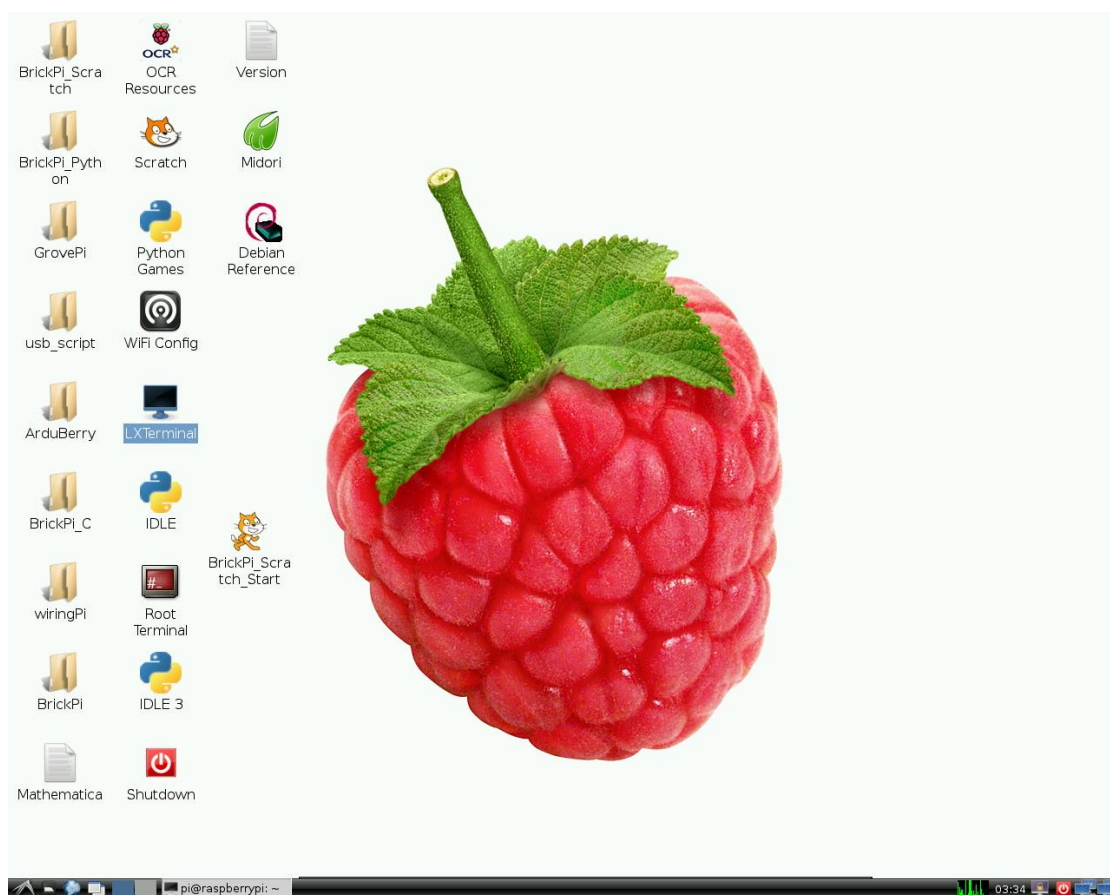


圖3.12 Raspberry Pi 圖形化使用者介面 X 視窗

以下為 Raspberry Pi 各種圖形化介面的功能介紹：

- **應用程式的 icon**

如 Windows 作業系統一樣，可以將常用的應用程式與檔案放在桌面，方便直接點選使用。如下圖 3.13

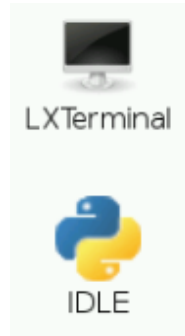


圖3.13 icon 圖示

- **系統選單 Menu Bar**

位於介面左下角，如 Windows 作業系統的 Start 開始，有系統常用的應用程式與功能。按鍵圖示如下圖 3.14。



圖3.14 系統選單按鍵圖示

- **檔案總管 File Manager**

位於介面左下角，可點選 icon 打開檔案總管 File Manager，如圖 3.15。



圖3.15 檔案總管按鍵圖示

- **切換虛擬畫面**

位於介面左下角，可轉換成另外一個桌面的功能，如圖 3.16。



圖3.16 切換虛擬畫面圖示

- **CPU 執行情況**

位於介面右下角，藉由右下角的小圖示，了解目前 CPU 的執行情況，如圖 3.17。



圖3.17 CPU 執行圖示

- **關機**

位於介面右下角，退出 Raspberry Pi 的系統，類似使電腦關機的功能，如圖 3.18。



圖3.18 關機圖示

3.1.4 遠端連線介紹

使用遠端連線的目的為，在沒有特殊需求的時候就不需為 Raspberry Pi 連接 HDMI 影音傳輸線與鍵盤，只要使用個人電腦透過遠端連線，就能執行各種需要的操作。

Raspberry Pi 內建 Secure Shell (SSH)的功能，提供遠端電腦較安全的連線和登入方式，使用 Raspberry Pi 的過程中會經常需要執行指令或修改檔案，而在使用 SSH 遠端登入之後，就能與直接在 Raspberry Pi 輸入指令一樣，在使用上更為方便。

本論文使用的遠端連線工具為「PuTTY」，它是一個 Telnet、SSH、rlogin、純 TCP 以及串列埠連線軟體。較早的版本僅支援 Windows 平台，後陸續增加對各類 Unix 平台和 Mac OS X 的支援。PuTTY 可以用來連線至許多不同的遠端電腦，能為每一台電腦建立一個連線設定資訊(Session)。圖 3.19 為其登入介面。

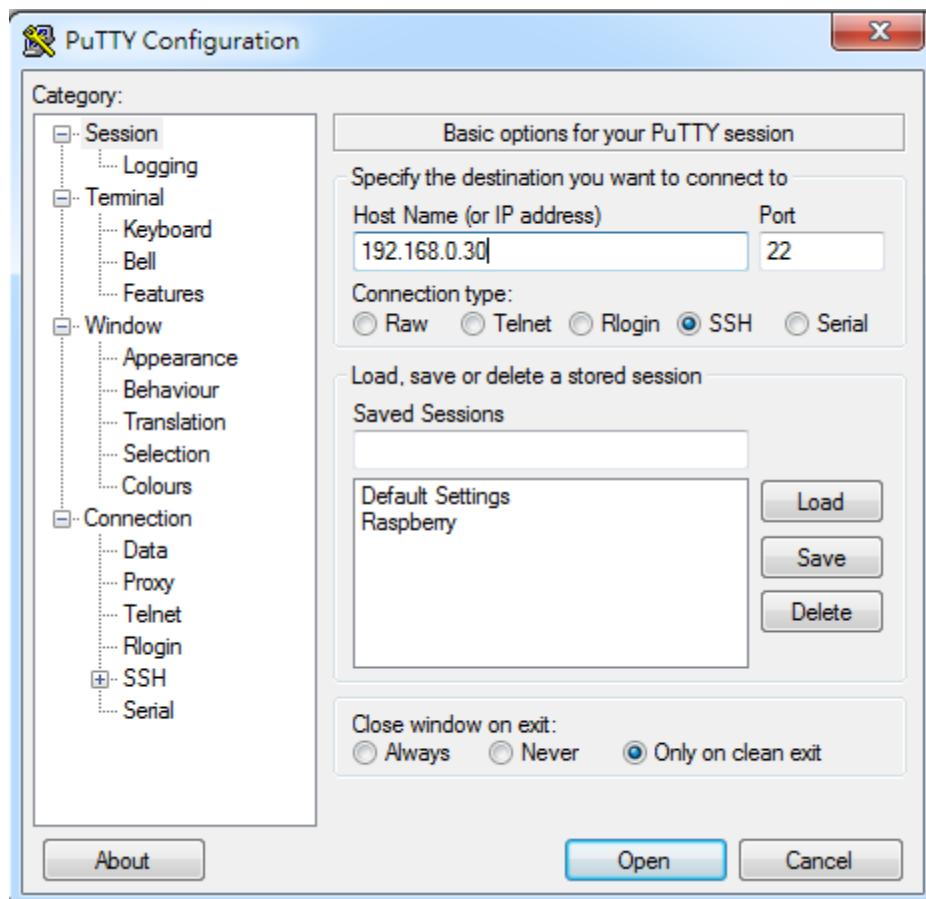


圖3.19 PuTTY 登入介面

3.1.5 硬體架構

圖 3.20 為本論文之硬體架構圖，其使用步驟為以下：

1. 先將作業系統燒錄至 SD 卡。
2. 再將 SD 卡插入 Raspberry Pi。
3. 再接上 HDMI 螢幕、鍵盤、滑鼠與 Webcam 網路攝影機。
4. 最後接上 5V 的電源，即可開機使用。

