**世新大學資訊管理學系**

**使用手冊**

**Recycling Robot (R2)**

組長：葉玫筠 (A104222039)

組員：嚴雪綾 (A104222020)

王品云 (A104222026)

林媛宣 (A105222006)

中華民國一○七年十二月

**目錄**

[一、 功能概述 4](#_Toc536640574)

[二、 系統操作環境 5](#_Toc536640575)

[三、 安裝說明 8](#_Toc536640576)

**圖目錄**

[圖一：機器人正面照 6](#_Toc536640577)

[圖二：機器人側面照 7](#_Toc536640578)

[圖三：硬體部屬圖 7](#_Toc536640579)

[圖四：多角度回收物品影像 8](#_Toc536640580)

[圖五：原始影像之相關資訊 8](#_Toc536640581)

[圖六：影像前處理之圖像 9](#_Toc536640582)

[圖七：影像前處理之相關資訊 9](#_Toc536640583)

[圖八：Anaconda IDE環境下載 9](#_Toc536640584)

[圖九：Anaconda環境下的Application 10](#_Toc536640585)

[圖十：存放回收物品圖像的資料夾架構 10](#_Toc536640586)

[圖十一：hfs 工具 11](#_Toc536640587)

[圖十二：putty 遠端管理工具 12](#_Toc536640588)

[圖十三：回收物品影像分類label 13](#_Toc536640589)

[圖十四：GPU硬體資訊 14](#_Toc536640590)

[圖十五：顯示卡驅動情況 15](#_Toc536640591)

[圖十六：CUDA Toolkit 9.0 下載資訊 15](#_Toc536640592)

[圖十七：在Window上下載cuda\_9.0.176\_384.81\_linux.run 16](#_Toc536640593)

[圖十八： 16](#_Toc536640594)

[圖十九：機器學習示意圖 17](#_Toc536640595)

[圖二十：機器學習訓練畫面 17](#_Toc536640596)

**表目錄**

[表一：功能說明表 4](#_Toc536640597)

[表二：軟體規格表 5](#_Toc536640598)

[表三：硬體規格表 6](#_Toc536640599)

[表四：PaddingImage.py程式所需的套件 10](#_Toc536640600)

[表五：本系統所需命令 11](#_Toc536640601)

[表六：檢查GPU資訊 13](#_Toc536640602)

[表七：安裝python 環境與tensorflow、keras套件 14](#_Toc536640603)

[表八：檢查顯示卡指令 14](#_Toc536640604)

[表九：檢查cuda指令 15](#_Toc536640605)

[表十：安裝cuda指令 16](#_Toc536640606)

### 功能概述

實作視覺辨識移動式環保機器人系統，主要是藉由裝設在機器人身上的webcam (攝影機)，拍攝照片後將影像傳遞至樹梅派中的分類器模組，進行被拍攝的單一回收物品的分類辨識；並藉由視覺作為移動和機器手臂的控制、抓取，與此同時，也會將機器人所看見的影像透過nginx網站伺服器將即時監控的影像回傳至我們的管理者網站，為此來達成我們的目的。

以下是四大功能的細項及說明，包含各項軟硬體，如表一所示：

表一：功能說明表

|  |  |  |
| --- | --- | --- |
| 四大功能 | 功能細項 | 功能說明 |
| 機器人控制 | 移動 | 透過攝影機朝特定物品前進，行進時會自動避開障礙物，若是沒有目標物會沿著牆壁移動 |
| 抓取 | 依照物品所擺放的重心，進行抓取 |
| 投遞 | 將手臂抓取到的物品經過分析後投遞到相對應的位置 |
| 視覺辨識 | 位置 | 透過攝影機定位物品 |
| 外觀 | 利用攝影機拍攝到的畫面判斷抓取點 |
| 物品分類 | 無 | 透過機器學習進行回收物的分類 |
| 後台管理 | 系統監控 | 監控機器人 |

### 系統操作環境

系統主要分為影像辨識、機器人控制與後端管理平台三個部分，在Linux的環境下結合開源軟體Python語言進行資料的前處理以及模組的訓練，經過機器學習後，我們會產生一個分類器的模組，將分類器的模組放進Raspberry Pi來進行影像辨識，而機器人控制亦均使用Open Source的軟硬體來進行開發，使用Arduino控制板進行機器人的移動、手臂的抓取、投遞回收物以及超音波等感測器數值回傳，搭配Raspberry Pi獨立的作業平台，內建支援多 USB Port 此主要用於傳遞訊息給Arduino控制板，作為影像與機器人間溝通的依據，機器人前方的攝影機會傳送影像至Raspberry Pi並定時拍照進行影像辨識，其辨識方法以單一物品辨識為主。

後端管理平台我們架設一台網站伺服器，並運用Micro Services的架構，使用Docker技術建立兩台Container來管理不同的服務，分別是系統管理以及帳號管理，而即時監控則要在Raspberry Pi上架設nginx網站伺服器，並在設定檔中加上RTMP的設定，再將後端管理平台與Raspberry Pi做連結便可進行影像的串流。

