配備手機遙控鏡頭之小型飛行器

A small flying device with a camera remotely controlled by a smartphone

[1] 侯承安　　[1] 黃永廣

[1] Cheng-An Hou　　[1] Wing-Kwong Wong

[1] 雲林科技大學電子與光電工程研究所晶片與電通組

[1] Graduate School of Electronic Engineering, National Yunlin University of Science & Technology, Douliou, Yunlin, 64002, Taiwan

摘要

空中拍攝在現今的小型無人飛行裝置上已有許多的應用，在某些場合下可以節省成本和執行任務期間減少生命損失的風險。本文中，我們使用遠程遙控的四軸直升機，並搭載 ARM 系列開發板控制有伺服馬達的 CMOS 攝像頭與藍芽模組。透過直升機將剛拍下來的照片經由藍芽模組傳送到智慧型手機上，而智慧型手機可以遠端操控 CMOS 鏡頭以便捕捉垂直或是水平的畫面。實地拍攝的結果顯示飛行時的震動會影響拍攝照片與在空氣中傳送照片的品質。如何減少震動並穩定拍攝優質的照片是一個重要的問題，仍有待解決。

關鍵字：小型飛行裝置、空拍、Android 應用、無線影像傳輸。

Abstract

In some applications, aerial imaging with a small unmanned flying device could save flying cost and reduce the risk of life loss during a flying mission. In this paper, we propose to use a remotely controlled quad-motor helicopter with an onboard ARM processor that controls a CMOS camera with a servo motor, and interacts with a Bluetooth module. Snapshots are transferred from the helicopter to smartphone with a Bluetooth module. Smartphone controls the rotation of the CMOS camera in order to capture a picture vertically down or horizontally. Unfortunately, the vibration of the air-borne chopper degrades the quality of the photos. How to reduce the vibration of the chopper for taking good-quality photos is an important issue that remains to be tackled.

Keywords: small flying device, aerial imaging, Android application, wireless image transmission.

1. 前言

近年來的科技蓬勃發展，許許多多的產品也日新月異，從人工到電子化，東西越做越小，功能卻越來越強大，而手機的發展更是極為快速，從普通的接聽電話功能到目前的智慧型手機具有方便的 APP (Application)，加上智慧型手機越來越普及化，幾乎人人一手一台。目前大家熟知的有兩種，一個是伴隨著開放式原始碼與可移植性的 Android 作業系統，另一個則是由蘋果公司所開發的操作系統 iOS (iPhone OS)。在 2012 年 Android 與 iOS 的市佔率高達 87% ，但使用 Google 的 Android 系統的智慧型手機的市佔率更高，反觀蘋果的 iOS 系統卻越來越低迷，因為 Android 有許多特點。第一，Android 是一個開放性程式開發的平台，因此許多學者願意去投入研究，利用此特性與生活周遭的電器產品結合，創造出無限的可能。例如可以用手機去控制家裡的燈，這樣天氣冷也可以不用起床去開關它，或是記錄家中各個電器的瓦特數，這樣就可以很明瞭的知道說那些電器可以適度的調節它，達到省電的作用，還可以用在醫療方面，照顧獨居老人，方便監控他們的行為舉止，將影像傳送到手機上，有太多太多的例子可以舉例了，這也是 Android 的魅力所在。第二，便利性，只要一支手機你就能完成很多事情，從中可以得到更多更快的資訊，提升了人類的生活品質。

現今嵌入式系統[10]開發已廣泛的應用在生活周遭，根據英國電器工程師協會的定義，嵌入式系統為控制、監視或輔助設備、機器的裝置。其優點為高穩定性、應用軟體多、可跨平台、省電及無線通訊軟體成熟，這些技術應用在我們生活當中已經成為不可或缺的部分。而嵌入式系統內的微電腦控制組，透過網路可以輕易的控制遠在天邊的機器，不但節省成本更提高效率。而藍芽雖然速度沒有網路的快速，但是也受到眾多的應用，例如藍芽耳機、點對點的檔案傳輸以及有線裝置無線化，就算沒有網路的狀況下也是可以使用，省電又方便。而這些軟體與硬體的結合，也就是大家俗稱的韌體，都廣泛的運用在我們生活周圍，若是善加利用便能提高生活水準。