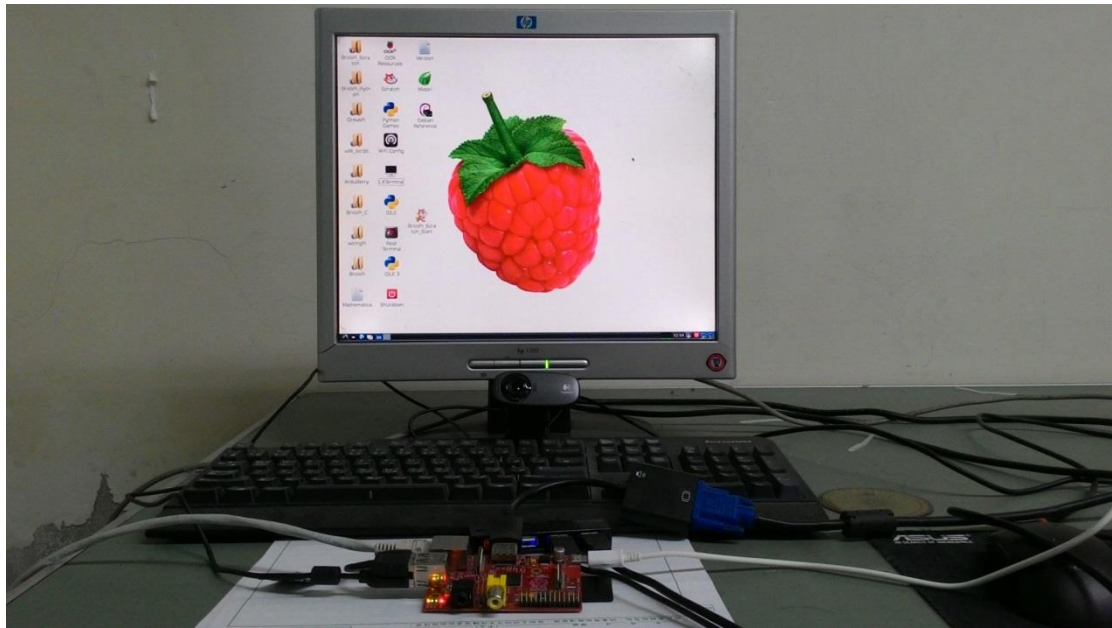


圖3.20 Raspberry Pi 硬體架構圖

3.1.6 Linux 介面介紹

點選 Raspberry Pi 圖形化使用者介面中的 LXTerminal 圖示，可開啟終端機介面，如下圖 3.21。

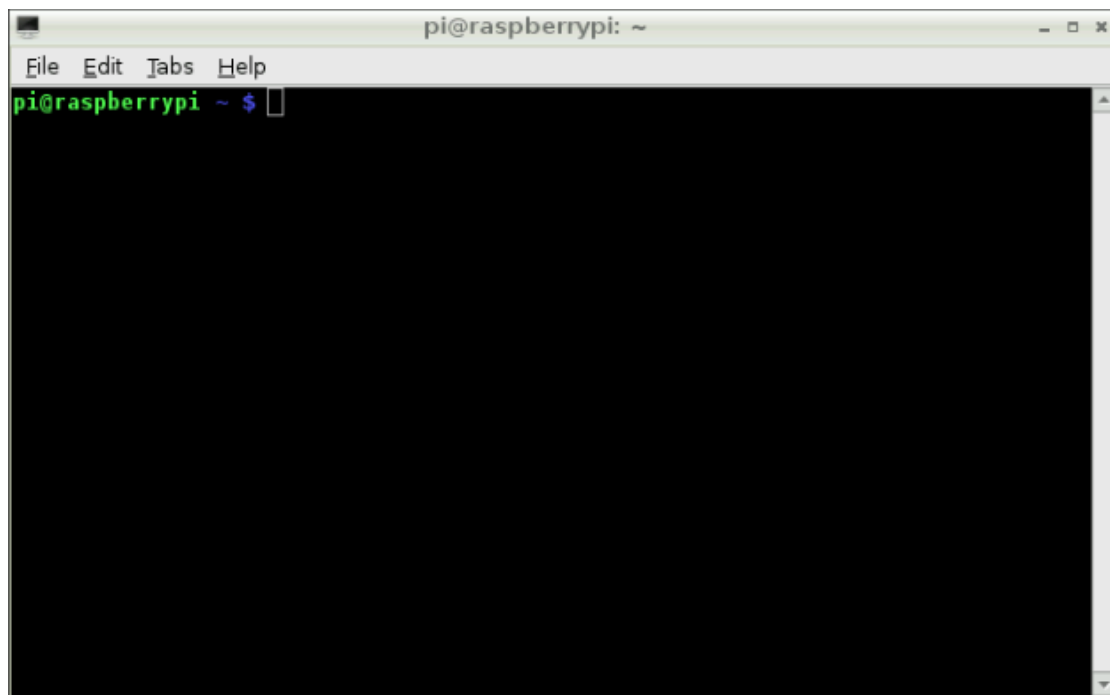
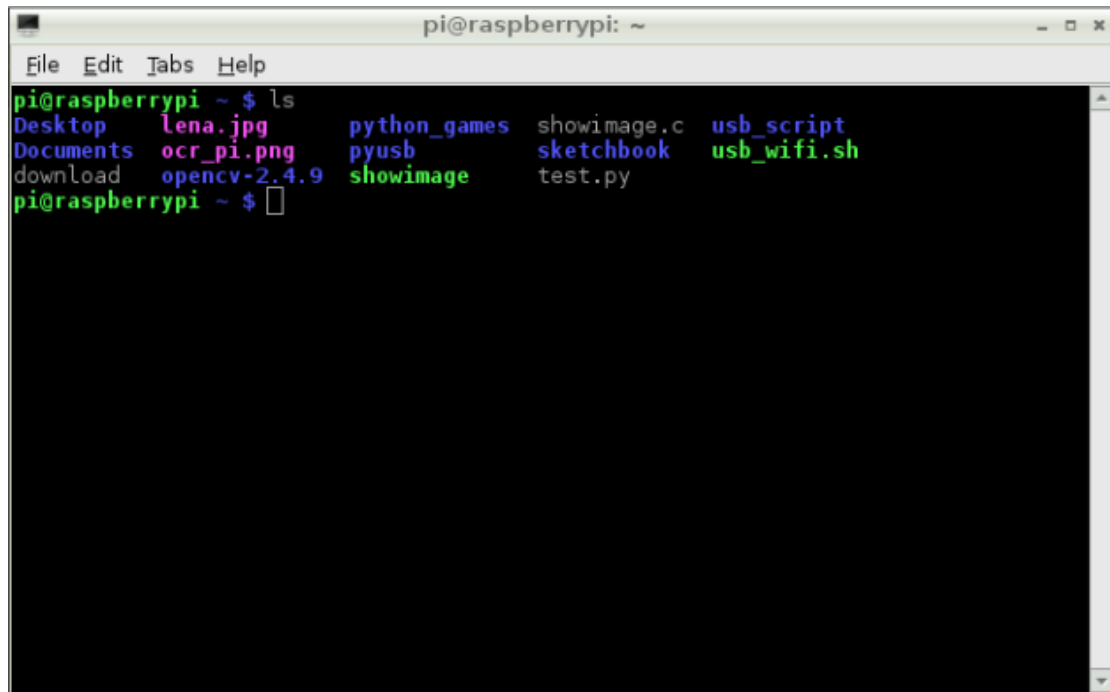


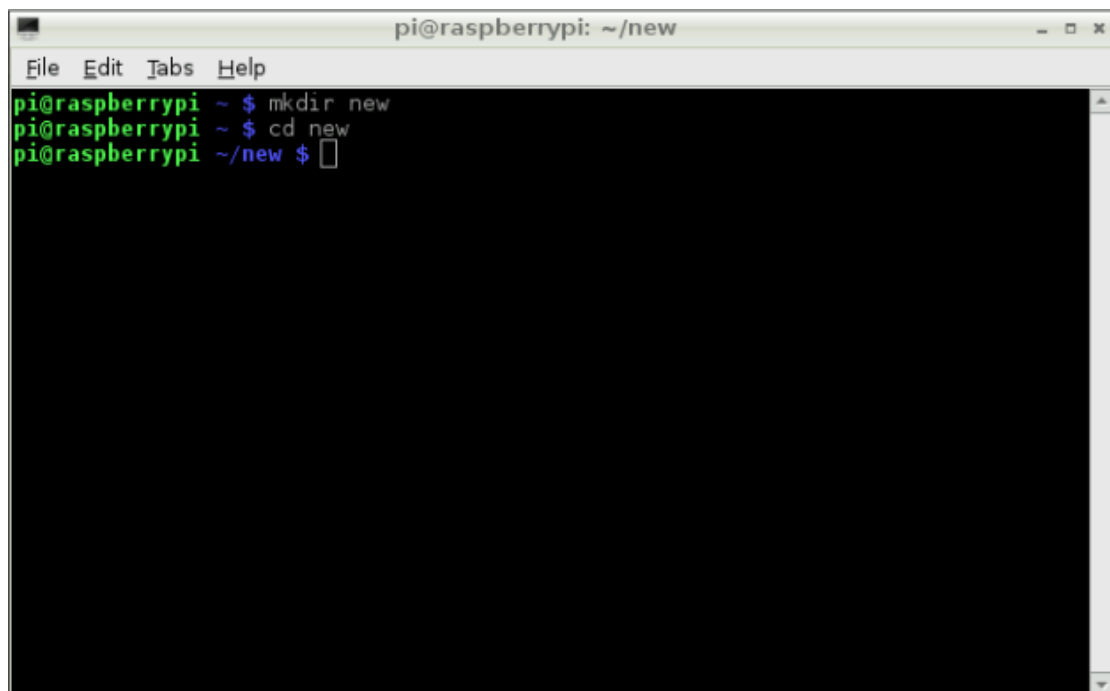
圖3.21 Raspberry Pi 終端機介面

輸入 ls 可顯示檔案的相關資訊，如圖 3.22。若要建立一個名為 new 的資料夾，則可輸入 mkdir new，再輸入 cd new 就能進入此資料夾的路徑，如圖 3.23。



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi ~ $ ls  
Desktop      lena.jpg      python_games  showimage.c   usb_script  
Documents    ocr_pi.png    pyusb         sketchbook    usb_wifi.sh  
download     opencv-2.4.9  showimage     test.py  
pi@raspberrypi ~ $
```

圖3.22 Raspberry Pi 檔案資訊



