實作 OpenCV 於 Android 照相應用

黃俊詠
靜宜大學資訊工程系
e-mail：
s1012581@pu.edu.tw

謝孟諺
靜宜大學資訊工程系
e-mail：
mengyen@pu.edu.tw

陳錫民
逢甲大學資訊工程系
e-mail：
hsiminc@fcu.edu.tw

蘇翊翔
靜宜大學資訊工程系
e-mail:
g1040306@pu.edu.tw

摘要

隨著智慧型手機的蓬勃發展，各式的 App 不斷推陳出新，使用者藉由智慧型手機及搭配其照相 App 即可拍出各式不同效果的照片，使生活上的樂趣大大提升。本研究利用 OpenCV 開發照相的 App，提供使用者即時的影像處理如增加影像亮度、過濾顏色等，藉由即時處理而非後製來達到隨修隨拍的效果。於拍照後更能藉由 WiFi 或行動網路連結到後端 Web Server 或 Facebook 上，上傳照片並進行儲存和分享照片。此論文除了討論 OpenCV 的發展，也進一步強化照相 App 發展的成果。此外本論文亦提供照片瀏覽網頁服務。App 實作的部分畫面，能有效展現出 OpenCV 在 Android 的應用。

關鍵字：OpenCV、Android、Camera

Abstract

Because of the rapid development of smartphone, mobile Apps are advancing daily. Users can create their enjoyments by taking amazing pictures from smartphone camera with the related Apps.. This research is focused on developing a Camera application using OpenCV on Android. The App has the functions of real-time image processing, for examples, light increment and color filtration functions. Users can use the functions during the period of picturing, not after picturing, when they take a picture.After taking a picture, users can upload the photos to a particular website or Facebook for sharing by wireless connection. Besides the descriptions of how to apply OpenCV into the App, this paper also advances the development of camera applications. In additions, a website is proposed to supply users with browsing photos by web browsers. Parts of screen captures of the App are presented in the paper to appear the advantages of OpenCV on Android Apps.

Keywords: OpenCV、Android、Camera

1. 前言

現今人手一支智慧型手機已是十分普遍，智慧型手機已具備多鏡頭，並搭配廠商製作的簡易的照相 App，因此智慧型手機即可成為拍照或者錄影設備。基於照相 App 的流行，本論文發展以 OpenCV[4]為基礎的 Android 照相 App，除了對影像做處理，拍照時能進行影像的追蹤等功能。在拍照後本研究使用 Android 的 HttpURLConnection[6]與已架設好的 Web Server 做連線，提供使用者將拍完後的照片上傳到 Web Server 的空間，同時 Web Server 也提供使用者連線並觀看其所上傳的圖片或照片。

OpenCV 其全名為 Open Source Computer Vision Library，由 Intel 於 1999 年所發起並開發的電腦視覺函式庫，早期 OpenCV 以推進機器視覺之研究以及提供共同的基礎 Library 使程式碼的閱讀能夠更簡易等為目標。其 1.0 的版本於 2006 年時釋出，於 2009 年釋出 OpenCV 2.0，每約半年的時間便會釋出新的正式版本，而目前 OpenCV 已正式釋出 3.0 版本。OpenCV 目前提供許多影像處理的 APIs 供開發者使用，藉由使用各個 APIs 裡面提供的函式可處理包括人機互動、圖像分割、人臉辨識、物體識別等問題。目前在作業系統上面的支援也十分廣泛，可以在包括 Windows、Linux、Android 以及 iOS 等等平台上執行。

在 2011 年，正是智慧型手機開始起飛的時刻，各個手機廠商無不積極的開發相關的手機，當時手上拿的手機，其內建的照相 App 上除了基礎的照相功能外，並沒有什麼其他特殊的功能，不過隨著時間以及硬體的進步，各式的照相 App 相繼出現，如 Cymera[7]以及 Camera360[8]等等，在 2012 年，其除了有基本的拍照功能外也提供照片後製的功能，而隨著功能的增加，其使用也更加的多樣化。而後 Camera360 除了提供後製外，也提供了即時的影像處理，許多的照相 App 著手開發相關的影像處理。Camera360 於最近更加入了雲端的系統，讓使用者能夠拍照並選擇將照片存入由 Camera360 所提供的雲端，藉由將照片存放在雲端上使用者可以不需要將許多的照片、圖片等放置於自身的裝置上，造成存放空間的浪費。在手機的硬體部分，照相手機第一次出現

