



UNIVERSITÀ DI BRESCIA
FACOLTÀ DI INGEGNERIA
Dipartimento di Elettronica per l'Automazione

Laboratorio di Robotica Avanzata **Advanced Robotics Laboratory**

Corso di Robotica
(Prof. Riccardo Cassinis)

Uso di webcam iSight su Apple
Mac mini

Elaborato di esame di:

**Andrea Pulvirenti, Stefano
Rinaldi**

Consegnato il:

11 luglio 2005

Sommario

Obiettivo del progetto "ISIGHT" è la comprensione delle strutture software alla base della comunicazione firewire tra webcam iSight e Apple Mac mini. In particolare il laboratorio di robotica avanzata ha acquisito due telecamere iSight (firewire), che si vogliono usare su un sistema Apple Mac mini da montare su un robot.

Esistono in commercio programmi per acquisizione d'immagini, anche da due dispositivi "contemporaneamente". Lo scopo del progetto è la ricerca d'informazioni riguardanti i driver di queste periferiche, in particolare delle librerie di funzioni utilizzabili nella loro comunicazione. Questo know how potrà essere utilizzato in applicazioni future che, grazie all'acquisizione delle immagini dalle telecamere, potranno sfruttare le potenzialità della visione stereoscopica su un robot autonomo.

1. Introduzione

In questo primo capitolo verranno introdotti alcuni concetti ritenuti importanti per la piena comprensione del problema e del contesto in cui si deve operare. Il primo argomento trattato è *QuickTime*, ovvero l'insieme di interfacce che permettono la comunicazione con la telecamera. In seguito verrà spiegato cosa è un *Sequence Grabber* e quale è il suo compito.

1.1. QuickTime

1.1.1. Cos'è QuickTime

QuickTime è un marchio registrato da Apple Computer e si riferisce ad una modalità di memorizzazione ed esecuzione di sequenze video e audio su personal computer diversi e con differenti sistemi operativi (in particolare Macintosh e Windows). Si tratta nella pratica di un formato di dati complesso, divenuto uno standard per i prodotti multimediali. Tra le sue caratteristiche principali vi sono il supporto di tecniche di compressione/decompressione di immagini e suoni, la capacità di sincronizzazione di sequenze audio-video ed un'interfaccia utente standard ed intuitiva.

1.1.2. Architettura

Un file *QuickTime* è suddiviso in *tracce* composte da audio, video o testo. Le tracce al loro interno contengono informazioni decodificabili da codec adeguati. All'interno delle tracce vi è una lista che associa la traccia ed il codec in grado di decodificarla. La *traccia media* contiene altre tracce che possono risiedere in zone diverse del file, su altri file o in rete. Internamente il formato mantiene i dati organizzati in una struttura ad albero di *atomi*. Ogni atomo è composto da 4 byte che vengono utilizzati per identificare la sua funzione. Un atomo può essere genitore di altri atomi oppure puntare a dati, ma non può fare entrambe le cose.

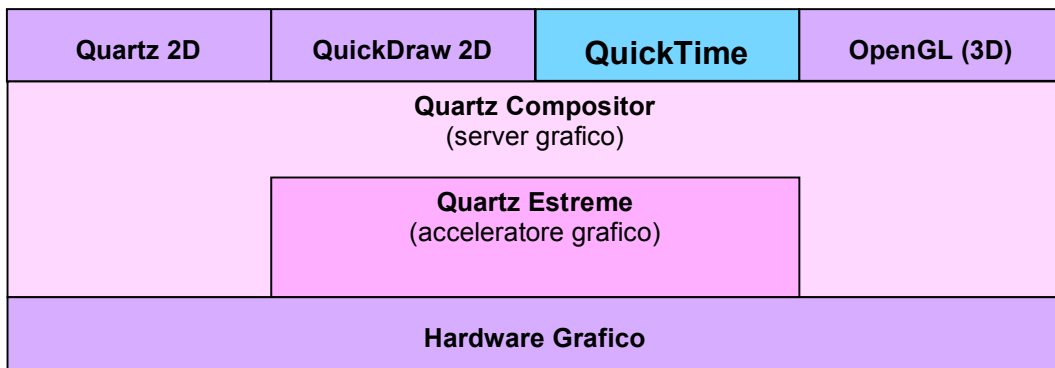


Figura 1.1: QuickTime nel Mac OS X

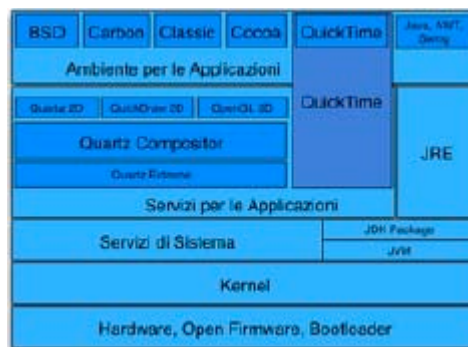


Figura 1.2: QuickTime nella struttura generale del Mac OS X

Le funzionalità di QuickTime possono essere estese fruttando la sua architettura modulare. Nella programmazione è possibile utilizzare delle interfacce, le *QuickTime API*, che permettono di aggiungere una serie di capacità multimediali ad un'applicazione senza doversi preoccupare dei dettagli riguardanti gli specifici formati di dati multimediali. Le QuickTime API includono più di 2500 funzioni, suddivise in tool secondo i compiti svolti, con funzioni speciali e tipi di dati per virtualizzare qualsiasi task. Ad ogni modulo (o *componente*) creato si può accedere attraverso un *Component manager*.

I principali tool delle QuickTime API sono:

- *Movie Toolbox*: è usato per inizializzare QuickTime, aprire, visualizzare, salvare filmati;
- *Image Compression Manager*: è un tool di compressione e decompressione di immagini, indipendente sia dal dispositivo sia dal driver associato;
- *Sequence Grabber*: è una struttura di componenti in grado di catturare e registrare campioni da una sorgente di dati real-time, come può essere una telecamera digitale;
- *QuickTime Streaming API* : permettono di inviare o ricevere flussi dati real-time usando protocolli standard come RTP o RTSP.

Esistono altri tool, tra cui *QuickTime VR*, *Sprite Toolbox*, e le *Wired Movies API*.

Quando si lavora con le QuickTime API, quasi tutte le operazioni sono eseguite su una struttura dati detta *movie*, si tratta della descrizione di una presentazione multimediale.

1.2. Il Sequence Grabber

1.2.1. Componenti Sequence Grabber

I *Sequence Grabber component* permettono all'applicazione di ottenere dati digitali da una sorgente esterna, per esempio una telecamera digitale. Questi dati possono essere visualizzati e/o salvati come filmati QuickTime. I Sequence Grabber component permettono inoltre all'applicazione di catturare audio e video senza preoccuparsi dei dettagli riguardanti l'acquisizione dei dati, infatti utilizzano servizi forniti da componenti di basso-livello, chiamati Channel Components, per ottenere dati da sorgenti differenti. Per esempio un Sequence Grabber component può fornire ad un'applicazione dati video e audio, usando i servizi di differenti Channel Component (video e audio). I Channel Component, a loro volta, possono affidarsi a servizi di livello ancora più basso forniti dai *Video Digitizer Component* (che verranno introdotti nel seguito).

Quindi l'intero processo di cattura dati risulta abbastanza complesso, ma lo sviluppatore QuickTime evita questa complessità utilizzando le Sequence Grabber API di più alto livello. Il diagramma in figura mostra le relazioni tra applicazioni QuickTime, Sequence Grabber component, e Channel Component.

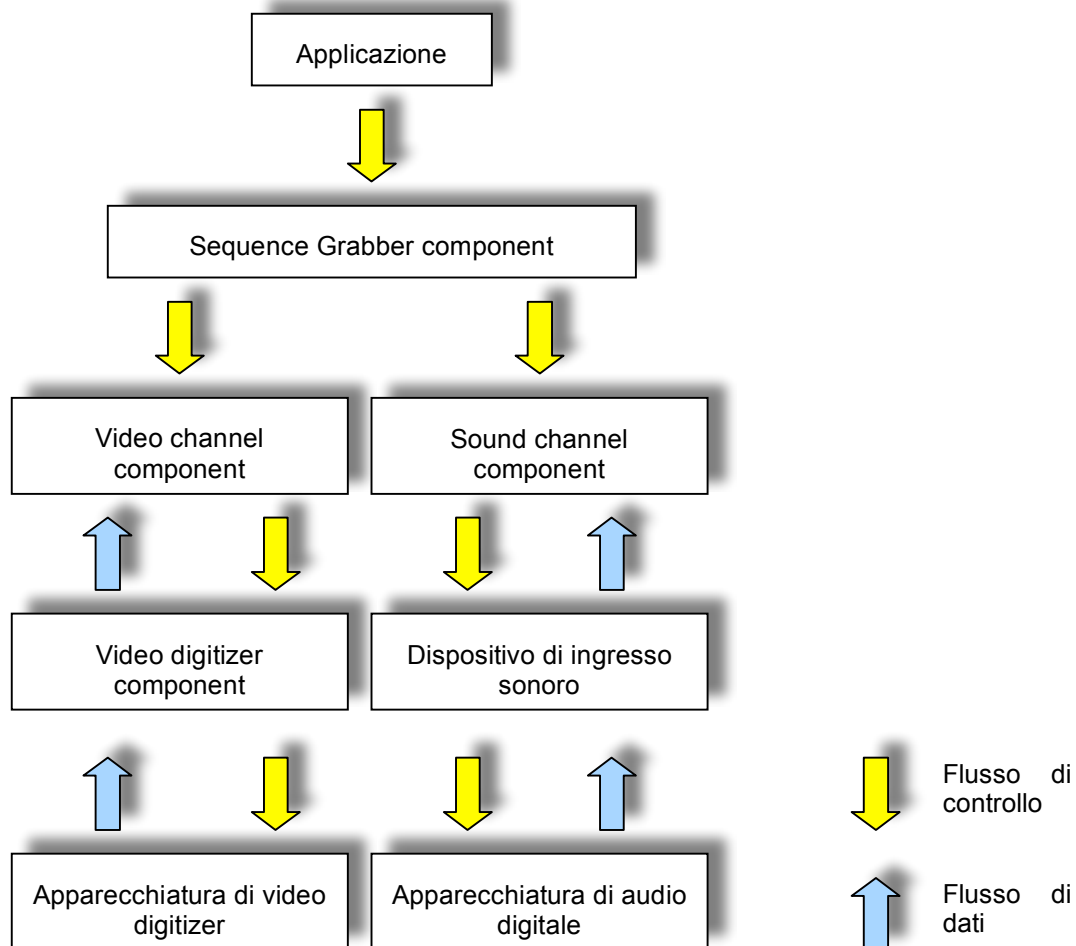


Figura 1.3: Processo di cattura dati digitali provenienti da differenti sorgenti

1.2.1.1 Channel Component

I *Sequence Grabber Channel Component*, spesso chiamati semplicemente Channel Component sono usati dai componenti di alto-livello del Sequence Grabber, e hanno il compito di svincolarlo dai dettagli sul tipo di dati presi in considerazione. I Channel Component, a loro volta, possono affidarsi ai servizi di

livello ancora più basso forniti dai Video Digitizer Component. I programmatori di applicativi QuickTime usano i servizi forniti dai Sequence Grabber Channel Component, ma solitamente non vi interagiscono direttamente.