使用者操控所需的介面，如表二所示：

表二：軟體規格表

|  |  |  |
| --- | --- | --- |
| 軟體名稱 | 格式說明 | 備註 |
| Windows作業系統 | 執行應用程式的作業系統 | Windows 10 中文家用隨機版 |
| Linux 作業系統 | 執行應用程式的作業系統 | Ubuntu 18.04 |
| Arduino IDE | 開源硬體控制程式開發軟體 | 自由軟體 |
| Raspberry Pi作業系統 | 使用於Raspberry Pi控制板 | 自由軟體 |
| Python | 開發軟體 | 配合Keras、Tensorflow等套件 |
| Tomcat Web Server | 開源伺服軟體 | 自由軟體 |
| Notepad++ | Java網頁編輯程式 | 自由軟體 |

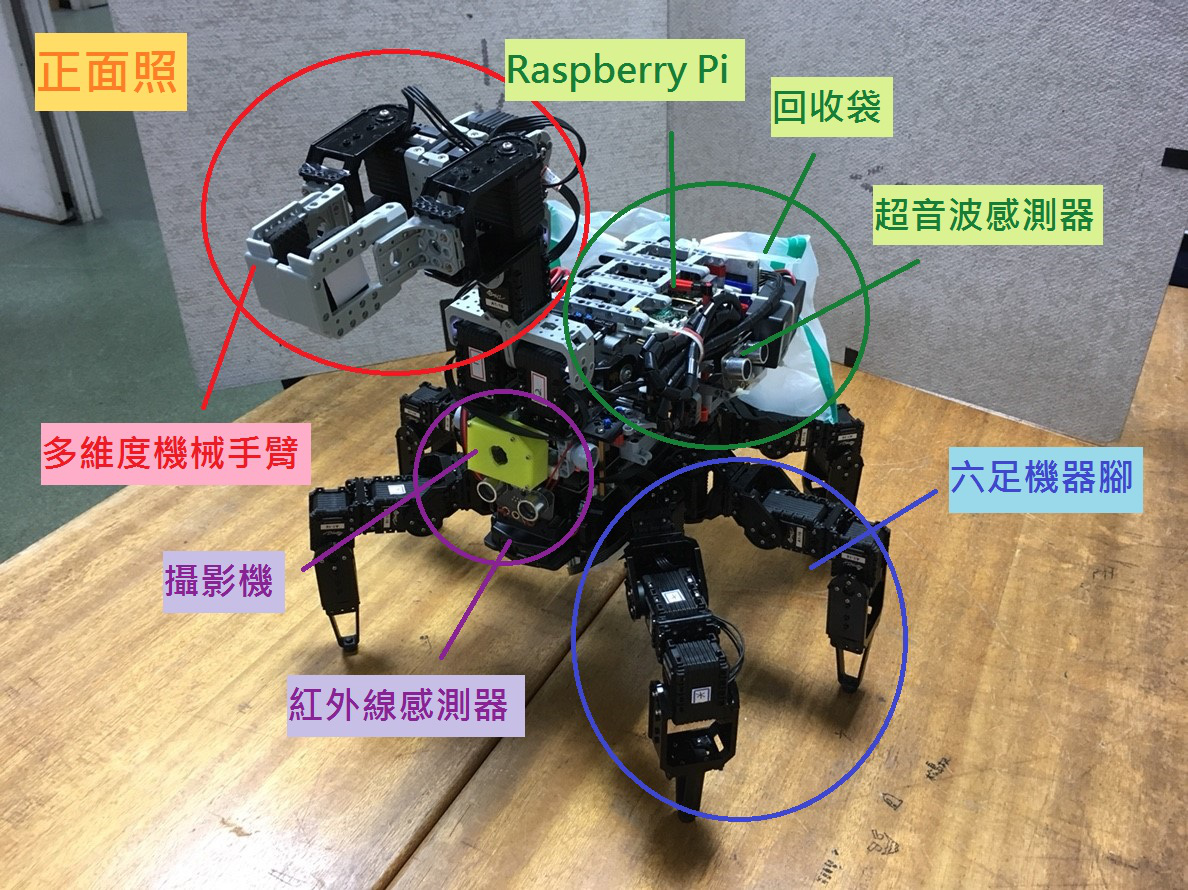
使用者控制機器人的裝置，如表三所示：

表三：硬體規格表

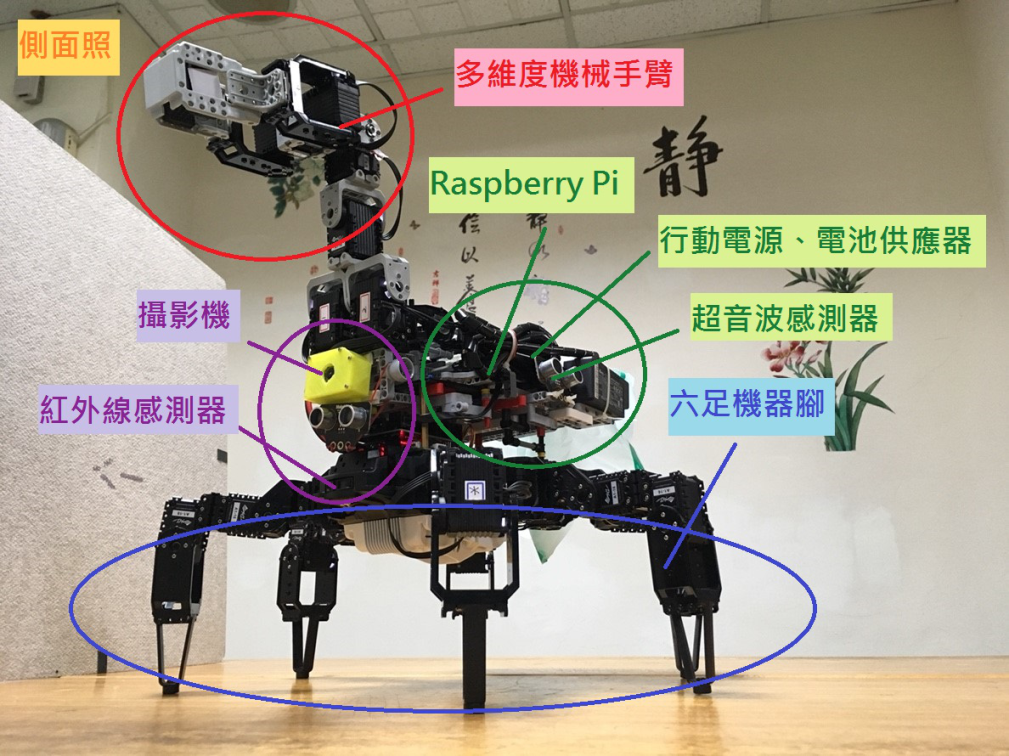
|  |  |  |
| --- | --- | --- |
| 硬體名稱 | 格式說明 | 備註 |
| 客製化六足機器人 | 伺服馬達數個、Arduino、藍牙、紅外線發射器以及接收器與超音波感測器、Raspberry Pi 單板電腦 | 一台，具有辨識物品的攝影機與感測器以及機械手臂可夾取物品 |
| 電腦 | Intel Core i7第四代、RAM 8GB、儲存可用空間20GB以上 | 一台：撰寫硬體程式  一台：伺服器主機  一台：撰寫網頁 |