本文中所使用智慧型手機 Android 系統與 ARM Cortex-M3 嵌入式開發板的應用，利用四旋翼遙控飛機搭載的開發板、CMOS 鏡頭與伺服馬達，並將影像在開發板上處理後透過藍芽傳送到手機上，或許速度上不是用最快的方式，但仍是值得研究的議題，由於畫質低再加上傳送速率的提高，便能加速了整體的傳送時間。手機方面能弄控制四旋翼上的鏡頭進行上下的移動與畫面的捕捉和接收。

2. 開發平台

2.1. Android

Android 是一個以 Linux 為核心並用 Java 開發為基礎的開放手持設備平台作業系統[7]，具備了許多驅動程式(Drivers)用於控制硬體的工作。在 2005 年 Google 收購了 Android 公司並積極的發展，為了不讓 Android 過度依賴 Linux kernel，便使用硬體抽象層 (HAL)，目的是為了把 Android framework 與 Linux kernel 隔開，以達成「核心獨立」的概念，也讓 Android framework 的開發能在不考量驅動程式實作的前提下發展。也因為它的開放性且提供免費且跨平台的整合開發環境，讓許多學者可以廣泛地開發並應用。

(1) 架構

圖 1，從上到下共分為五種，分別為應用層 (Applications)、應用框架層(Application Framework)、函式庫 (Libraries)、系統執行層(Android Runtime)、Linux 核心層(Linux Kernel)。其中應用框架層提供了完整手機上所有功能支援的 API，如視圖(View)、訊息管理員(Notification Manager)、活動管理員(Activity Manager)等由開發人員直接呼叫使用，極為方便，我們所實作的部分也有應用於此層。而函式庫提供了標準 C 函式庫、OpenGL/ES 等，其中 Java 本身無法直接執行硬體，若想讓 Java 執行硬體必須透過 NDK，而 NDK 是由 C/C++語言編寫的函式庫所組成的，再將 Java 的程式碼轉換成 dex，並使用 Dalvik 虛擬機器去執行。對於開發者來說，必須瞭解這兩大部分才能在行動裝置上設計出得心應手的應用程式。Android 的核心是基於 Linux 2.6 版，主要作為硬體抽象層之目的，包含了安全管理、記憶體管理、網路協定等。但我們只是做應用開發，就不太需要深入了解 Linux Kernel。

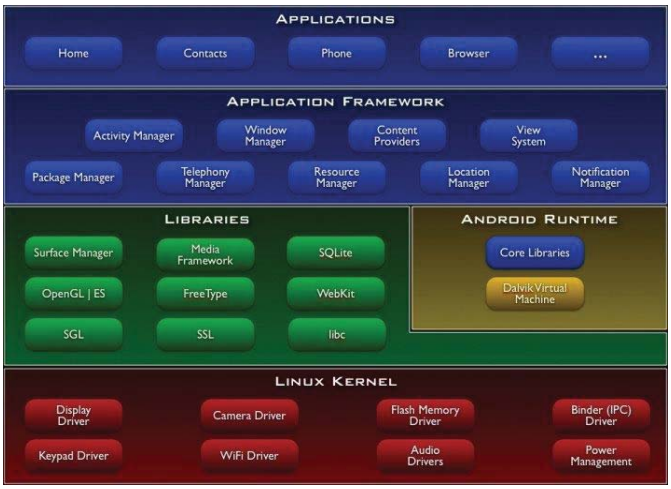

圖1 Android架構圖

(2) 版本

Android 作業系統的軟體圖案是一隻綠色的機器人。在 Android 1.5 版之後，Google 為每個版本上都附上一個甜點系列的英文字母開頭。例如 1.5 版叫做杯子蛋糕（Cupcake）、1.6 版為甜甜圈（Donut）、2.0/2.1 版為閃電泡芙（Eclair）、2.3 版為薑餅（Gingerbread）、3.0 版為蜂窩（Honeycomb）、4.0 版為冰淇淋三明治（Ice Cream Sandwich）等版本[5]。這些是以 A、B、C、D 字頭順序來排列的，十分有趣。