1.2.1.2 Panel Component

Un Sequence Grabber Panel Component crea una finestra di dialogo dove è possibile impostare i comportamenti di un Channel Component. Per esempio controllare la frequenza d'immagini di un Video Digitizer e la qualità dell'immagine di un coprocessore di immagini. I Sequence Grabber Panel Component non sono mai chiamati direttamente da un'applicazione. Gli sviluppatori di applicazioni QuickTime application usano il Panel Component indirettamente chiamando il Sequence Grabber component. Quest'ultimo usa un Panel Component per ottenere le caratteristiche desiderate dall'utente per la configurazione dei Channel Component.

1.3. Il processo di cattura video

Il processo per realizzare *movie* coinvolge diversi componenti.

Il **Sequence Grabber component** gioca un ruolo fondamentale in quanto responsabile del coordinamento delle attività dei componenti di basso livello per ottenere i risultati prefissati, che possono essere ad esempio la visualizzazione di video in una finestra o l'acquisizione di una singola immagine o un filmato. Il Sequence Grabber component evita allo sviluppatore di applicazioni di dover gestire a basso livello il Video Digitizer e rende più semplice l'acquisizione video all'interno delle applicazioni. Inoltre gestisce anche dispositivi di ingresso audio e la sincronizzazione tra immagini e suono. Il flusso di dati ha inizio ad esempio con una sorgente video, ad esempio una videocamera. L'hardware del video digitizing gestisce la conversione delle sorgenti video in forma digitale e può eseguire il ridimensionamento, la conversione dei colori o il clipping delle immagini. Il Sequence Grabber può eseguire queste funzioni anche in assenza di un supporto hardware.

Lo scopo principale del **Video Digitizer** ('vdig') è fornire a QuickTime un'interfaccia software consistente usata per interagire con l'hardware del Video Digitizer. I dati forniti dal componente 'vdig' possono essere visualizzati sullo schermo o ulteriormente processati con i componenti di Sequence Grabber. Dalla prospettiva di un video digitizing, la visualizzazione di video in una finestra è detta *play-through*, e la cattura di video in un movie è detto *capturing* o *grabbing*. Il Sequence Grabber chiama queste operazioni rispettivamente *previewing* e *recording*.

Un compressore d'immagine può ulteriormente processare il dato e il risultato viene memorizzato nella memoria di sistema o su disco.

1.3.1. Il Sequence Grabber

Il Sequence Grabber facilita il lavoro dei programmatori di applicazioni collegando tutti i dettagli di controllo dei Video Digitizer, dei dispositivi audio e dei compressori. L'esempio di *HackTVCarbon* rappresenta una buona introduzione al gruppo di API del Sequence Grabber.

1.3.1.1 Previewing

La fase di *previewing* di un video con il Sequence Grabber è equivalente a settare il 'vdig' nella modalità *play-through* per visualizzare video sullo schermo del computer. Si ricorda che il Sequence Grabber fornisce il video *play-through* e che alcuni dispositivi potrebbero non gestire questa funzionalità via hardware o tentare di disegnare direttamente sullo schermo su MacOS X.

In generale i passi sono:

- Stabilire una connessione con il componente di Sequence Grabber nel modo usuale, con *OpenDefaultComponent*.
- Inizializzare il Sequence Grabber, impostare l'ambiente grafico, e allocare un nuovo canale per il video preview (usando il flag *seqGrabPreview*). Per esempio, se si vuole visualizzare video ad un quarto della sua dimensione, si deve chiamare *SGGetSrcVideoBounds* per conoscere la dimensione della sorgente video. Questa funzione chiama *VDGetDigitizerRect*, che ritorna la dimensione del video sorgente pari al digitizer rectangle. È possibile scalare l'altezza e la

larghezza mantenendo le proporzioni, e quindi inviare la nuova dimensione al Sequence Grabber attraverso la routine *SGSetChannelBounds*.

- Chiamare *SGStartPreview*, che attiva il Video Digitizer con *VDSetCompressionOnOff*.

Dopo queste operazioni la fase di previewing ha inizio. È necessario assicurarsi sempre di chiamare *SGSetGWorld* al fine di stabilire la porta grafica per il componente di Sequence Grabber. Questo è richiesto anche se non si intende lavorare con un canale video.

1.3.1.2 Recording

La funzione di *recording* è molto simile a quella di previewing, ma esistono alcune differenze tra le due operazioni. Per esempio si può impostare il Sequence Grabber per eseguire le fasi di record e play-through mentre si esegue il recording (usando il *seqGrabRecord*/ flag *seqGrabPlayDuringRecord*). È necessario specificare un file dove poter scrivere il movie, indicando che il movie sarà grabbato direttamente su disco. Per un breve movie, è possibile grabbare in memoria. Il tempo di registrazione è limitato dalle risorse di sistema disponibili (in questo caso lo spazio su disco). La chiamata a *SGStartRecord* setta il grab su disco. *SGIdle* è chiamato ripetitivamente per fornire il tempo di processing al Sequence Grabber.

Si dovrebbe chiamare *SGIdle* il più spesso possibile mentre si registra. Quando l'utente clicca il bottone del mouse, o quando il disco è pieno, si arresta la registrazione e viene chiamata *SGStop* per completare il processo di registrazione.

Se si desidera sviluppare task più sofisticati è possibile consultare la documentazione QuickTime riguardante il Sequence Grabber ed il sample code sui siti:

- <http://developer.apple.com/documentation/QuickTime/RM/CreatingMovies/SeqGrabComp/index.html>
- http://developer.apple.com/samplecode/Sample_Code/QuickTime/Capturing.htm

2. Il problema affrontato

Come spiegato precedentemente, l'obiettivo di questa relazione è verificare la possibilità di utilizzare più telecamere iSight in un sistema basato su hardware Apple Mac mini e di indicare quali devono essere i passi per poter raggiungere questo obiettivo. Verranno riportate le funzioni più importanti per la gestione e la configurazione delle telecamere, senza per questo volersi sostituire al manuale di QuickTime, a cui si rimanda per avere maggiori dettagli sulla programmazione. Maggior peso viene dato agli esempi (diverso codice funzionante è allegato alla relazione), in quanto nella fase di sviluppo questi forniscono dettagli e consigli che spesso non sono riportati su alcun manuale, ma sono frutto dell'esperienza del programmatore.

3. La soluzione adottata

In ambiente Macintosh per la gestione del flusso video in arrivo dalle telecamere viene utilizzato, come già detto nella fase introduttiva, il *Sequence Grabber*, che attraverso le sue funzioni (che sfruttano i servizi di più basso livello del *Video Digitizer*), permette al programmatore di configurare rapidamente la comunicazione con la telecamera. Attraverso queste funzioni è possibile acquisire delle immagini dal flusso video, impostare i parametri di configurazione della telecamera ed utilizzare più telecamere contemporaneamente.

Questa relazione si pone l'obiettivo di fornire supporto nella fase di programmazione e di gestione dei programmi che utilizzano le telecamere iSight. Invece di riportare solo una sequenza di funzioni importanti (che comunque sono presenti), si è pensato di suddividere il tutto nei diversi sottoproblemi che si incontrano durante la programmazione:

- La connessione fisica delle telecamere e i componenti necessari

- Le operazioni di inizializzazione di un Sequence Grabber
- Le operazioni di controllo di un Sequence Grabber
- L'acquisizione del flusso video da più telecamere
- L'acquisizione di un frame dal flusso video
- La configurazione dei parametri di una telecamera

Ad ognuno di questi argomenti è stato dedicato uno specifico paragrafo in cui si sono spiegate le operazioni e le funzioni principali utili alla risoluzione del problema. Inoltre è stato allegato anche del codice ricavato da esempi, in quanto si ritiene che spesso questo aiuti più di un semplice elenco di funzioni.

4. Modalità operative

4.1. Componenti necessari

Il sistema realizzato nell'ambito del progetto (schematizzato in **figura 4.1**) consta delle seguenti parti:

Apple Mac mini. È un calcolatore potente inglobato in un case in alluminio dalle dimensioni ridotte, un quadrato di 16,51cm per 5,08cm di altezza; Mac mini racchiude un processore G4 a 1,25GHz (o 1,42GHz), un disco rigido da 40GB (o 80GB), un'unità ottica CD-R/DVD-ROM con caricatore automatico, 256MB di SDRAM DDR e una scheda grafica ATI Radeon 9200 con 32MB di SDRAM DDR dedicata su un bus AGP 4x. È possibile collegarvi diversi dispositivi digitali, come fotocamera, iPod, stampante, videocamera o tastiera attraverso la porte USB 2.0 o FireWire. Supporta la tecnologia Ethernet BASE-T 10/100 e dispone di un modem fax v.92 da 56K. Il sistema operativo installato sul Mac mini è il **Mac OS X Tiger**.