| 置物袋 | 組合 | 兩組（塑膠類、紙類） |

首先，先介紹機器人中各個重要的部件：

圖一與圖二分別為機器人的正面照與側面照，我們以六足機器人為基底，於上方搭建樂高以固定放置相關的部件，分別是Raspberry Pi、行動電源以及XYZRobot Arduino板、供應XYZRobot Arduino板的電池。在機器人前方的攝影機，用於影像辨識與即時監控，側邊、前方的超音波感測器，用於偵測距離以及避障功能；Raspberry Pi主要用於影像辨識並傳指令給Arduino控制板讓手臂與六足進行動作，回收袋於機器人後方，右邊放置塑膠類、左邊放置紙類，最上方則是機械手臂，用於夾取回收物與投遞至後方回收袋。



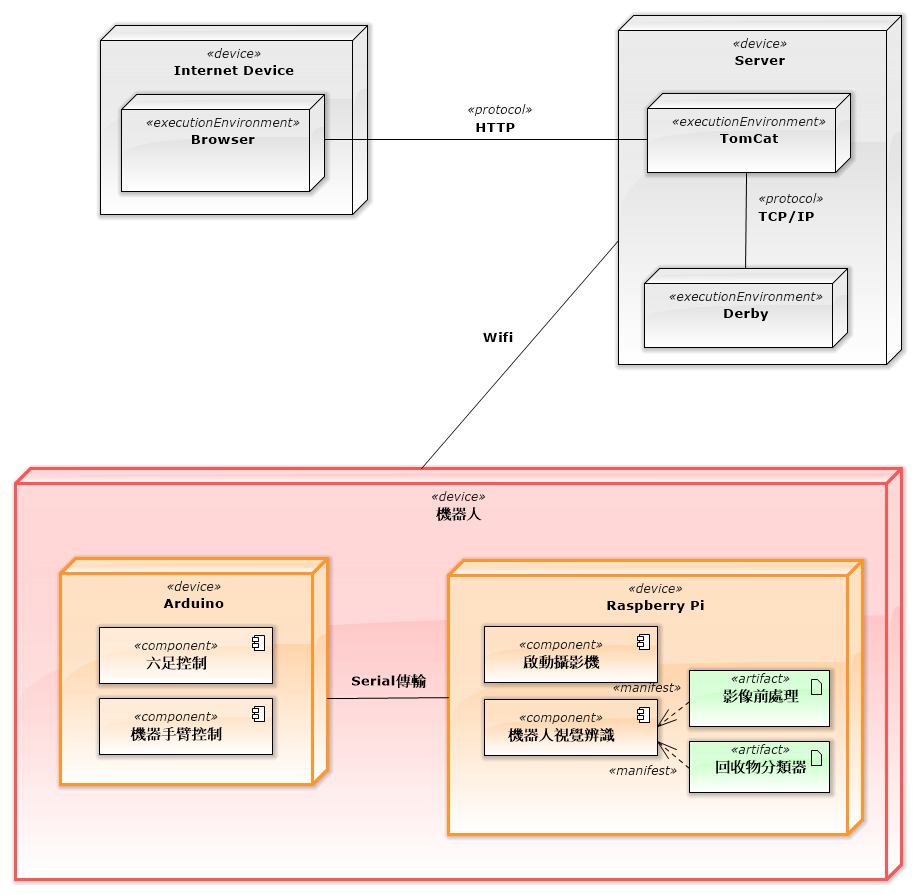
圖一：機器人正面照



圖二：機器人側面照

本系統於Raspberry Pi作業系統Raspbian，執行Python 3.5環境開發之程式。

圖三顯示對外通訊基於Wi-Fi無線網路，如果遇到通訊時常中斷的情況，須增設Wi-Fi基地台加強訊號穩定度。



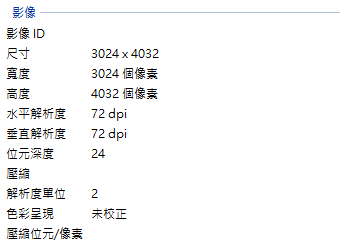
圖三：硬體部屬圖

### 安裝說明

首先利用照相設備(這裡我們使用一台iphone手機)進行多角度回收物(如圖四所示)的拍攝，採用拍攝多角度的回收物影像，目的是希望機器可以辨識出更加多元且更符合實境的回收物品，原始回收物圖像的像素是72 dpi，影像大小為3024 pixel \* 4032 pixel。