```
pi@raspberrypi: ~/new  
File Edit Tabs Help  
pi@raspberrypi ~ $ mkdir new  
pi@raspberrypi ~ $ cd new  
pi@raspberrypi ~/new $
```

圖3.23 Raspberry Pi 新資料夾的建立

若要在 new 資料夾中建立一個名為 pi 的檔案，則可輸入 nano pi，可看到以下如圖 3.24 的畫面，可在此介面中編譯程式。

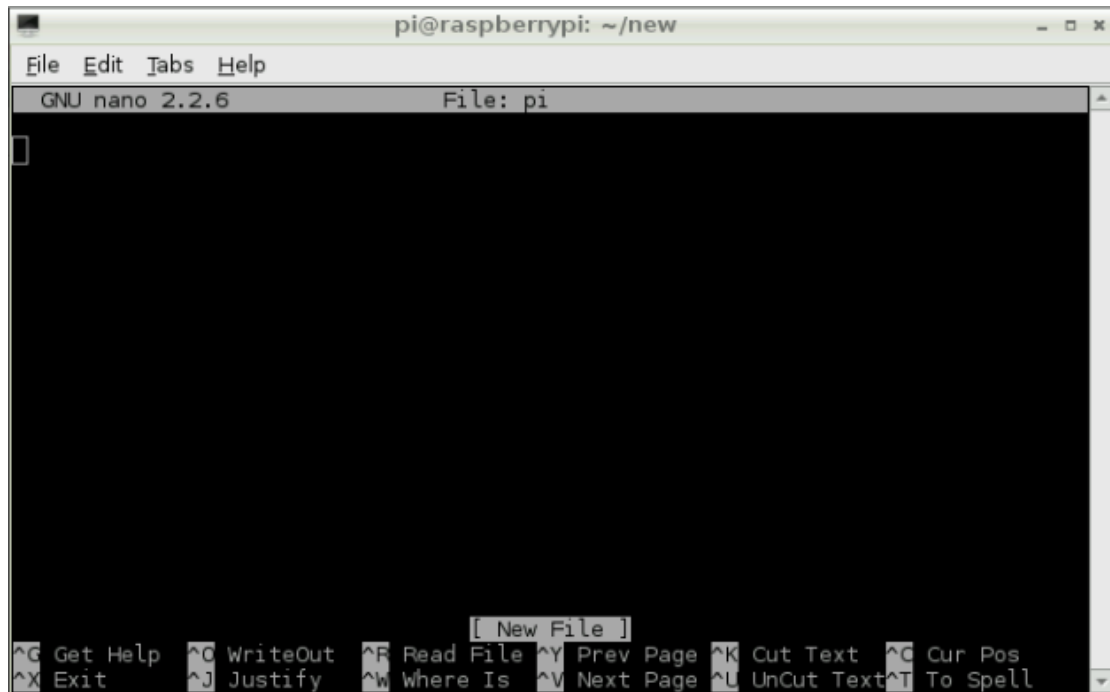


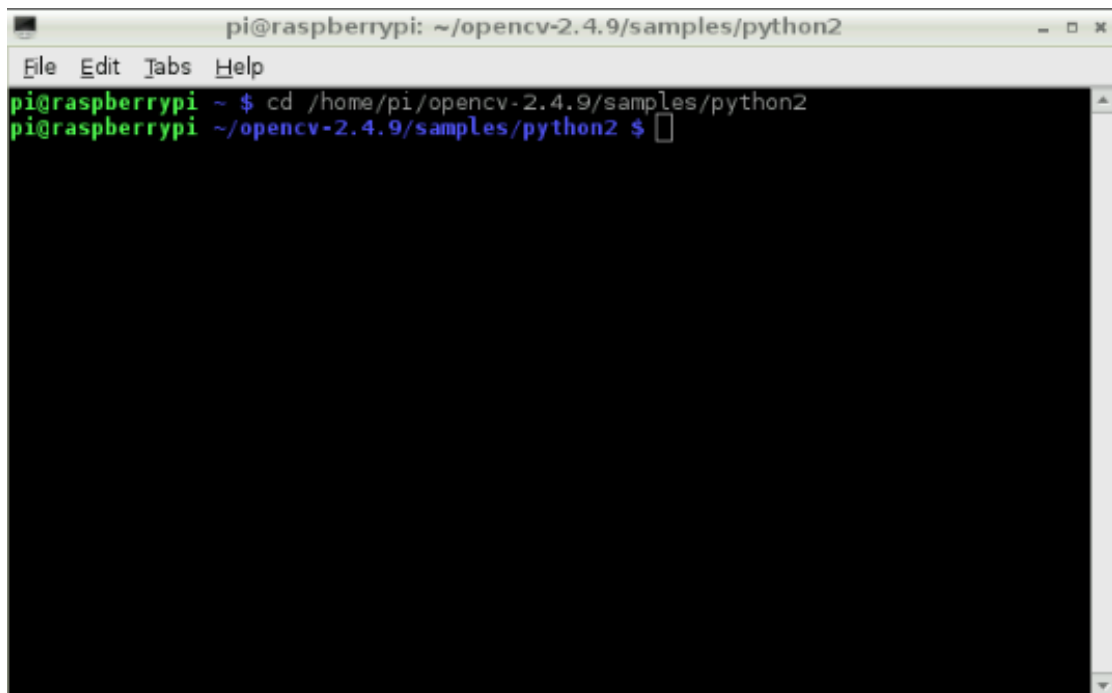
圖3.24 Raspberry Pi 程式編譯介面

圖 3.25 為編譯介面中，使用 OpenCV 函式庫，人臉特徵點辨識的.xml 檔之使用介面。

```
args, video_src = getopt.getopt(sys.argv[1:], '', ['cascade=', 'nested-cascade='])
try: video_src = video_src[0]
except: video_src = 0
args = dict(args)
cascade_fn = args.get('--cascade', "../../data/haarcascades/haarcascade_frontalface_alt.xml")
nested_fn = args.get('--nested-cascade', "../../data/haarcascades/haarcascade_eye.xml")
```

圖3.25 使用 OpenCV 函式庫中的.xml 檔之介面

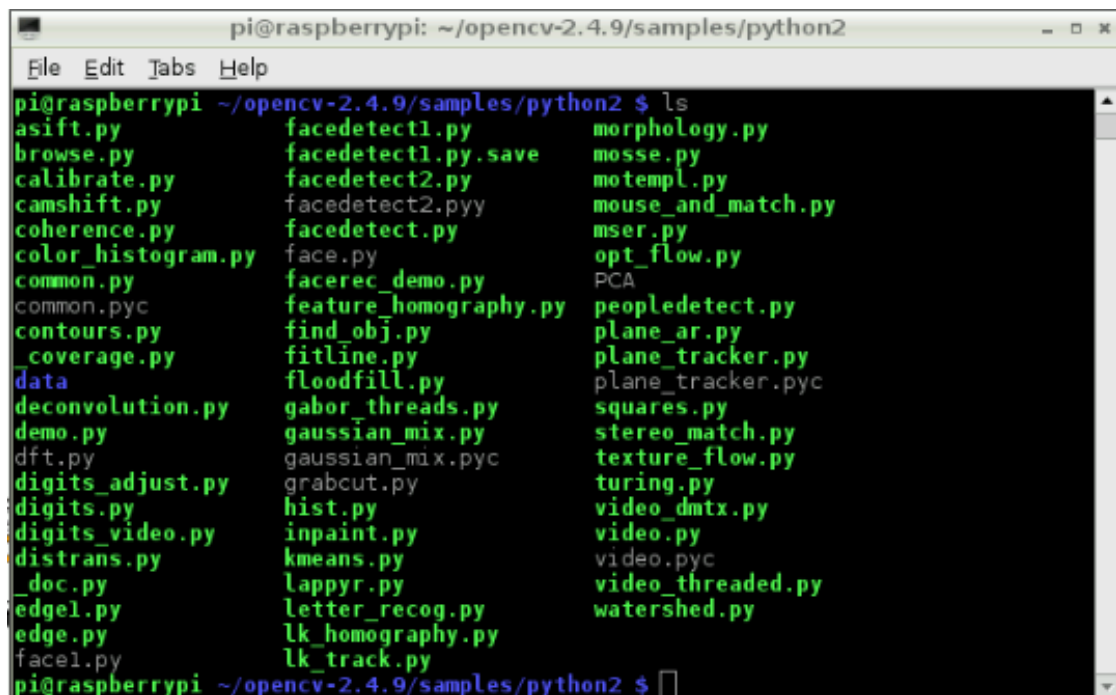
若要使用 OpenCV 中的執行檔，可先進入位於 Raspberry Pi 的 OpenCV 路徑，如圖 3.26。



```
pi@raspberrypi: ~/opencv-2.4.9/samples/python2
File Edit Tabs Help
pi@raspberrypi ~ $ cd /home/pi/opencv-2.4.9/samples/python2
pi@raspberrypi ~/opencv-2.4.9/samples/python2 $
```

圖3.26 進入 OpenCV 路徑之介面

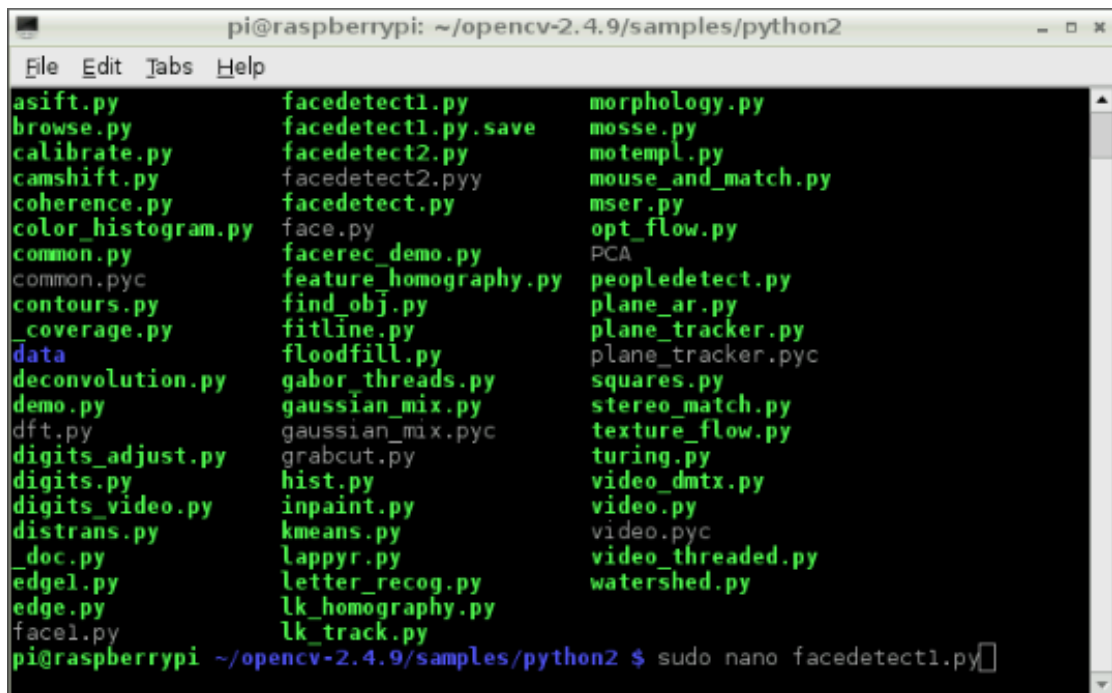
輸入 ls 就可顯示所有執行檔的相關資訊，圖 3.27 為 OpenCV 中內建的.py 執行檔。



