第四章 研究與實做方法

在上一個章節中，我們介紹了整個遠端監控系統在 PDA 端上實現的基本架構。本章節中，我們將對此架構的實作方法做詳細的介紹，讓讀者能對此系統有更深入的了解。我們主要分四個部分來探討實作的細節，第一節將探討 SIP-Cam 的 Viewer 在 PDA 端的實現方法，第二節將探討如何在 PDA 端與 SIP-Cam 端加入 SIP 的功能，第三節將對使用者介面做簡單的介紹，第四節則針對解碼速度較慢的 PDA 提出一套 DropFrame 的機制，以避免因為解碼速度過慢導致系統的不穩定或者故障。

這個系統的實作上，主要是透過修改、整合開放原始碼的專案(Open Source Project)來達到系統的主要功能。SIP-Cam Viewer 的部分，我們使用 MPlayer 這套開放原始碼的程式。SIP User Agent 的功能，我們則使用 Sipsak 與 Pjsip 這兩套 SIP 的開放原始碼程式。下面的章節，也會對這幾個重要的開放原始碼套件做簡單的介紹。

4.1 SIP-Cam Viewer 的實現

SIP-Cam 的是透過 HTTP 的通訊協定將壓縮好的 MPEG-4 串流傳送給接收端。接收端如果要正確的觀看或監控到 SIP-Cam 所擷取的影像，就必須正確的解碼 SIP-Cam 的 MPEG-4 串流，但是 SIP-Cam 的 MPEG-4 串流和一般標準的

MPEG-4 有些不太一樣的地方。這個差異點主要是 SIP-Cam 的 MPEG-4 在檔頭(Header)的資訊中多加了自己定義的 Prediction Bits。換句話說，如果要以標準的 MPEG-4 Decoder 解碼，就必須對這些 Prediction Bits 做處理，也才能利用開放原始碼的程式來達成我們的目標。

SIP-Cam 的 Viewer 是將 MPlayer 修改並且移植到 KS_PXA270 開發板並且再以一個簡單的使用者介面整合 SIP 功能。MPlayer 在 Viewer 中扮演最重要的角色，因為 MPlayer 可以解碼標準的 MPEG-4 影像，負責解碼與顯示 SIP-Cam 影像的功能。

MPlayer 是一個免費、開放原始碼的多媒體播放器。MPlayer 可以在相當多的系統平台上播放，例如 Linux、Mac OSX、Microsoft Windows。MPlayer 可以播放的影像格式相當多，他可以播放 MPEG2、AVI、RealMedia、ASF、MPEG-4 等檔案格式。此外，MPlayer 有個特點，MPlayer 可以支援廣泛的 Video Output Driver，包括 X11、OpenGL、fbdev…等等。

我們會選擇 MPlayer 當作是開發個工具，除了 MPlayer 是一個功能廣大的多媒體播放器，能夠滿足我們必須解碼 MPEG-4 影像格式的需求，另一個特點就是因為他能夠支援 fbdev 的影像顯示。因為在 PDA、Smart Phone…等的手持式設

備上不可能支援 X11 這類龐大的圖形函式庫，一般的嵌入式系統都是直接透過 framebuffer(framebuffer 代表一些視訊硬體的 frame buffer，並且允許應用層的軟體透過一些定義好的介面來對顯示卡做存取)來顯示圖形。換句話說，在我們的 PDA 開發板(KS_PXA270)上，我們可以使用 MPlayer 播放 MPEG-4 影像。

在先前的介紹中曾提到，SIP-Cam 的 MPEG-4 在檔頭資訊中加入了自己定義的 Prediction Bits，因此我們必須對這些 Prediction Bits 做處理。Prediction Bits 的處理方式有兩種，原先我們以播放端的角度去解決這個問題，所以在 MPlayer 的解碼過程中，我們會直接忽略掉 Prediction Bits，這樣的做法雖可以正常播放 SIP-Cam 影像，但後來發現穩定性不足。後來我們改以 SIP-Cam 的角度去解決，在 SIP-Cam 編碼時，我們把 Prediction Bits 的定義除去，不但能正確解碼，比起原先的做法，穩定度也更高了。

在我們的架構中，因為 PDA 的 CPU 能力明顯比不上一般 X86 電腦，所以在監控 SIP-Cam 影像時，解碼速度一定會比較慢，很可能因為來不及解碼而使得 MPlayer 以及 TCP 的緩衝區過飽和。這將有可能造成 SIP-Cam 壓縮的資料無法及時傳送給 PDA 端播放，而影響影像的品質或者 SIP-Cam 的穩定度。為了避免此類情形的發生，我們提出了一個 FrameDrop 的機制，希望能夠透過 Drop 部分 Frame 的方法提高整個遠端監控系統解碼的效率。這個 FrameDrop 的機制，