圖四：多角度回收物品影像

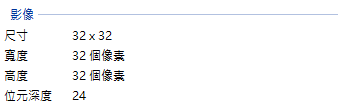


圖五：原始影像之相關資訊

接著將原始照片一共2742張進行影像的前處理(將影像進行resize與padding)後，會產生影像大小為32 pixel \* 32 pixel的訓練用影像集。影像的前處理目的是在將影像送入機器學習前，進行適當的處理。我們是採用CNN的深度學習演算法作為機器學習的訓練模組，因此我們會將原始的長方形影像藉由我們撰寫的影像前處理程式，轉成矩形並變更原始影像的大小為32 pixel \* 32 pixel (如圖七所示)。



圖六：影像前處理之圖像



圖七：影像前處理之相關資訊

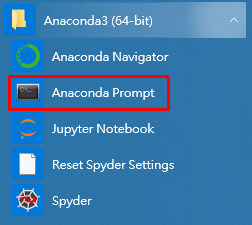
我們是在Windows的電腦內使用Notepad ++文字編輯器撰寫Python的影像前處理程式PaddingImage.py，以下是執行Python程式所需安裝的IDE環境—Anaconda的安裝流程。

Google搜尋Anaconda的官網，下載Anaconda IDE 環境(<https://www.anaconda.com/download/>)，選擇Windows作業系統64-bit的Python 3.7 version進行安裝(如圖八所示)。



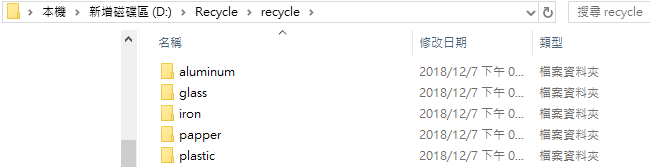
圖八：Anaconda IDE環境下載

Anaconda安裝完成後會，會產生以下幾個Application，包含Anaconda Navigator、Anaconda Prompt、Jupyter Notebook、Spyder (如圖九所示)；在這裡我們使用Anaconda Prompt來執行Python程式。



圖九：Anaconda環境下的Application

將預進行影像前處理的所有回收物品圖像放在D:\Recycle\recycle的資料夾內(如圖十所示)，影像前處理程式PaddingImage.py放在D:\Recycle中，開啟Anaconda Prompt後change directory至PaddingImage.py程式所屬的目錄下來執行程式。



圖十：存放回收物品圖像的資料夾架構

執行影像前處理程式前，需要按照

表四載入本程式所需要的相關套件來運作。

表四：PaddingImage.py程式所需的套件

|  |  |
| --- | --- |
| 名稱 | 指令 |
| from PIL import Image, ImageOps | Built-in |
| import os | Built-in |
| import sys | Built-in |

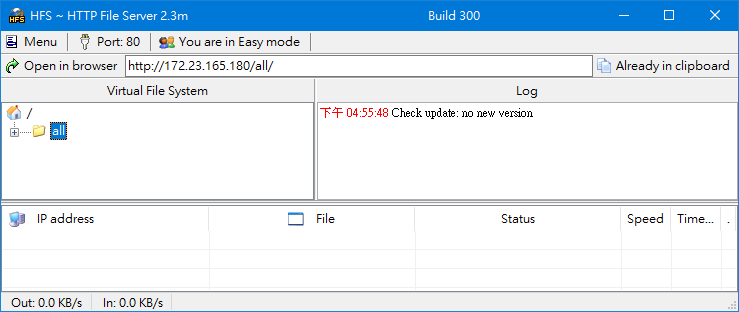
載入所需使用的套件後，即可以在Anaconda Prompt上執行PaddingImage.py程式，將影像進行Padding與Resize，完成我們機器學習訓練用的回收物品樣本。

接下來我們使用實驗室內的一台GPU電腦來作為我們機器學習的設備，我們所使用的作業系統環境為Linux Ubuntu 18.04 Desktop版本，作業系統安裝完畢後，會輸入「sudo apt-get update」與「sudo apt-get upgrade -y」來更新系統的套件清單，之後依照表五安裝本系統所需要的相關命令。

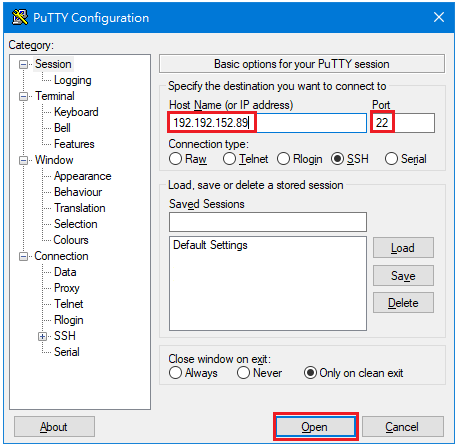
表五：本系統所需命令

|  |  |
| --- | --- |
| 名稱 | 指令 |
| nano | sudo apt-get install nano |
| wget | sudo apt-get install wget |
| net-tools | sudo apt-get install net-tools |
| openssh-server | sudo apt-get install openssh-server |

