逢 甲 大 學

資訊工程學系

專題研究報告

**基於影像辨識與網路爬蟲推薦加油站系統APP**

指導教授:陳錫民 老師

學生:蔡尚霖 D0514534

葉佑薪 D0542159

中 華 民 國 一 百 零 七 年 十 月 六 號

誌謝

很榮幸這次可以邀請到陳錫民老師,來當這次的指導教授,感謝老師在專題期間的教導,在專題技術發生困難的時候,給予適時的幫助我們,在每一次的討論之中,一步步修正了專題的錯誤,在最後才能成功製作出這個專題。在這裡也要特別感謝葉佑薪學弟幫忙製作後台的部分,沒有他的犧牲奉獻是做不出這個專題的。

摘要

在這個經濟不好的時代中,辛苦努力外出工作,總是少不了交通工具,例如:機車和汽車之類的,免不了的汽機車一定要加油,而台灣有這麼多的加油站,這麼多的品牌,各家為了搶生意通常都會打出一些加油折扣,而通常都會與信用卡結合,這麼多加油站加上這麼多信用卡,使用者要如何加到最省的油哩?

本研究主要將影像辨識與網路爬蟲結合在APP上,利用手機上的相機鏡頭實現影像辨識功能,來找出使用者所擁有的信用卡,接著加上使用者的設定,來跟後台的伺服器作匹配,進而回傳適合使用者的加油站與加油折扣資訊,最後再由APP顯示出來。

**關鍵字:OpenCv影像辨識,Android APP,Python爬蟲**

目錄

[誌謝 i](#_Toc529041895)

[摘要 ii](#_Toc529041896)

[目錄 iii](#_Toc529041897)

[圖目錄 v](#_Toc529041898)

[表目錄 vii](#_Toc529041899)

[第一章 導論 1](#_Toc529041900)

[1.1研究動機 1](#_Toc529041901)

[1.2研究目的 1](#_Toc529041902)

[1.3工作分配與甘特圖 1](#_Toc529041903)

[第二章 系統開發平台 1](#_Toc529041904)

[2.1 Android系統平台介紹 1](#_Toc529041905)

[2.2 開發工具 1](#_Toc529041906)

[2.2.1 Android SDK 1](#_Toc529041907)

[2.2.2 JAVA JDK 2](#_Toc529041908)

[2.2.3 Android Studio 2](#_Toc529041909)

[2.2.4 Android 架構 2](#_Toc529041910)

[2.2.5 OpenCv Android 3](#_Toc529041911)

[2.2.6 OpenCv Contrib 3](#_Toc529041912)

[2.2.7 Python 3](#_Toc529041913)

[2.3 開發環境 3](#_Toc529041914)

[2.3.1 Android模擬器 3](#_Toc529041915)

[2.3.2 Jupyter 4](#_Toc529041916)

[2.3.3 MySql 4](#_Toc529041917)

[第三章 文獻回顧與探討 5](#_Toc529041918)

[3.1圖像辨識SURF 5](#_Toc529041919)

[3.2序列化Kryo 6](#_Toc529041920)

[3.3 Android線程池(FixedThreadPool) 6](#_Toc529041921)

[3.4 Flask 6](#_Toc529041922)

[第四章 研究方法與步驟 7](#_Toc529041923)

[4.1 系統架構 7](#_Toc529041924)

[4.2系統流程 8](#_Toc529041925)

[4.3前置處理 8](#_Toc529041926)

[4.3.1 提出特徵點 8](#_Toc529041927)

[4.3.2 Mat轉型 9](#_Toc529041928)

[4.3.3 序列化 9](#_Toc529041929)

[4.3.4反序列化 10](#_Toc529041930)

[4.4相機掃描 10](#_Toc529041931)

[4.5圖片處理 11](#_Toc529041932)

[4.5.1 圖片切割 11](#_Toc529041933)

[4.5.2 圖片縮小 12](#_Toc529041934)

[4.6圖片比對 13](#_Toc529041935)

[4.6.1 偵測清晰度 13](#_Toc529041936)

[4.6.2 比對相似度 14](#_Toc529041937)

[第五章 實作結果 16](#_Toc529041938)

[5.1系統介面 16](#_Toc529041939)

[5.2功能介紹 18](#_Toc529041940)

[5.3 APP比較 20](#_Toc529041941)

[第六章 結論 21](#_Toc529041942)

[6.1遭遇問題 21](#_Toc529041943)

[6.1.1前端部分 21](#_Toc529041944)

[6.1.2 後端部分 21](#_Toc529041945)

[6.2未來展望 21](#_Toc529041946)

[第七章 心得 22](#_Toc529041947)

[參考文獻 23](#_Toc529041948)

圖目錄

[圖1.1甘特圖 2](#_Toc529041949)

[圖2.1 Android系統架構圖 2](#_Toc529041950)

[圖4.1 系統架構圖 7](#_Toc529041951)

[圖4.2系統流程圖 8](#_Toc529041952)

[圖4.3掃描流程圖 8](#_Toc529041953)

[圖4.4特徵點範例圖 9](#_Toc529041954)

[圖4.5提出特徵點程式碼 9](#_Toc529041955)

[圖4.6 Mat轉型程式碼 9](#_Toc529041956)

[圖4.7 序列化程式碼 10](#_Toc529041957)

[圖4.8 反序列化程式碼 10](#_Toc529041958)

[圖4.9相機掃描圖 11](#_Toc529041959)

[圖4.10相機掃描程式碼 11](#_Toc529041960)

[圖4.11掃描原始大小圖 12](#_Toc529041961)

[圖4.12預設比例程式碼 12](#_Toc529041962)

[圖4.13縮小圖 13](#_Toc529041963)

[圖4.14縮小程式碼 13](#_Toc529041964)

[圖4.15清晰度程式碼 14](#_Toc529041965)

[圖4.16相似度範例圖 14](#_Toc529041966)

[圖4.17相似度程式碼 15](#_Toc529041967)

[圖5.1權限詢問 16](#_Toc529041968)

[圖5.2登入 16](#_Toc529041969)

[圖5.3註冊 17](#_Toc529041970)

[圖5.4主畫面 17](#_Toc529041971)

[圖5.5油價表 18](#_Toc529041972)

[圖5.6地圖 18](#_Toc529041973)

[圖5.7設定 19](#_Toc529041974)

[圖5.8手選信用卡 19](#_Toc529041975)

[圖5.9掃描信用卡 20](#_Toc529041976)

表目錄

[表1.1工作分配表 1](#_Toc529041977)

[表5.3.1 APP比較表 20](#_Toc529041978)

第一章 導論

1.1研究動機

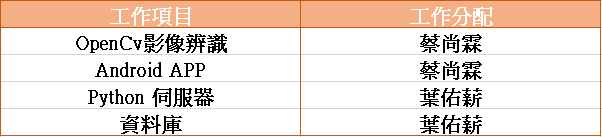
根據交通部2017年10月份統計資料，全台總共有七百九十餘萬輛汽車與一千三百餘萬輛機車，選擇附近加油站為汽機車加油已成為人們日常不可或缺的事項。此外根據經濟部能源局統計2017年全台有二千四百多家加油站，屬於高密度的分佈，雖然目前中油與台塑石油的油價訂定是一致地，但各家加油站會根據加油的日期、自助加油與否以及信用卡的種類等提供各式各樣的優惠，這些優惠也會隨著時間動態調整，而人們常常必須到了加油站才開始選擇較為優惠方案。