```
pi@raspberrypi: ~/opencv-2.4.9/samples/python2
File Edit Tabs Help
pi@raspberrypi ~/opencv-2.4.9/samples/python2 $ ls
asift.py          facedetect1.py    morphology.py
browse.py         facedetect1.py.save mosse.py
calibrate.py      facedetect2.py    motempl.py
camshift.py       facedetect2.pyy   mouse_and_match.py
coherence.py      facedetect.py     mser.py
color_histogram.py face.py           opt_flow.py
common.py         facerec_demo.py  PCA
common.pyc        feature_homography.py peopledetect.py
contours.py       find_obj.py       plane_ar.py
_coverage.py      fitline.py        plane_tracker.py
data              floodfill.py      plane_tracker.pyc
deconvolution.py  gabor_threads.py  squares.py
demo.py           gaussian_mix.py   stereo_match.py
dft.py            gaussian_mix.pyc  texture_flow.py
digits_adjust.py  grabcut.py        turing.py
digits.py         hist.py           video_dmtx.py
digits_video.py   inpaint.py        video.py
distrans.py       kmeans.py         video.pyc
_doc.py           lappyr.py         video_threaded.py
edge1.py          letter_recog.py   watershed.py
edge.py           lk_homography.py
facel.py          lk_track.py
pi@raspberrypi ~/opencv-2.4.9/samples/python2 $
```

圖3.27 OpenCV 內建的.py 執行檔

再輸入 `sudo nano 執行檔名稱.py`，就可將此執行檔開啟，如圖 3.28。