Hub Firewire. Viene utilizzato nel sistema per collegare l'unica porta firewire del calcolatore, accessibile dalla parte posteriore del case, alle due telecamere.

iSight. Le due telecamere sono digitali. L'obiettivo (in tre parti) è formato da due elementi asferici con messa a fuoco su un sensore CCD da 1/4" con risoluzione pari a 640x480 (VGA). L'apertura dell'obiettivo è F/2,8. iSight supporta inoltre un'ampia gamma di tempi di otturazione. È possibile un'acquisizione di alta qualità a 30 fotogrammi al secondo e true colour a 24 bit. Altre caratteristiche sono la messa a fuoco automatica da 50mm all'infinito, il frame-rate Video full-motion a un massimo di 30 FPS, le porte I/O FireWire per audio, video e alimentazione, il microfono integrato con soppressione del rumore ambientale.

Il sistema Apple Mac mini + telecamere iSight ha nella compattezza il suo punto di forza. Infatti il primo è inglobato in un case compatto che pesa solo 1,4kg, mentre la singola telecamera 63,8g. Quindi il sistema può operare agevolmente a bordo di un robot mobile, implementando ad esempio applicazioni basate su visione stereoscopica.

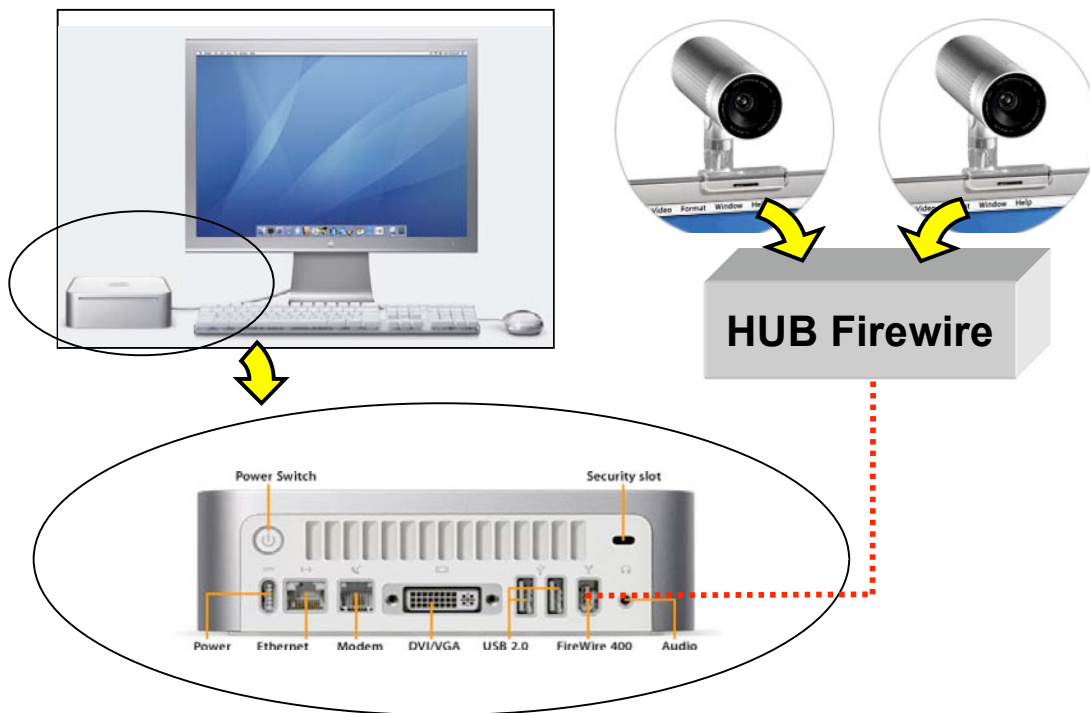


Figura 4.1: Sistema realizzato

4.2. Inizializzazione di un canale video

4.2.1. La sequenza delle operazioni da eseguire

Per inizializzare correttamente il Sequence Grabber vanno eseguite le seguenti operazioni (utilizzate negli esempi riportati):

- aprire il componente Sequence Grabber di default con *OpenDefaultComponent*
- inizializzare il componente Sequence Grabber di default con *SGInitialize*
- associare un canale ad componente Sequence Grabber con *SGNewChannel*
- configurare la porta grafica da utilizzare ed il dispositivo (Video Digitizer) da utilizzare con *SGGWorld*
- impostare la dimensione del video con *SGGetSrcVideoBounds* (qualora necessario)
- specificare come un canale deve essere usato da un componente di Sequence Grabber con *SGSetChannelUsage*
- eseguire le operazione sul flusso video fornito dal Video Digitizer: anteprima (*SGStartPreview*), registrazione(*SGStartRecord*) e controllo del flusso video (*SGStop*)
- si chiude il componente Sequence Grabber aperto in precedenza con *CloseComponent*.

Nel seguito vengono riportate alcune informazioni relative alle funzioni più importanti che vengono utilizzate per l'inizializzazione di un Sequence Grabber; ovviamente questa relazione non si vuole sostituire al manuale QuickTime ([1]), a cui si rimanda per ottenere dettagli più approfonditi, ma è si è comunque ritenuto opportuno riportare le funzioni più critiche per la gestione del Sequence Grabber, che si ritroveranno nel codice commentato di seguito.

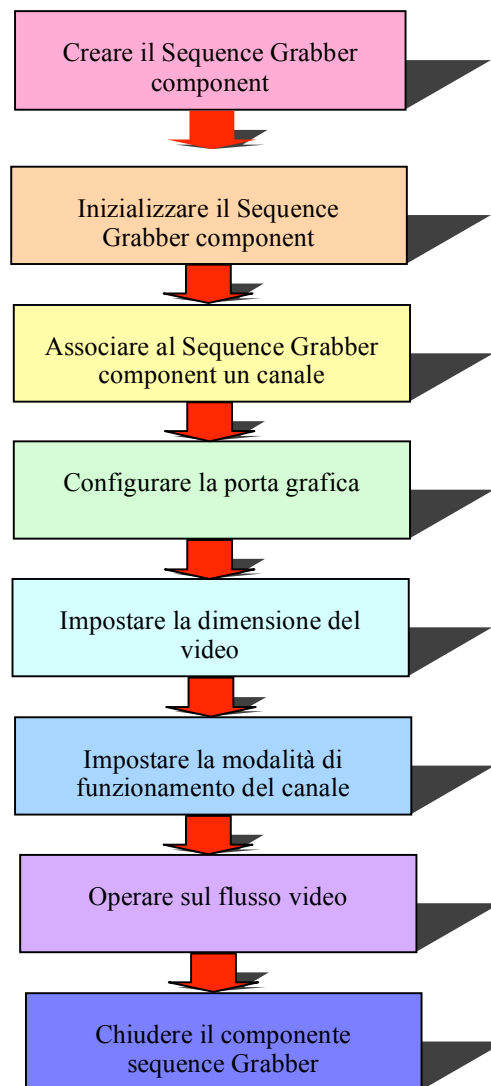


Figura 4.2: Passi per l'inizializzazione del Sequence Grabber

- **FindNextComponent**

Restituisce l'identificatore del componente per il prossimo componente registrato che soddisfa i criteri specificati da un'applicazione.

```
Component FindNextComponent (  
Component aComponent,  
ComponentDescription *looking );
```

Descrizione dei parametri

- **aComponent.** Il punto di partenza per la ricerca. Settare a 0 questo campo per far partire la ricerca all'inizio della lista del componente. Se si sta continuando una ricerca è possibile specificare un identificatore di componente restituito in precedenza da *FindNextComponent*. La funzione cerca quindi i restanti componenti.
- **Looking.** E' una struttura *ComponentDescription*. L'applicazione specifica i criteri per la ricerca di componenti nel campo di questa struttura. Il Component Manager ignora campi che sono settati a 0. Per esempio, se si settano tutti i campi a 0, tutti i componenti soddisfano i criteri di ricerca. In questo caso, l'applicazione può ricavare informazioni riguardanti tutti i

componenti registrati nel sistema chiamando ripetitivamente *FindNextComponent* e *GetComponentInfo* fino a quando la ricerca è completata. Se si settano tutti i campi a 0 ad eccezione del campo *componentManufacturer*, il Component Manager ricerca tutti i componenti registrati del *manufacturer* specificato. Si noti che la funzione *FindNextComponent* non modifica i contenuti della struttura *ComponentDescription*. Per ottenere informazioni dettagliate riguardanti un componente si può usare la funzione *GetComponentInfo* che restituisce un record di descrizione.

Risultati della funzione

Questa funzione restituisce l'identificatore del componente di un componente che soddisfa i criteri di ricerca oppure 0 se non esistono componenti che le soddisfano. Il seguente codice di esempio ne illustra l'utilizzo:

```
// FindNextComponent coding example
// See "Discovering QuickTime," page 281
void findGraphicsExporterComponents()
{
    Component c =0;
    ComponentDescription cd;
    cd.componentType =GraphicsExporterComponentType;
    cd.componentSubType =0;
    cd.componentManufacturer =0;
    cd.componentFlags =0;
    cd.componentFlagsMask =graphicsExporterIsBaseExporter;
    while((c =FindNextComponent(c, &cd) !=0) {
        ... // add component c to the list.
    }
}
```

Informazioni sulla programmazione

C interface file: Components.h

Carbon: Supportato

- **OpenComponent**

Permette di accedere ai servizi forniti da un componente specificato dall'identificatore di componente.

```
ComponentInstance OpenComponent (Component aComponent );
```

Descrizione dei parametri

- **aComponent** E' l'identificatore di componente che specifica il componente da aprire. L'applicazione può ottenere questo identificatore utilizzando la funzione *FindNextComponent*. Se l'applicazione registra un componente è possibile ottenere l'identificatore dalla funzione *RegisterComponent* o dalla *RegisterComponentResource*.

Informazioni sulla programmazione

C interface file: Components.h

- **SGInitialize**

Inizializza il componente Sequence Grabber.

```
ComponentResult SGInitialize (SeqGrabComponent s );
```

Descrizione parametri

- **s.** è l'istanza di componente che identifica la connessione al componente Sequence Grabber. Si ottiene questo valore dalla *OpenDefaultComponent* o dalla *OpenComponent*.