將準備好的訓練資料集從Windows的作業系統透過hfs (如圖十一所示)這個http file server將資料以此命令「sudo wget http://172.23.165.180/all/」傳輸至Linux的作業系統所指定的目錄中，下載putty工具，用以遠端操作Linux這台電腦使用(如圖十二所示)。



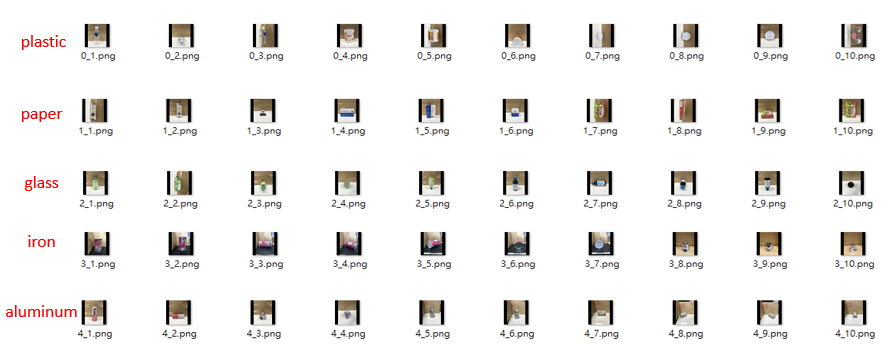
圖十一：hfs 工具



圖十二：putty 遠端管理工具

在Linux電腦中，訓練用的影像集，合計共2742張，存放在/home/rec\_robot/recycle/keras/all的目錄下，而測試用的影像集，合計共150張，存放在/home/rec\_robot /recycle/keras/testing的目錄下。我們拍攝回收物品時，一共分成5個類別，分別有塑膠類(plastic)、紙類(paper)、玻璃類(glass)、鋁罐類(aluminum)、鐵罐類(iron)，同時也分成5個不同的資料夾儲存以上類別的影像。

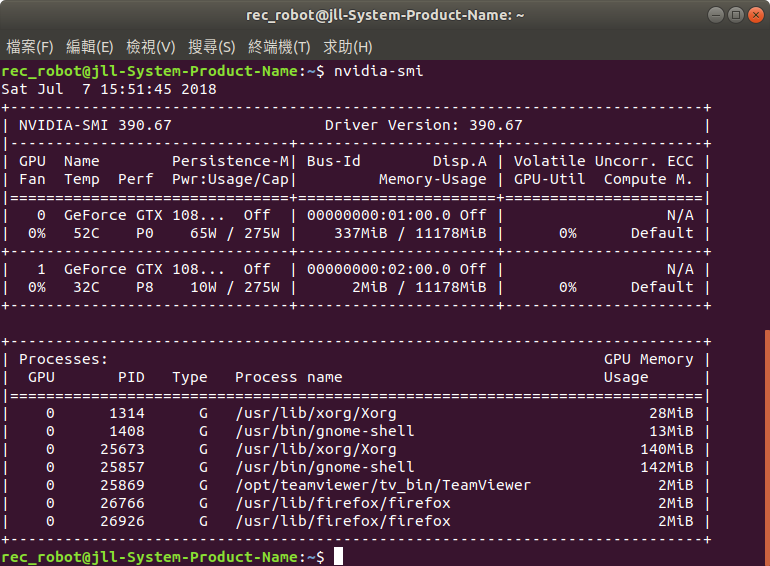
在經過我們的PaddingImage.py程式處理後會根據類別檔案名稱，增加類別label，塑膠類(label 0)、紙類(label 1)、玻璃類(label 2)、鐵罐類(label 3)以及鋁罐類(label 4)，如下圖十三所示。

圖十三：回收物品影像分類label

接著我們需要在GPU (如圖十四所示)的電腦裡安裝使用GPU的相關模組和套件，使我們的程式在執行時可以使用安裝在Linux作業系統中的2顆GPU，而非使用CPU來進行機器學習。透過表六指令的檢查可以知道系統中存在的GPU資訊。

表六：檢查GPU資訊

|  |  |
| --- | --- |
| 名稱 | 指令 |
| NVIDIA System Management Interface  查詢驅動程式與顯示卡的資訊 | nvidia-smi |
| 列出所有插在電腦上的GPU卡 | nvidia-smi -L |



圖十四：GPU硬體資訊

經過以上指令進行GPU檢查後，我們可以開始我們的機器學習訓練，首先根據表七安裝環境所需的python環境與tensorflow、keras套件。Linux內建已經具有python 2.7版本的開發環境；基於我們所使用的tensorflow與keras等套件所需python版本的安裝需求(3.5以上)，因此根據以下表格來安裝、檢查python3的環境。

