單板 NVIDIA JETSON NANO 人臉辨識

國立勤益科技大學 電機工程系
指導老師：簡伯霖老師
班級：四電三丙
學生：高天儀、江嘉哲

摘要

本專題使用 NVIDIA JETSON NANO 單板以及 Ubuntu 的作業系統來進行人臉辨識，Ubuntu 是以桌面應用為主的 Linux 發行版，Ubuntu 是著名的 Linux 發行版之一，也是目前最多使用者的 Linux 版本。運用 Visual Studio Code（簡稱 VS Code）跨平台的免費原始碼編輯器，程式方面使用 C++和 OpenCV，使用已經訓練好的 caffemodel 檔以及執行檔 prototxt 來進行簡單的人臉辨識。其中 Caffe 全稱為 Convolutional Architecture for Fast Feature Embedding，是一個被廣泛使用的開源深度學習框架。

關鍵字：NVIDIA JETSON NANO、Visual Studio Code、人臉辨識

一、簡介

隨著時代的進步，人們的生活也越來越便利，現在人手一機的時代，手機的功能也越來越便利，而在大部分的手機都有一個功能叫做「人臉辨識」，而我們的專題則是使用 NVIDIA JETSON NANO 來執行。

本專題應用 NVIDIA JETSON NANO 搭配攝影機，將攝影機鏡頭內的畫面即時顯示，對畫面中的所有人像進行即時人臉辨識，並將所有的人臉偵測數據顯示出來，而利用能夠同時偵測多數目標的功能，就能夠在鏡頭的範圍內辨識出有多少人會在這一個空間，進一步的達到人流管制的功能，也應用到現今社會在公共場合的特殊需求。

二、理論基礎

本專題使用 NVIDIA JETSON NANO，輸入裝置有鍵盤、滑鼠還有攝影機，輸出裝置只有螢幕，作業系統是使用 Ubuntu，軟體方面我們用 Visual Studio Code 來寫，使用 C++語言配合 OpenCV 來進行即時人臉辨識，程式中我們有用已經訓練好的 Caffemodel 檔，在 Visual Studio Code 中 OpenCV 需要三個負責環境配置的檔案，主要重點是 OpenCV 的位置要設置好，如果沒設置好將無法執行有使用 OpenCV 的程式檔。

三、系統架構

本專題研究使用 NVIDIA JETSON NANO 當作主體，並使用 Ubuntu 作業系統、C++程式、VS code、Opencv、Caffe 模型來執行，使用攝影機即時顯示畫面，進行人臉辨識。

3.1 硬體

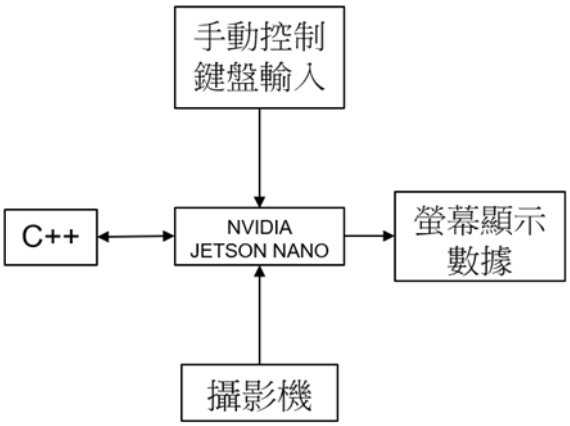

圖 3-1 硬體架構圖

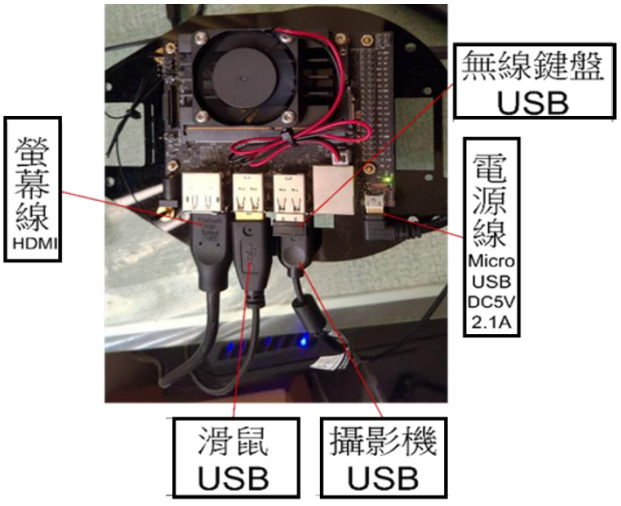

圖 3-2 設備接線圖

如圖 3.1 所示。本專題之硬體部份分別為：1.