Prima di chiamare questa funzione si deve stabilire una connessione al componente Sequence Grabber. È possibile utilizzare *OpenDefaultComponent* oppure *OpenComponent* per stabilire la connessione del componente come mostrato dal codice che segue:

```
// SGInitialize coding example
// See "Discovering QuickTime," page 262
SeqGrabComponent MakeMySequenceGrabber (WindowRef pMacWnd)
{
    SeqGrabComponent seqGrab =NIL;
    OSErr nErr =noErr;
    // open the default sequence grabber
    seqGrab =OpenDefaultComponent(SeqGrabComponentType, 0);
    if (seqGrab !=NIL) {
        // initialize the default sequence grabber component
        nErr =SGInitialize(seqGrab);
        if (nErr ==noErr) {
            // set its graphics world to the specified window
            nErr =SGSetGWorld(seqGrab, (CGrafPtr)pMacWnd, NIL);
        }
        if (nErr && (seqGrab !=NIL)) { // clean up on failure
            CloseComponent(seqGrab);
            seqGrab =NIL;
        }
        return seqGrab;
    }
}
```

Informazioni sulla programmazione

C interface file: QuickTimeComponents.h

Carbon: Supportato

- **CloseComponent**

Termina l'accesso dell'applicazione ai servizi forniti da un componente.

```
OSErr CloseComponent (
    ComponentInstance aComponentInstance );
```

Descrizione dei parametri

- **AComponentInstance** E' l'istanza di un componente. Si ottiene da *OpenComponent* o da *OpenDefaultComponent*.

Il seguente codice mostra come *CloseComponent* viene chiamato quando non si ha bisogno a lungo di un componente.

```
// CloseComponent coding example
// See "Discovering QuickTime," page 279
void writeGWorldToImageFile(GWorldPtr gw, const FSSpec *fileSpec)
{
    GraphicsExportComponent ge =0;
    OpenADefaultComponent(GraphicsExporterComponentType,
        kQTFileTypePNG, &ge);
    GraphicsExportSetInputGWorld(ge, gw);
    GraphicsExportSetOutputFile(ge, fileSpec);
    GraphicsExportDoExport(ge, NIL);
    CloseComponent(ge);
}
```

Informazioni sulla programmazione

C interface file: Components.h

- **SGNewChannel**

Crea un canale di Sequence Grabber e assegna un componente al canale.

```
ComponentResult SGNewChannel (
    SeqGrabComponent s,
```

```
OSType channelType,
SGChannel *ref );
```

Descrizione parametri

- **s** E' l'istanza di un componente che identifica la connessione al componente di Sequence Grabber. Si ottiene questo valore da *OpenDefaultComponent* o da *OpenComponent*.
- **channelType** Il tipo di canale da aprire. Questo valore corrisponde al sottotipo di componente del componente di canale.
- **ref** E' un puntatore al campo *frameChannel* nella struttura *SeqGrabFrameInfo* che riceve un riferimento al canale associato al componente di Sequence Grabber component. Se questo componente è localizzato con successo e collegato ad un appropriato componente di canale, il componente di Sequence Grabber ritorna un riferimento al componente di canale di questo campo.

Risultato della funzione

Ritorna un codice di errore appropriato; se non c'è alcun errore ritorna noErr.

Costanti channelType

VideoMediaType: Canale video.

SoundMediaType: Canale audio.

Il componente di canale fornisce dati digitali al componente di Sequence Grabber.

Si specifica il tipo di componente di canale da associare al componente di canale come mostrato dal codice d'esempio seguente:

```
// SGNewChannel coding example
// See "Discovering QuickTime," page 263
void MakeMyGrabChannels (SeqGrabComponent seqGrab,
SGChannel *sgchanVideo,
SGChannel *sgchanSound,
const Rect *rect,
Boolean bWillRecord)
{
OSErr nErr;
long lUsage;
// figure out the usage
lUsage =seqGrabPreview; // always previewing
if (bWillRecord)
lUsage |=seqGrabRecord; // sometimes recording
// create a video channel
nErr =SGNewChannel(seqGrab, VideoMediaType, sgchanVideo);
if (nErr ==noErr) {
// set boundaries for new video channel
nErr =SGSetChannelBounds(*sgchanVideo, rect);
// set usage for new video channel
if (nErr ==noErr)
nErr =SGSetChannelUsage(*sgchanVideo, lUsage |
seqGrabPlayDuringRecord);
if (nErr !=noErr) {
// clean up on failure
SGDisposeChannel(seqGrab, *sgchanVideo);
*sgchanVideo =NIL;
}
}
// create a sound channel
nErr =SGNewChannel(seqGrab, SoundMediaType, sgchanSound);
if (nErr ==noErr) {
// set usage of new sound channel
nErr =SGSetChannelUsage(*sgchanSound, lUsage);
if (nErr !=noErr) {
// clean up on failure
SGDisposeChannel(seqGrab, *sgchanSound);
*sgchanSound =NIL;
}
}
}
```

```
}  
}
```

Informazioni sulla programmazione

C interface file: QuickTimeComponents.h

Carbon: Supportato

- **SGSetGWorld**

Stabilisce la porta grafica ed il dispositivo per un componente di Sequence Grabber.

```
ComponentResult SGSetGWorld (  
    SeqGrabComponent s,  
    CGrafPtr gp,  
    GDHandle gd );
```

Descrizione parametri

- **s** è un'istanza del componente di Sequence Grabber connesso al componente di canale. Il componente di Sequence Grabber fornisce questo valore attraverso *SGInitChannel*.
- **gp** La porta grafica (a colori) di destinazione. Il componente di Sequence Grabber setta sempre questo parametro ad un valore valido. Per usare la porta grafica corrente il parametro è settato a *NIL*.
- **gd** è un collegamento al dispositivo grafico di destinazione. Il componente di Sequence Grabber setta sempre questo parametro ad un valore valido.

Risultato della funzione

Ritorna un codice di errore appropriato; se non c'è alcun errore ritorna noErr.

Si deve chiamare questa funzione quando si sta lavorando con un canale che raccoglie dati visuali. Se si sta lavorando solo con dati che non hanno rappresentazione visuale non è necessario chiamare questa funzione. Il componente di Sequence Grabber esegue questa operazione in modo implicito quando si chiama *SGInitialize* ed il componente usa la corrente porta grafica dell'applicazione. Per impostarla ad una finestra specificata usare il codice come mostrato di seguito:

```
// SGSetGWorld coding example  
// See "Discovering QuickTime," page 262  
SeqGrabComponent MakeMySequenceGrabber (WindowRef pMacWnd)  
{  
    SeqGrabComponent seqGrab =NIL;  
    OSErr nErr =noErr;  
    // open the default sequence grabber  
    seqGrab =OpenDefaultComponent(SeqGrabComponentType, 0);  
    if (seqGrab !=NIL) {  
        // initialize the default sequence grabber component  
        nErr =SGInitialize(seqGrab);  
        if (nErr ==noErr) {  
            // set its graphics world to the specified window  
            nErr =SGSetGWorld(seqGrab, (CGrafPtr)pMacWnd, NIL);  
        }  
    }  
    if (nErr && (seqGrab !=NIL)) { // clean up on failure  
        CloseComponent(seqGrab);  
        seqGrab =NIL;  
    }  
    return seqGrab;  
}
```

Non è possibile chiamare questa funzione durante un'operazione di registrazione o visualizzazione, o dopo aver preparato un componente di Sequence Grabber ad operare in tal modo chiamando *SGPrepare*.

Informazioni sulla programmazione

C interface file: QuickTimeComponents.h

Carbon: Supportato

La funzione `SGGetGWorld` permette di ottenere la porta grafica e il dispositivo che vengono utilizzati da un componente `Sequence Grabber`.

- **SGGetSrcVideoBounds**

Determina la dimensione del rettangolo di contorno della sorgente video.

```
ComponentResult SGGetSrcVideoBounds (
    SGChannel c,
    Rect *r );
```

Descrizione parametri

- **c** E' l'identificatore della connessione per il canale di questa operazione. Si ottiene questo valore da `SGNewChannel` o da `SGNewChannelFromComponent`.
- **r** E' un puntatore ad una struttura `Rect` che riceve informazioni circa il rettangolo di contorno della sorgente video del canale.

Risultato della funzione

Ritorna un codice di errore appropriato; se non c'è alcun errore ritorna `noErr`.

Per i componenti di canale video che lavorano con i componenti `Video Digitizer`, il rettangolo di contorno della sorgente video corrisponde al rettangolo del `Video Digitizer` attivo.

Informazioni sulla programmazione

C interface file: `QuickTimeComponents.h`

Carbon: Supportato

- **SGSetChannelUsage**

Specifica come un canale deve essere usato da un componente di `Sequence Grabber`.

```
ComponentResult SGSetChannelUsage (
    SGChannel c,
    long usage );
```

Descrizione parametri

- **c** E' l'identificatore di connessione del canale per questa operazione. Si ottiene questo valore da `SGNewChannel` o da `SGNewChannelFromComponent`.
- **usage** contiene i flag che specificano come il canale deve essere usato. Il componente di `Sequence Grabber` potrebbe settare più di uno di questi flag a 1 e setta i flag non utilizzati a 0.

Risultato della funzione

Ritorna un codice di errore appropriato; se non c'è alcun errore ritorna `noErr`.

Costanti

- **seqGrabRecord**: settare questo flag a 1 se il canale è usato per operazioni di recording.
- **seqGrabPreview**: settare questo flag a 1 se il canale è usato per operazioni di previewing.
- **seqGrabPlayDuringRecord**: settare questo flag a 1 se si vuole usare il canale per visualizzare dati catturati durante un'operazione di registrazione.
- **seqGrabLowLatencyCapture**: settare questo flag a 1 se si vuole usare il canale per restituire il frame più recente durante videoconferenza, trasmissione dal vivo, e live image processing.
- **seqGrabAlwaysUseTimeBase**: settare questo flag a 1 se si vuole usare il canale per dire a `Video Digitizer` di usare sempre tempo base al fine di creare durata di frame uniforme.

Informazioni sulla programmazione

C interface file: QuickTimeComponents.h e Carbon: Supportato

- **SGSetChannelBounds**

Specifica un rettangolo di contorno della visualizzazione del canale

```
ComponentResult SGSetChannelBounds (  
    SGChannel c,  
    const Rect *bounds );
```

Descrizione parametri

- **c** E' l'identificatore di connessione per il canale per questa operazione. Si ottiene questo valore da *SGNewChannel* o da *SGNewChannelFromComponent*.
- **bounds** è un puntatore ad una struttura *Rect* che definisce il rettangolo di contorno del canale visualizzato.