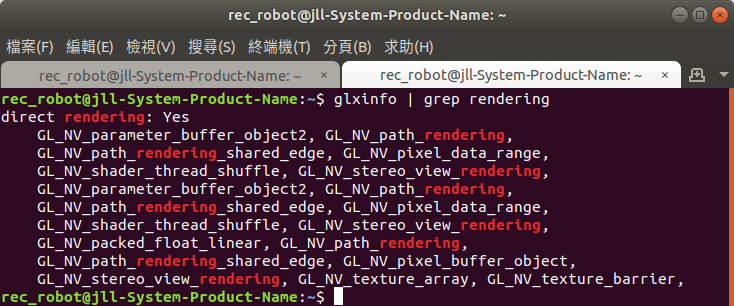
表七：安裝python 環境與tensorflow、keras套件

|  |  |
| --- | --- |
| 名稱 | 指令 |
| 更新現有作業系統的環境 | sudo apt-get update |
| 安裝python 3.6 | sudo apt-get install python3.6 |
| 查看現有python的版本 | python3 --version |
| 安裝pip以及建構python模塊所需的開發工具 | sudo apt-get install python3-pip python3-dev python-virtualenv |
| 列出已安裝的套件 | pip list |
| 查看tensorflow相關資訊 | sudo pip3 show tensorflow |
| 卸載tensorflow (CPU版) | sudo pip3 uninstall tensorflow |

Python 3.6環境已安裝完畢後，我們需要檢查顯卡驅動是否已啟動，顯示yes表示已經正常被啟動，如圖十五所示。

表八：檢查顯示卡指令

|  |  |
| --- | --- |
| 名稱 | 指令 |
| 檢查顯示卡是否已啟動 | glxinfo | grep rendering |

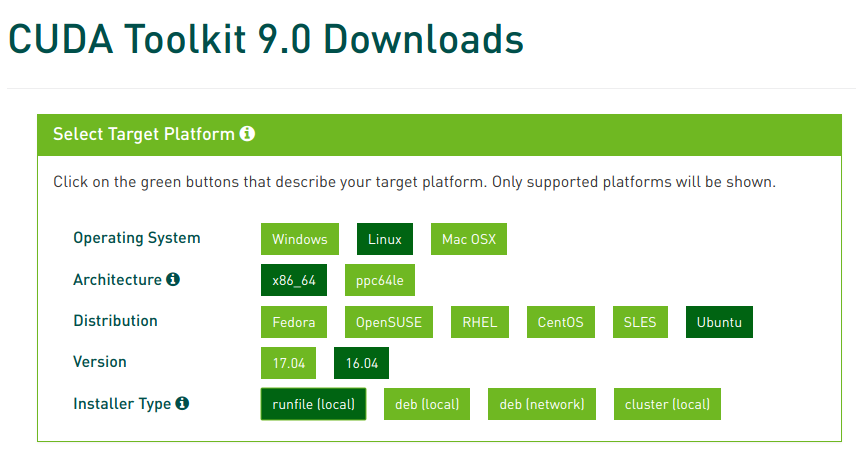
****

圖十五：顯示卡驅動情況

檢查完顯示卡驅動情況後，需要安裝[CUDA Toolkit 9.0](http://nvidia.com/cuda)，安裝方式如下表九所示。

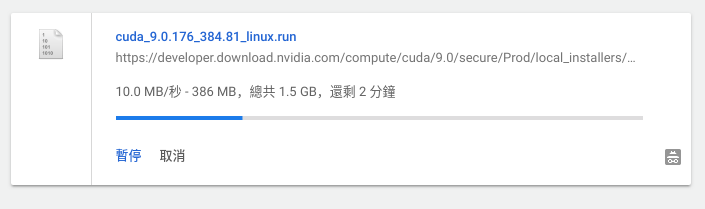
表九：檢查cuda指令

|  |  |
| --- | --- |
| 名稱 | 指令 |
| 查看cuda的版本 | cat /usr/local/cuda/version.txt |
| 查看是否已安裝cuda | cd /usr/local |



圖十六：CUDA Toolkit 9.0 下載資訊

要安裝哪一個版本的CUDA可以上網查詢，我們電腦的作業系統是安裝Ubuntu 18.04，根據官網上的資料顯示，現階段尚未釋出關於Ubuntu 18.04的CUDA支援版本，因此這裡我們選擇Ubuntu 16.04的CUDA版本來操作。

****

圖十七：在Window上下載cuda\_9.0.176\_384.81\_linux.run

將下載下來的檔案cuda\_9.0.176\_384.81\_linux.run放在/cuda的目錄底下後，並執行檔案，指令如下表十所示。

表十：安裝cuda指令

|  |  |
| --- | --- |
| 名稱 | 指令 |
| 創建目錄/cuda | mkdir cuda |
| 執行安裝cuda | sudo sh ~/cuda/cuda\_9.0.176\_384.81\_linux.run |



圖十八：

再使用tensorflow的keras模組，將訓練照片以RGB三個channel的方式放入訓練模組進行機器學習來產生分類器，分類結果包含塑膠類、紙類、玻璃類、鋁罐類以及鐵罐類等五項輸出類別，如圖十九所示，此外最後也會計算出訓練的精準度，如圖二十所示，經過多方嘗試後，每次訓練時間約3分鐘，訓練精準度最高為56%。