我們將在 4.4 節做詳細的介紹。

以下則為整個 SIP-Cam Viewer 架構的圖示(圖 4.1)與解釋 ：

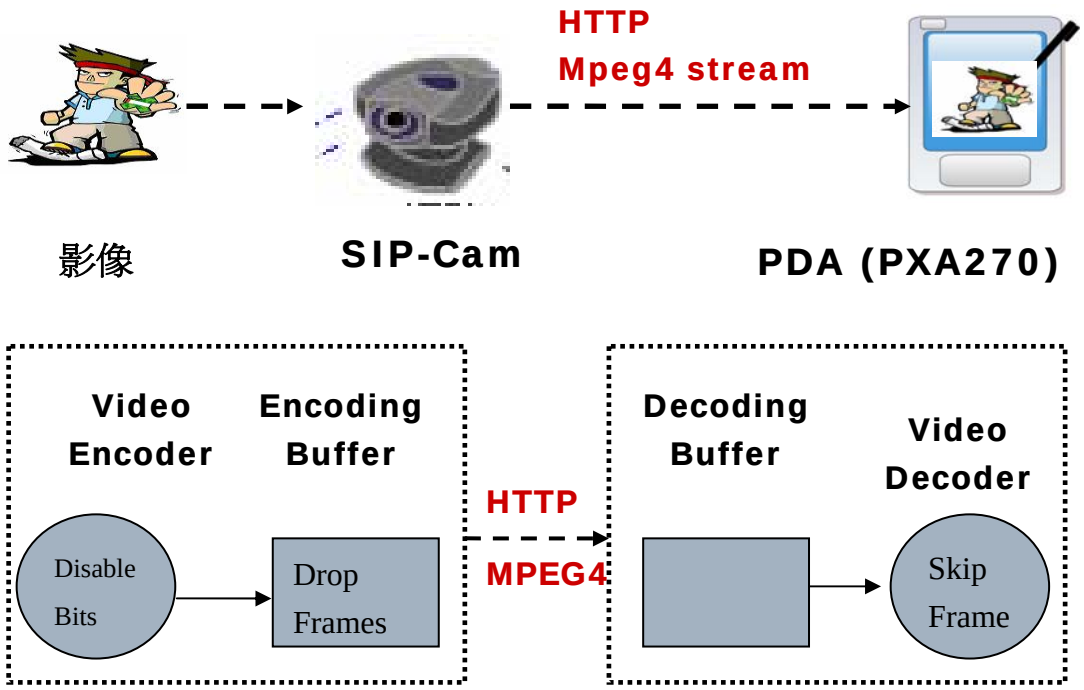

圖 4.1

SIP-Cam 擷取影像後，SIP-Cam 會將影像進行壓縮的動作，壓縮成 MPEG-4 的檔案格式，儲存在緩衝區中，當 PDA 端想要監控 SIP-Cam 的影像時，SIP-Cam 就會透過 HTTP 的通訊協定將壓縮好的 MPEG-4 影像傳送到 PDA 端，PDA 端會先將接收到的資料先存在一個緩衝區中，再將存在緩衝區的影像資訊進行解壓縮的動作。這整個過程中，編碼端編碼時會取消 Prediction Bits 的定義，編碼出來

的影像就不再含有 Prediction Bits。此外，SIP-Cam 會提供四種不同程度的 FrameDrop，以減輕 PDA 解碼時的負擔。PDA 端也會依據解碼緩衝區(Decoding Buffer)的情況，適時地放棄解碼部分的 Frame，提高解碼的效率。SIP-Cam 與 PDA 的 FrameDrop 機制，我們將會在 4.4 節再做詳細的介紹。

4.2 SIP 功能的實現

由前一章的系統架構可知，以 SIP 帳號取代 IP 位址是我們的目標之一。整合 SIP 的功能，PDA 端可以在不知道 SIP-Cam IP 位址的情況下，依然進行影像監控的動作，有效的避免了因為 DHCP 可能造成 IP 位址變動所帶來的麻煩與困擾。SIP 功能的實作上，我們使用 Sipsak 與 Pjsip 這兩個開放式原始碼程式，並加以修改，而達到我們系統架構上的需求。

Sipsak 是一個簡單的 SIP 測試工具。主要是透過指令控制模式(Command line)進行簡易的測試。Sipsak 可以對一些 SIP 的程式(Application)或者設備(Device)做一些簡單的量測，例如，我們可以利用 Sipsak 測量 SIP User Agent Client 對 SIP Server 進行註冊所花的時間。Sipsak 是個簡單的 SIP 測量工具，而且是開放原始碼的程式，我們可以很容易的修改 Sipsak，使修改過後的 Sipsak 具有 SIP User Agent Client 的功能。