```
pi@raspberrypi: ~/opencv-2.4.9/samples/python2
File Edit Tabs Help
asift.py          facedetect1.py    morphology.py
browse.py         facedetect1.py.save mosse.py
calibrate.py      facedetect2.py    motempl.py
camshift.py       facedetect2.py.py mouse_and_match.py
coherence.py      facedetect.py     mser.py
color_histogram.py face.py           opt_flow.py
common.py         facerec_demo.py  PCA
common.pyc        feature_homography.py peopledetect.py
contours.py       find_obj.py      plane_ar.py
_coverage.py      fitline.py       plane_tracker.py
data              floodfill.py     plane_tracker.pyc
deconvolution.py  gabor_threads.py squares.py
demo.py           gaussian_mix.py  stereo_match.py
dft.py            gaussian_mix.pyc texture_flow.py
digits_adjust.py  grabcut.py       turing.py
digits.py         hist.py          video_dmtx.py
digits_video.py   inpaint.py       video.py
distrans.py       kmeans.py        video.pyc
_doc.py           lappyr.py        video_threaded.py
edgel.py          letter_recog.py  watershed.py
edge.py           lk_homography.py
facel.py          lk_track.py
pi@raspberrypi ~/opencv-2.4.9/samples/python2 $ sudo nano facedetect1.py
```

圖3.28 開啟執行檔之輸入指令

3.2 網路攝影機 webcam 介紹

網路攝影機（Webcam）一般具有視訊攝影、傳播和靜態圖像捕捉等基本功能，它是藉由鏡頭採集圖像後，由網路攝影機內的感光元件電路及控制元件對圖像進行處理並轉換成電腦所能識別的數位訊號，然後藉由並列埠、USB 連接，輸入到電腦後由軟體再進行圖像還原。有些則支援乙太網路或 WiFi，內建有處理器及網頁伺服器，接上網路後可連線檢視畫面。

目前網路攝影機的使用相當普遍，大多是透過 USB 介面連接到電腦上，因 USB 的方便性，所以整合方面也相當方便，相對於其他需使用影像擷取卡的影像裝置而言，降低了許多實驗的成本。市面上網路攝影機分為兩種，一種為直接連接電腦可用於視訊通話的消費型網路攝影機（Webcam），另一種為保全監控專用的網路監控攝影機（IP Camera/Network Camera）。

本論文使用之網路攝影機為 Logitech Webcam C310，其功能為拍攝及時之影像並透過 USB 介面連接至 Raspberry Pi。而在攝影機的感光元件上，通常都會使用 COMS (Complementary Metal Oxide Semiconductor) 晶片作為感光元件，原因是因為與另一個感光元件 CCD (Charge-coupled Device) 相比，CMOS 價格便宜、耗電量低，雖然影像品質沒有 CCD 來的好，不過也在可以接受的範圍內，相較之下會具有絕對的優勢。表 3.2 為 CMOS 與 CCD 的優缺點比較。圖 3.29 為網路攝影機 webcam C310 的實體圖。

表3.2 CMOS 與 CCD 之比較

	CCD	CMOS
影像品質	優	範圍可接受
感光度	優	範圍可接受
耗能比	高	低
雜訊比	低	高
晶片整合	難	易
生產成本	高	低



圖3.29 Logitech Webcam C310 的外觀

第四章 實驗結果與分析

本論文利用 2.6 節所提出的基於膚色偵測和特徵點辨識擷取方法，進一步辨識出位於臉部的各個特徵點，接著運用已建立的資料庫與 2.7 節所介紹的特徵臉進行人臉識別，將這些結果進行整合分析，成為一套辨識率良好的人臉辨識系統。

4.1 人臉特徵點偵測結果

本論文使用的方法為先偵測出影像中所有的膚色區塊，一般臉部膚色區塊會比其他部位膚色區塊面積較大，用此特徵搜尋出影像中人臉可能的位置，並取出人臉的區塊影像，當取得人臉之後，再做更近一步的特徵點分析。圖 4.1 為膚色偵測所使用的色塊圖。



圖4.1 膚色的色塊圖示

4.1.1 膚色偵測結果

以下將測試人臉相對於鏡頭不同距離的人臉膚色偵測，利用 OpenCV 函式庫中的 Haar 矩形特徵，分類出人臉的膚色，測試人臉相對於鏡頭，一般與最遠距離的膚色偵測。圖 4.2 為一般距離的偵測，圖 4.3 則為距鏡頭最遠能辨識的範圍。