單板 NVIDIA JETSON NANO　2. Logitech C270 HD 網路攝影機 3. 鍵盤 4. 螢幕。應用 Visual Studio Code 編譯軟體，用鍵盤手動控制程式，將攝影機畫面即時顯示至螢幕，由程式抓取人像位置，進行人臉辨識並顯示數據。

3.1.1 介紹 NVIDIA JETSON NANO

NVIDIA JETSON 是 NVIDIA 為嵌入式系統設計的人工智慧平台，而我們專題所使用的是 NVIDIA JETSON NANO，硬體外觀的尺寸只有 70 x 45 mm，並且具備了 472 GFLOP(每秒 10 億的浮點運算次數)的運算能力,能夠同時執行多個神經網路(例如 Caffe)，以及同步處理數個高解析度感應器，而且只需要耗費 5 到 10 瓦特。軟體方面則支援深度學習框架，以及 OpenCV 等電腦視覺應用[1]。

模組技術規格	
GPU	NVIDIA Maxwell™ 架構，配備 128 個 NVIDIA CUDA® 核心，運算效能可達 0.5 兆次浮點運算 (FP16)
CPU	四核心 ARM® Cortex®-A57 MPCore 處理器
記憶體	4 GB 64 位元 LPDDR4 1600 MHz | 25.6 GB/秒
儲存空間	16 GB eMMC 5.1 快閃記憶體
影片編碼	4Kp30 | 4x 1080p30 | 9x 720p30 (H.264/H.265)
影片解碼	4Kp60 | 2x 4Kp30 | 8x 1080p30 | 18x 720p30 (H.264/H.265)
攝影機	12 個通道 (3x4 或 4x2) MIPI CSI-2 D-PHY 1.1 (18 Gbps)
連線能力	Wi-Fi 需外接晶片
	10/100/1000 BASE-T 乙太網路
顯示器	HDMI 2.0 或 DP1.2 | eDP 1.4 | DSI (1x2) 2 個同步
UPHY	1 個 1/2/4 PCIE、1 個 USB 3.0、3 個 USB 2.0
I/O	3 個 UART、2 個 SPI、2 個 I2S、4 個 I2C、GPIO
尺寸	69.6 公釐 x 45 公釐
機械規格	260-pin 邊緣接頭

圖 3-3 NVIDIA JETSON NANO 模組技術規格
（圖片擷取 NVIDIA 官網）

開發套件技術規格	
GPU	128-core NVIDIA Maxwell™
CPU	Quad-core ARM A57 @ 1.43 GHz
記憶體	4GB 64-bit LPDDR4 25.6GB/s
儲存空間	microSD (不包含記憶卡)
影片編碼器	4Kp30 | 4x 1080p30 | 9x 720p30 (H.264/H.265)
影片解碼器	4Kp60 | 2x 4Kp30 | 8x 1080p30 | 18x 720p30 (H.264/H.265)
連線能力	Gigabit Ethernet, M.2 Key E
攝影機	2x MIPI CSI-2 connectors
顯示器	HDMI 與 DP
USB	4x USB 3.0, 1x USB 2.0 Micro-B
其他	40 pin 排針連接器 (GPIO、I2C、I2S、SPI、UART) 12 pin 排針連接器 (電源及相關訊號、UART) 4 pin 風扇排針連接器
機械規格	100 mm x 80 mm x 29 mm