Risultato della funzione

Ritorna un codice di errore appropriato; se non c'è alcun errore ritorna noErr.

Informazioni sulla programmazione

C interface file: QuickTimeComponents.h

Carbon: Supportato

Per effettuare la corrispondente funzione get si veda *SGGetChannelUsage* *SGGetChannelBounds*.

4.2.2. Un esempio concreto: HACKTVCarbon

Questo software, il cui sorgente è allegato alla relazione, è un esempio (un po' complesso), di come poter configurare il Sequence Grabber per acquisire i dati da una telecamera, di come poter visualizzare a video l'anteprima del flusso video e di come poter salvare i dati in un movie (dopo aver impostato il tipo di acquisizione che il Video Digitizer deve compiere sui dati in arrivo). Tutto questo viene effettuato attraverso una chiara interfaccia grafica. In questa relazione viene commentato solo parte del codice, quella riguardante la configurazione del Sequence Grabber, in quanto l'unica veramente utile per chiarire alcuni aspetti sulla gestione del Sequence Grabber. Per avere maggiori chiarimenti riguardanti il salvataggio dei dati e le modalità di acquisizione del segnale video si rimanda al manuale e al codice allegato alla relazione.

La funzione che nel codice si occupa dell'inizializzazione del Sequence Grabber è *InitializeSequenceGrabber*. All'interno di questa funzione vengono ripercorse tutte le tappe precedentemente spiegate:

```
void InitializeSequenceGrabber(void)  
{  
    ComponentDescription theDesc;  
    ComponentResult      result = noErr;  
  
    /*Puntatore alla porta grafica  
    GrafPtr              savedPort;  
    Component            sgCompID;  
  
    /*Definisco alcuni flag che saranno utilizzati dalle funzioni successive  
    gQuitFlag = false;  
    gSeqGrabber = 0L;  
    gVideoChannel = 0L;  
    gSoundChannel = 0L;  
    gMonitorPICT = NULL;  
    #if __MWERKS__  
    gPrintRec = (TPrint) NewHandleClear (sizeof (TPrint));  
    #endif  
    gCurrentlyRecording = false;
```



```

// Find and open a sequence grabber
theDesc.componentType = SeqGrabComponentType;
theDesc.componentSubType = 0L;
theDesc.componentManufacturer = 'appl';
theDesc.componentFlags = 0L;
theDesc.componentFlagsMask = 0L;

/*Cerco il componente Sequence Grabber con le caratteristiche che voglio
sgCompID = FindNextComponent (NULL, &theDesc);
if (sgCompID != 0L)

    /*apro il componente Sequence Grabber
    gSeqGrabber = OpenComponent (sgCompID);

// If we got a sequence grabber, set it up
if (gSeqGrabber != 0L)
{
    // Get the monitor
    CreateMonitorWindow();
    if (gMonitor != NULL)
    {
        // Display the monitor window
        GetPort (&savedPort);
        MacSetPort ((GrafPtr)GetWindowPort(gMonitor));
        MacMoveWindow(gMonitor, 10, 30 + GetMBarHeight(), 0);
        MacShowWindow (gMonitor);

        // Initialize the sequence grabber

        /*Inizializzazione del Sequence Grabber
        result = SGInitialize (gSeqGrabber);
        CheckError(result,"SGInitialize");
        if (result == noErr)
        {

            /*Imposto la porta grafica ed il dispositivo
            result = SGSetGWorld (gSeqGrabber,
                                GetWindowPort(gMonitor), NULL);
            CheckError(result,"SGSetGWorld");

            // Get a video channel

            /*Associo al Sequence Grabber il canale video
            result = SGNewChannel (gSeqGrabber, VideoMediaType,
                                &gVideoChannel);
            CheckError(result,"SGNewChannel for video");
            if ((gVideoChannel != NULL) && (result == noErr))
            {
                short width;
                short height;

                gQuarterSize = false;
                gHalfSize = true;
                gFullSize = false;

                /*Chiedo la dimensione attuale della sorgente video
                result = SGGetSrcVideoBounds (gVideoChannel,
                                                &gActiveVideoRect);
                CheckError(result,"SGGetSrcVideoBounds");
                width = (gActiveVideoRect.right -
                        gActiveVideoRect.left) / 2;
                height = (gActiveVideoRect.bottom -
                        gActiveVideoRect.top) / 2;
                SizeWindow (gMonitor, width, height, false);

                /*Imposto alcune opzioni del Sequence Grabber
                result = SGSetChannelUsage (gVideoChannel,
                                            seqGrabPreview | seqGrabRecord |
                                            seqGrabPlayDuringRecord);
                CheckError(result,"SGSetChannelUsage for vid");
                #if TARGET_API_MAC_CARBON
                {
                    Rect portRect;

```

```

        GetPortBounds (GetWindowPort (gMonitor),
                        &portRect);

        /*Imposto la dimensione della sorgente video
        result = SGSetChannelBounds (gVideoChannel,
                                    &portRect);
    }
#else
/*Imposto la dimensione della sorgente video
result = SGSetChannelBounds (gVideoChannel, &(gMonitor-
                            >portRect));
#endif
    CheckError (result, "SGSetChannelBounds");
}

// Get a sound channel

/*Associo al Sequence Grabber il canale sonoro
result = SGNewChannel (gSeqGrabber, SoundMediaType,
                      &gSoundChannel);
CheckError (result, "SGNewChannel for sound");

if ((gSoundChannel != NULL) && (result == noErr))
{

    /*Impostazione dei parametri del canale audio
    if (gSoundChannel != NULL)
    {
        Handle sampleRates = NULL;

        /*Imposto alcune opzioni del canale sonoro

        result = SGSetChannelUsage (gSoundChannel,
                                    seqGrabPreview | seqGrabRecord);
        CheckError (result, "SGSetChannelUsage for
                    sound");

        // Set the volume low to prevent feedback
        //when we start the preview, in case the
        //mic is anywhere near the speaker.
        result = SGSetChannelVolume (gSoundChannel,
                                    0x0010);
        CheckError (result, "SGSetChannelVolume ");
        //result = SGSetSoundRecordChunkSize
        //(gSoundChannel, -((Fixed) (.25*65536.0)
        //sampleRates=NewHandleClear (5*
        //                                sizeof(Fixed));
        if (sampleRates) {
            OSErr tempErr;
            *(long*) (*sampleRates) = 8000<<16;
            // add 8kHz rate
            *((long*) (*sampleRates)+1) =
            11025<<16; // add 11kHz rate
            *((long*) (*sampleRates)+2) =
            16000<<16; // add 16kHz rate
            *((long*) (*sampleRates)+3) =
            22050<<16; // add 22kHz rate
            *((long*) (*sampleRates)+4) =
            32000<<16; // add 32kHz rate
            tempErr = SGSetAdditionalSoundRates
            (gSoundChannel, sampleRates);
            CheckError (result,
                "SGSetAdditionalSoundRates ");
            DisposeHandle (sampleRates);
        }
    }
}

// Get the alignment proc (for use when dragging the
//                                monitor)

```

```

        result = SGGetAlignmentProc (gSeqGrabber,
                                    &gSeqGrabberAlignProc);
        CheckError(result, "SGGetAlignmentProc ");
    }

    // Go!
    if (result == noErr) {

        /*Faccio partire l'anteprima
        result = SGStartPreview (gSeqGrabber);
        CheckError(result, "SGStartPreview ");
    }
    MacSetPort (savedPort);
}
}
}
}

```

4.2.3. Un altro esempio: BigEasyVideoGrabber.c

Di seguito viene riportato un ulteriore esempio, tratto da un altro codice contenuto nel file Easy_video_Grabber.zip, su come può essere inizializzato il Sequence Grabber. Rispetto al caso visto in precedenza, questo programma è molto più semplice e lineare in quanto privo d'interfaccia grafica.

```

EasyVideoGrabber NewEasyVideoGrabber(Rect* outputSize)
/*
 * Gets the default video digitizer,
 * and returns the output size of the video image.
 */
{
    EasyVideoGrabber evg;
    ComponentResult thisError;

    evg = (void*)NewPtr(sizeof(EasyVideoGrabberRecord));
    if (!evg)
        goto fail;

    evg->sg = 0;
    evg->vc = 0;

    /* Apro il Sequence Grabber di default
    evg->sg = OpenDefaultComponent(SeqGrabComponentType, 0);
    if (!evg->sg)
        goto fail;

    /*Inizializzo il componente Sequence Grabber di default
    thisError = SGInitialize(evg->sg);
    if (thisError)
        goto fail;

    /*Associo un nuovo canale video al componente Sequence Grabber
    thisError = SGNNewChannel(evg->sg, VideoMediaType, &evg->vc);
    if (thisError)
        goto fail;

    if (outputSize)
    {

        /*Imposto la dimensione della sorgente video
        thisError = SGGetSrcVideoBounds(evg->vc, outputSize);
        if (thisError)
            goto fail;
    }

    /*Imposto il Sequence Grabber per visualizzare l'anteprima
    thisError = SGSetChannelUsage(evg->vc, seqGrabPreview);
    if (thisError)
        goto fail;

    goto goHome;

fail:if (evg)

```

```

    {
        if (evg->sg)
            /*Nel caso di errore chiudo il componente Sequence Grabber aperto
            CloseComponent (evg->sg) ;
            DisposePtr ((void*) evg) ;
            evg = 0;
        }
    goHome:return evg;
}

```

4.3. Il controllo del Sequence Grabber

Prima di passare ad analizzare gli aspetti più legati a questa relazione, è necessario dare uno sguardo alle funzioni che sono messe a disposizione per la gestione del flusso video. Dato che questo aspetto non è direttamente collegato all'argomento principale di questa relazione, qui vengono riportate solo le principali funzioni e il loro compito; per avere maggiori dettagli si rimanda al relativo manuale.

QuickTime mette a disposizione diverse funzioni per permettere di effettuare il controllo dell'anteprima o delle operazioni. Con queste funzioni si può iniziare o fermare le operazioni, si può mettere in pausa l'acquisizione dei dati.