是在 2000 年由夏普所製造的 J-SH04，開啟了照相手機的時代，而在 2004 年左右，當時照相手機的畫素多為 30 萬畫素到 200 萬畫素。

而後以三星公司開發的手機為例子，其在 2011 年所開發製造的 Galaxy S2 搭載了 8 百萬畫素相機，2013 年所開發製造的 Galaxy S4 搭載 1 千 3 百萬畫素相機，隨時間及硬體技術的成熟，手機的相機持續的在成長，藉由高畫素相機搭配照相 App 所拍出來的畫面也就更加的清楚、動人。

本論文的組織架構如下所述。在第二節介紹一些與 OpenCV 相關的應用及開發。而第三節說明及介紹與本研究相關的 OpenCV APIs 及 App 的相關設計。而在第四節的部分則介紹本研究所實作的案例，並討論相關的實作內容。最後在第五節做整個研究的結論以及未來可能持續進行的方向。

2. 相關應用

目前 App 商店中已有許多照相應用，以下將介紹六個主要相關的應用。

Guardian Angel[1]是一款使用 OpenCV 所開發的嬰兒照顧的智慧型手機軟體，提供的服務包括環境噪音偵測、嬰兒趴睡偵測、空床偵測等等。藉由一支智慧型手機置於旁並與家長手上的智慧型手機做連線，當發生狀況時即告知家長，其中嬰兒趴睡偵測利用 OpenCV 的圓形辨識來判斷偵測到的圓形中膚色所占比例多寡來判斷是否趴睡，當偵測到趴睡狀況時便會以電話鈴聲來通報。而空床偵測則是防止嬰兒從床上離開，以 OpenCV 的圓形辨識及人臉偵測來判別嬰兒是否仍在床上，若不在偵測範圍內即會發出通知，讓使用的家長能獲的訊息得知小孩可能在危險中。

利用 Android 智慧型手機遙控 NXT 樂高機器人。Android 手機加上 OpenCV 的影像辨識功能，讓 NXT 樂高機器人也可以具備 ”視覺”，進行簡單的自主判斷。例如追逐光源，且能夠依照光現的強弱來改變速度、記算和目標物之間的距離，在快碰撞至目標物時，能夠自行停止，改變方向，還有分辨物體的顏色及形狀等。隨著程式複雜度的提升，機器人所能做的事情也越來越多，甚至像俄羅斯方塊這種難度較高的計算，它也能輕鬆達成。

人臉識別功能。利用 OpenCV 技術來鑑別人的臉部特徵，應用包含門禁系統、攝像監視系統、網路應用、學生考勤系統等。此技術已經廣泛使用，隨著智慧型手機的普及，大家對於行動裝置的安全問題也日益重視，開始有人把 OpenCV 臉部識別技術轉移到手機上，透過每個人外觀上的特徵來分辨使用者是否為此手機的所有者，增加安全上的保障。

手勢識別功能。目前 Android 系統人機交互方式是以屏幕觸控為主，比起之前只能透過物理按鍵來操控手機，觸屏明顯方便了不少，但是使用者和行動裝置之間仍然有距離，OpenCV 手勢識別技術讓使用者不需要觸碰螢幕，只要用手勢，想往上就往上，想往下就往下，移動也能用手直接捏取物鍵，這個技術的轉移更加的拉近人機之間的距離。

王煜紘[2]以 Android 系統為平台，藉由 OpenCV 提供的邊緣偵測演算法，針對彎道與直線車道進行辨識，計算出道路中間導引線，開發出行車駕駛輔助系統，駕駛者能夠藉由此系統，隨時注意到自己目前的駕駛狀況，降低危險發生的可能性。