(3) 開發工具

而 Eclipse IDE 開發軟體提供了 Android 所需的程式開發，若安裝額外的外掛模組便能支援各種程式語言。ADT (Android Development Tools)，基於 Eclipse 開發環境的 Android 工具擴充套件，而 Android SDK (Sofewave Development Kit)為 Android 程式開發套件，包含了 Android 模擬器，有了這些開發軟體與工具更能得心應手。

(4) 程式語言

Java 是 Android 重要的程式語言，擁有跨平台及物件導向的特性[3]，1990 年代初由昇陽電腦公司的詹姆斯·高斯林（James Gosling）等人開發。Java 程式語言的風格與 C++語言十分接近，也繼承了 C++語言物件導向技術的核心。Java 語言中改以參照取代 C++語言中容易引起錯誤的指標，改用介面取代多重繼承的特性，同時移除了原 C++與原來運算子多載特性，增加垃圾回收器機制，可將無用的物件所佔用的記憶體釋放並重新被利用。Java 將原始碼編譯成位元組碼（bytecode），並依賴各種不同平台上的虛擬機器來執行位元組碼，實作了「一次編譯、到處執行」的跨平台特性。

2.2. 硬體設備

(1) ARM 開發板與零件

ARM 廣泛地使用在許多嵌入式系統設計，它是基於 32 位元架構的微處理器[6]。由於節能的特點，ARM 處理器非常適合使用在行動通訊領域，例如行動電話、電子產品、遊戲機、多媒體設備等等，從周遭生活到軍用設施，如飛彈、監控儀器都有它的影子。這些產品幾乎都有符合主要的設計目標，為低成本、高效能、低耗電的特性，成為處理器中佔全世界最多數的架構之一，也因為它的使用率極高，故我們也採用 ARM 系列的開發板作為我們的研究之一。此開發板型號為 HY-SmartSTM32，由浩宇(THAOYU)電子所開發，CPU 採用 ARM 系列的，時脈為 72MHz/90MIPS，同時具備了許多 GPIO (General Purpose I/O)腳位，如 UART (Universal Asynchronous Receiver/Transmitter)、PWM (Pulse Width Modulation)等等，重量輕及耗電量低，正是我們所需要選購的門檻之一。鏡頭我們選用了 OV7725 的 CMOS 鏡頭，成本低並省電，電力影響到續航力及穩定性，對我們來說是很重要的關鍵之一。

(2) 藍芽

藍芽技術最初由易利信所開發，用於低功率消耗、低成本的無線通訊方式[8]。與當時的紅外線來比較，藍芽的速度是優於紅外線的，也不需要進行面對面的連線，但當時還不普及，應用裝置也少且價格昂貴。但過了幾年後，藍芽技術越來越成熟，其優點為傳輸時，每分鐘跳頻 1600 次，故資料傳送安全較高，每個封包都在不同的頻率下傳輸，不容易受到環境干擾，且功率低又省電，所以在手機、醫療器材、汽車、PDA 等等方面的應用也增加了不少。此研究也選用了由英棒公司所製造的藍芽 2.1+EDR (Enhanced data rate)，支援 UART 介面，速度上 Baud Rate 可從 1.2k 至 921.6k bps (bits per second)，且距離可在無遮蔽物下達到 100 公尺，當然在耗電量上和重量上也是較低。