1.2研究目的

有鑑於此，在本計畫中我們將研發一套最佳加油站推薦系統，讓使用者使用APP可以搜尋到附近最便宜的加油站。在本系統中，我們將採用影像辨識技術提供較佳的使用者體驗，讓使用者方便地登錄所擁有的信用卡，同時我們也將開發網路爬蟲自動化抓取各家加油站在網路上公告的優惠。藉此，當使用者透過智慧手機尋找周遭的加油站時，系統會根據使用者所在的位置、使用者的喜好、當下的日期、登錄的信用卡卡別以及各家加油站的優惠等，推薦使用者周遭油價便宜的加油站。

1.3工作分配與甘特圖

表1.1工作分配表



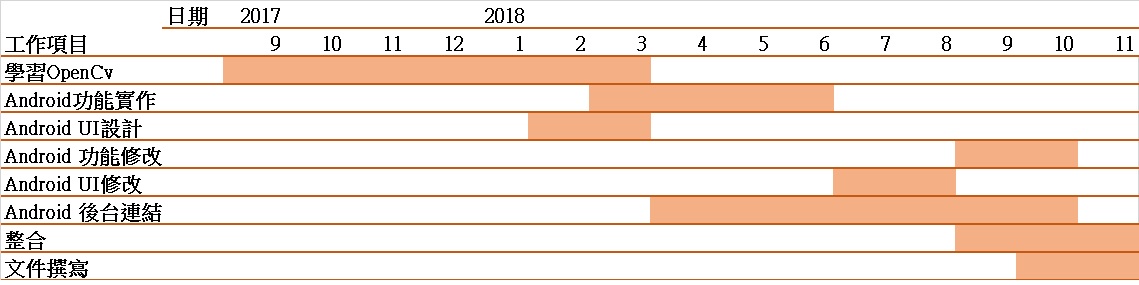
****

圖1.1甘特圖

第二章 系統開發平台

2.1 Android系統平台介紹

Android，常見的非官方中文名稱為安卓，是一個基於Linux核心的開放原始碼行動作業系統，由Google成立的Open Handset Alliance（OHA，開放手機聯盟）持續領導與開發，主要設計用於觸控螢幕行動裝置如智慧型手機和平板電腦與其他可攜式裝置。

Android Inc.於2003年10月由Andy Rubin、Rich Miner、Nick Sears和Chris White 在加州帕羅奧圖創建。Android最初由安迪·魯賓等人開發製作，最初開發這個系統的目的是創建一個數位相機的先進作業系統，這是2004年4月該公司向投資者介紹的基礎；但是後來發現市場需求不夠大，加上智慧型手機市場快速成長，於是Android成為一款面向智慧型手機的作業系統。於2005年7月11日Android Inc.被美國科技企業Google收購。2007年11月，Google與84家硬體製造商、軟體開發商及電信營運商成立開放手機聯盟來共同研發改良Android，隨後，Google以Apache免費開放原始碼許可證的授權方式，發布了Android的原始碼，開放原始碼加速了Android普及，讓生產商推出搭載Android的智慧型手機，Android後來更逐漸拓展到平板電腦及其他領域上。

2010年末資料顯示，僅正式推出兩年的Android作業系統在市場佔有率上已經超越稱霸逾十年的諾基亞Symbian系統，成為全球第一大智慧型手機作業系統。

在2014年Google I/O開發者大會上Google宣布過去30天裡有10億台活躍的安卓裝置，相較於2013年6月則是5.38億。

2017年3月，Android全球網路流量和裝置超越Microsoft Windows，正式成為全球第一大作業系統。[1]

2.2 開發工具

2.2.1 Android SDK

Android提供的軟體開發套件（Android SDK），是一套開發Android應用程式需要的基本套件，裡面包含許多在開發應用程式必須使用的基本工具程式，也包含一些用來管理與下載資源的工具程式。Android在2014年12月發表全新的Android Studio，取代原來使用的Eclipse開發工具。Android Studio整合所有開發Android應用程式需要的工具，讓開發人員可以使用這些特別為Android設計的功能，快速的開發與測試Android應用程式。

2.2.2 JAVA JDK

Java Development Kit（JDK）是昇陽電腦針對Java開發人員發布的免費軟體開發套件（SDK，Software development kit）。自從Java推出以來，JDK已經成為使用最廣泛的Java SDK。由於JDK的一部分特性採用商業許可證，而非開源。因此，2006年昇陽電腦宣布將發布基於GPL的開源JDK，使JDK成為自由軟體。在去掉了少量閉源特性之後，昇陽電腦最終促成了GPL的OpenJDK的發布。

2.2.3 Android Studio

Android Studio是一個為Android平台開發程式的整合式開發環境。2013年5月16日在Google I/O上發布，可供開發者免費使用。2013年5月發布早期預覽版本，版本號為0.1。2014年6月發布0.8版本，至此進入beta階段。第一個穩定版本1.0於2014年12月8日發布。Android Studio基於JetBrains IntelliJ IDEA，為Android開發特殊客製化，並在Windows、OS X和Linux平台上均可執行。

Android Studio目前是Google正式支援IDE，提供視覺化布局、內建Android SDK和AVD管理器、Android特定程式碼重構和快速修復、內建布局編輯器，可讓開發者拖放UI元件，並預覽在不同尺寸裝置上的UI顯示效果…。

2.2.4 Android 架構

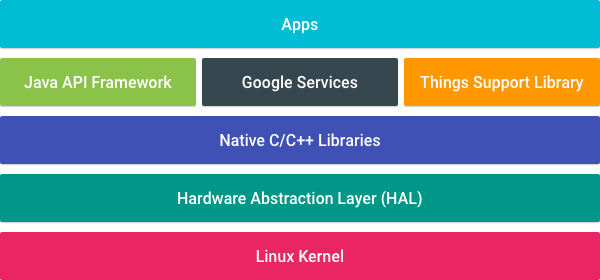


圖2.1 Android系統架構圖

2.2.5 OpenCv Android

OpenCV的全稱是Open Source Computer Vision Library，是一個跨平台的電腦視覺庫。OpenCV是由英特爾公司發起並參與開發，以BSD授權條款授權發行，可以在商業和研究領域中免費使用。OpenCV可用於開發實時的圖像處理、電腦視覺以及模式識別程式。

原本OpenCv要導入到Android需要透過JNI，編譯成C/C++才能在Android裡面使用,爾且還需要在手機裡預先裝設OpenCv Manager才能使用OpenCv的功能。但是現在OpenCv已經推出以JAVA編譯的OpenCv，只需要導入Module然後將.os檔和.a檔匯入Android Project之後，就不在需要安裝OpenCv Manager，也可以在手機上運行。

2.2.6 OpenCv Contrib

自從進入3.X時代以後，OpenCV將代碼庫分成了兩部分，分別是穩定的核心功能庫和試驗性質的contrib庫。並且從3.1以後，編譯好的版本中就沒有現成的x86平台的庫了，想用就必須自己從源碼編譯，這次專案中就有使用到了contrib的surf演算法

2.2.7 Python

Python是完全物件導向的語言。函式、模組、數字、字串都是物件。並且完全支援繼承、重載、衍生、多重繼承，有益於增強原始碼的複用性。Python支援重載運算符，因此Python也支援泛型設計。