多旋翼機目前多用於影像監控及蒐集，尚俊宏[3]藉由搭配 Android 系統的智慧型手機作為酬載，並且在手機中加入 OpenCV 做影像即時辨識以及影像目標追蹤，透過 WiFi 將即時影像回傳到地面的 Android 裝置上，透過地面上的裝置可自行觸控螢幕選擇欲做追蹤的對象。並藉由每張回傳的影像，即時辨識目標在螢幕上的位置，同時地對旋翼機下達飛行命令，使目標能夠保持在畫面的中心。

3. 系統設計

OpenCV 是一個主要以 C 以及 C++所撰寫的 Library，而目前 OpenCV 也支援了包括 Java 以及 Python。其提供許多與影像處理相關的模組，包括 core、imgproc、highgui、calib3d、features2d 等等，而本研究主要應用到其中的 core 以及 imgproc 的類別部分，實作在 Android 上。

3.1. 資料型態與 Camera 應用

使用 OpenCV 時，必須先了解 OpenCV 的基本資料型態。Mat 是 OpenCV Library 所提供的資料型態，其為一個多維矩陣的陣列。本研究主要使用 Mat 儲存相關的影像，而研究中的影像多以 RGBA 格式來進行儲存。

OpenCV 在使用 Android 裝置的 Camera 上，提供了相關的方法給開發者使用，其中 CameraBridgeViewBase 繼承自 Android 的 SurfaceView，其主要實作的部分為 Camera 與 OpenCV Library 之間的互動。也就是控制裝置

的 Camera 以及相關的處理。由於繼承自 SurfaceView，與 Android 開啟 Camera 相同地也必須覆寫其 onPause、onDestory 等 Class，與 Android 開啟 Camera 不同的在於，除了覆寫上述的 Class 外，還必須覆寫的 Class 包括有，(1)onCameraFrame()-主要用於 Camera Frame 的資料傳輸，(2)onCameraViewStarted()-用於開啟相關的 Camera View 時調用，和(3)onCameraViewStopped()-關閉 Cameara View 時調用。

3.2. Core 類別

OpenCV 的 core 模組主要提供基本的資料結構，而影像便是儲存於這些資料結構中、陣列的處裡、影像的明亮以及像素處理、檔案格式的轉換處理等等。其中提供了包括 Mat 陣列的切割以及合併、計算 Mat 陣列之間的最小值或者最大值、Mat 陣列內的數值重新分配等等。Core 類別裡面的方法包括了:
(1)Core.split(Mat src, List<Mat> dst)-將 src 的資料進行分割後存於 dst 中(2) Core.min(Mat src1, Mat src2, Mat dst):計算比較 src1 及 src2 每個元素的最小值，結果儲存於 dst 中(3)Core.max(Mat src1, Mat src2, Mat dst):計算比較 src1 及 src2 每個元素的最大值，結果儲存於 dst 中(4)Core.merge(List<Mat> src, Mat dst):將 List 的資料做合併後存至 dst 中(5)Core.addWeighted(Mat src1, double alpha, Mat src2, double beta, double gamma, Mat dst):重新分配目標的各個數值間的比例。(src1*alpha)+(src2*beta)+gamma，將其值存於 dst 中。(6)Core.bitwise_not(Mat src, Mat dst):計算 src 內每個位元值，將其作倒置後，其結果存於 dst 中。(7)Core.multiply(Mat src1, Mat src2,Mat dst, double scale):將 src1 與 src2 做矩陣的相乘，後將其值存於 dst 中。