Deep Learning

**塑膠類**

**紙類**

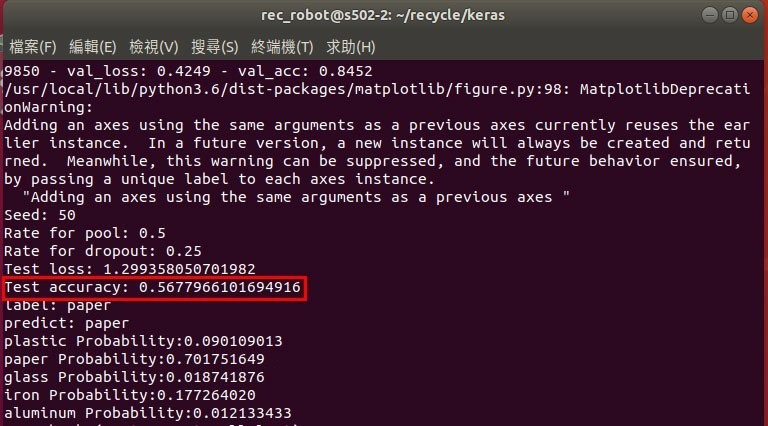
**玻璃類**

**鋁罐類**

**鐵罐類**



圖十九：機器學習示意圖



圖二十：機器學習訓練畫面

攝影機自動拍照

|  |  |
| --- | --- |
| 名稱 | 指令 |
| 安裝fswebcam軟體 | $sudo apt-get install fswebcam |
| 安裝motion軟體 | $sudo apt-get install motion |
| 修改設定檔motion.conf | $sudo nano /etc/motion/motion.conf |
| 修改 /etc/default/motion | $sudo nano /etc/default/motion |
| 啟動motion service | $sudo service motion start |
| 安裝影片製作套件 gstreamer | $sudo apt-get install gstreamer1.0-tools |

在修改motion.conf檔時，需修改以下設定：

* 設定相機自動啟用：將daemon off改成daemon on
* 設定Web連線時使用8081 port：stream\_port 8081
* 設定不從Raspberry Pi連線至Webcam：將stream\_localhost on改成stream\_localhost off
* 設定只能從localhost觀看webcam畫面的限制：將webcontrol\_localhost on改成webcontrol \_localhost off
* 設定管理用的連線使用8080 Port：webcontrol\_port 8080
* 設定Webcam解析度的寬：width 640
* 設定Webcam解析度的高：height 480
* 設定Webcam影像品質（百分比）：webcam\_quality 100

在修改/etc/default/motion時，需修改以下設定：

* 設定自動啟用相機：將start\_motion\_daemon=no改成start\_motion\_daemon=yes

安裝套件且修改完設定檔後，撰寫一份python程式可以直接自動拍照且儲存在固定路徑。

參考網址:

<https://blog.everlearn.tw/%E7%95%B6-python-%E9%81%87%E4%B8%8A-raspberry-pi/%E5%88%A9%E7%94%A8-raspberry-pi-3-model-b-%E8%88%87-usb-webcam-%E9%80%B2%E8%A1%8C%E7%B8%AE%E6%99%82%E6%94%9D%E5%BD%B1>

http://atceiling.blogspot.tw/2014/04/raspberry-pi-webcam.html

http://tekibrain.blogspot.tw/2013/05/raspberry-pi-webcam-motion-webcam-server.html

https://blog.gtwang.org/iot/raspberry-pi-usb-webcam/

https://tutorials-raspberrypi.com/raspberry-pi-security-camera-livestream-setup/

即時監控

|  |  |
| --- | --- |
| 名稱 | 指令 |
| 安裝 nginx 伺服器 | $sudo apt-get update  $sudo apt-get -y install nginx |
| 移除 nginx 伺服器安裝好的套件，並留下系統環境 | $sudo apt-get -y remove nginx  $sudo apt-get clean |
| 移除 nginx 伺服器設定檔 | $sudo rm -rf /etc/nginx/\* |
| 安裝編譯用的套件 | $sudo apt-get install -y curl build-essential libpcre3-dev libpcre++-dev zlib1g-dev libcurl4-openssl-dev libssl-dev |
| 建立系統上放置網頁的目錄 | $sudo mkdir -p /var/www |
| 建立編譯用的目錄，並移動到該目錄 | $mkdir -p ~/nginx\_src  $cd ~/nginx\_src |
| 下載 nginx 與 nginx-rtmp-module 的原始碼 | $git clone <https://github.com/arut/nginx-rtmp-module.git>  $git clone https://github.com/nginx/nginx.git |
| 設定編譯參數(從 nginx 的官方網站下載打包好的原始碼，然後解壓縮) | $wget <http://nginx.org/download/nginx-1.9.3.tar.gz>  $tar zxvf nginx-1.9.3.tar.gz  $cd nginx-1.9.3/  $./configure --prefix=/var/www --sbin-path=/usr/sbin/nginx --conf-path=/etc/nginx/nginx.conf --pid-path=/var/run/nginx.pid --error-log-path=/var/log/nginx/error.log --http-log-path=/var/log/nginx/access.log --with-http\_ssl\_module --without-http\_proxy\_module --add-module=/home/pi/nginx\_src/nginx-rtmp-module |
| 編譯與安裝 nginx | $make  $sudo make install |

安裝完後可以使用$ nginx –v來測試nginx是否安裝成功，若成功會顯示nginx的版本。

再來修改/etc/nginx/nginx.conf 設定檔與/home/pi/nginx\_sr /nginx/conf/nginx.conf設定檔，直接在這兩個檔案最後加上RTMP的設定：

rtmp {

server {

listen 1935;

ping 30s;

notify\_method get;

application rtmp {

live on;