Python本身被設計為可擴充的。並非所有的特性和功能都整合到語言核心。Python提供了豐富的API和工具，以便程式設計師能夠輕鬆地使用C、C++、Python來編寫擴充模組。Python編譯器本身也可以被整合到其它需要指令碼語言的程式內。因此，有很多人把Python作為一種「膠水語言」使用。使用Python將其他語言編寫的程式進行整合和封裝。在Google內部的很多專案，例如Google應用服務引擎使用C++編寫效能要求極高的部分，然後用Python或Java/Go調用相應的模組。[2]

2.3 開發環境

2.3.1 Android模擬器

Android SDK內有包含一套完整模擬Android系統的手機或平板硬體環境之AVD模擬器，主要是給Android系統開發應用程式用的。當你裝完Android Studio時預設會安裝Android SDK，也就是說成功安裝了AVD模擬器，有了這個模擬器，不用實體的手機就能輕鬆開發Android的App，唯一的條件就是你的機器校能不能太差。

2.3.2 Jupyter

Jupyer Notebook（以前稱為IPython notebook） 是一個介於IDE(Pycharm, Spider)以及Editor(Sublime text, Atom, VScode, 記事本)之間的一個讓你可以寫code的工具。並利用直譯語言的特性，可以很容易的逐行執行並且做到資料視覺化，而且可輸出成 Html5嵌入到任何網頁或是Blog上，或是用notebook的形式分享。由於容易分享，許多資料分析、機器學習的平台(ex: Kaggle)上面的討論都是以Jupyter notebook的形式出現，並且支援多種主流的直譯語言像是Python,R,Julia。[3]

2.3.3 MySql

MySQL原本是一個開放原始碼的關聯式資料庫管理系統，原開發者為瑞典的MySQL AB公司，該公司於2008年被昇陽微系統（Sun Microsystems）收購。2009年，甲骨文公司（Oracle）收購昇陽微系統公司，MySQL成為Oracle旗下產品。

MySQL在過去由於效能高、成本低、可靠性好，已經成為最流行的開源資料庫，因此被廣泛地應用在Internet上的中小型網站中。隨著MySQL的不斷成熟，它也逐漸用於更多大規模網站和應用，比如維基百科、Google和Facebook等網站。非常流行的開源軟體組合LAMP中的「M」指的就是MySQL。

與其他的大型資料庫例如Oracle、IBM DB2、MS SQL等相比，MySQL自有它的不足之處，如規模小、功能有限等，但是這絲毫也沒有減少它受歡迎的程度。對於一般的個人使用者和中小型企業來說，MySQL提供的功能已經綽綽有餘，而且由於MySQL是開放原始碼軟體，因此可以大大降低總體擁有成本。[4]

第三章 文獻回顧與探討

3.1圖像辨識SURF

SURF（加速版的具有魯棒性的特徵，SpeededUp Robust Features），SURF是尺度不變特徵變換算法（SIFT算法）的加速版。 SURF最大的特徵在於採用了harr特徵以及積分圖像的概念。

SURF原理：（1）構建Hessian矩陣構造高斯金字塔尺度空間SURF採用的是Hessian矩陣（SURF算法核心）行列式近似值圖像。在數學中，Hessian矩陣是一個自變量為向量的實值函數的二階偏導數組成的方塊矩陣，即每一個像素點都可以求出一個2x2的Hessian矩陣，可計算出其行列式detH，可以利用行列式取值正負來判別該點是或不是極值點來將所有點分類。在SURF算法中，選用二階標準高斯函數作為濾波器，通過特定核間的捲積計算二階偏導數，從而計算出Hessian矩陣，但是由於特徵點需要具備尺度無關性，所以在進行Hessian矩陣構造前，需要對其進行高斯濾波，即與以方差為自變量的高斯函數的二階導數進行卷積。通過這種方法可以為圖像中每個像素計算出其H的行列式的決定值，並用這個值來判別特徵點。

（2）利用非極大值抑制初步確定特徵點，將經過hessian矩陣處理過的每個像素點與其三維鄰域的26個點進行大小比較，如果它是這26個點中的最大值或者最小值，則保留下來，當作初步的特徵點。檢測過程中使用與該尺度層圖像解析度相對應大小的濾波器進行檢測。

（3）精確定位極值點採用三維線性插值法得到亞像素級的特徵點，同時也去掉那些值小於一定閾值的點，增加極值使檢測到的特徵點數量減少，最終只有幾個特徵最強點會被檢測出來。

（4）選取特徵點的主方向而在SURF中，不統計其梯度直方圖，而是統計特徵點鄰域內的harr小波特徵。即在特徵點的鄰域（比如說，半徑為6s的圓內，s為該點所在的尺度）內，統計60度扇形內所有點的水平haar小波特徵和垂直haar小波特徵總和，haar小波的尺寸變長為4s，這樣一個扇形得到了一個值，然後60度扇形以一定間隔進行旋轉，最後將最大值那個扇形的方向作為該特徵點的主方向。

（5）構造surf特徵點描述算子，是在特徵點周圍取16x16的鄰域，並把該鄰域化為4x4個的小區域，每個小區統計8個方向的梯度，最後得到4x4x8=128維的向量。

在SURF中，也是在特徵點周圍取一個正方形框，框的邊長為20s（s是所檢測到該特徵點所在的尺度）。該框帶方向，方向當然就是第（4）步檢測出來的主方向了。然後把該框分為16個子區域，每個子區域統計25個像素的水平方向和垂直方向的haar小博特徵，這裡的水平和垂直方向都是相對主方向而言的。該haar小波特徵為水平方向值之和，水平方向絕對值之和，垂直方向之和，垂直方向絕對值之和。這樣每個區域就有4個值，所以每個特徵點就是16x4=64維向量，這在特徵匹配過程中會大大加快匹配速度。[5]

3.2序列化Kryo

Kryo 是一個快速序列化/反序列化工具，其使用了字節碼生成機制（底層依賴了 ASM 庫），因此具有比較好的運行速度。

Kryo 序列化出來的結果，是其自定義的、獨有的一種格式，不再是JSON 或者其他現有的通用格式；而且，其序列化出來的結果是二進制的（即byte[]；而JSON 本質上是字符串String）；二進制數據顯然體積更小，序列化、反序列化時的速度也更快。

Kryo 一般只用來進行序列化（然後作為緩存，或者落地到存儲設備之中）、反序列化，而不用於在多個系統、甚至多種語言間進行數據交換。目前kryo 也只有java 實現。[6]

3.3 Android線程池(FixedThreadPool)

在使用圖片比對的時候,對於APP來說是一個耗時的工作,所以不能放在一般的UI線程上,這樣會造成UI卡頓或是凍結。那麼這些事情應該交給子線程去做，但對於一個系統而言，創建、銷毀、調度線程的過程是需要開銷的，所以我們並不能無限量地開啟線程，那麼對線程的了解就變得尤為重要了。

FixedThreadPool它是個線程數量固定的線程池，該線程池的線程全部為核心線程，它們沒有超時機制且排隊任務隊列無限制，因為全都是核心線程，所以響應較快，且不用擔心線程會被回收。[7]