Attraverso la funzione *SGStartPreview* (pag. 1909 [1]) si può iniziare l'operazione di anteprima. *SGStartRecord* (pag. 1910 [1]) permette di iniziare le operazioni di registrazione. *SGStop* (pag. 1910 [1]) blocca un componente Sequence Grabber. Si può mettere in pausa il Sequence Grabber chiamando *SGPause* (pag. 1866 [1]). Per sapere se il sequenze Grabber è nello stato di pausa si può utilizzare *SGGetPause* (pag. 1830 [1]). Si può garantire il tempo di elaborazione al sequenze Grabber attraverso *SGIdle* (pag. 1848 [1]). Questa funzione va chiamata spesso durante le operazioni di registrazione e di anteprima. Se l'applicazione riceve un nuovo evento durante la fase di anteprima o di registrazione va chiamata la funzione *SGUpdate* (pag. 1912 [1]). Se si vuole preparare il sequenze Grabber ad un'operazione di anteprima o di registrazione va chiamata la funzione *SGPrepare* (pag. 1867 [1]). Questa funzione permette di verificare se il Sequence Grabber supporta tutti i parametri specificati in precedenza. Verificando la correttezza dei parametri utilizzati si possono migliorare le operazioni di registrazione e di anteprima. Per rilasciare le risorse di sistema viene chiamata *SGRelease* (pag. 1868 [1]). Si può ricavare un riferimento al movie creato con un'operazione di registrazione attraverso *SGGetMovie* (pag. 1825 [1]). Si può determinare l'identificativo assegnato all'ultimo movie creato con *SGGetLastMovieResID* (pag. 1823 [1]). Si può estrarre un frame dal flusso video usando *SGGrabPic* (pag. 1846 [1]).

4.4. Acquisizione dei dati da più telecamere

Nella guida alla programmazione di QuickTime non vi è alcun riferimento esplicito, né limitazione, sulla acquisizione di dati da diverse telecamere. Per questo compito le operazioni di inizializzazione del Sequence Grabber non devono essere modificate in modo radicale; infatti è sufficiente aprire delle istanze multiple del Video Digitizer QuickTime e permettere ai diversi flussi video di essere attivi contemporaneamente (ovvero vanno creati diversi componenti Sequence Grabber, ognuno associato ad una telecamera diversa). Il numero dei canali video che possono essere attivi è limitato solamente dal numero delle telecamere che sono presenti sul bus FireWire.

In realtà alcune limitazioni relative all'utilizzo di più telecamere esistono: infatti il bus FireWire, pur essendo molto veloce, non può sopportare un numero eccessivo di dati, ovvero il flusso di dati non può (per ovvi motivi) essere maggiore della larghezza di banda del canale. Qualora questo si verifichi è possibile risolvere il problema cercando di diminuire il flusso di dati proveniente dalle telecamere connesse al bus. Ad esempio, se è possibile, si può ridurre la dimensione dei frame o la frequenza con cui vengono inviati dalle telecamere; qualora ciò non fosse sufficiente si può pensare di allargare la banda di trasmissione del canale connettendo al bus PCI un altro bus FireWire. Un'altra limitazione riguarda le telecamere che utilizzano per lo scambio dei dati il *DV Video Digitizer* (non è il caso di iSight), per le quali non è prevista la possibilità di utilizzare più telecamere.

Non è stato trovato alcun esempio relativo a questo argomento, ma attraverso il codice presentato al punto precedente è possibile rendersi conto della possibilità di acquisire i dati da due diverse telecamere,

senza dover modificare il codice; infatti basta installare due diverse copie del programma compilato e lanciarle in parallelo. Una volta aperte le due finestre di anteprima si possono impostare le opzioni in modo da far acquisire alle due istanze del Video Digitizer (invocate ognuna da una copia del codice) i dati che provengono da telecamere diverse.

4.5. Come acquisire un frame dal flusso dei dati

QuickTime mette a disposizione una funzione di alto livello che permette di ottenere da un Sequence Grabber una struttura dati **Picture**. In questa struttura viene allocato il frame acquisita dal canale video secondo diverse modalità, a seconda dei parametri che vengono forniti alla funzione.

- **SGGrabPict**

Questa funzione, come già ricordato in precedenza, permette di ottenere un puntatore ad una struttura Picture da un Sequence Grabber.

```
ComponentResult SGGrabPict (
    SeqGrabComponent s,
    PicHandle *p,
    const Rect *bounds,
    short offscreenDepth,
    long grabPictFlags );
```

Descrizione dei campi

- **s** L'istanza del componente che identifica la connessione al Sequence Grabber; questo valore può essere ottenuto attraverso le funzioni *OpenDefaultComponent* o *OpenComponent*.
- **p** Puntatore a un campo che riceve l'handle della struttura Picture. Se la funzione non può creare la struttura dati, l'handle viene fissato a NIL.
- **bounds** Puntatore alla regione di confine della struttura Picture. Di default questo rettangolo giace nella porta grafica corrente. Se viene fissato il flag di *grabPictOffScreen* a 1 nel campo *grabPictFlags*, il Sequence Grabber posiziona la struttura in una porta grafica esterna e in questo caso il rettangolo viene interpretato in quel contesto.
- **offscreenDepth** Determina il pixel depth (ovvero il numero di bit associati ad un pixel) se il mondo grafico è esterno. Questo parametro viene di solito fissato a 0, ovvero seleziona in automatico il numero di bit per pixel migliore. Se si sta visualizzando la figura, questo parametro è ignorato.
- **grabPictFlags** Contiene un flag (elencati di seguito) per il controllo dell'operazione. Questa funzione ritorna un errore (pag. 2705 [1]); se non viene individuato nessun errore, ritorna no error.

Costanti

- **grabPictOffScreen** In questa modalità il Sequence Grabber posiziona la struttura picture in un mondo grafico esterno. Questo flag va posto a 1 se si vuole utilizzare questa modalità; in questo caso si utilizza il campo *offscreenDepth* per impostare il pixel depth e il rettangolo specificato dal campo *bounds* viene interpretato per il mondo grafico esterno.
- **grabPictIgnoreClip** Indica al Sequence Grabber di ignorare qualsiasi regione di ritaglio che è stata definite per il canale del Sequence Grabber. Deve essere fissato a 1 per far sì che il Sequence Grabber ignori qualunque regione di interesse impostata in precedenza.
- **grabPictCurrentImage** Impostare a 1 questo flag causa l'acquisizione più rapida possibile dell'immagine. Sebbene questo flag sotto certe condizioni possa fallire, l'errore può essere recuperato; infatti se non ritorna alcuna struttura picture, è possibile richiamare di nuovo questa funzione senza avere impostato questo flag. Questa funzione non causa l'arresto dell'anteprima del Sequence Grabber né l'acquisizione del frame successivo. Permette di acquisire la prima immagine che viene visualizzata del Sequence Grabber. Di solito conviene utilizzare *SGPause* (che permette di bloccare o riavviare temporaneamente l'operazione precedente) prima di invocare questa funzione con questo flag.

Risultato della funzione

Ritorna un codice di errore appropriato; se non c'è alcun errore ritorna noErr.

Informazioni sulla programmazione

C interface file: QuickTimeComponents.h e Carbon: Supportato

4.5.1. La struttura dati picture

Per poter correttamente interagire con il frame fornito dalla funzione precedente è necessario poter conoscere con precisione la sua struttura altrimenti non è possibile effettuare alcuna operazione utile.

- **La struttura dati Picture**

Questa struttura definisce una Picture Mac OS:

```
struct Picture {
    short picSize;
    Rect picFrame;
};
```

Descrizione dei campi

- **picSize** Questo campo contiene la dimensione, in byte, del resto del record per una picture della versione 1. L'informazione in questo campo è utile solo per le picture della versione 1, le cui dimensioni non possono superare i 32 KB. La versione 2 e la versione 2 estesa possono supportare record con dimensione maggiori dei 32 KB, e quindi non è sufficiente lo spazio di 2 byte per poter indicare la loro dimensione.
- **picFrame** Indica il rettangolo che rappresenta la frontiera della picture definita nel resto del record. Questo rettangolo viene utilizzato per riscaldare l'immagine qualora venga graficata in rettangolo di dimensioni diverse.

Informazioni aggiuntive sulla Picture

Le strutture Picture nei sistemi Mac OS sono delle sequenze di comandi di grafica. Questi permettono di graficare in modo semplice le immagini di applicazioni diverse; ovvero garantiscono uniformità nella rappresentazione delle immagini a livello di sistema operativo.

Pur non costituendo l'argomento di questo progetto, si è ritenuto opportuno riportare alcune informazioni relative alla gestione e all'utilizzo di questa struttura dati; per ulteriori informazioni si rimanda al manuale di *QuickDraw*, ovvero alle funzioni che tale interfaccia offre per la elaborazione delle immagini in ambiente Mac OS.

Per poter creare una struttura Picture (oltre alla già citata funzione *SGrabPict* che permette di ottenere un Picture a partire dal flusso video) si utilizza la funzione *OpenPicture*; in questo modo QuickDraw alloca un nuovo puntatore, immagazzina alcuni dati nel record, imposta il campo *picSave* per la porta corrente e invoca *HidePen* (ovvero le modifiche non verranno visualizzate sull'immagine); i dati iniziali sono costituiti dalla lunghezza del record, dal rettangolo che costituisce l'immagine, dalla versione e dalla regione di interesse dell'immagine.

Quando si procede alla modifica dell'immagine (ad esempio disegnando un ovale invocando l'opportuna funzione di QuickDraw), QuickDraw agisce andando a aggiungere ai dati della Picture i comandi opportuni. La lunghezza dei dati iniziale viene incrementata per riflettere lo stato dei nuovi dati. Alla fine delle operazioni, invocando *ClosePicture* (che chiama lo *ShowPen*), viene posto alla fine della struttura Picture un indicatore di fine della figura e il campo *picSave* è posto a *NIL*. Attraverso la funzione *DrawPicture* è possibile visualizzare a video l'immagine immagazzinata nella struttura. La struttura infine può essere salvata in un opportuno file 'PICT' invocando le apposite routine che permettono il salvataggio dei dati. Per poter caricare da file una struttura Picture si deve utilizzare la funzione *GetPicture*.