Pjsip 也是開放原始碼的 SIP 程式，可以在許多平台上運行，例如 arm、i386、mips 等。Pjsip 容易使用，而且執行效率好，適合當成 Server 使用。此外，Pjsip 比起 Sipsak 有更完整的功能，我們可以利用 Pjsip 作為 SIP User Agent Server，可以監聽 SIP User Agent Client 所發出的邀請，使得 SIP 的功能整合上更加完善。

在我們的架構中，PDA 與 SIP-Cam 都要有 SIP User Agent 的功能。PDA 需要具備的是 Client 的功能，而 SIP-Cam 需要的則是 Server 的功能。PDA 端，我們將 Sipsak 加入 SIP INVITE 的功能，再利用使用者介面與 Viewer 整合，使得我們的 PDA 具備 SIP User Agent Client 的功能。SIP-Cam 端，我們將 Pjsip 移植到 SIP-Cam 中，再修改部分已定義的常數，讓 Pjsip 能夠在我們的架構中運行。如此，我們的 SIP-Cam 就成了一個 Daemon，可以監聽 SIP User Agent Client 所發出的邀請。

PDA 與 SIP-Cam 都具有 SIP User Agent 的功能時，PDA 端就可以透過 SIP Server 向 SIP-Cam 端發出一個 INVITE 的動作，而 SIP-Cam 端如果收到來自 PDA 端的 INVITE 要求時，也可以透過 SIP Server 回覆一個 200 OK 的訊息給 PDA 端，代表會談已經成功被開啟。下圖 4.2 即為 PDA 端與 SIP-Cam 端的連線流程:

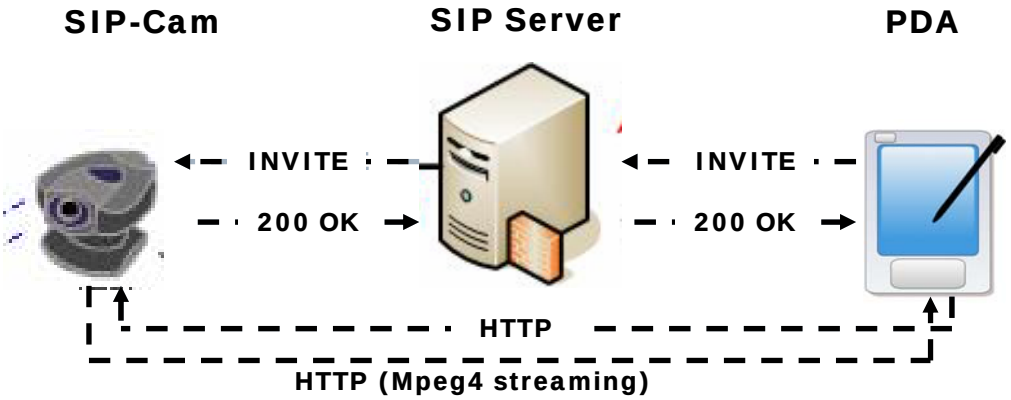

圖 4.2

當 PDA 端透過 SIP 的架構向 SIP-Cam 進行影像的監控時，PDA 端會先透過 SIP Server 對 SIP-Cam 的 SIP 帳號發出 INVITE 的要求，當 SIP-Cam 接收到來自 PDA 端的 INVITE 要求時，也會透過 SIP Server 向 PDA 端回覆 200 OK 的訊息，代表 SIP-Cam 接受這個請求。就在這些訊息的傳遞中，我們可以藉由訊息的內容得到 SIP-Cam 的真實 IP 位址資訊。PDA 端得到 SIP-Cam 的真實 IP 位址後，可以利用這個 IP 發出一個建立 HTTP 連線的請求，以進行影像的監控。當 SIP-Cam 收到這個 HTTP 連線建立的請求後，便可利用 HTTP 的連線把影像的資訊傳送給 PDA 端。

4.3 User Interface 的整合

使用者介面可以讓使用者容易了解系統的操作，降低使用者需要具備的知識門檻。本系統中，我們就利用使用者介面整合 Viewer 與 SIP 的功能，除了讓架

構更完整，也讓使用者能簡單的使用我們的介面

第三章的系統架構中，曾經提到我們的 PDA(PXA270 開發板)使用的圖形化介面是 QT/Embedded。因此，在使用者介面的整合上，我們是透過 QT/Embedded 的函式庫實現視窗化的介面，以方便使用者使用我們的系統。