圖 3-4 NVIDIA JETSON NANO 開發套件技術規格
（圖片擷取 NVIDIA 官網）

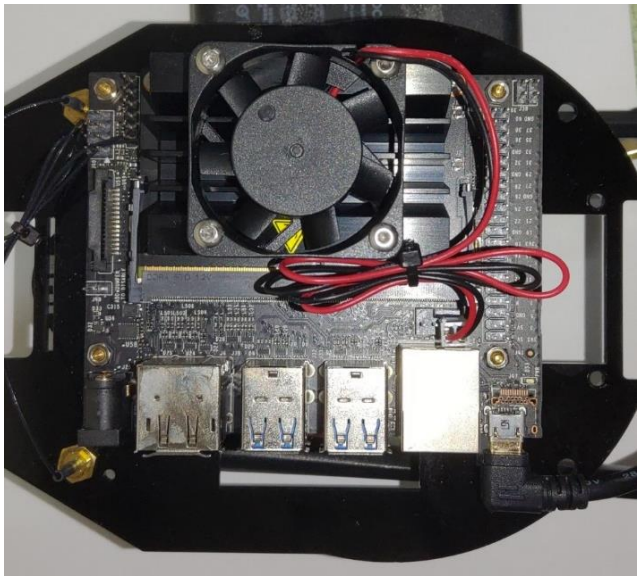

圖 3-5 NVIDIA JETSON NANO 正視圖

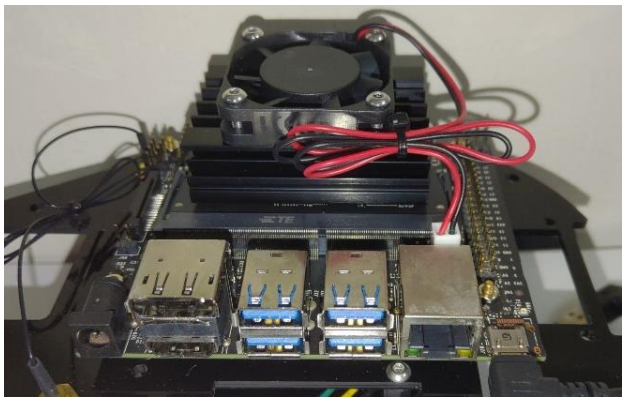

圖 3-6 NVIDIA JETSON NANO 側視圖

3.2 軟體

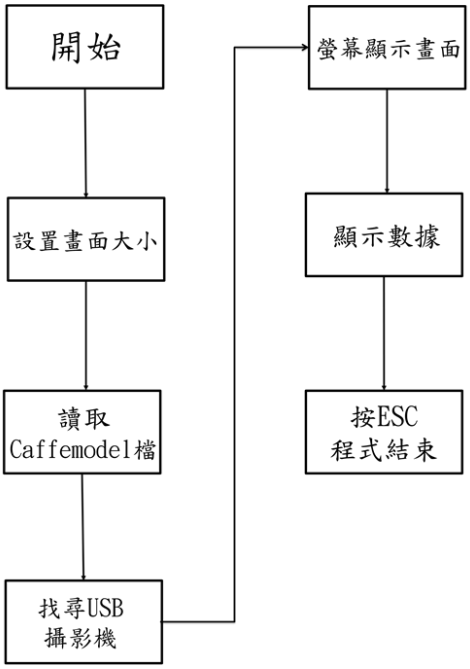

圖 3-7 軟體流程圖

圖 3.7 所示為軟體流程圖，當 Visual Studio Code 編譯軟體按下執行鍵，會將預先設置好的畫面大小設置完成，再讀取已學習完成的 Caffemodel 檔，接著找尋已接上 USB 接孔的攝影機，完成預設好的攝影機設定（攝影機畫面長寬、自動對焦），進行人臉辨識運算以及資料處理加工，螢幕顯示攝影機畫面及相關數據（頭像在畫面中的大小、頭像畫面左上角的位置、頭像編號、分辨機率、分辨頭像在畫面中的位置(左右)、頭像中間點），按 ESC 結束程式所有動作。

3.2.1 Ubuntu 作業系統

Ubuntu 是著名的 Linux 發行版之一，也是目前最多使用者的 Linux 版本，而 Ubuntu 專案完全遵從開源軟體的開發原則[2]。Ubuntu 和我們常用的 windows 10 有些不一樣，其中最明顯的就是操作介面，剛開始用有些不熟悉，不過 Ubuntu 有很多事都可以靠在 terminal（終端機）內輸入指令完成，下載、解壓縮、安裝、卸載等等，都是用 terminal 完成，terminal 幾乎可以做到所有的事。