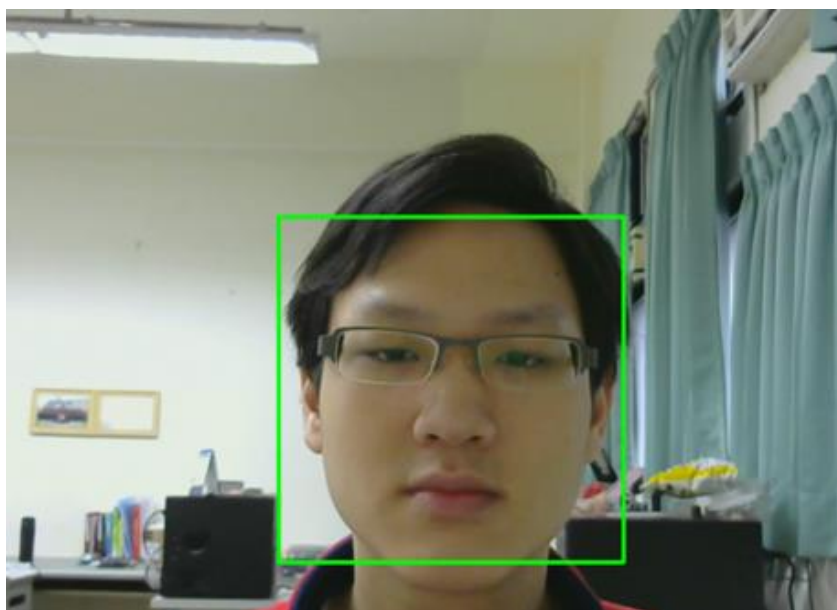


圖4.2 距離鏡頭 0.3 公尺的人臉膚色偵測



圖4.3 距離鏡頭 3.3 公尺的人臉膚色偵測

其最遠的人臉膚色偵測距離為距鏡頭約 3.3 公尺處。接著找出臉上的局部特徵，順序是眼睛、嘴巴和鼻子，根據這些局部特徵判別是否為正確的人臉。

4.1.2 眼睛特徵點偵測結果

以下為眼睛特徵點偵測的結果，分為只偵測一隻眼睛與同時辨識雙眼，圖 4.4 為偵測單眼之左眼的辨識，圖 4.5 為同時偵測雙眼之左眼的辨識。

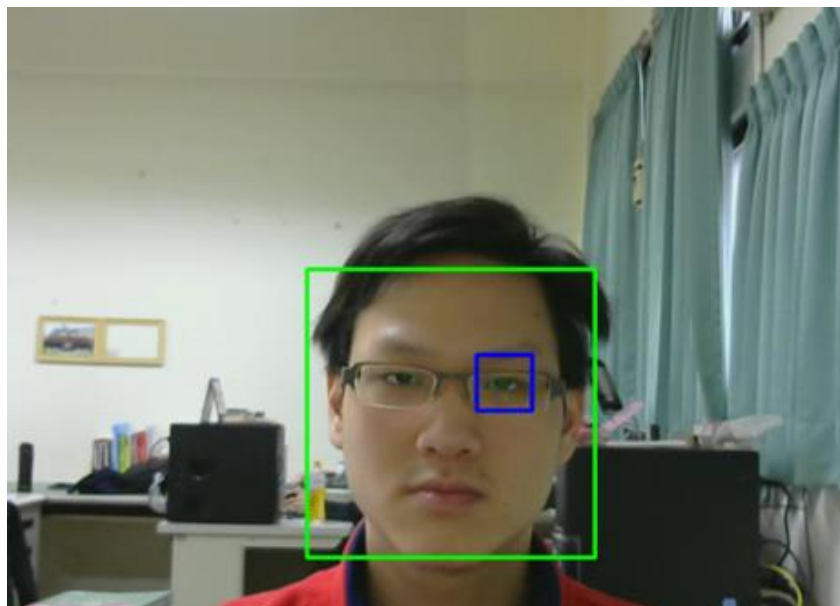


圖4.4 單眼偵測之左眼的辨識

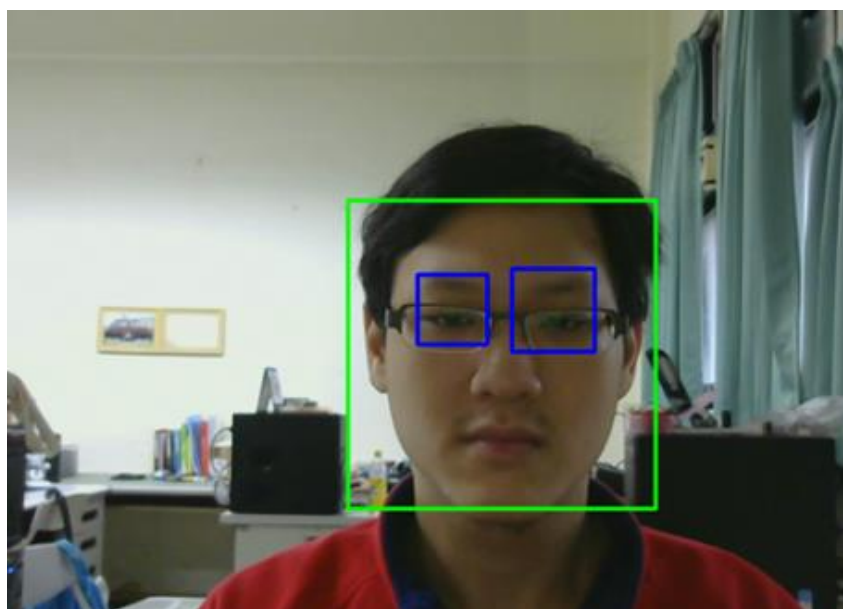


圖4.5 雙眼偵測之左眼的辨識

4.1.3 嘴唇特徵點偵測結果

圖 4.6 為嘴唇特徵點偵測的結果，圖 4.7 為同時偵測雙眼及嘴唇特徵點的偵測結果，此時則可判定為一個完整的人臉。

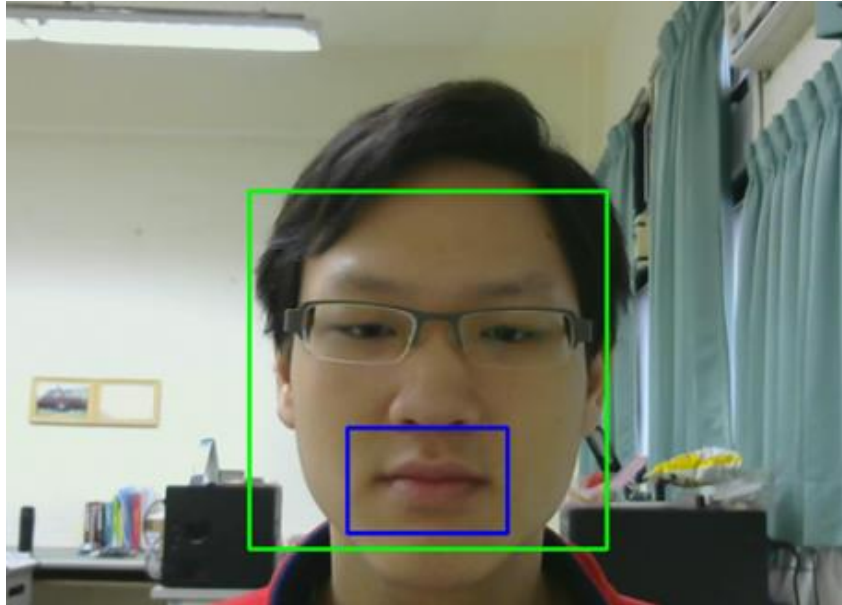


圖4.6 嘴唇偵測的辨識

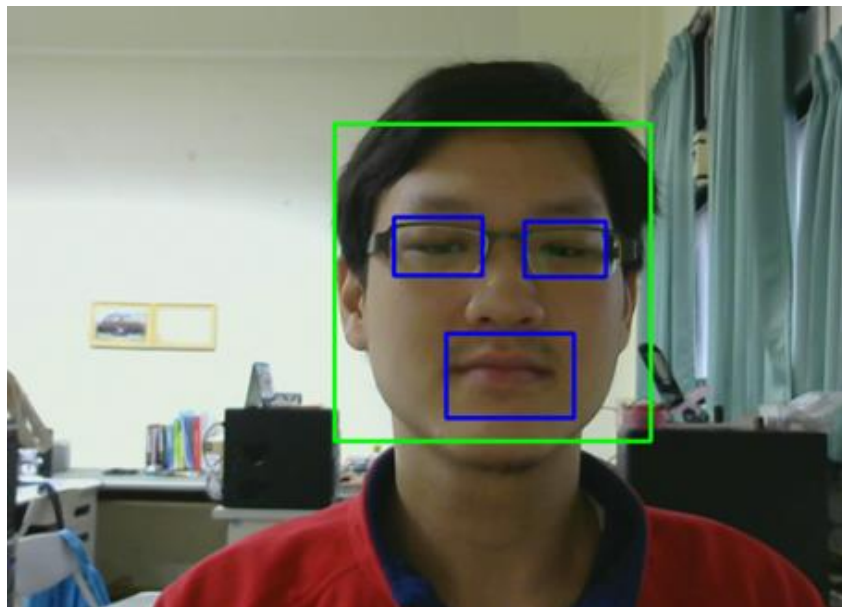


圖4.7 雙眼及嘴唇偵測的辨識

4.1.4 鼻子特徵點偵測結果

若要做更精確的人臉偵測，則可加入鼻子特徵點的偵測，圖 4.8 為其辨識結果。

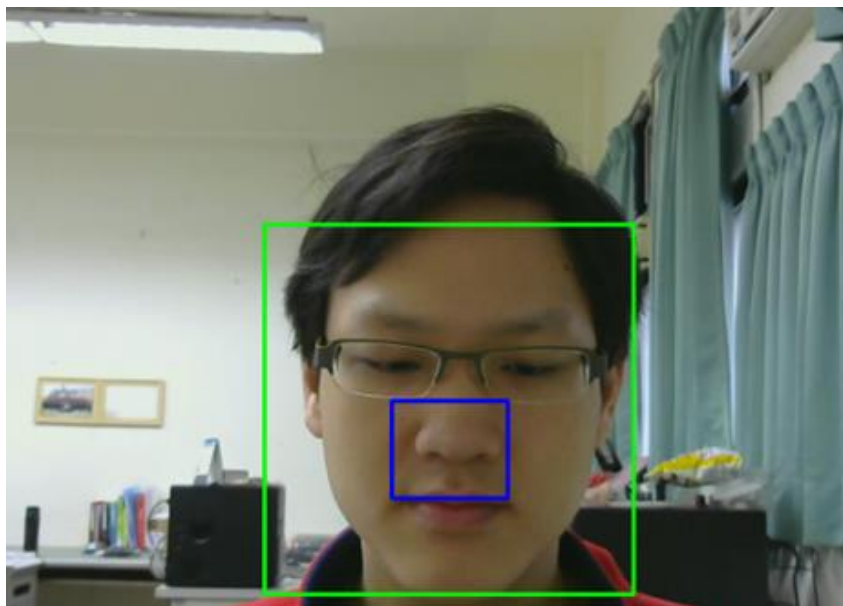


圖4.8 鼻子偵測的辨識

4.1.5 眼鏡特徵點偵測結果

圖 4.9 為人臉配戴眼鏡，以眼鏡做為特徵點的結果，其位置約為眼睛上方。

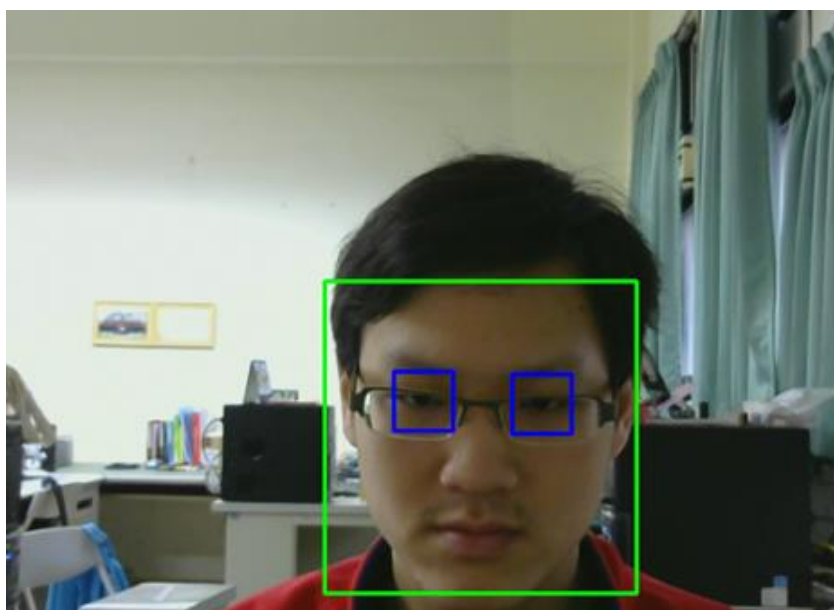


圖4.9 眼鏡偵測的辨識

4.2 臉部出現干擾之偵測結果

假設部分的臉部被遮住，是否還能辨識出一個完整的人臉，以下為人臉配戴口罩的偵測結果，圖 4.10 是口罩遮住下巴部分，仍然能辨識為一個完整的人臉。

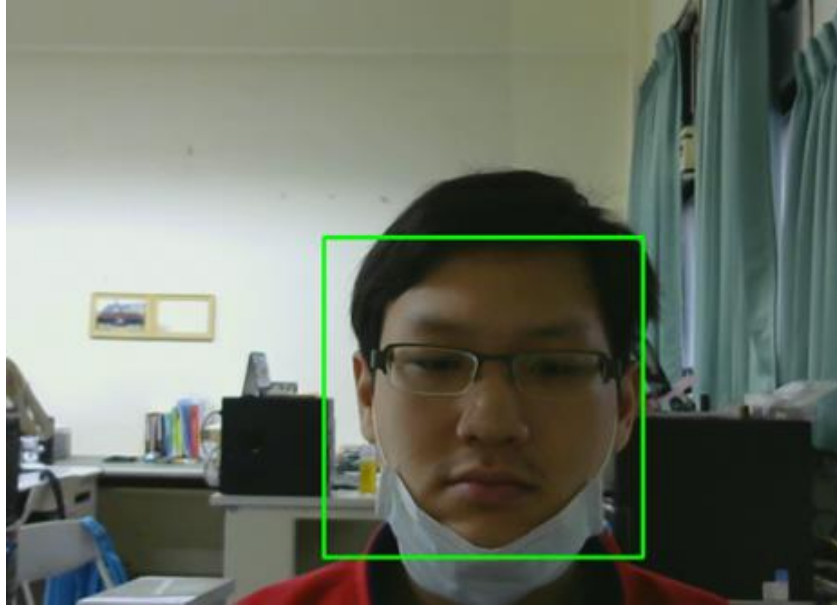


圖4.10 口罩遮住部分臉部之偵測

圖 4.11 為口罩遮住一半的臉部，此時無法偵測出完整的人臉，可得知影像中的特徵點偵測，必須要有眼睛、鼻子與嘴巴，才能辨識出一個完整的人臉影像。



圖4.11 口罩遮住一半臉部之偵測

4.3 人臉識別與檢測

本論文為達成即時人臉識別的功能，因而建立一套人臉資料庫，在資料庫中收集多位學生的人臉圖像，再結合以上所介紹的特徵點擷取與特徵臉分析，加以整合後，可藉此判斷出在辨識的人臉影像中，是否為想要辨識的人臉，成為一套辨識率良好的人臉識別系統。

4.3.1 人臉識別與檢測介紹

人臉識別為一種對於已知人臉，進行分類的過程，人臉識別通常包含四個主要步驟，下圖 4.12 為人臉識別流程圖。

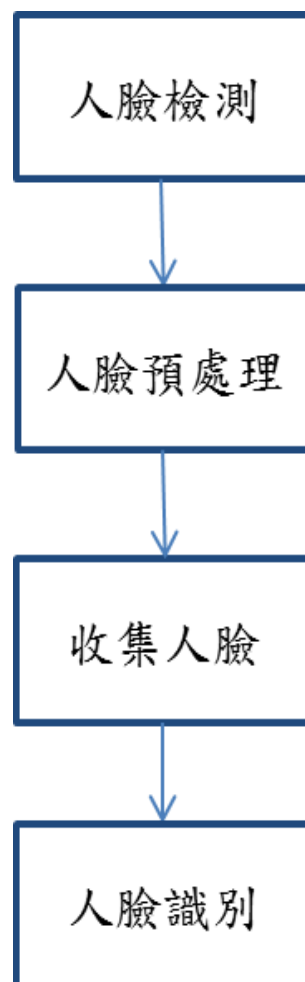


圖4.12 人臉識別流程圖

- 人臉檢測

先在偵測到的影像中，搜尋出人臉的影像，此時先不辨識是誰，而是只找出人臉，此步驟使用到 OpenCV 的 CXCORE 部分中之.xml 函數。

- **人臉預處理**

此步驟為調整人臉圖像，目的是為了讓圖像更為清晰，此步驟使用到 OpenCV 的 CV 部分中之.css 函數。

- **收集人臉**

此步驟將收集到的人臉圖像，放入建立人臉資料庫中，之後要讓系統去辨識與判別，此步驟使用到 OpenCV 的 MLL 部分中之.js 函數。

- **人臉識別**

此步驟會在建立的人臉資料庫中，找出與 Webcam 偵測的人臉最為相似的，如此就能判斷出 Webcam 偵測到的人是否為想找的人，此步驟使用到 OpenCV 的 HighGUI 部分中之.htm 函數。

4.3.2 常用的人臉資料庫介紹

無論是人臉偵測、臉部特徵輪廓、或是性別辨識及年齡辨識等，各種人臉相關的演算法，都需要建立擁有大量數據的人臉資料庫去加以辨識。例如表情辨識的部分，有些人會使用規則判斷來決定，而有些則也會使用資料庫是先分類訓練。人臉識別也是必須使用大量人臉資料庫去尋找較強健的特徵當作比對依據，楊煒達[9]整理出一些常用的人臉資料庫。

- **FERET 人臉資料庫**

此資料庫由美國國防部的 Counterdrug Technology Transfer Program (CTTP) 所發起的人臉識別技術 (Face Recognition Technology 簡稱 FERET)工程，它包括了一個通用人臉庫以及通用測試標準。同一人的包括不同表情，光照，姿態和年齡的照片，到 1997 年，它已經包括了 1000 多人的 10000 多張照片，並不斷得到擴充，FERET 定期對不同識別演算法進行測試。是人臉識別領域應用最廣泛的人臉資料庫之一.其中的多數人是西方人,每個人所包含的人臉影象的變化比較單一。

- **CMU-PIE 人臉資料庫**

此資料庫由卡耐基梅隆大學所收集的，所謂 PIE 就是姿態 (POSE)、光照 (ILLUMINATION) 和表情 (EXPRESSION) 的縮寫，CMU PIE 人臉庫建立於 2000 年 11 月，它包括來自 68 個人的 40000 張照片，其中包括了每個人的 13 種姿態

條件，43 種光照條件和 4 種表情下的照片，現有的多姿態人臉識別的文獻基本上都是在 CMU PIE 人臉庫上測試的。

- **YALE 人臉資料庫**