第三章所提到的系統架構有兩種，第一種架構是在使用者已知 SIP-Cam IP 位址的情況下，PDA 的使用者直接以知道的 IP 位址作為連線的依據。第二種架構則是使用者不知道 SIP-Cam 的 IP 位址，則 PDA 的使用者透過 SIP Server 向 SIP-Cam 的 SIP 帳號發出 SIP INVITE 的要求，而達到連線監控的目的。因此，在我們的使用者介面設計上，允許使用者自由選擇任何一種模式進行影像監看的動作。接著我們將對介面的操作上做簡單的敘述。

圖 4.3

圖 4.4

上圖 4.3 與 4.4 是我們的使用者介面，使用者可以透過選擇【IP Address】或者【Sip Account】的 TAB 頁面切換不同模式的輸入畫面。

由圖 4.3 我們可知，如果 PDA 的使用者已經知道 SIP-Cam 的 IP 位址為 140.122.185.205 時，我們可以選擇【IP Address】這個 TAB 頁面，然後在輸入格輸入這個已經知道的 IP 位址，確定以後再按下【OK】的按鈕，即可進行影像監看的動作，萬一輸入錯誤的時候，則可以利用【Clear】的按鈕清除錯誤的 IP 位址。

而圖 4.4 則是另外一種模式，當使用者不曉得 SIP-Cam 的 IP 位址時，我們可以利用 SIP-Cam 的 SIP 帳號，進行影像監控的要求。使用者如果想使用這個模式，首先必須要先選擇【Sip Account】這個 TAB 頁面，接著在輸入格的地方輸入 SIP-Cam 的 SIP 帳號(圖 4.4 的例子，為 1111)，最後再按下【OK】的按鈕進行影像的監看，相同地，一旦使用者輸入錯誤，也可以透過【Clear】的按鈕清除錯誤的輸入，然後再重新輸入正確的帳號。

4.4 FrameDrop 的機制

透過之前的介紹，我們可知 SIP-Cam 的 Viewer 解碼時，會先把 SIP-Cam 傳送過來的影像存在解碼緩衝區(Decoding Buffer)中，再進行解碼、顯示的動作。

因為在我們的架構中，我們使用 PDA 作為監控影像的平台。PDA 雖然擁有高移動性的優點，卻也存在著低運算速度的缺點，導致解碼速度比較慢。解碼速度太慢可能會導致緩衝區堆積太多未解碼的影像而造成無法繼續接受新的影像，使得 SIP-Cam 已經壓縮好的影像無法傳送到 PDA 端，而影響到影像監控的品質。為了改善 PDA 解碼速度不足造成的影響，我們提出一套 DropFrame 的機制，希望能改善這個問題。

在 SIP-Cam 端，我們有四種等級的丟棄封包動作，由丟棄封包的多寡，分成 Level1、Level2、Level3 以及 Level4。丟棄封包則以影像資訊較不重要的 P-Frame 為優先，以下則是四種等級的封包丟棄法則 。

1. Level1 ： 代表解碼端速度良好，我們不會進行任何封包的丟棄。如下圖 4.5 所示，SIP-Cam 端會把所有的影像資訊傳送到 Client 端。
2. Level2 : 代表解碼速度尚可，我們會丟棄一半 P-Frame。如下圖 4.5 所示，SIP-Cam 端會將 8 個 P-Frame 的前 4 個 P-Frame 丟棄。
3. Level3 : 代表解碼端速度不足，我們會丟棄全部的 P-Frame。如下圖 4.5 所示，SIP-Cam 端只會傳送 I-Frame，而把全部的 P-Frame 都丟棄。
4. Level4 ： 代表解碼速度很差，我們除了丟棄 P-Frame 外，還會丟棄部分的 I-Frame。如下圖 4.5 所示，SIP-Cam 除了丟棄所有的 P-Frame 外，每