3.4 Flask

Flask 是一個使用 Python 撰寫的輕量級 Web 應用程式框架，由於其輕量特性，也稱為 micro-framework（微框架）。Flask 核心十分簡單，主要是由 Werkzeug WSGI 工具箱和 Jinja2 模板引擎所組成，Flask 和 Django 不同的地方在於 Flask 給予開發者非常大的彈性（當然你也可以說是需要思考更多事情），可以選用不同的用的 extension 來增加其功能。相比之下，Django 雖然完善但技術選擇相對不彈性，不論是 ORM、表單驗證或是模版引擎都有自己的作法。事實上沒有最好的框架，只有合適的使用情境，Django 相比之下適合需要快速的開發大型的應用程式，和 Ruby 中的 Ruby on Rails 相似，而 Flask 則是相對輕量彈性，更像是 Ruby 界的 Sinatra。[8]

第四章 研究方法與步驟

4.1 系統架構

整個系統大概分成五個部分:

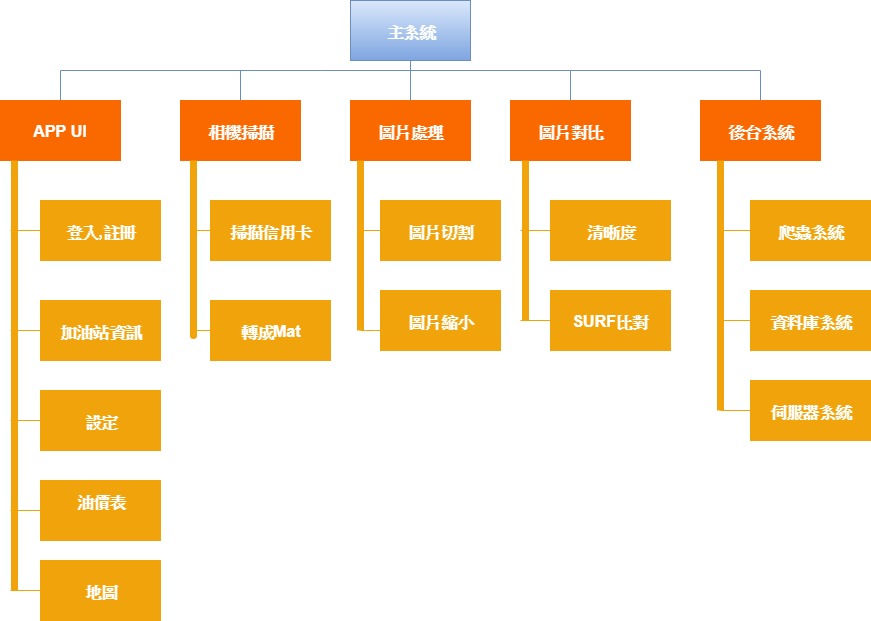


圖4.1 系統架構圖

4.2系統流程

首先開啟APP，接著登入或是註冊帳號，接下來進入設定畫面，使用掃描信用卡功能，開啟相機掃描，開始辨識信用卡跳出可能的信用卡選項，選擇你的信用卡，最後返回主畫面更新加油站資訊。

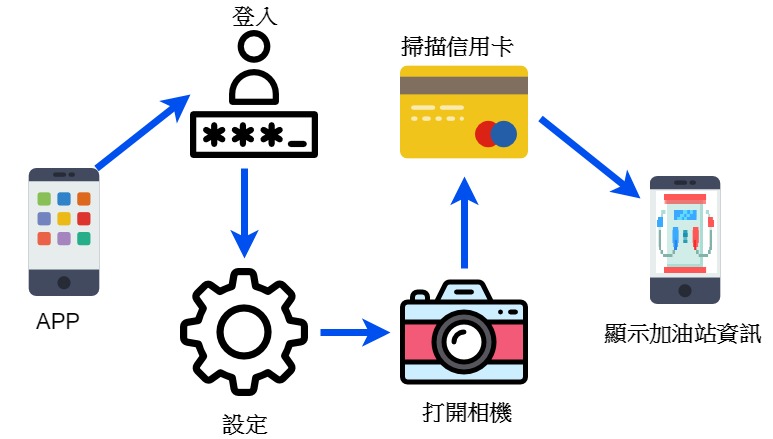


圖4.2系統流程圖

按下相機掃描按鈕之後，首先相機會先擷取畫面，接著會裁切成預設的大小,然後將圖像縮小，送進線程池使用SURF做圖像比對，最後更新使用者資料和顯示的加油站資訊。

**C:\Users\Shang\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\34402D89.tmp**

圖4.3掃描流程圖

4.3前置處理

4.3.1 提出特徵點

要提出圖片的特徵點，我們需要一個Detector(檢測器) ，這裡我們選擇SURF檢測器，首先將圖片轉型成Mat，接下來使用SURF做detect()和compute()，最後就能產出一張具有特徵點的Mat檔案。