此資料庫由耶魯大學計算視覺與控制中心建立，資料庫中有十五個不同的人，每人十一張照片，共有一百六十五張影像。每個人的十一張影像各有不同的情緒與光照條件，分別有中央打光、戴眼鏡與不戴眼鏡、快樂表情、左邊打光、正常表情、右邊打光、難過表情、想睡覺的表情、驚訝表情與眨眼表情。每張影像為 320×243 像素，含大量背景的 GIF 檔。

- **ORL 人臉資料庫**

此資料庫由劍橋大學 AT&T 實驗室建立，包含 40 人共 400 張面部影像，每個人有十張影像，都是在不同時間狀況下拍的。十張影像中，又各種不同的表情（睜眼、閉眼、微笑或扁嘴）和光照條件，以及一些外在的因素（戴眼鏡與否）等，都不盡相同。而拍攝的背景固定為黑色，且皆為正面的人臉（允許一點小角度的偏差）。所有原始圖檔皆以 256 色灰階 PGM 檔的形式儲存，其大小為 92×112 像素。

- **Cohn Kanade 人臉資料庫**

此資料庫由 Takeo Kanade, Ph.D. 與 Jeffrey F. Cohn, Ph.D. 的實驗室所收集的。其中共有兩百一十個不同的人，年齡集中在十八到五十歲之間；裡面包含百分之六十九的女生（一百四十五），和百分之三十一的男生（六十五）；其中白種人佔百分之八十一、亞洲人佔百分之十三，以及其他種人佔百分之六。原始影像大小皆為 640×490。

4.3.3 人臉資料庫收集與建立

本論文所建立的人臉資料庫，為自行拍攝與收集的學生人臉圖像，其資料庫中包含六位學生的訓練人臉圖像，每個人包含九種角度與六種表情，總共九十張人臉圖像。將利用 Webcam 所拍攝到的人臉圖像通過預處理之後，建立於資料庫中，而由於人臉檢測會受到光線的影響，所以只適用於灰階圖像，因此需將其轉換成灰階影像，下圖 4.13 為本實驗成員的灰階人臉圖像。



圖4.13 灰階人臉圖像

為了增加辨識率，則必須考慮實際多種情形，例如人臉不一定是正面，有可能為偏左、偏右、偏上或偏下，另外表情也會有多種變化。一個良好的資料庫應包含大量的人臉圖像與人臉變化的各種情形，但假設每張圖像的外觀幾乎一樣，其辨識率依然會很差，所以需在資料庫中放入各種不同角度的拍攝，與不同人臉的表情以提高辨識率，使系統能判斷此人是否為想找的人。圖 4.14 與圖 4.15 為不同角度與不同表情所拍攝的人臉圖像，表 4.1 為自製人臉資料庫的規格。

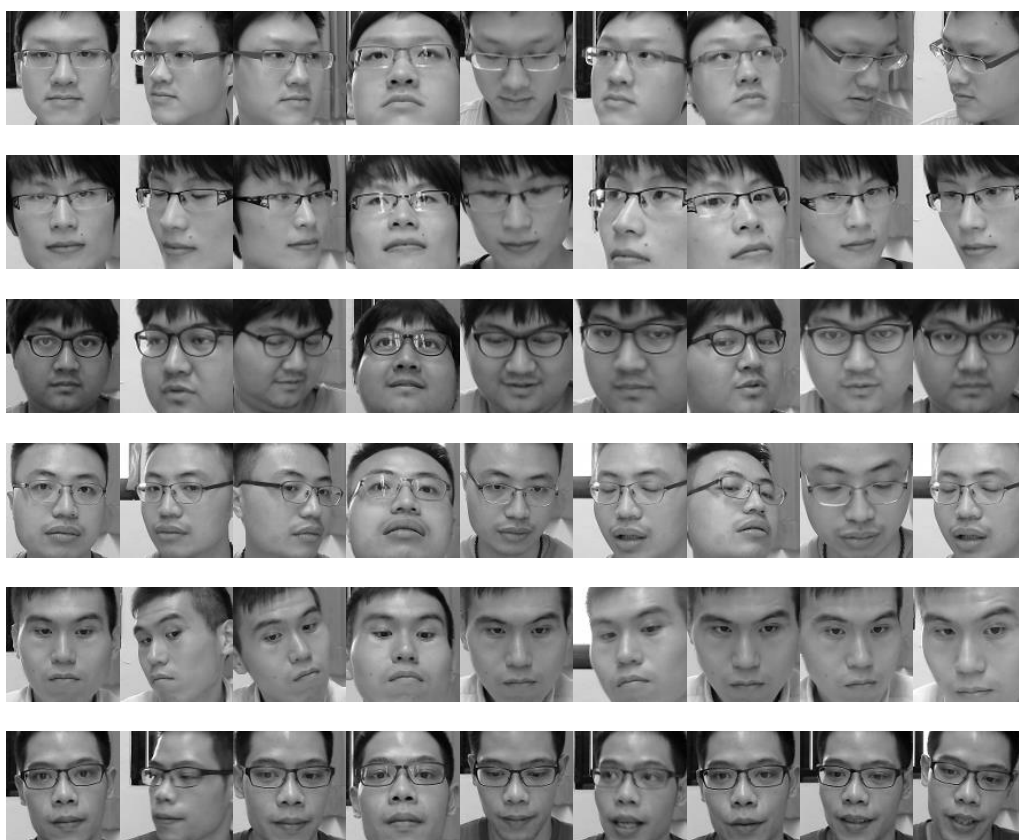


圖4.14 不同角度的人臉圖像



圖4.15 不同表情的人臉圖像







表4.1 自製人臉資料庫規格

辨識裝置	Webcam
辨識環境	室內日光燈
實驗成員人數	6 人
不同角度樣本	9 張/人，共 54 張
不同表情樣本	6 張/人，共 36 張
資料庫總樣本	15 張/人，共 90 張

4.3.4 人臉識別結果

本實驗使用自製人臉資料庫，收錄六位實驗成員，每個人收集九種不同角度與六種不同表情，共九十張人臉影像，表 4.2 為每位實驗成員的實驗代碼，為了顯示辨識結果所使用。

表4.2 實驗成員與其代碼

實驗成員代碼	實驗成員
no.1	
no.2	
no.3	
no.4	
no.5	
no.6	

為了達到本論文的目的，即時人臉辨識的功能，因此需要利用一套人機介面系統來顯示其辨識結果，圖 4.16 為使用 C#結合 OpenCV 函式庫，所實現的人臉識別系統介面，以下將介紹此人機介面的操作步驟。



圖4.16 人臉識別之人機介面

● 此人機介面的操作步驟:

1. 首先開啟右下方「檢測與識別」，使 Webcam 開始偵測人臉影像。
2. 偵測到的影像，會顯示於介面上方的視窗，若偵測到人臉則會框出其位置。
3. 接著輸入要建立於資料庫中，此位實驗成員的名稱代碼，位於此介面左下方。
4. 開啟「加入人臉」，視窗中會出現此實驗成員的灰階影像，能將其名稱代碼與人臉圖像存入資料庫中。
5. 當 Webcam 再次偵測到此位成員時，就會顯示其名稱代碼於介面上，達成人臉識別的功能。

以下圖 4.17～圖 4.22 為參與本實驗的六位實驗成員，其辨識的結果，每位成員皆能顯示出，代表各自的實驗代碼。

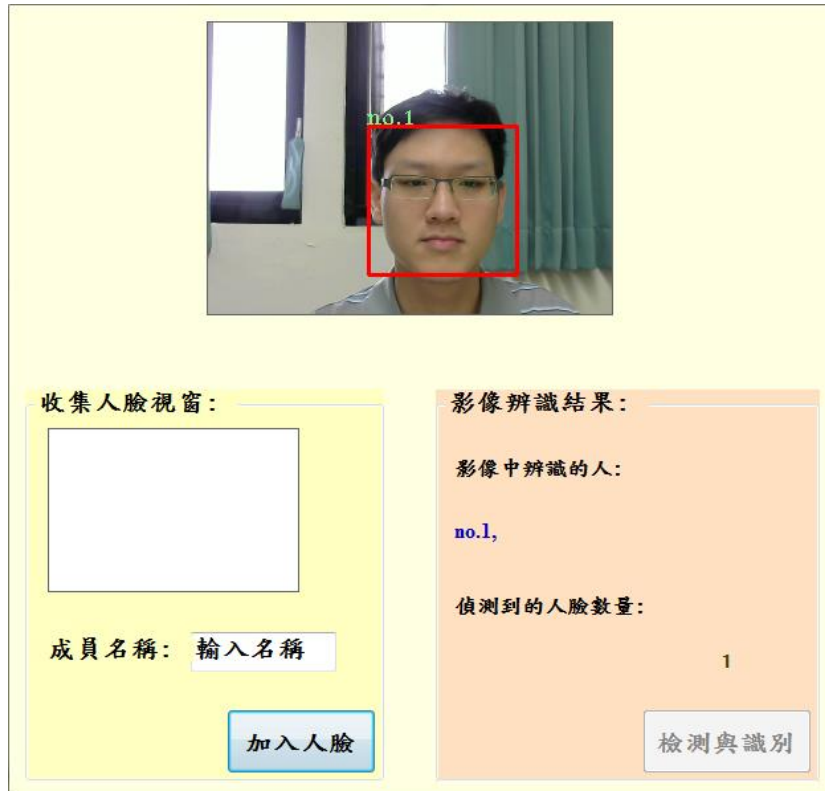


圖4.17 實驗成員 no.1 之辨識結果

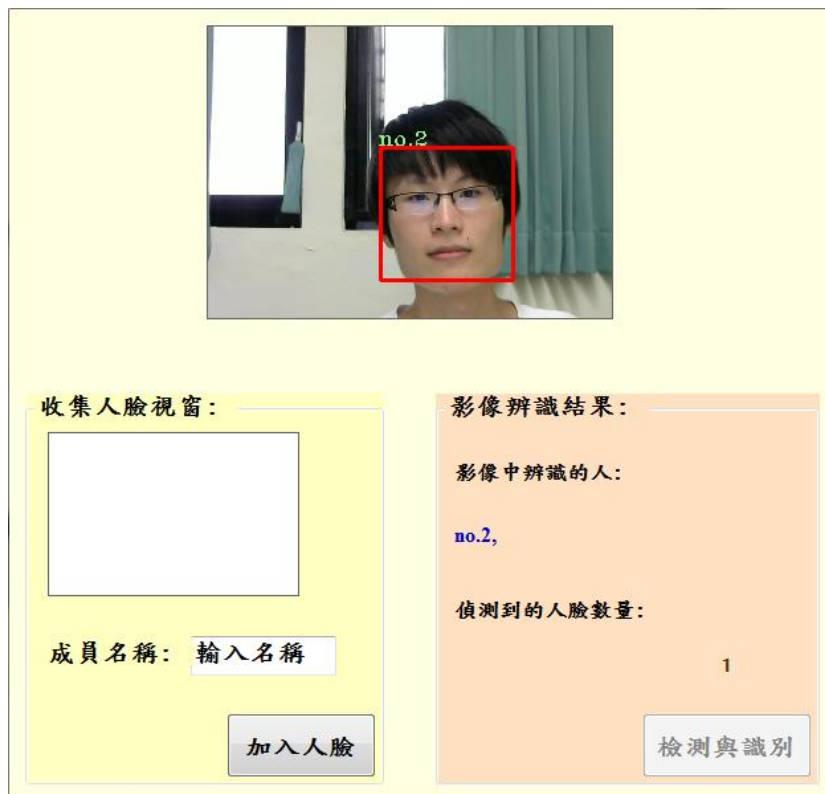


圖4.18 實驗成員 no.2 之辨識結果

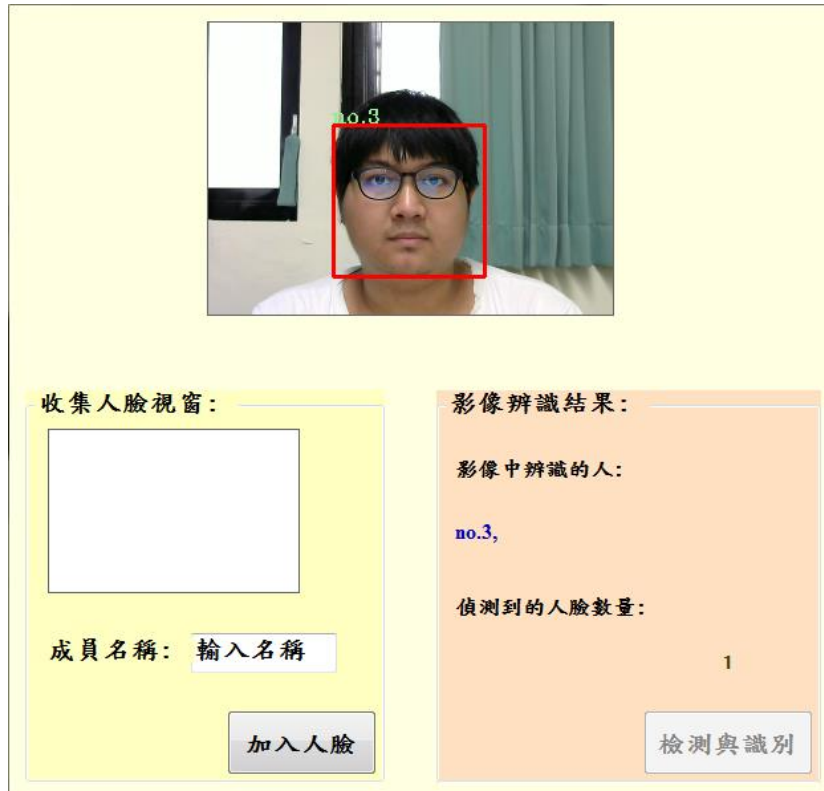


圖4.19 實驗成員 no.3 之辨識結果

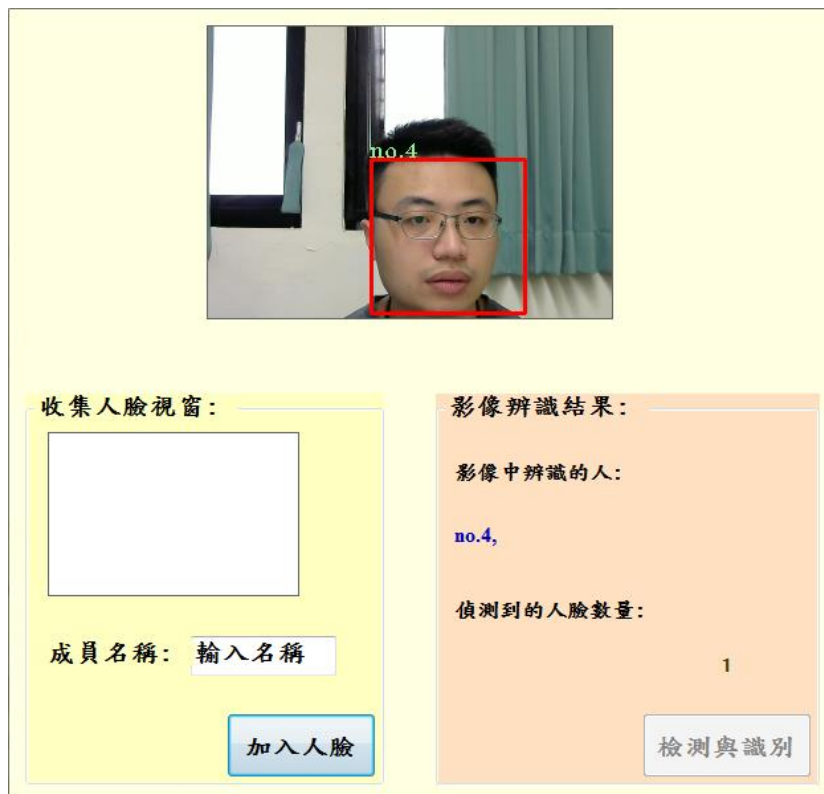


圖4.20 實驗成員 no.4 之辨識結果



圖4.21 實驗成員 no.5 之辨識結果



圖4.22 實驗成員 no.6 之辨識結果

本實驗將每位實驗成員，分別進行各一百次的辨識，將結果記錄下來並計算正確的次數，可得到其辨識率如表 4.3。

表4.3 本實驗之辨識率

實驗成員代碼	正確次數	錯誤次數	辨識率
no.1	94	6	94%
no.2	99	1	99%
no.3	98	2	98%
no.4	96	4	96%
no.5	95	5	95%
no.6	95	5	95%

由實驗結果可得知，本實驗的辨識率皆可達到 90% 以上，而當中出現辨識錯誤的部分，可能因為辨識距離的不同、些微角度的誤差或是出現資料庫中未有的表情，以上皆會對辨識結果造成影響，若要再提高辨識率，則可增加資料庫中的人臉樣本，收集越多實驗成員的表情，與不同角度的拍攝，其辨識率就能更加提高。另外由於辨識的距離，也會對實驗造成影響，因此每位實驗成員，都應盡量在與 Webcam 保持一致的距離進行實驗，如此也能使辨識率更為精準。