(3) 機體

無人飛行載具一直都是很夯的話題,在成本有限的環境下實現遠程遙控並在載具上裝置器材以回報數據，一來不但可以節省人力資源的開銷，二來可降低風險。此研究採用泰世科技 (TSH GAUI)所開發的的 330X 四旋翼產品，之所以用四旋翼是因為它飛行的穩定性其實蠻好的，在開發上的空間也算大，若是用一般的遙控直升機，就只能在腳架的位置架設配備，空間不夠大。

3. 研究方法

此研究共分兩大區塊，藍芽的左半邊為開發板、伺服馬達、鏡頭與機體，右半邊則為手機端，它們之間溝通的橋樑是用藍芽的方式去傳遞，整體的架構圖如圖 2。

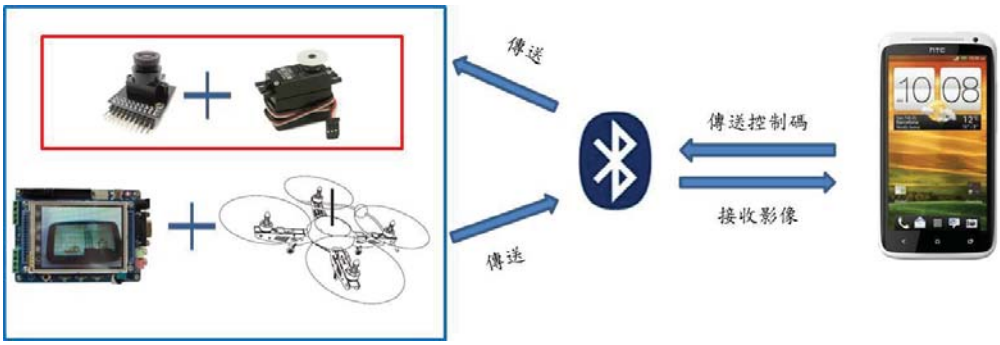

圖2 兩大區塊架構圖

3.1. 開發板與伺服馬達

晶片使用 ARM Cortex-M3 系列的，Device 型號為 STM32F103ZE，使用的編譯軟體為 Keil uVision，版本為 4.14，燒入方式是用意法半導體的 ST-LINK 開發工具，整體的流程圖如圖 3。

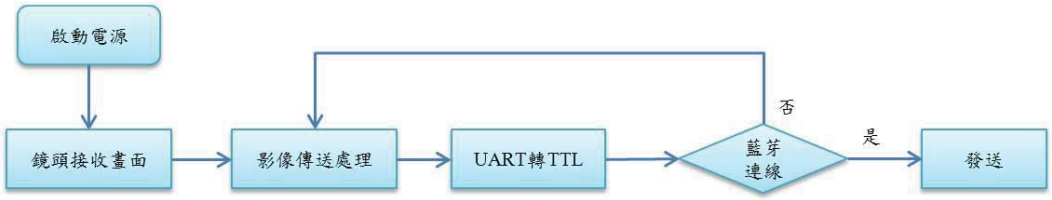

圖3 開發板流程圖

在影像傳輸的部分，則是使用了藍芽的方式去傳遞資料。SmartSTM32 的 UART 接口是 RS232 系列的，跟我們所購買的藍芽 UART 接口是 TTL 系列的不同，所以要透過電壓準位轉換的方式將準位校正到藍芽電壓準位正 3.3V 到正 5.0V 的範圍，為了解決 RS232 電位差與 Bluetooth 模組的問題，因此透過 MAX3232 這塊 IC 來做轉換[12]，轉換之後再將資料由一筆一筆送出。影像本身的解析度大小是 320*240，而每一個點都是由 RGB565 的 16bit 所組成，為了能更快速得到影像，我們直接選用了跳點取值的方法將行列式(1)f(x,y)裡的 x 與 y 的基數排忽略[9]，只選擇偶數排，再將每一個偶數排裡的藍色值擷取出來傳送到手機上，這樣原本有 76800 筆資料，但透過此方法化簡為 19200 筆，為了要讓接收端知道發送端已經傳送完畢了，故在最後一筆資料取代了原有的數值，影像中是絕對看不出來最後一個點有何差異，雖然畫面變得比較不清楚一點，但乃是可見範圍。