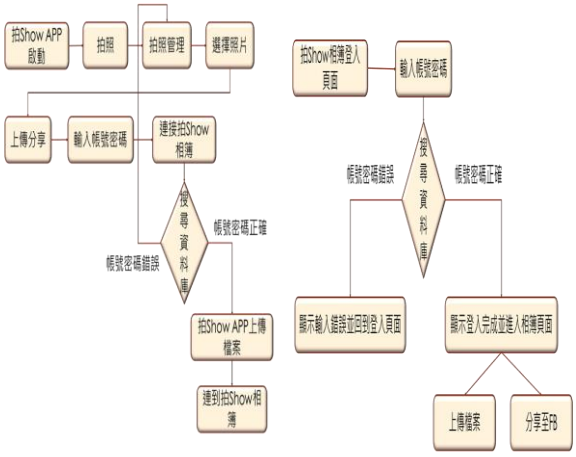
3.3. Imgproc 類別

OpenCV 的 imgproc(image processing)顧名思義就是主要做影像處裡的模組，其提供包括幾何影像的轉換(影像的縮放及翻轉等)、影像的濾波器、影像結構的分析、物件追蹤、特徵偵測等等。其中包括影像的色彩空間轉換、卷積(convolution)應用、高斯模糊、重新調整影像大小等等。類別裡面的方法包括了:(1)Imgproc.cvtColor():轉換影像的色彩空間如 RGB 轉換成 GRAY 等等。(2)Imgproc.filter2D():應用數學的卷積矩陣，將開發者設定好的相關 kernel 效果套用到目標上。(3)Imgproc.resize():對影像的大小做重新的調整。

3.4. 設計理念

一般使用者在使用照相 App 時，拍照是最為重要的，本研究因此將拍照按鈕首要擺於 menu 上，使用者可以在開啟 App 時便能清楚了解何處可以進行拍照。而使用者在使用 App 時也會搭配一些不同的效果來進行拍照，這些效果的按鈕一般來說使用者是不希望擺在畫面中的，故本研究將其隱藏於 menu 中的"..."按鈕中，當使用者希望使用不同的效果時，藉由點選"..."按鈕便能彈跳出各式效果的選項供使用者點選。而在拍完照後，提供使用者拍照後的相關功能。使用者在進行後製時是不需要去注意影像變動的，故設計將所有功能一字排開，使其可以在 menu 上一覽本研究提供的功能。

使用者在開啟本研究的 App 後可以在拍照的同時進行影像處理，在拍完照後使用者可自行選擇是否刪除此照片或選擇裝置內的其他照片，若使用者想要另外儲存在裝置外，本研究也提供上傳至 Web Server 的功能，其流程如圖 1(a)。Web Server 藉由簡單的註冊登入，使用者便能將照片上傳至 Server 上，同時 Web Server 也提供相簿的功能，使用者可以在登入 Server 後上傳電腦或者裝置上的照片或圖片以及觀看自己上傳的照片等，使用者也能透過分享功能將相簿分享到自己的 FaceBook 上供好友觀賞，其流程如圖 1(b)。



(a)照相 app 流程圖　(b)網頁相簿流程圖

圖 1 研究流程圖

圖 2 照相 App 畫面

4. 實作案例

本研究使用 Eclipse 將 OpenCV 應用到 Android 的 App 上對影像做處理，由於 Android 本身並不支援 OpenCV，需先於網路上下載 OpevCV4Android SDK，在開發環境及 Android 相關專案的設定部分也必須做處理，OpenCV 的程式才可寫於 Android 的程式內。

由於 OpenCV 主要是由 C 以及 C++撰寫，而開發者若需要使用到 C 或 C++來撰寫相關程式則 Eclipse 上需要另外安裝 Android NDK(Android Native Development Kit)並在系統的環境變數上新增 Android NDK 路徑才能對 C 與 C++的相關程式做編譯動作。完成後將前面所提到的 OpenCV SDK 資料夾 import 到 Eclipse 中，其內容包括一些 OpenCV 應用的範例以及必要的 OpenCV Library。完成上述步驟後於建置的專案內容中 Properties 的 Library 部分加入方才 import 到 Eclipse 的 OpenCV Library。

4.1 於 Android 專案使用 OpenCV 與利用 OpenCV 開啟 Camera

完成相關的環境設置後，在使用 OpenCV 之前須在專案中撰寫 BaseLoaderCallback 去實作 LoaderCallbackInterface，由於要在 Android 上執行 OpenCV 需要有其相關的 App(OpenCV Manager)，若裝置上沒有此部分則會要求使用者下載安裝 OpenCV Manager。