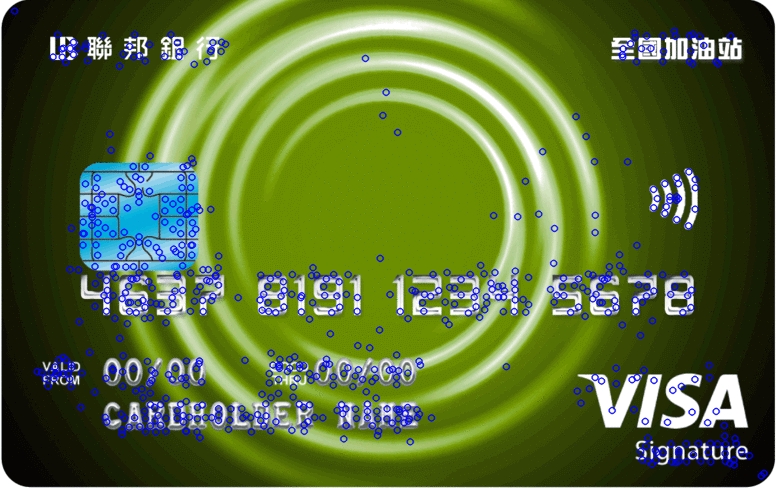


圖4.4特徵點範例圖

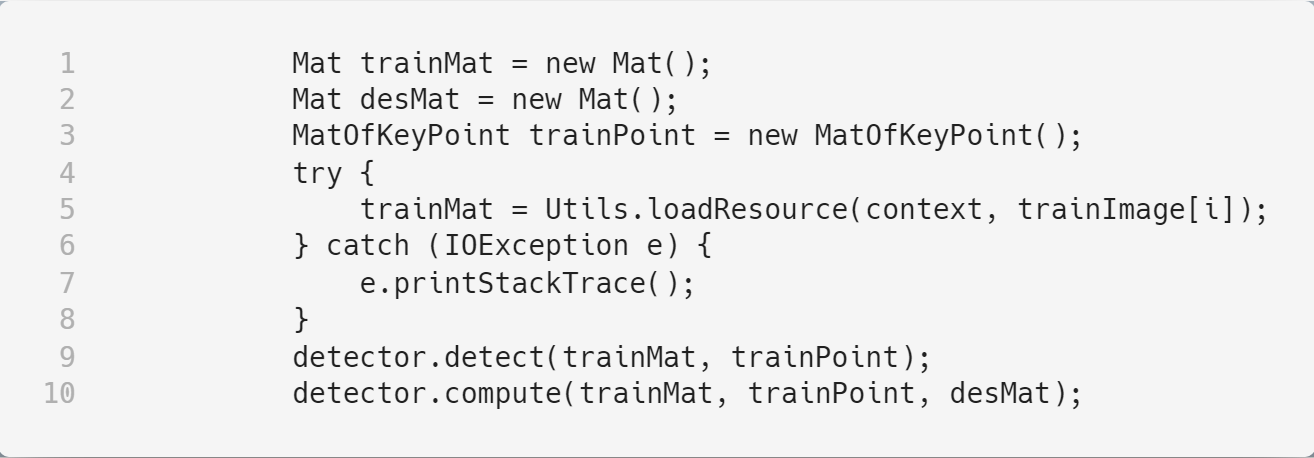


圖4.5提出特徵點程式碼

4.3.2 Mat轉型

要比對用的數據是Mat格式，可是Mat並不能直接輸出成檔案，所以先要對Mat進行一個轉型，將Mat轉型成float[]格式,以便可以輸出成檔案。

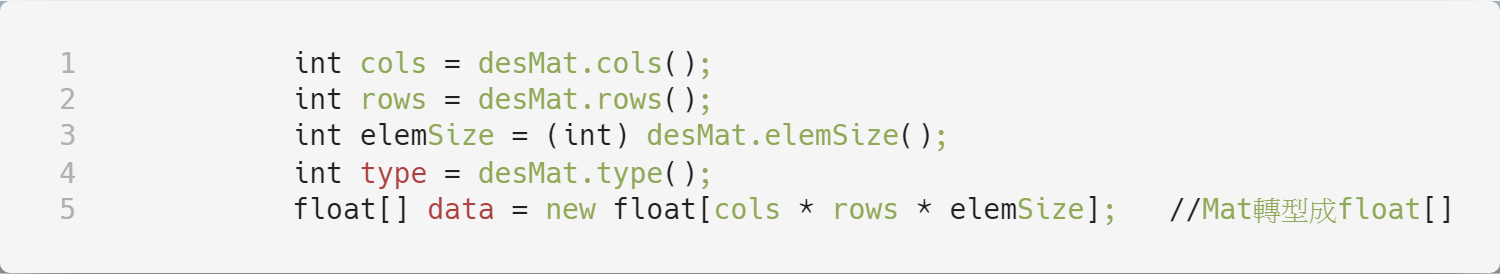


圖4.6 Mat轉型程式碼

4.3.3 序列化

使用Kryo這個函式庫,將信用卡的資訊跟float[]一起序列化成.bin檔。

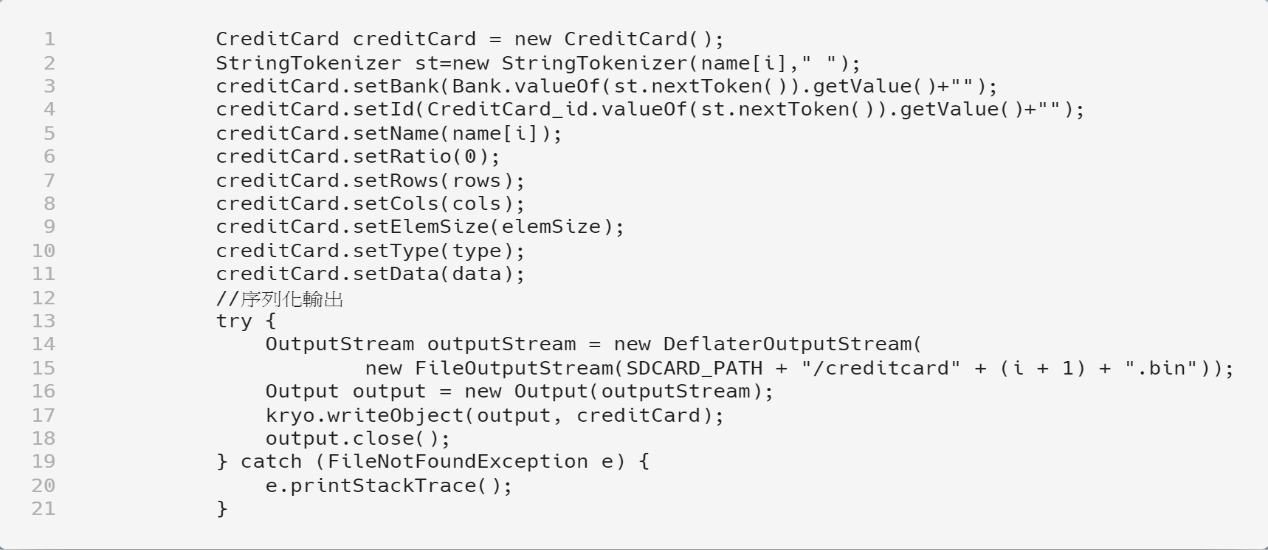


圖4.7 序列化程式碼

4.3.4反序列化

當進入相機掃描之前,需要先把手機端儲存的序列化檔案,反序列化回來。

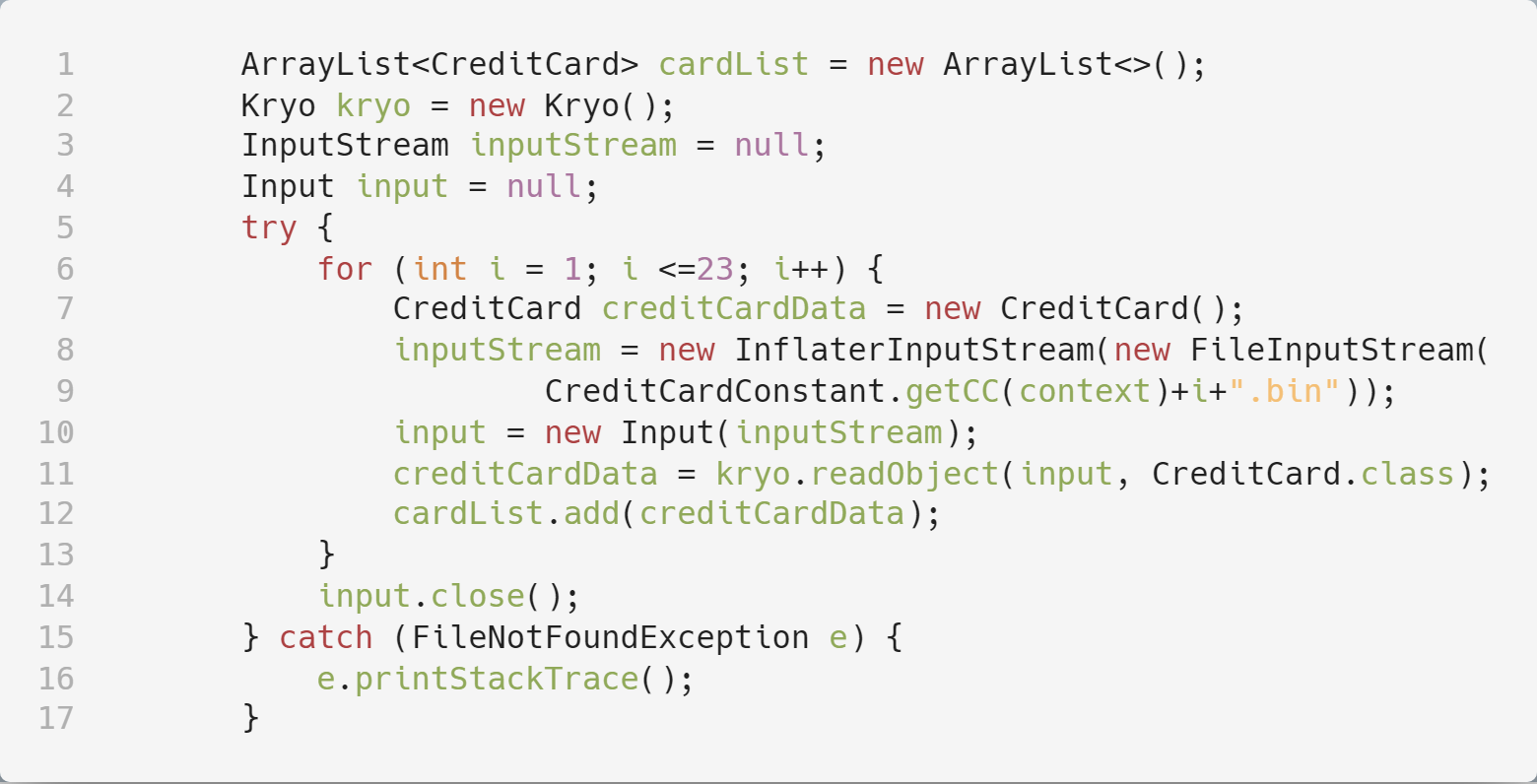


圖4.8 反序列化程式碼

4.4相機掃描

先透過OpenCv所提供的CameraBridgeViewBase將相機所掃描到的圖像轉成Mat格式。