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(M-1,0) & f(M-1,0) & \cdots & f(M-1,N-1) \end{bmatrix} \quad (1)$$

而伺服馬達的部分則是由手機發送一筆字元，字元傳送到開發板會轉為十六進位，而這個字元絕對不能與影像傳送的值有所衝突，否則會造成誤判。控制的方向共分三種，為 0 度、45 度與 90 度，並透過 PWM 去控制馬達轉動的角度，方便增加拍攝角度。

3.2. 手機

本論文中使用的 Android 版本是 2.2 以上的，不需要用過高的版本就可以執行了。介面圖如圖 4。

圖4 手機介面

圖 4 裡編號(1)的為應用程式的圖，開啟之後會進入下一個 Activity，即為(2)，等待兩秒之後會再進入下一個 Activity，即為(3)，在這邊做了一個密碼保護的機制，就好像使用者付費的感覺，保障開發者，輸入完畢之後最下面一欄可建立一個資料夾名稱，將影像與照片儲存在建立的資料夾內，之後進入最後一個 Activity，即為(4)，紅色方框才是最主要影像接收的地方，藍色即為拍照顯示的畫面，而中間區域中的各個按鈕，功能依序為拍照、往前預覽，恢復剛剛拍照畫面、往後預覽、連拍與結束連拍。最後，上中下代表的是移動鏡頭的角度。