圖 3-8 Ubuntu 作業系統執行畫面

3.2.2 安裝 Ubuntu 作業系統步驟

Step1 準備一片 SD 卡，官方建議最低要 16GB，我們取用 64GB。

Step2 使用有內建 SD 卡讀卡機的筆記型電腦。

Step3 下載官方提供的映像檔，下載後請將檔案解壓縮，並記得存檔路徑。

Step4 安裝映像檔燒錄軟體 balenaEtcher。

Step5 執行 balenaEtcher，進行燒錄。

Step6 將 SD 卡插入 NVIDIA JETSON NANO 卡槽。

Step7 啟動NVIDIA JETSON NANO並設定以下內容：同意條款、選擇語言、選擇鍵盤排列方式、選擇時區、設定帳號及密碼。

圖 3-9 64GB SD 卡

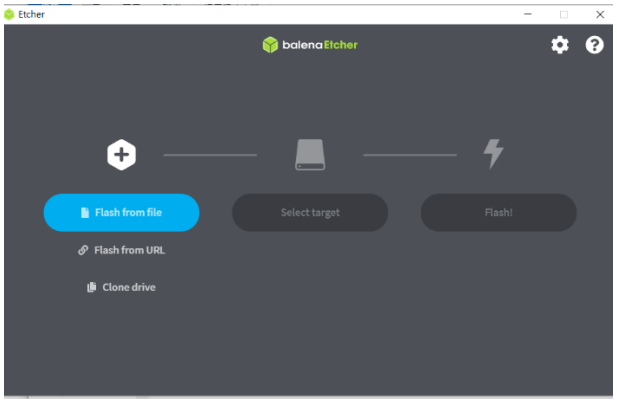

圖 3-10 balenaEtcher 燒錄軟體

3.2.3 介紹 Sudo 指令

sudo 是 Unix/Linux 平臺上的一個非常好用的工具，其中 Linux 系統最高權限的管理者帳號為 root，而 sudo 指令就可以用來取得 root 的權限，在使用 sudo 指令時，是輸入自己的密碼而不是輸入 root 號的密碼，以確保安全性 [3] 。

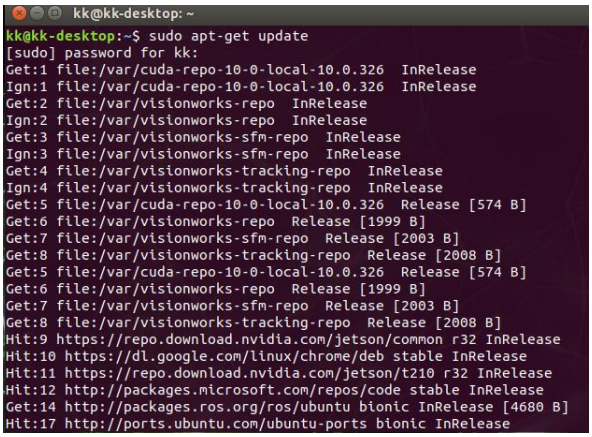

圖 3-11 Sudo 軟體更新包

3.2.4 介紹 Visual Studio Code（VS code）

Visual Studio Code 是一款免費原始碼編輯器，Visual Studio Code 預設支援非常多的程式語言，也可以通過下載擴充支援 Python、C/C++[4]。Visual Studio Code 如果要使用 opencv 的話，就必須要再使用另外三個檔案，這三個檔案分別是 c_cpp_properties.json、tasks.json、launch.jason，主要是把 opencv 的檔案位置設置完成，如果沒設置好，程式在一開始的宣告就會出現錯誤，導致沒辦法使用。

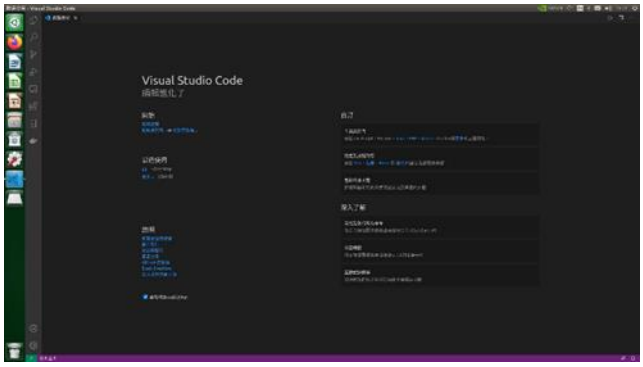

圖 3-12 Visual Studio Code 編譯軟體

3.2.5 介紹 OpenCV

Open Source Computer Vision Library(簡稱 OpenCV)是一個跨平台的電腦視覺庫[5]。OpenCV 可用於開發即時的圖像處理、電腦視覺以及圖形識別程式。OpenCV 可以在 Windows、Linux 等平台上執行，它的主要介面是 C++語言。在圖像和影像處理方面使用 OpenCV 可以節省很多時間，我們專題使用的版本是 OpenCV 4.1。