而 OpenCVLoader 的部分則是做同步 OpenCV 的動作。

```
// The OpenCV loader callback.
private BaseLoaderCallback mLoaderCallback = new BaseLoaderCallback(this)
{
    @Override
    public void onManagerConnected(final int status)
    {
        switch (status)
        {
            case LoaderCallbackInterface.SUCCESS:
                Log.d(TAG, "OpenCV loaded successfully");

                break;
            default:
                super.onManagerConnected(status);
                break;
        }
    }
};
@Override
public void onResume()
{
    super.onResume();
    OpenCVLoader.initAsync(OpenCVLoader.OPENCV_VERSION_3_0_0, this, mLoaderCallback);
}
```

圖 3 BaseLoaderCallback 與 OpenCVLoader

與其他 Android 照相 App 相同的，要在 Android 裝置上使用 Camera 必須在專案的 AndroidManifest.xml 檔案的中寫上可以使用 Camera 裝置的權限，而本研究主要是探討在 Android 上使用 OpenCV 開啟 Camera。

本研究開啟 Camera 的部分以實作 OpenCV Library 所提供的 CameraBridgeViewBase 來呈現。在專案的 MainActivity 需 implements 其底下的 CvCameraViewListener2 或其他與 Camera 相關的 OpenCV Interface。

4.2 使用 OpenCV 過濾影像色彩

在知道如何使用 OpenCV 來開啟 Camera 後。本研究藉由使用 OpenCV 的 Core.spilt Function 對獲取到 Camera 的相關影像的 Mat 做分割，而藉由分割此 Mat 出來的各個 Channel(分割原先的 RGBA)成單一 Channel 做調整便能使影像產生色彩的變化，之後再藉由 Core.merge 將做完調整的 Channels 合併到研究最後要呈現到畫面上的 Ma 中。

以將 Camera 的影像從彩色調整為灰階畫面為例，先藉由前面提到的 Core.spilt 做分割 Channel，由一個型態為 Mat 的 ArrayList 去存放各個 Channel 的值(RGBA)。後使用 ArrayList 提供的 get function 將其中一個 Channel(研究中取得的為 Green,綠色)的值設定給一個 Mat 型態的變數。利用 ArrayList 提供的 set function 將其值設定給另外兩個 Channels，此時三原色 RGB 的 Channels 都設定成同樣的資料後，將其做合併動作。(此處不須對 A,透明度的部分做修改)。其呈現的畫面就如同一般常見的灰階畫面的效果，畫面中所有的物體皆為黑色與白色間的顏色所組成。

```
private final ArrayList<Mat> mChannels = new ArrayList<Mat>(4);
@Override
public void apply(final Mat src, final Mat dst)
{
    Core.split(src, mChannels);
    final Mat g = mChannels.get(1);
    mChannels.set(2, g);
    mChannels.set(0, g);
    Core.merge(mChannels, dst);
}
```

圖 4 彩色轉灰階部分程式

與轉灰階相同地，藉由 ArrayList 取得 RGB 個別的 Channel，並且使用 Core.min 最小化某個顏色如綠色(G)，其結果便顯示綠色的部分被過濾而呈現黑色的狀態如圖 5。

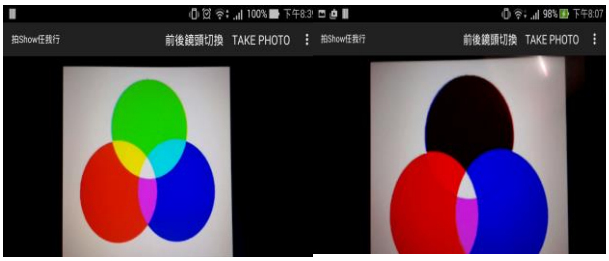


圖 5 過濾綠色對照圖

而使用 Core.max 最大化某個顏色如綠色(G)，其結果則與 Core.min 不同的是其最大化的顏色的對比色受影響而被過濾掉，以本研究為例，綠色的對比顏色為品紅色，其為藍色與紅色的混合。於最大化綠色後，品紅色即被過濾掉形成白色，而紅色與藍色則間接地受到影響分別形成與綠色混合時的顏色，青色與黃色。