圖4.9相機掃描圖



圖4.10相機掃描程式碼

4.5圖片處理

4.5.1 圖片切割

將從相機取得的Mat圖檔,切割成預設好的比例,目的是為了減少不相關的背景,原始大小1033x649。

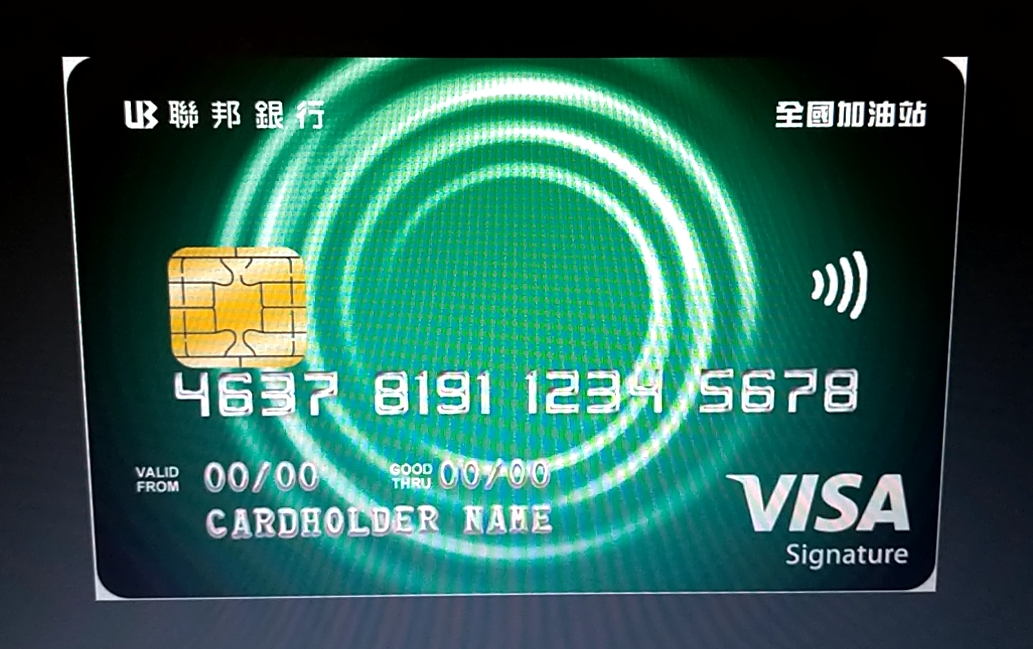


圖4.11掃描原始大小圖

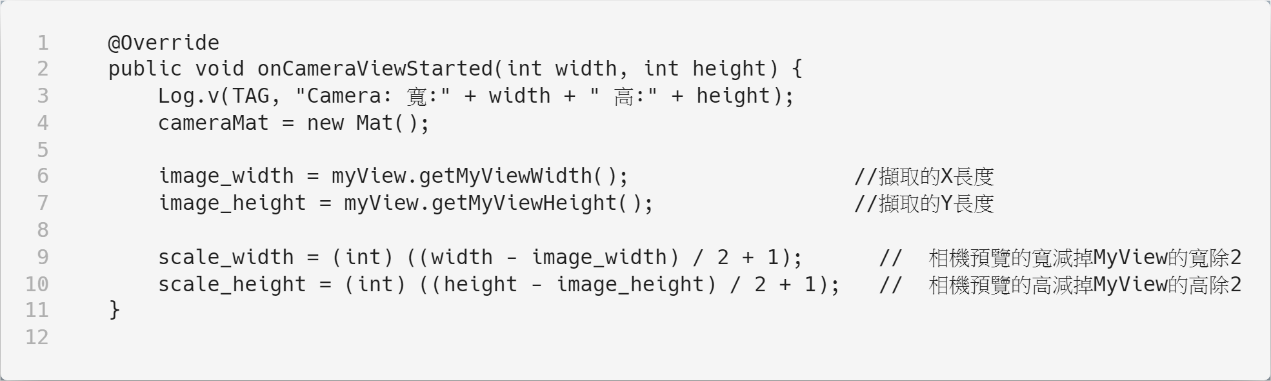


圖4.12預設比例程式碼

4.5.2 圖片縮小

接著將切割好的圖片,進行reSize來縮小圖片,目的是為了在對比的時候可以加快速度,現在大小430x270。



圖4.13縮小圖

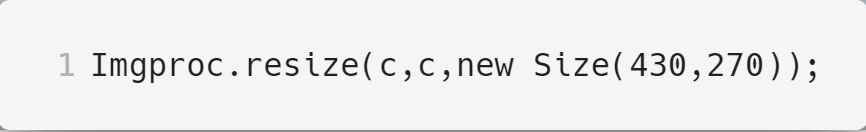


圖4.14縮小程式碼

4.6圖片比對

4.6.1 偵測清晰度

將處理好的Mat進行比對,首先將先掃描到的圖片,偵測它的清晰度,這是為了減少比對的次數,因為掃描時使用者可能會手抖或是相機還沒對焦到。

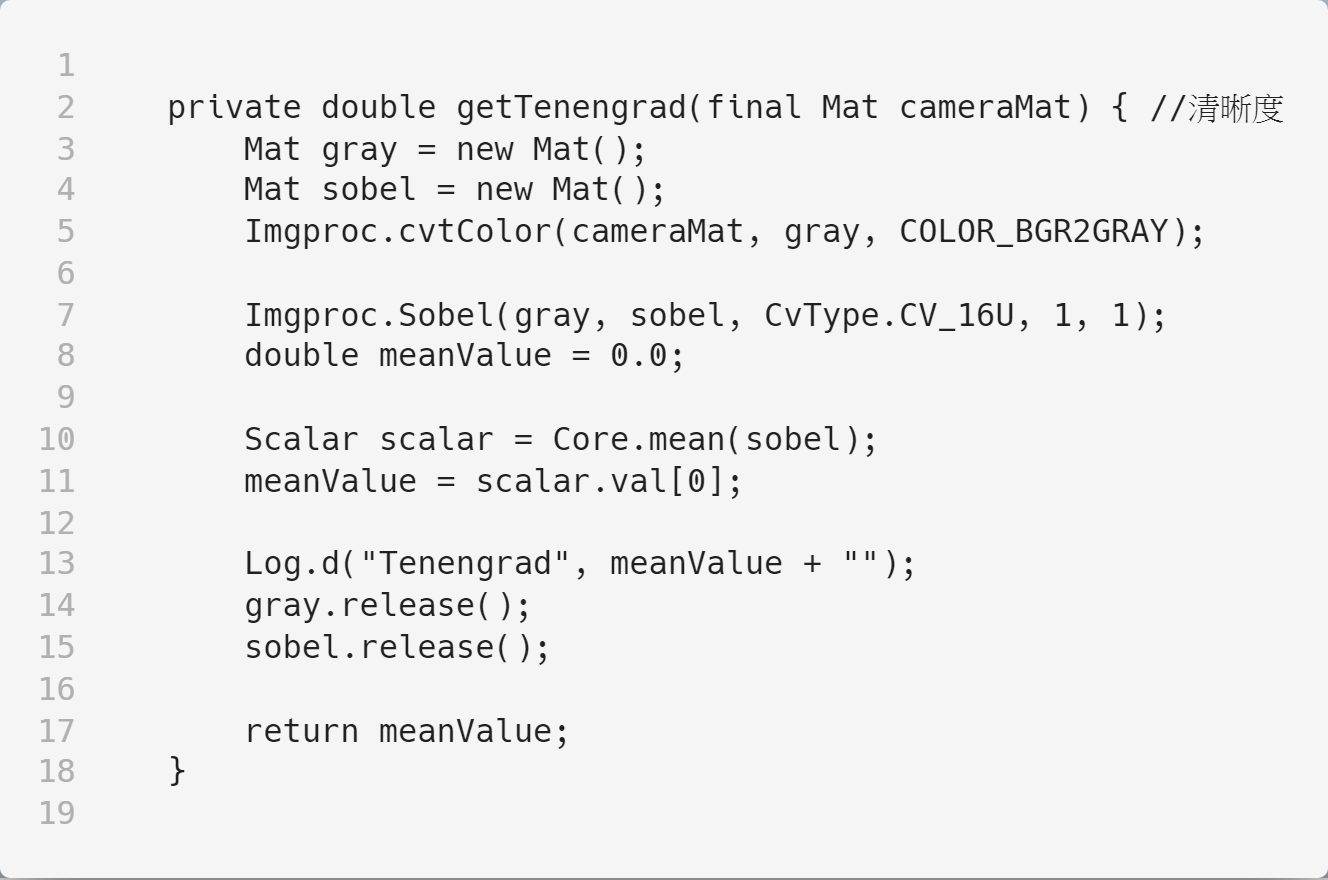


圖4.15清晰度程式碼

4.6.2 比對相似度