3.2.6 介紹 C++

C++是一種被廣泛使用的電腦程式設計語言，它是一種通用程式設計語言[6]。支援多重程式設計模式。C++是由 C 語言衍生而來，因此它包含了 C 語言的所有功能，幾乎所有的 C 語言程式在 C++裡只需要修改部分的程式碼，甚至不需要修改程式碼的情況下，便可正確的執行。

3.2.7 介紹 caffe

Caffe（快速特徵嵌入的卷積結構）是一個清晰、高效的深度學習框架[7]。核心語言是 C++，與支援 Python 和 Matlab，是一個經過認證後的開放原始碼軟體。我們專題所使用的 caffemodel 檔是使用別人已經訓練好的檔案，在執行的時候必須搭配 prototxt 執行檔，否則無法使用。

四、專題程式

4.1 專題程式介紹

```
1
2   #include <opencv2/opencv.hpp>
3   #include <opencv2/dnn.hpp>
4   #include <iostream>
5   #include <string>
6   using namespace cv::dnn;
7   using namespace cv;
8   using namespace std;
9   int main(int argc, char** argv)
```

圖 4-1　2-8 行 宣告

```
10  {
11      const int inWidth = 640;
12      const int inHeight = 480;
13      cout << "Run" << endl;
```

圖 4-2 11-12 行 設定畫面大小

```
14    Net net = readNetFromCaffe("deploy.prototxt", "res10_300x300_ssd_iter_140000.caffemodel");
```

圖 4-3 14 行 讀取 caffemodel 檔

```
15
16      VideoCapture cap(0);
17      cap.set(CAP_PROP_FRAME_WIDTH, inWidth);
18      cap.set(CAP_PROP_FRAME_HEIGHT, inHeight);
19      cap.set(CAP_PROP_AUTOFOCUS, 1);
```

圖 4-4 16-19 行 相機設定

```
20
21      string window_name = "Display" + to_string(inWidth) + "x" + to_string(inHeight);
22      namedWindow(window_name);
```

圖 4-5 21-22 行 視窗設定

```
23
24      while (getWindowProperty(window_name, WND_PROP_AUTOSIZE) >= 0)
25      {
26          Mat color_mat;
27          cap >> color_mat;
28
29          Mat inputBlob = blobFromImage(color_mat, 1.0, color_mat.size(), Scalar(104.0, 177.0, 123.0), false);
30          net.setInput(inputBlob, "data");
31          Mat detection = net.forward();
32
33          Mat detectionMat(detection.size[2], detection.size[3], CV_32F, (float*)detection.data);
34
35          float confidenceThreshold = 0.5;
36          Vec3b color[] = { Vec3b(255,255,0),Vec3b(0,255,0), Vec3b(0,255,255), Vec3b(255,255,127) };
37          int num = 0;
```

圖 4-6　24-37 行 人臉辨識運算

```
tt1.cpp > main(int, char **)
38          for (int i = 0; i < detectionMat.rows; i++)
39          {
40
41              float confidence = detectionMat.at<float>(i, 2);
42              if (confidence > confidenceThreshold)
43              {
44
45                  int x0 = (int)(detectionMat.at<float>(i, 3) * color_mat.cols);
46                  int y0 = (int)(detectionMat.at<float>(i, 4) * color_mat.rows);
47                  int x1 = (int)(detectionMat.at<float>(i, 5) * color_mat.cols);
48                  int y1 = (int)(detectionMat.at<float>(i, 6) * color_mat.rows);
49                  Rect object(x0, y0, x1 - x0 + 1, y1 - y0 + 1);
50                  cout << object;
51                  string ss = "#" + to_string(num++) + " Prob=" + to_string(confidence);
52                  cout << ss ;
53
54                  if ((x0 + (x1 - x0 + 1) / 2) < inWidth / 2)
55                  {
56                      cout << "   L   " ;
57                      cout << x0 + (x1 - x0 + 1) / 2 << endl;
58                  }
59                  else
60                  {
61                      cout << "   R   " ;
62                      cout << x0 + (x1 - x0 + 1) / 2 << endl;
63                  }
64                  rectangle(color_mat, object, color[num % 4], 2);
65                  int baseLine = 0;
66                  Size labelSize = getTextSize(ss, FONT_HERSHEY_SIMPLEX, 0.5, 1, &baseLine);
67                  Point LB(x0, y1 + labelSize.height);
68                  rectangle(color_mat, Rect(Point(LB.x, LB.y - labelSize.height),
69                      Size(labelSize.width, labelSize.height + baseLine)), color[num % 4], -1);
70                  putText(color_mat, ss, LB, FONT_HERSHEY_SIMPLEX, 0.5, Scalar(0, 0, 0));
71              }
```