本研究除了使用上述最大化及最小化外，也使用 Core.addWeighted 調整其中兩種顏色之間的數值分配，將結果套用回原本的三原色上，藉由數值分配的變動便能改變影像的相關配色，研究以將紅色以及藍色做重新分配為例子，藉由 Core.addWeighted 將其比例改成紅藍各半後，結果分別套用到藍色及紅色上。其結果便呈現藍色與紅色被修改成藍色與紅色混合的顏色，品紅。而間接受到影像地，與綠色混合部分的顏色則呈現淡青色。

4.3 使用 OpenCV 調整曲線與應用卷積進行影像處理

本研究利用曲線的內插函數調整影像的曲線，不過由於 OpenCV 本身並不支援這類的函數，不過慶幸的是 Apache Commons Math Library 有提供，藉由將其 Library 的.jar 檔放入專案的 libs 中便可利用之。先建立一個建構值其參數為 8 個 double 陣列所組成，藉此帶入包括 RGBA 的輸入及輸出數值。而 Apache Commons Math Library 中主要使用到的 Function 包括 UnivariateFunction、SplineInterpolator 及 LinearInterpolator，藉此建立相關的內插函數，再藉由 Mat 的 create 及 Core.LUT 建立查找表，藉由填入不同的數值便可改變查找表的數據，以此改變影像的曲線數值。

圖 6(a) 建立一個 Class，其應用於建立相關的內插函數，由於研究可能輸入多個或一個數據，故先判斷其帶入的 double 陣列中元素個數來決定建立線性或樣條的內插函數。

```
private UnivariateFunction newFunc(final double[] valIn, final double[] valOut)
{
    UnivariateInterpolator interpolator;
    if (valIn.length > 2)
    {
      interpolator = new SplineInterpolator();
    } else {
      interpolator = new LinearInterpolator();
    }
    return interpolator.interpolate(valIn, valOut);
```

圖 6 (a) 建立相關的內插函數

圖 6(b)藉由使用圖 6(a)所建立的相關內插函數，產生曲線的查找表。

```
    // Create and populate the lookup table.
    mLUT.create(256, 1, CvType.CV_8UC4);
    for (int i = 0; i < 256; i++)
    {
        final double v = vFunc.value(i);
        final double r = rFunc.value(v);
        final double g = gFunc.value(v);
        final double b = bFunc.value(v);
        mLUT.put(i, 0, r, g, b, i); // alpha is unchanged
    }
}
@Override
public void apply(final Mat src, final Mat dst)
{
  // Apply the lookup table.
  Core.LUT(src, mLUT, dst);
}
```

圖 6 (b) 影像曲線調整部分 code

完成上述便可開始使用測試，建立繼承自此 class 的另外一個 class，後其內容即為利用方才提到的建構值得部分，以將影像曲線調整到類似於復古的畫面為例：

```
public class TestCurveFilter extends CurveFilter
{
    public TestCurveFilter()
    {
        super
        (
                new double[] { 0, 255 }, // vValIn
                new double[] { 0, 255 }, // vValOut
                new double[] { 0, 54, 100, 180, 255 }, // rValIn
                new double[] { 0, 40, 130, 176, 255 }, // rValOut
                new double[] { 0, 52, 100, 189, 255 }, // gValIn
                new double[] { 0, 40, 120, 196, 255 }, // gValOut
                new double[] { 0, 41, 100, 120, 180, 255 }, // bValIn
                new double[] { 0, 20, 105, 106, 170, 255 } // bValOut

        );
    }
}
```