將清晰度足夠的圖片,使用SURF去偵測和計算它的特徵點,接下來在和反序列化之後的信用卡進行比對,最後在將把相似度高於70%紀錄回傳。

圖4.16相似度範例圖

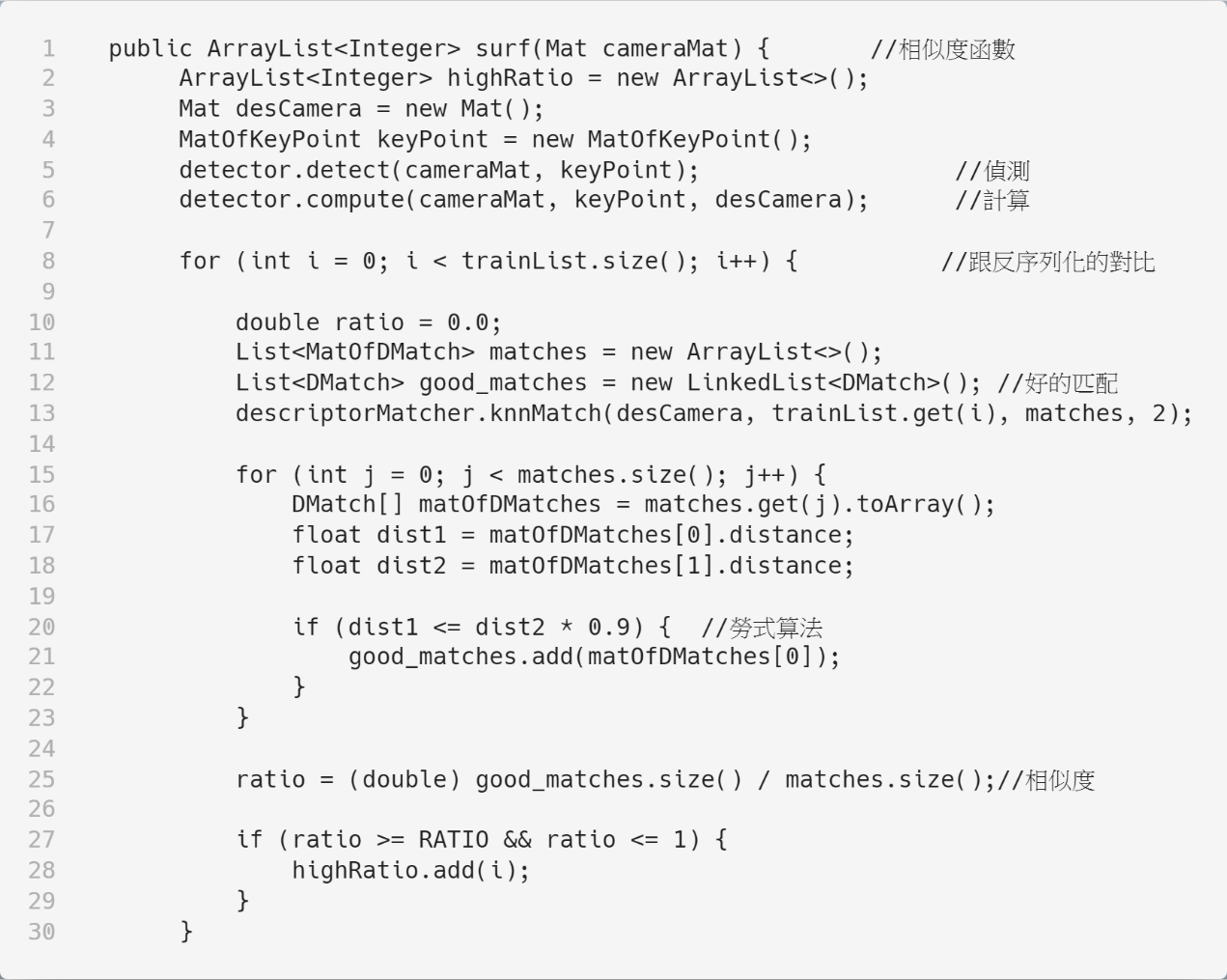


圖4.17相似度程式碼

第五章 實作結果

5.1系統介面

一開始打開APP,如果沒有開起權限的話,會跳出詢問權限方塊,接著如果是第一次開起APP的話,會下載信用卡資料。



圖5.1權限詢問

接著如果已有帳戶可以直接登入。



圖5.2登入

如果沒有帳戶的話可以選擇註冊。



圖5.3註冊

進入主畫面之後,會顯示加油站資訊,以距離,價格和使用者最愛排序顯示,同時也有油價表,設定和地圖可以選擇



圖5.4主畫面

5.2功能介紹

如圖X.X當點擊油價表時,就會跳出今日的油價和下周的油價漲幅預測



圖5.5油價表

如圖X.X當點擊加油站資訊時,會切換至地圖模式,如圖X.X在地圖模式中當,點擊Marker之後,會顯示加入最愛按鈕和導航按鈕



圖5.6地圖

如圖X.X當點擊設定時,會進入設定畫面如圖X.X,在設定畫面裡可以設定油品,加油模式,與加油站距離,預設加油公升數,也可以新增使用者擁有的信用卡,方法有兩種,1.手選信用卡,2.掃描信用卡