圖 4-7 38-72 資料處理加工顯示

```
76          imshow(window_name, color_mat);
```

圖 4-8 76 行 顯示畫面

```
79          if (waitKey(100) == 27)break;
80      }
81      destroyAllWindows();
82      return 0;
83  }
```

圖 4-9 79-83 行 按 ESC 結束程式關閉畫面

4.2 專題環境設定

```
gg.cpp   tt1.cpp ●   {} c_cpp_properties.json ×   {} task
.vscode > {} c_cpp_properties.json > [ ] configurations > {} 0 > [ ] includePa
1   {
2       "configurations": [
3           {
4               "name": "Linux",
5               "includePath": [
6                   "${workspaceFolder}/**",
7                   "/usr/include/opencv4",
8                   "/usr/include"
9               ],
10              "defines": [],
11              "compilerPath": "/usr/bin/gcc",
12              "cStandard": "c11",
13              "cppStandard": "c++17",
14              "intelliSenseMode": "clang-x64"
15          }
16      ],
17      "version": 4
18  }
19
```

圖 4-10 c_cpp_properties.json

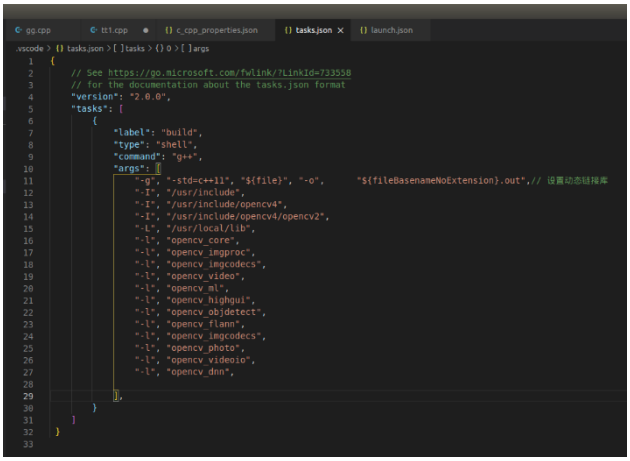

圖 4-11 tasks.json

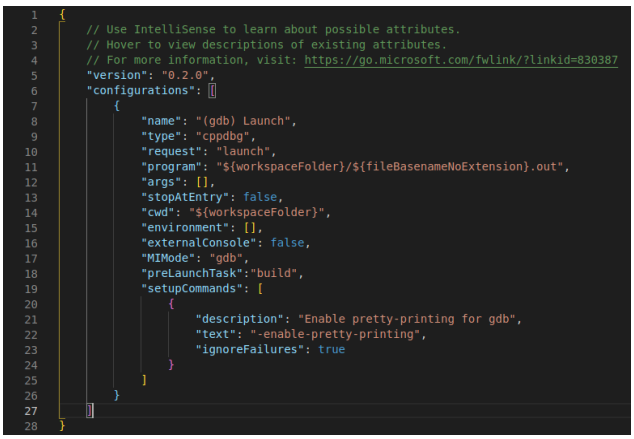

圖 4-12 launch.jason

五、實驗結果

數據顯示頭像在畫面中的大小、頭像畫面左上角的位置、頭像編號、分辨機率、分辨頭像在畫面中的位置（左右）、頭像中間點（如圖 5-1），並在人臉下方顯示頭像編號以及分辨機率（如圖 5-2）。