圖 7 (a)復古模式相關數值

圖 7 (b)復古模式畫面

藉由類似於復古模式的修改方式，另外建立了調整包括整體影像的明亮度、對比度、以及陰影等等的效果。

本研究藉由 Mat 建立相關的 Kernel 效果(參考 wiki)，利用 Imgproc.filter2D 將此效果套用到研究的影像上，這邊須注意，此時的效果

是增加白色的效果於黑色的部分，而這並非研究所要的，故藉由 Core.bitwise_not 做反轉動作，藉此將效果改變成增加黑色的效果於白色的部分。

```
private final Mat Kernel = new MatOfInt
(
        0, 4 , 0,
        4, -16, 4,
        0, 4, 0
);
```

圖 8(a) Kernel 相關數值

圖 8(b) 銳化後畫面

4.4 使用 HttpURLConnection 上傳照片與應用 JQuery 製作滑動相簿

而在拍完照後，本研究也提供上傳功能，藉由 HttpURLConnection 進行與 WebServer 間的資料傳遞，首先透過使用者輸入帳號密碼後先行傳送到 Server 端進行基本的會員認證機制，認證成功後以 intent.getStringExtra 獲取拍照後取得的照片路徑，使用 URL 的 openConnection 開啟與 Server 端的連線，同時設定一些相關的 input、output 設定，後使用 DataOutputStream 進行檔案的寫入，同時藉由 Server 回傳的相關資訊，顯示不同的資訊給使用者知道傳送的狀況如：傳送成功、網路狀況確認、上傳失敗等等。

本研究使用 woothemes 所提供的滑動插件使每張圖片的切換可以產生滑動效果，從其網站中取得 jquery.flexslider-min 後，將其放入 Web Server 的資料夾中，後藉由 JavaScript 的 Function 讀取相關的檔案並做其動畫相關的設定。使用者可以藉由滑鼠點選或者使用鍵盤的向左向右鍵來滑動切換圖片

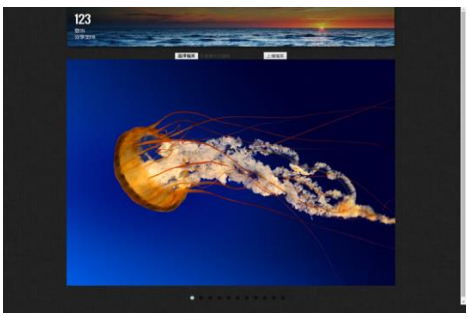


圖 9 相簿主頁面

5. 結論

本研究使用 OpenCV 及 Apache Commons Math Library 開發一套簡易照相 App，使用者藉由此 App 可隨時享受拍照樂趣，使用不同的即時影像處理，拍出獨一無二的照片。除了拍照外，使用者也可使用其行動網路將照片上傳至 Web Server，Web Server 除了供使用者上傳外也包含播放照片及將相簿分享至臉書的功能。未來研究將使用 OpenCV 對 App 做相關的影像處理更新，提供更多不同的即時影像處理使 App 具有更多元的功能，新增如擴增實境(Augmented Reality)、物件辨識、人臉辨識等。另外於 Web Server 端除了提供使用者上傳外，新增供使用者觀看其上傳的照片同時也能從 Server 上將自己的照片下載到裝置上，達到雲端相簿的功能。於此同時隨著時間的前進，OpenCV 正持續的在進行更新，本 App 未來也會持續地跟隨其腳步，提供使用者更多樣化的即時處理。

6. 參考文獻

[1] 王志強、邱奕鈞、吳致宏、陳洳瑾、楊隆翔、盧彥儒，”使用 OpenCV 開發嬰兒安全監控軟體之研究”，OSET2013 自由軟體與教育科技研討會

[2] 王煜紘，”使用 OpenCV4Android 開發車道偏移偵測系統”，

[3] 尚俊宏，” OpenCV 於 Android 系統下實現即時 UAV 地面影像辨識與目標追蹤之可行性研究”

[4] OpenCV，[Online] Available: http://docs.opencv.org/

[5] Joseph Howse，”Android Application Programming with OpenCV3”，2015

[6] HttpURLConnection, http://developer.android.com/reference/java/net/HttpURLConnection.html

[7] Cymera, https://play.google.com/store/apps/details?id=com.cyworld.camera

[8] Camera360, https://play.google.com/store/apps/details?id=vStudio.Android.Camera360