圖5.7設定

當選擇手選信用卡時如圖X.X,先選擇銀行再選擇信用卡

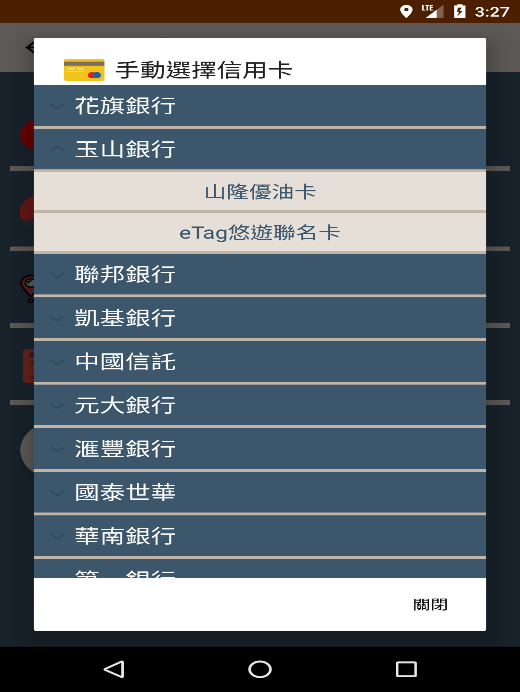


圖5.8手選信用卡

當選擇掃描信用卡時如圖X.X,畫面會進入到相機模式,按下右下角的掃描按鈕,即會開啟掃描模式,當使用者的信用卡對比到資料庫的信用卡時,高於70%相似圖的信用卡名稱,將會提供給使用者選擇。



圖5.9掃描信用卡

5.3 APP比較

表5.3.1 APP比較表

|  |  |  |  |
| --- | --- | --- | --- |
| APP名稱 | Cheap加油 | 油價公告 | 油價查詢 |
| 加油價格查詢 | ● | ● | ● |
| 下周油價預測 | ● | ● | ● |
| 信用卡優惠 | ● | ● |  |
| 信用卡圖像辨識 | ● |  |  |
| 加油站查詢 | ● | ● | ● |
| 加油站排序 | ● |  |  |

第六章 結論

6.1遭遇問題

6.1.1前端部分

1.相機掃描辨識度無法達到90%以上,原因是因為當在掃描信用卡時,圖片會含有其他的背景,即便進行切割過後還是回殘留一些,這樣會稍微影響到辨識度。

2.由於資安和信用卡資訊安全問題,不使用拍照信用卡回傳後台進行辨識,所以採用掃描的方法,所以OpenCv是放在前端上,所以導致APP的容量過大。

3.UI設計美感不夠,由於本人美感不佳又沒有其他隊友,所以UI畫面不太能夠吸引使用者使用。

6.1.2 後端部分

後端爬蟲目前只爬到加油站資訊和信用卡的加油優惠資訊,但是在加油優惠資訊不夠完整,原因是因為有些折扣訊息用使用圖檔的方式呈現,這樣無法直接使用。所以這裡有使用過OCR(光學文字辨識)文字辨識功能,可是辨識率實在不行,原因是因為中文辨識很難,即便是使用了工具去校正它,辨識率還是相當的低,所以根本無法使用,只能夠過人工的方式去輸入。

6.2未來展望

1.現在機器學習,深度學習和AI已經發展到了一定程度了,現在已經有人把OpenCv跟機器學習結合了,以來提高辨識率,可惜目標題目訂在前端,不然未來如果想要提高辨識率的話可以考慮機器學習。

2.目前後端爬蟲只做有關於加油站的資訊,可是信用卡其實在很多地方也有折扣,例如:電影院,便利超商……之類的,未來如果要擴充的話也是可以。

3.UI介面優化。

第七章 心得

**蔡尚霖:**

一開始接觸這個專題時，知道要寫OpenCv時一度想要放棄，因為OpenCv主要使用C++撰寫，可是我並不會C++爾且還要把OpenCv實作在Android上，讓我覺得非常困難，幸虧OpenCv也有推出專門給Android使用的Library，爾且在導入上也變成比較簡易，由於是專門給Android使用的Library所以可以使用JAVA撰寫。在使用OpenCv實作圖片相似度時，我嘗試過了許多的方法，例如:ORB、直方圖、二值化、腐蝕化、陣點化……之類的，可是等到用相機掃描實作時才發現之前的方法都達不到目標，最後才找到SURF這個演算法來處理圖片相似度的問題，OpenCv可以說是這個專案花費最多時間的部分，不過我認不還是不夠滿意,應該還有許多地方可以優化。

很開心當初可以實作這個專題，因為可以學到很多東西，像是OpenCv、Android和製作流程，雖然我原先就有Android的基礎不過，不過在這個專題上我第一次與後台溝通，知道了前端與後端溝通上的困難和要注意的點，還有學習許多Android實作上的技巧，雖然很可惜沒有找到會寫ios的隊友。在製作這個專題時，經常與老師和學弟一起討論，一起修改需求雖然痛苦但是也是學習到了很多有關流程規劃之類的。

這個專題我原先是要一個人實作全部，不過指導教授幫我找了一個學弟來實作後台，他原先是做大專生計畫之類的，剛好缺少一個做前端Android APP的,所以我們兩個就合作一起實作這個專案，真的非常感謝學弟幫忙製作後端，沒有他這個專題可能很難實作出來。

參考文獻

[1]Android : [https://zh.wikipedia.org/wiki/Android](https://zh.wikipedia.org/wiki/Android%20)

[2]python : <https://zh.wikipedia.org/wiki/Python>

[3]Jupyer : <https://medium.com/@yehjames/%E8%B3%87%E6%96%99%E5%88%86%E6%9E%90-%E6%A9%9F%E5%99%A8%E5%AD%B8%E7%BF%92-%E7%AC%AC1-2%E8%AC%9B-jupyter-notebook%E4%BB%8B%E7%B4%B9-705f023e3720>

[4]MySQL : <https://zh.wikipedia.org/wiki/MySQL>

[5]Surf: <https://blog.csdn.net/qq_31531635/article/details/73798398>

[6]kryo : <https://www.cnblogs.com/hntyzgn/p/7122709.html>

[7]線程池 : <https://juejin.im/entry/593109e72f301e005830cd76>

[8]Flask : <https://zh.wikipedia.org/wiki/Flask>