第五章 結論與未來展望

5.1 結論

本論文運用網路攝影機 webcam 並結合 Raspberry Pi 單板機電腦，建構一個及時人臉偵測與辨識系統。先從偵測到的影像中搜尋膚色，並使用 Viola 人臉偵測方法判斷是否為人臉，將搜尋到的人臉做特徵點辨識，並加以擷取和分析。

並配合一套自製的人臉資料庫，在資料庫中收集多位學生的人臉圖像，其中包含各種不同角度與不同表情的人臉圖像，運用主成分分析（PCA）找出每個人臉不同的特徵臉，識別出不同的人臉。

再經由取樣多次的識別實驗，紀錄其判別結果的正確率與錯誤率，可計算出此系統的辨識率，得知辨視距離與部分表情，會影響系統的識別結果。因此若要提升識別精確度，可增加資料庫中的人臉樣本，以及實驗成員的位置，皆要和 Webcam 保持一致的距離。

5.2 未來展望

人臉辨識系統在近幾年來，應用於偵測、辨識、追蹤等相關的領域中的研究項目越來越豐富，人與機器的接觸也越來越緊密，對於使用者的感受逐漸受到重視。

以電腦視覺幫助人類雙眼來從事各種活動也越來越廣泛。未來人臉辨識系統將不再單純辨識人臉，而是與人類的日常生活息息相關，其未來的應用包含如下：

1.娛樂

可在體感遊戲中，加入玩家表情辨識結合體感動作，增加與玩家互動性與娛樂性。

2.醫療

可結合家庭看護，藉由偵測病人的臉部變化回傳至醫院，藉此判斷病人的身體狀況與生理反應。

3.商業

可運用一般公司的員工上下班打卡系統，只要辨識人臉就能記錄此員工的上下班時間，節省更多時間。

4.警方辦案

結合人車盤查登錄查詢系統，巡邏盤查遇有可疑人、車時，這時只要將相關人的年齡、性別及地址等資料登錄，並利用嫌疑人的臉型五官來作為身分識別，供預防犯罪及偵辦刑案之查詢與運用參考，對維護治安有極大的功能與重要性。

參考文獻

- [1] Paul Viola, Michael Jones, "Robust Real-time Object Detection", Second International Workshop on Statistical and Computational Theories of Vision - Modeling, Learning, Computing, and Sampling, 2001.
- [2] 梁振升, “以 OpenCV 實現即時之人臉偵測與辨識系統”, 銘傳大學電腦與通訊工程學系碩士學位論文, 2010。
- [3] 張哲維, “視覺迴授之即時物體追蹤系統”, 國立海洋大學機械與機電工程學系碩士學位論文, 2007。
- [4] 陳永霖, “互補式金屬氧化物半導體影像感測器(CMOS image sensor) 產業關鍵成功因素之探討—以系統單晶片(SoC)為例”, 國立臺灣大學商學研究所碩士論文, 2002。
- [5] 陳繼棠, “結合膚色區域分割與主要成份分析於多人臉部辨識”, 國立台灣海洋大學機械與機電工程學系碩士學位論文, 2006。
- [6] 楊勝文 “使用膚色和臉部特徵的人臉偵測”, 逢甲大學通訊工程學系碩士班碩士論文, 2012。
- [7] 吳展宇, “應用立體視覺迴授於機械臂路徑規劃”, 國立台灣海洋大學機械與機電工程學系碩士學位論文, 2010。
- [8] 張循鋁、林開榮、余誌強, “應用於學習者專注力分析的人臉偵測法之研究”, 教育科技與學習研究論文, 2015, pp.45-70。
- [9] 楊煒達 “簡易方法之少量人臉辨識系統”, 國立中央大學資訊工程研究所碩士論文, 2007。
- [10] 維基百科, 特徵臉, <http://zh.wikipedia.org/>。
- [11] 維基百科, 主成分分析, <http://zh.wikipedia.org/>。
- [12] Daniel Lelis Baggio, Shervin Emami, David Millan Escrive, Khvedchenia Ievgen, Naureen Mahmood, Jason Saragih, Roy Shilkrot, 深入理解 OpenCV 實用計算機視覺項目解析, 劉波 譯, 機械工業出版社, 2015。
- [13] 維基百科, Raspberry Pi, <http://zh.wikipedia.org/>。
- [14] 林建男, “以樹莓派單板電腦為基礎的人群分流之研究”, 國立屏東大學資訊工程學系碩士班碩士論文, 2015。
- [15] 柯博斌, “嵌入式感測盒的實作議題-Raspberry Pi 系統模組案例”, 國立台灣科技大學自動化及控制研究所碩士學位碩士論文, 2014。

- [16] 謝俊彥，“基於 Raspberry pi 之人臉偵測系統”，聖約翰大學資訊與通訊系碩士班碩士學位論文，2016。
- [17] 張益裕，Raspberry Pi 嵌入式應用程式開發:使用 Java，松崗出版股份有限公司，2015。
- [18] 河野悅昌，Raspberry Pi 實力應用集，許郁文 譯，碁峰資訊股份有限公司，2015。
- [19] 柯博文，Raspberry Pi 超炫專案與完全實戰，碁峰資訊股份有限公司，2015。
- [20] Samarth Shah，Raspberry Pi 專案製作:物聯網、機器人、圖像辨識，江良志 譯，碁峰資訊股份有限公司，2016。
- [21] Richard Grimmett，Raspberry Pi 機器人自造專案，曾吉弘 譯，碁峰資訊股份有限公司，2014。
- [22] 淺析人臉檢測之 Haar 分類器方法，<http://alex-phd.blogspot.tw/>。