本文實驗的手機使用的是 HTC OneX，版本為 Android 4.0，搭載四核心處理器，而影像接收的整體流程圖如圖 5。

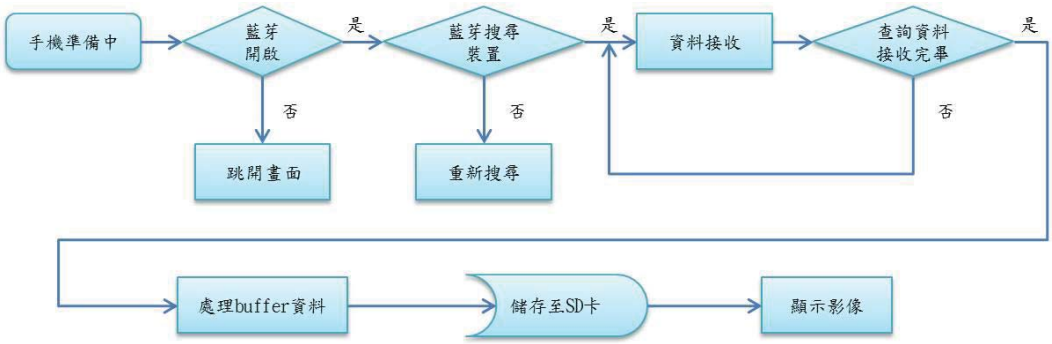

圖5 手機影像流程圖

藍芽連線有四大步驟，設定藍芽通訊、發現藍芽設備並配對、連接藍芽設備與藍芽設備之間資料的傳遞。其中一旦配對設備，藍芽會保存基本資訊(設備名稱、MAC 地址)和允許藍芽通訊 API 讀取相關資料，在成功建立之間的連線後，藍芽設備之間會有一條 RFCOMM 協定的加密管道，而每一個藍芽設備也都有 BluetoothSocket 的 InputStream 和 OutputStream 接口來傳遞資料，透過專門的執行緒(Thread)方式呼叫 read(byte[])和 write(byte[])來讀寫資料流，之後再建立一個 buffer 空間給資料填滿，因此 InputStream 要不斷地讀取資料，維持通訊的順暢。在此我們也改良了前年徐偉翔[4]的交握式 (Handshake)通訊協定[11]做法，Handshake 機制在於主控端必須等待接收端的訊號才會繼續傳送下一筆資料，如果一直等不到接收端發送回來的訊號，發送端就不會再傳送資料，呈現等待的動作。由於等待的空窗期過長，再加上資料在空氣中傳送有可能遺失，導致更新畫面的效率降低。為了改善這個問題，我們移除程式在等待的時間，將程式分成四大步驟，依序為接收資料、判斷訊號、處理資料與顯示畫面，而接收部分也改用輪詢的方式查詢資料內是否已經接收完畢，等接收完畢之後再使用 Android API 裡的 setPixel(int x, int y, int color)方式去填滿 160*120 的大小[1] [2]，最後將填滿的 Bitmap 儲存為 JPG 檔再顯示出來。此方法不用再等待接收端做完整個程式後再發送訊號，可以不斷的接收發送端所傳送的資料，降低程式在等待訊號的空窗期。

從手機開始接收一張圖片資料到接收完畢後所花費的最少時間為 0.003 秒，最多為 0.032 秒，平均時間為 11.2928 ms，即為 0.0112 秒多。而接收完資料處理後顯示圖片的時間最少為 0.09 秒，最多為 0.18 秒，平均時間為 135.8214 ms，即為 0.135 秒多。顯示的張數一分鐘平均為 135 張，即為一秒 2.25 張左右。本論文實作的結果圖如圖 6。



(a)鏡頭眺望操場　　(b)電扇與櫃子　　(c)水族缸

圖6 實體影像結果圖

圖 6 的(a)拍攝距離稍遠，故畫面稍微模糊，(b)與(c)是近距離拍攝，可以很清楚的知道畫面的重點。

4. 結論與未來展望

本論文所使用的 Bluetooth 無線傳輸速度設定為此模組最高包率 921.6k bps，若是使用更高級的藍芽 3.0+HS 版本，最快速度可高達 24Mbps，必定可以大幅提升傳輸速度。但空氣中與在飛行的過程中有些許不穩定的因素，例如飛行高度受限於 100 公尺以內、手機與飛行器的距離要保持在 80 公尺內、操作者控制的熟練度與飛行時震動會使線路彼此交互影響，導致畫面有偶爾短暫的異常，這是我們需要克服並改進的地方。我們也參考了前年徐偉翔的做法[4]，並加以突破改良。未來會加強線路的穩定性與畫面品質，並加速畫面更新的速率。

5. 致謝

感謝國科會科教處支持本研究，計畫編號為 NSC 101-2511-S-224 -001 -。

6. 參考文獻

[1] 余志龍、陳昱勛、鄭名傑、陳小鳳、郭秩均，Google Android SDK 開發範例大全 2，民國 99 年 2 月。

[2] 林城，Google Android 2.X 應用程式開發實戰，民國 100 年 2 月，初版。

[3] 洪維恩，Java 2 教學手冊 JDK 5/6，民國 100 年 2 月，第四版。

[4] 徐偉翔，「應用 SOPC 軟硬體協同設計於影像追蹤」，民國 101 年 6 月，頁 P43~P45。

[5] 蓋索林，Google ! Android 2 手機應用程式設計入門第三版，民國 99 年 9 月，三版 6 刷。

[6] ARM architecture, http://en.wikipedia.org/wiki/ARM_architecture。

[7] Android Developers, http://developer.android.com/reference/android/graphics/Bitmap.html。

[8] Bluetooth, http://en.wikipedia.org/wiki/Bluetooth。

[9] Determinant, http://en.wikipedia.org/wiki/Determinant。

[10] Embedded Systems, http://en.wikipedia.org/wiki/Embedded_system。

[11] Handshake,http://www.symantec.com/zh/tw/security_response/glossary/define.jsp?letter=c&word=chap-challenge-handshake-authentication-protocol。

[12] MAX3232, http://www.pdf.la/MAXIM/MAX3232_detail.html。