2 個 I-Frame 還會丟棄 1 個 I-Frame。

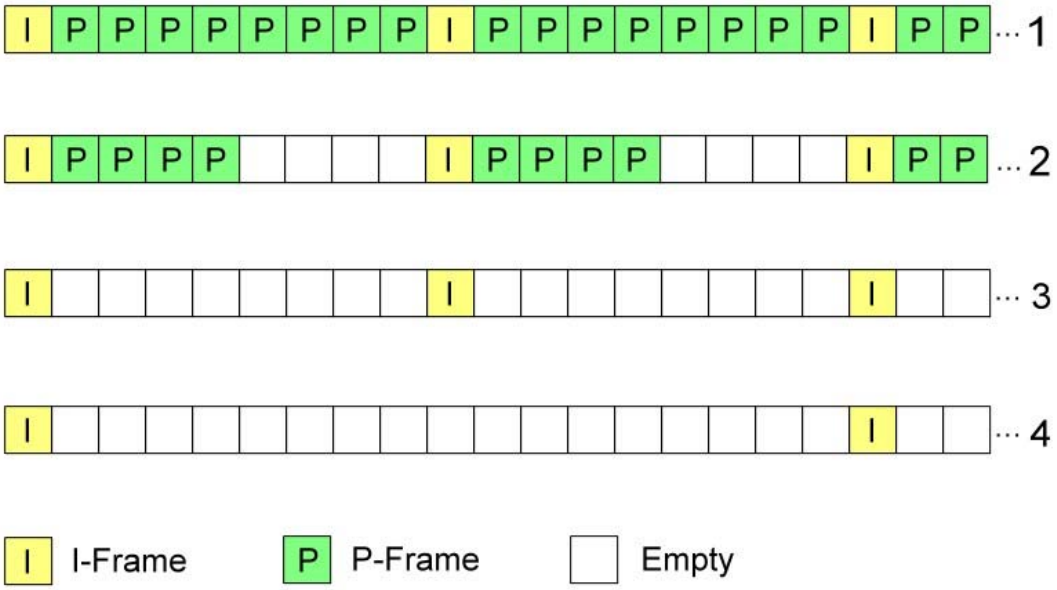

圖 4.5

我們針對 PDA 端解碼緩衝區使用程度也分成四個等級，第 1 等級代表緩衝區填滿 0%~25%，第 2 等級代表緩衝區填滿 26%~50%，第 3 等級代表緩衝區填滿 51%~75%，第 4 等級代表緩衝區填滿 76%~100%。

我們會依據 PDA 目前緩衝區使用的等級回報給 SIP-Cam，告知 SIP-Cam 可以進行不同程度的封包丟棄策略。下圖 4.6 就是我們 SIP-Cam 與 PDA 端的 DropFrame 機制與配合的情形。

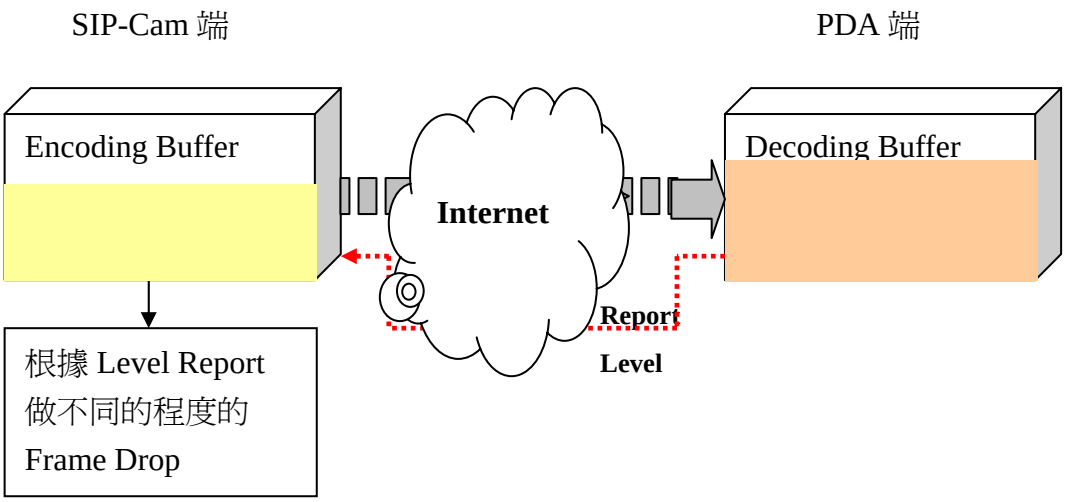

圖 4.6

由上圖 4.6 可知,PDA 端會將 SIP-Cam 的影像資訊放在解碼緩衝區(Decoding Buffer)內，再從緩衝區抓取資料進行解碼。此時，我們的 PDA 會針對目前緩衝區使用情況做出判斷，並且與 SIP-Cam 端會建立一個 Report Level 的連線，以告知 SIP-Cam 目前緩衝區使用的程度，SIP-Cam 會依據回報的四種等級，進行上述四種程級的封包丟棄策略，以減輕 PDA 端解碼的負擔。

下個章節我們將針對整體的架構進行結果與數據的探討。們將會分別針對 SIP-Cam 的四種封包丟棄機制進行實驗數據的探討，由實驗來驗證我們提出的 DropFrame 機制是否能有效地降低 PDA 解碼的負擔。