Pur essendo disponibile la struttura interna della Picture e l'elenco di tutti i codici che corrispondono alle operazioni che possono essere eseguite, non è possibile modificare direttamente la struttura Picture, altrimenti tale immagine non può più essere correttamente gestita dagli applicativi che ne permettono la visualizzazione.

Note relative alla versione della Picture

Esistono tre diverse versioni della struttura Picture. La versione 1 supporta solo le operazioni di grafica in bianco e nero a 72 dpi. La versione 2 può essere usata quando la porta grafica corrente è a colori. Le strutture così create supportano operazioni di grafica a colori a 72 dpi. La versione 2 estesa permette alle applicazioni di definire la risoluzione per le immagini a colori o in bianco e nero. Per mantenere la compatibilità con le strutture della versione 1, le versioni successive non hanno modificato il campo *picSize*, anche se per esse non ha significato. Per poter ottenere le informazioni relative alla dimensione si può utilizzare la funzione *getHandleSize*.

Informazioni sulla programmazione

C interface file: Quickdraw.h.

4.5.2. Un esempio applicativo: EasyVideoGrabberTest.c

Questo software, pur non utilizzando la funzione indicata in precedenza per l'acquisizione dei frame in formato picture, fornisce, oltre ad alcune utili indicazioni riguardanti l'utilizzo del Sequence Grabber, un esempio di acquisizione di un frame dal flusso video permettendo di visualizzarlo sulla porta grafica corrente. In questa sede si commenta solo la parte relativa all'acquisizione del frame, rimandando per le altre parti al codice che è allegato alla relazione (in quanto, in modo semplificato ricalca quanto già visto in precedenza con il codice HACKTVCarbon).

La funzione che permette di effettuare l'acquisizione di un frame dal flusso video è *GrabEasyVideoGrabber*. I parametri che tale struttura riceve in ingresso sono la struttura *EasyVideoGrabber* che contiene i parametri relativi al Sequence Grabber (quali l'istanza del Sequence Grabber, l'istanza del canale video e la dimensione dell'area digitalizzata) e la dimensione dell'area della porta grafica (il puntatore alla struttura *rect*).

```
typedef struct
{
    ComponentInstance sg;           /* Sequence Grabber */
    ComponentInstance vc;         /* Video Channel */
    Rect preferredRect;           /* Size of digitizing area */
} EasyVideoGrabberRecord, * EasyVideoGrabber;
```

La funzione ritorna il valore true se il frame è stato correttamente acquisito e visualizzato dalla funzione. La caratteristica principale di questo codice è la presenza di un ciclo while; questo pur sembrando molto inelegante è un metodo molto efficace (anche se non l'unico), per acquisire un frame; infatti il Sequence Grabber non permette di sapere quando un frame è stato acquisito; il Sequence Grabber risulta occupato solo per qualche ciclo del while (se il frame rate è sufficientemente alto).

```
Boolean GrabEasyVideoGrabber(EasyVideoGrabber evg, Rect* r)
{
    ComponentResult thisError;

    /*Puntatore alla porta grafica corrente
    GWorldPtr gw;

    /*Puntatore al dispositivo
    GDHandle gd;
    Boolean localGrabber;
    unsigned long t;
    Boolean result;

    result = false;

    localGrabber = (evg == 0);
    if (localGrabber)
        evg = NewEasyVideoGrabber(nil); /*Funzione che inizializza il Sequence grabber

    if (!evg)
        goto goHome;

    /*Determina la porta grafica corrente
    GetGWorld(&gw, &gd);

    /*Stabilisco la porta grafica e il dispositivo che il Sequence Grabber utilizza
```

```

thisError = SGSetGWorld(evg->sg, gw, gd);
if (thisError)
    goto goHome;

/*Specifico la dimensione della porta di uscita su cui visualizzare l'anteprima
thisError = SGSetChannelBounds(evg->vc, r);
if (thisError) /*in caso di errore l'esecuzione viene terminata
    goto goHome;

/*Faccio partire l'anteprima del flusso video
thisError = SGStartPreview(evg->sg);
if (thisError)
    goto goHome;

/*Contatore che tiene conto del numero di cicli eseguito
t = TickCount() + 8;

/*Ciclo while che viene utilizzato per acquisire un frame
do
{
    /*Fornisce il tempo di elaborazione per il Sequence Grabber
    thisError = SGIdle(evg->sg);
    if (thisError)
        goto goHome;
} while (TickCount() < t);

/*Ferma la fase di anteprima
thisError = SGStop(evg->sg);

if (localGrabber)
    DisposeEasyVideoGrabber(evg); /*Termina il Sequence Grabber

result = true;

goHome;;
return result;
}

```

4.6. Modalità di taratura

4.6.1. La taratura di una telecamera

La telecamera digitale è un sistema visivo complesso costituito da una sezione ottica e da una di elaborazione (digital image processing). L'obiettivo di questo strumento è l'acquisizione di un'immagine proveniente dall'ambiente circostante. Affinché questa sia caratterizzata da un alto contenuto informativo è necessario operare una taratura del dispositivo, cioè impostare (manualmente o via software) un insieme di parametri che influenzano la qualità dell'immagine acquisita. I principali sono: la tinta, il livello di saturazione, la nitidezza, il livello di luminosità, la dimensione dell'immagine, il trigger, il filtro ottico applicato, lo zoom, il fuoco dell'obiettivo, la gamma, il bilanciamento del bianco, il tempo di apertura dell'otturatore, e diversi altri. Le librerie QuickTime mettono a disposizione dello sviluppatore di applicazioni una serie di funzioni per l'impostazione via software di questi parametri, detti *feature*.

4.6.2. Funzioni IIDC Digitizer

Molte telecamere digitali basate su IEEE-1394 supportano le *Instrumentation and Industrial Control (IIDC) features*, alle quali si può avere accesso via software. In particolare in QuickTime esistono delle funzioni per comunicare con il Video Digitizer.

- **VDIIIDCGetFeatures** inserisce atomi in un QuickTime atom container che specifica le capacità correnti della telecamera e lo stato delle sue IIDC features;
- **VDIIIDCGetFeaturesForSpecifier** inserisce atomi in un QuickTime atom container che specifica lo stato corrente di una singola telecamera attraverso una feature o un gruppo di feature;
- **VDIIIDCSetFeatures** cambia lo stato delle IIDC features di una telecamera:

- **VDIIDCGetDefaultFeatures** inserisce atomi in un QuickTime atom container che specifica le capacità di default della telecamera e lo stato di default delle sue IIDC features;
- **VDIIDCGetCSRData** legge il registro CSR della telecamera direttamente;
- **VDIIDCSetCSRData** scrive nel registro CSR della telecamera direttamente;

Quasi tutte le IIDC features possono essere espresse usando la struttura dati `VDIIDCFeatureSettings`. Ma ne esistono alcune che invece richiedono altre strutture dati come mostrato dalla tabella seguente:

Feature	Definizione variabile	Struttura dati
Tinta	<code>vdIIDCFeatureHue</code> = 'hue'	<code>VDIIDCFeatureSettings</code>
Saturazione	<code>vdIIDCFeatureSaturation</code> = 'satu'	<code>VDIIDCFeatureSettings</code>
Nitidezza	<code>vdIIDCFeatureSharpness</code> = 'shrp'	<code>VDIIDCFeatureSettings</code>
Luminosità	<code>vdIIDCFeatureBrightness</code> = 'brit'	<code>VDIIDCFeatureSettings</code>
Guadagno	<code>vdIIDCFeatureGain</code> = 'gain'	<code>VDIIDCFeatureSettings</code>
Diaframma	<code>vdIIDCFeatureIris</code> = 'iris'	<code>VDIIDCFeatureSettings</code>
Otturatore	<code>vdIIDCFeatureShutter</code> = 'shtr'	<code>VDIIDCFeatureSettings</code>
Esposizione	<code>vdIIDCFeatureExposure</code> = 'xpsr'	<code>VDIIDCFeatureSettings</code>
Bilanciamento del bianco (U)	<code>vdIIDCFeatureWhiteBalanceU</code> = 'whbu'	<code>VDIIDCFeatureSettings</code>
Bilanciamento del bianco (V)	<code>vdIIDCFeatureWhiteBalanceV</code> = 'whbv'	<code>VDIIDCFeatureSettings</code>
Gamma	<code>vdIIDCFeatureGamma</code> = 'gmma'	<code>VDIIDCFeatureSettings</code>
Temperatura	<code>vdIIDCFeatureTemperature</code> = 'temp'	<code>VDIIDCFeatureSettings</code>
Zoom	<code>vdIIDCFeatureZoom</code> = 'zoom'	<code>VDIIDCFeatureSettings</code>
Fuoco	<code>vdIIDCFeatureFocus</code> = 'fcus'	<code>VDIIDCFeatureSettings</code>
Panoramica	<code>vdIIDCFeaturePan</code> = 'pan'	<code>VDIIDCFeatureSettings</code>
Tilt	<code>vdIIDCFeatureTilt</code> = 'tilt'	<code>VDIIDCFeatureSettings</code>
Filtro ottico	<code>vdIIDCFeatureOpticalFilter</code> = 'opft'	<code>VDIIDCFeatureSettings</code>
Trigger	<code>vdIIDCFeatureTrigger</code> = 'trgr'	<code>VDIIDCTriggerSettings</code>
Dimensione	<code>vdIIDCFeatureCaptureSize</code> = 'cpsz'	Undefined settings
Qualità	<code>vdIIDCFeatureCaptureQuality</code> = 'cpql'	Undefined settings
Punto di fuoco	<code>vdIIDCFeatureFocusPoint</code> = 'fpnt'	<code>VDIIDCFocusPointSettings</code>
Alterazione bordi	<code>vdIIDCFeatureEdgeEnhancement</code> = 'eden'	<code>VDIIDCFeatureSettings</code>
Lighting Hint	<code>vdIIDCFeatureLightingHint</code> = 'lhnt'	<code>VDIIDCLightingHintSetting</code>

4.6.3. Configurare le caratteristiche della telecamera

Gli sviluppatori possono operare seguendo due approcci differenti:

- Il più semplice consiste nell'usare: `SgSettingsDialog` per configurare il canale video, quindi `SGSetChannelSettings` e `SGGetChannelSettings` per salvare e caricare le impostazioni.
- Alternativamente si possono usare direttamente le API `VDIIDC` e sebbene sia leggermente più complesso, questo metodo potrebbe essere più appropriato in alcune applicazioni.