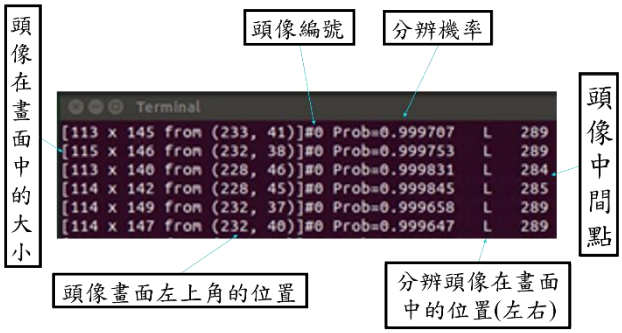

圖 5-1

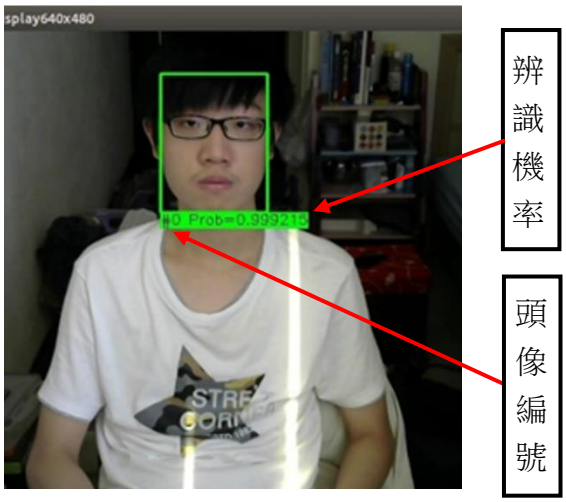

圖 5-2

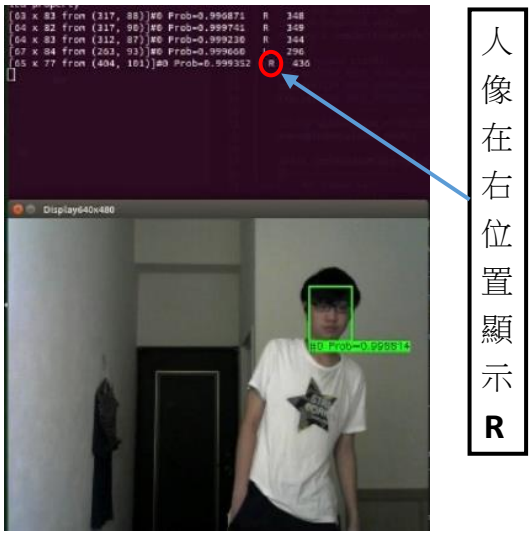

圖 5-3

圖 5-4

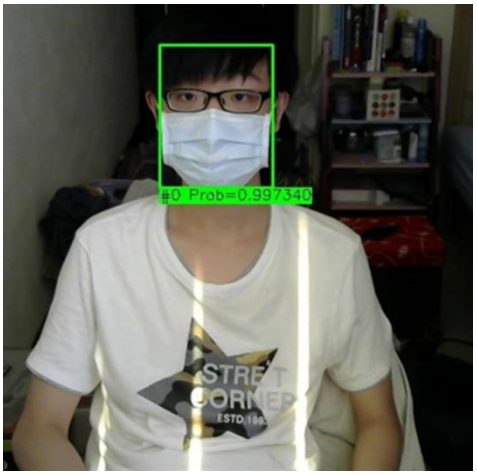

圖 5-5 戴口罩也能夠進行辨識

六、結論

本專題運用 NVIDIA JETSON NANO 單板進行即時人臉辨識，主要是以 C++來編寫該程式，其中運用了 Opencv 和 Caffe 的相關技術來執行，人臉辨識結果的機率很高即使有戴口罩也可以識別出來，畫面上也可以一次辨識多個人臉，也可以定位畫面的中間點。

六、參考資料

6.1 軟體介紹參考資料

1. https://www.nvidia.com/zh-tw/autonomous-machines/embedded-systems/jetson-nano/product-development/
2. https://zh.wikipedia.org/zh-tw/Ubuntu
3. https://blog.gtwang.org/linux/sudo-su-command-tutorial-examples/
4. https://zh.wikipedia.org/zh-tw/Visual_Studio_Code
5. https://zh.wikipedia.org/zh-tw/OpenCV
6. https://zh.wikipedia.org/zh-tw/C%2B%2B
7. https://zh.wikipedia.org/zh-tw/Caffe

6.2 程式參考資料

1. https://shengyu7697.github.io/opencv-4-1-1-build-install-in-ubuntu/
2. https://linuxize.com/post/how-to-install-visual-studio-code-on-ubuntu-18-04/
3. https://blog.csdn.net/sunzhao1000/article/details/103185875
4. https://zhuanlan.zhihu.com/p/363035298
5. https://www.youtube.com/watch?v=VYzzkHIvBhw&ab_channel=An-WenDengAn-WenDeng
6. https://github.com/gopinath-balu/computer_vision/tree/master/CAFFE_DNN
7. https://www.rs-online.com/designspark/jetson-nano-1-cn