}

}

}

且修改http server listen的埠號為8080

http {

include mime.types;

default\_type application/octet-stream;

sendfile on;

keepalive\_timeout 65;

server {

listen 8080;

server\_name localhost;

location /stat {

rtmp\_stat all;

rtmp\_stat\_stylesheet stat.xsl;

}

location /stat.xsl {

# you can move stat.xsl to a different location

root /usr/build/nginx-rtmp-module;

}

隨後啟用nginx系統服務$ sudo service nginx start，並可以直接在網頁上輸入樹莓派的ip位址以及使用埠號即可觀看即時影像。

樹梅派安裝

|  |  |
| --- | --- |
| 名稱 | 指令 |
| 更新 /etc/apt/sources.list 底下的套件清單 | $sudo apt-get update |
| 比對套件清單決定是否需要更新，但如果要更新的套件有相依性問題，則放棄更新 | $sudo apt-get upgrade |
| 會處理新版本套件與相依性套件的衝突，並試著安裝/移除有問題的套件來完成更新 | $sudo apt-get dist-upgrade |
| 刪除之前因為有相依性而安裝，但現在已經不再使用的套件(非必要) | $sudo apt-get autoremove |
| 清除下載到 /var/cache/apt/archives 的 .deb 套件檔(非必要) | $sudo apt-get autoclean |
| 更新核心和韌體到最新版本(可能不穩定)，因此更新前一定要先備份重要資料 | $sudo rpi-update |
| 安裝藍芽 | $sudo apt-get update  $sudo apt-get install bluetooth  $sudo apt-get install bluez  $sudo apt-get install python-bluez  $sudo python3 -m pip install pybluez  (上面四行裝完還是不行執行這個，python3要先退出再進去一次)  $sudo pip3 install pybluez |

安裝OpenCV

|  |  |
| --- | --- |
| 名稱 | 指令 |
| 擴展分割區 | $sudo raspi-config  Advanced Options > A1 Expand filesystem > Press “Enter”  $sudo shutdown -r now |
| 釋放空間 | $sudo apt-get purge wolfram-engine  $sudo apt-get purge libreoffice\*  $sudo apt-get clean  $sudo apt-get autoremove |
| 安裝Dependencies | $sudo apt-get update  $sudo apt-get upgrade  $sudo apt-get upgrade --fix-missing  $sudo shutdown -r now  $sudo apt-get install build-essential cmake pkg-config -y  $sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev –y  $sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev -y  $sudo apt-get install libxvidcore-dev libx264-dev -y  $sudo apt-get install libgtk2.0-dev libgtk-3-dev -y  $sudo apt-get install libatlas-base-dev gfortran -y |
| 安裝 Python 3, setuptools, dev和Numpy | $sudo apt-get install python3 python3-setuptools python3-dev -y  $wget https://bootstrap.pypa.io/get-pip.py  $sudo python3 get-pip.py  $sudo pip3 install numpy |
| 下載OpenCV 3.4與contrib額外模組 | $cd ~  $wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.4.1.zip  $unzip opencv.zip  $wget -O opencv\_contrib.zip https://github.com/Itseez/opencv\_contrib/archive/3.4.1.zip  $unzip opencv\_contrib.zip |
| 編譯與安裝適用於Python 3的 OpenCV 3.4.0 | $cd opencv-3.4.1  $mkdir build  $cd build  $cmake -D CMAKE\_BUILD\_TYPE=RELEASE \  -D CMAKE\_INSTALL\_PREFIX=/usr/local \  -D BUILD\_opencv\_java=OFF \  -D BUILD\_opencv\_python2=OFF \  -D BUILD\_opencv\_python3=ON \  -D PYTHON\_DEFAULT\_EXECUTABLE=$(which python3) \  -D INSTALL\_C\_EXAMPLES=OFF \  -D INSTALL\_PYTHON\_EXAMPLES=ON \  -D BUILD\_EXAMPLES=ON\  -D OPENCV\_EXTRA\_MODULES\_PATH=~/opencv\_contrib-3.4.0/modules \  -D WITH\_CUDA=OFF \  -D BUILD\_TESTS=OFF \  -D BUILD\_PERF\_TESTS= OFF .. |
| 編譯前交換空間增加虛擬記憶體 | $sudo nano /etc/dphys-swapfile  # CONF\_SWAPSIZE=100  CONF\_SWAPSIZE=1024  $sudo /etc/init.d/dphys-swapfile stop  $sudo /etc/init.d/dphys-swapfile start |
| 開始編譯 | $make -j4  $make clean  $make |
| 安裝與建置 | $sudo make install  $sudo ldconfig |
| 驗證OpenCV的建置 | $ls -l /usr/local/lib/python3.5/dist-packages/  $cd /usr/local/lib/python3.5/dist-packages/  $sudo mv /usr/local/lib/python3.5/dist-packages/cv2.cpython-35m-arm-linux-gnueabihf.so cv2.so |
| 測試OpenCV | $python3  >>> import cv2  >>> cv2.\_\_version\_\_  '3.4.0' |
| 移除壓縮檔 | $cd ~  $rm opencv.zip opencv\_contrib.zip |
| 將交換空間的設定檔還原 | $sudo nano /etc/dphys-swapfile  CONF\_SWAPSIZE=100  # CONF\_SWAPSIZE=1024  $ sudo /etc/init.d/dphys-swapfile stop  $ sudo /etc/init.d/dphys-swapfile start |