Nota: Le API `VDIIDC` possono essere usate solo con l'IIDC Video Digitizer. Questo componente è identificato dal sottotipo `vdSubtypeIIDC`.

Spesso le diverse impostazioni IIDC sono dipendenti tra loro e la modifica di una può causare cambiamenti anche di un'altra; questo si verifica ad esempio per il guadagno (Gain), l'otturatore (Shutter) e le impostazioni di esposizione (Exposure settings). Quindi, se non si opera correttamente, è possibile impostare la telecamera in modo da ottenere immagini completamente inutilizzabili. Inoltre telecamere differenti possono rispondere in modo diverso allo stesso set di parametri.

Una telecamera IIDC può supportare o meno alcune feature, è possibile sapere quali usando una delle API VDIIDCGetFeatures, tale informazione viene restituita sotto forma gerarchica di atomi QuickTime come mostrato in **figura 4.3**.

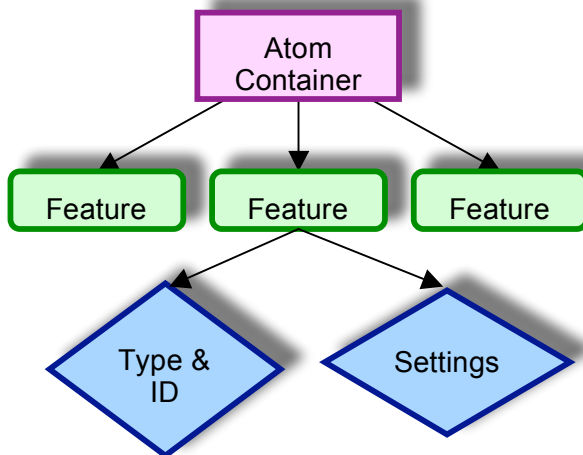
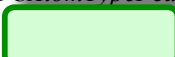



Figura 4.3: IIDC Atom Hierarchy.

<p><i>VdIIDCAtomTypeFeature</i></p> 	<p>Al primo livello della gerarchia c'è una serie di atomi VdIIDCAtomTypeFeature, uno per ogni caratteristica che l'atom container descrive.</p>
	<p>All'atomo feature sono associati due tipi di atomi figli : uno riguarda informazioni generali sulla feature e l'altro informazioni sulla sua impostazione.</p> <p>VdIIDCAtomType<xxx>Settings VdIIDCAtomID<xxx>Settings</p> <p>questi atomi mantengono le impostazioni per una data feature. La maggior parte delle feature usa lo stesso tipo di atomo per contenere le loro impostazioni, comunque quello può variare molto da feature a feature come dimostrato dalle feature Trigger e Focus Point.</p>

Segue, a titolo di esempio, il codice per la configurazione del guadagno utilizzando le API VDIIDC.

```

ComponentResult ConfigureGain(SGChannel inChannel)
{
    QTAtomContainer    atomContainer;
    QTAtom             featureAtom;
    VDIIDCFeatureSettings settings;
    VideoDigitizerComponent vd;
    ComponentDescription desc;
    ComponentResult    result = paramErr;

    if (NULL == inChannel) goto bail;
    
```

```

// get the digitizer and make sure it's legit
vd = SGGetVideoDigitizerComponent(inChannel);
if (NULL == vd) goto bail;

GetComponentInfo((Component)vd, &desc, NULL, NULL, NULL);
if (vdSubtypeI IDC != desc.componentSubType) goto bail;

// return the gain feature in an atom container
result = VDIIDCGetFeaturesForSpecifier(vd, vdI IDCFeatureGain,
                                       &atomContainer);

if (noErr == result) {

    // find the feature atom
    featureAtom = QTFindChildByIndex(atomContainer, kParentAtomIsContainer,
                                    vdI IDCAtomTypeFeature, 1, NULL);
    if (0 == featureAtom) { result = cannotFindAtomErr; goto bail; }

    // find the gain settings from the feature atom and copy the data
    // into our settings
    result = QTCopyAtomDataToPtr(atomContainer,
                                QTFindChildByID(atomContainer, featureAtom,
                                                vdI IDCAtomTypeFeatureSettings,
                                                vdI IDCAtomIDFeatureSettings, NULL),
                                true, sizeof(settings), &settings, NULL);

    if (noErr == result) {
        /* When indicating capabilities, the flag being set indicates that
           the feature can be put into the given state.
           When indicating/setting state, the flag represents the
           current/desired state. Note that certain combinations of flags
           are valid for capabilities(i.e. vdI IDCFeatureFlagOn |
           vdI IDCFeatureFlagOff) but are mutually exclusive for state.
        */
        // is the setting supported?
        if (settings.capabilities.flags & (vdI IDCFeatureFlagOn |
                                           vdI IDCFeatureFlagManual |
                                           vdI IDCFeatureFlagRawControl)) {

            // set state flags
            settings.state.flags = (vdI IDCFeatureFlagOn |
                                   vdI IDCFeatureFlagManual |
                                   vdI IDCFeatureFlagRawControl);
            // set value - will either be 500 or the max value supported by
            // the camera represented in a float between 0 and 1.0
            settings.state.value = (1.0 / settings.capabilities.rawMaximum) *
                                   ((settings.capabilities.rawMaximum > 500)
                                    ? 500 :
                                    settings.capabilities.rawMaximum);
            // store the result back in the container
            result = QTSetAtomData(atomContainer,
                                  QTFindChildByID(atomContainer,
                                                  featureAtom,
                                                  vdI IDCAtomTypeFeatureSettings,
                                                  vdI IDCAtomIDFeatureSettings, NULL),
                                  sizeof(settings), &settings);

            if (noErr == result) {
                // set it on the device
                result = VDIIDCSetFeatures(vd, atomContainer);
            }
        } else {
            // can't do it!
            result = featureUnsupported;
        }
    }
}

bail:
return result;
}

```

5. Conclusioni e sviluppi futuri

Il sistema formato da Apple Mac mini e telecamere iSight ha nella compattezza il suo punto di forza. Inoltre il peso non raggiunge i 2Kg. Queste caratteristiche permettono al sistema di operare agevolmente a bordo di un robot mobile.

Fondamentalmente sono due i possibili sviluppi che potrà avere questo lavoro:

- Migliorare le prestazioni sviluppando un Video Digitizer
- Utilizzo del sistema per la visione stereoscopica

Il primo di questi sviluppi non è necessario qualora le prestazioni raggiungibili utilizzando le funzioni base del Sequence Grabber siano compatibili con il sistema che si vuole realizzare. Comunque, qualora si intenda procedere lungo questa strada (ampiamente sconsigliata dal manuale), è allegato a questa relazione anche un codice (*SoftVDigX*) che, pur non riguardando direttamente quanto è stato descritto in questa relazione, si propone di fornire una traccia guida per coloro che intendano sviluppare dei Video Digitizer.

Il secondo punto è quello che può avere un più ampio sviluppo; infatti la presenza di due telecamere rende possibile la realizzazione di applicazioni basate sulla visione stereoscopica, ovvero la possibilità di ricavare a partire dalle immagini informazioni sulla terza dimensione spaziale, che risultano molto utili in campo robotico.

Bibliografia

- [1] QuickTime API Reference:
<http://developer.apple.com/documentation/QuickTime/APIREF/QuickTimeAPIReference.pdf>
- [2] Sito Apple: <http://www.apple.com>
- [3] Informazioni sul Apple Mac mini: <http://www.apple.com/macmini/>
- [4] Sito sviluppatori applicazioni QuickTime: <http://developer.apple.com/quicktime/>
- [5] Documentazione Sequence Grabber Components:
<http://developer.apple.com/documentation/QuickTime/APIREF/UtilityFunctionsforSequenceGrabberChannelComponents.htm>
- [6] Documentazione Picture:
<http://developer.apple.com/documentation/mac/QuickDraw/QuickDraw-333.html>
- [7] Utilizzo delle Picture: <http://www.mactech.com/articles/mactech/Vol.02/02.04/Pictures/>
- [8] Utilizzo delle Picture:
http://www.mactech.com/articles/develop/issue_20/063-064_Graphical_Truffles.html
- [9] Documentazione funzioni IIDC:
http://developer.apple.com/documentation/QuickTime/WhatsNewQT6_4/Chap1/chapter_1_section_14.html#apple_ref/doc/uid/TP30000902-CH240-BBCDAECB
- [10] Software HackTV:
Carbon:<http://developer.apple.com/samplecode/HackTVCarbon/HackTVCarbon.html>
- [11] Software SoftVDigX:
<http://developer.apple.com/samplecode/SoftVDigX/SoftVDigX.html>
- [12] Software BigEasyVideoGrabber:
<http://developer.apple.com/samplecode/BigEasyVideoGrabber/BigEasyVideoGrabber.html>

SOMMARIO	1
1. INTRODUZIONE	1
1.1. QuickTime	1
1.1.1. Cos'è QuickTime	1
1.1.2. Architettura	1
1.2. Il Sequence Grabber	3
1.2.1. Componenti Sequence Grabber	3
1.3. Il processo di cattura video	4
1.3.1. Il Sequence Grabber	4
2. IL PROBLEMA AFFRONTATO	5
3. LA SOLUZIONE ADOTTATA	5
4. MODALITÀ OPERATIVE	6
4.1. Componenti necessari	6
4.2. Inizializzazione di un canale video	7
4.2.1. La sequenza delle operazioni da eseguire	7
4.2.2. Un esempio concreto: HACKTVCarbon	14
4.2.3. Un altro esempio: BigEasyVideoGrabber.c	17
4.3. Il controllo del Sequence Grabber	18
4.4. Acquisizione dei dati da più telecamere	18
4.5. Come acquisire un frame dal flusso dei dati	19
4.5.1. La struttura dati picture	20
4.5.2. Un esempio applicativo: EasyVideoGrabberTest.c	21
4.6. Modalità di taratura	22
4.6.1. La taratura di una telecamera	22
4.6.2. Funzioni IIDC Digitizer	22
4.6.3. Configurare le caratteristiche della telecamera	23
5. CONCLUSIONI E SVILUPPI FUTURI	26
BIBLIOGRAFIA	26