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

Laboratorio di Robotica Avanzata
Advanced Robotics Laboratory

Corso di Robotica Mobile
(Prof. Riccardo Cassinis)

Sistema di ausilio al puntamento delle telecamere

Elaborato di esame di: **Paolo Lucchetti**

Consegnato il: 21 luglio 2007

Sommario

In questo documento vengono analizzati due possibili metodi di implementazione che si possono adottare per la realizzazione di un programma per l'ausilio al puntamento delle telecamere. Il software è stato sviluppato in ambiente Linux ed è stata utilizzata, modificata e ampliata la libreria vislib-1.9.1 disponibile per il linguaggio C. Durante l'elaborato verranno fatti riferimenti espliciti al manuale della stessa.

In appendice, inoltre, verrà trattato il problema relativo all'installazione di webcam su sistemi dotati di schede grafiche che supportano una profondità di 32 bit riscontrato in fase iniziale di progetto.

1. Programma per l'ausilio al puntamento delle telecamere

1.1. Introduzione

Il sistema realizzato, che consiste nel programma *acquisizionePiccola.c* o *acquisizioneGrande.c* (che si trovano nella directory */example/* della libreria vislib-1.9.2, vedi paragrafo 2.7), consente, partendo da un'immagine precedentemente salvata oppure no, di confrontare continuamente la prima immagine con le successive, restituendo un valore percentuale che esprime il grado di correlazione e quindi di somiglianza delle immagini in esame, fino all'interruzione manuale dello stesso. In questo modo è possibile innanzitutto capire di quanto si sia spostata la webcam dall'immagine di origine e quindi riportarla intuitivamente e in modo visuale, grazie ad un escamotage adottato, nella posizione iniziale (si veda il paragrafo 1.4).

1.2. Problema affrontato

Poter calcolare in modo efficiente ed ottimale la correlazione che esiste tra due immagini, per capire di quanto si sia spostata la webcam dalla posizione di partenza, non è un problema semplice da affrontare in quanto vi sono diverse alternative che possono portare ad una soluzione più o meno precisa.

Sono state valutate due possibili soluzioni; la seconda risulterà essere la migliore.

1.3. Correlazione bidimensionale

La prima alternativa è stata quella di calcolare il grado di somiglianza delle immagini a confronto utilizzando la correlazione bidimensionale descritta dalla formula:

$$\varphi(m,n) = \sum_{y=1}^{M} \sum_{x=1}^{N} [a(x,y) * b(x+n, y+m)]$$

dove a e b sono le immagini di riferimento, M e N l'altezza e la lunghezza e la coppia (m,n) determina la traslazione della seconda immagine rispetto alla prima.

La funzione di correlazione è stata calcolata per *(m,n) = (0,0)* in quanto si è voluto valutare semplicemente il valore della correlazione nell'origine, in modo da ridurre il numero di confronti e per

evitare di porsi il problema relativo ai bordi. Tale questione, che si introduce facendo appunto variare m e n, crea infatti delle difficoltà nella stima della correlazione, in quanto la zona presente nella prima immagine ma non più inquadrata nella seconda risulta essere determinante nella stima della correlazione stessa per riconoscere il movimento della telecamera attorno ad un punto. Per piccoli spostamenti i risultati sono migliori, perché non ci si aspetta che le immagini abbiano molte discordanze tra loro, per cui il matching dato dalla correlazione è abbastanza affidabile.

Avendo optato per il calcolo della correlazione bidimensionale in *(m,n) = (0,0),* anche per rendere più semplice e veloce l'algoritmo, si è riscontrato un ulteriore problema. Nasce infatti la possibilità che, in instanti successivi, vengano inquadrate zone che inizialmente non rientrano nell'immagine di riferimento. Essendo l'autocorrelazione di un'immagine calcolata come la somma dei contributi dei singoli pixel (come mostra la formula riportata sopra), può succedere che l'immagine traslata abbia autocorrelazione maggiore rispetto a quella di partenza; si ottiene in questo caso in uscita un valore di correlazione percentuale maggiore del 100%.

L'algoritmo implementato è risultato corretto solo in un intorno molto piccolo dell'immagine presa in considerazione. Per questo si è pensato di utilizzare, con opportune modifiche, la *motion detection (vedi paragrafo 3.9 del manuale della vislib-1.9.1 a pag. 24),* che implementa tecniche più sofisticate e permette così di migliorare le prestazioni del software realizzato.

1.4. Motion Detection

La seconda alternativa, come anticipato nel paragrafo precedente, ha previsto l'utilizzo della motion detection e delle relative funzioni definite nella vislib-1.9.1 nel file */src/motion.c.* Tali routine sono in grado di analizzare due frame acquisiti consecutivamente e, grazie alla funzione *vlMotionRGBImage (vedi paragrafo 3.9 del manuale della vislib a pag.24),* di mostrarne la variazione di pixel, ammettendo una certa soglia di errore stabilita con la *vlMotionThreshold*.

Grazie all'utilizzo di queste funzioni di libreria, opportunamente modificate per far sì che potessero funzionare correttamente anche partendo da un'immagine precedentemente acquisita e salvata su Hard-Disk (per le modifiche vedi paragrafo 2.1) e grazie a delle macro definite *ad-hoc*, sono stati implementati due programmi, *correlazioneGrande.c* (per immagini 640x480 pixel) e *correlazionePiccola.c* (per immagini 320x240 pixel). Questi permettono di calcolare in modo molto preciso la correlazione tra due immagini, mantenendo in trasparenza sempre il contorno dell'immagine di riferimento in modo tale da poter ricalibrare la webcam in modo visuale e poter quindi ovviare anche a quei fastidiosi problemi dovuti al riflesso o alla variazione di luce che alterano il valore della correlazione anche se le due immagini sono perfettamente sovrapponibili.

2. Modifiche apportate alla libreria vislib-1.9.1

Le modifiche apportate hanno interessato gran parte dei file della libreria vislib-1.9.1 e quindi si è giunti ad una nuova versione chiamata vislib-1.9.2. Nei paragrafi successivi verranno commentate solo le parti aggiunte quindi, per comprendere a fondo il corretto funzionamento delle funzioni modificate, meglio riferirsi al manuale allegato alla libreria.

2.1. Modifiche in /src/motion.c

In questo file sono state modificate le seguenti parti:

- Modifica della funzione ***int*** *vlMotionInit(****void****)* in modo tale da poter caricare un'immagine da file.
- Inserimento della funzione ***int*** *vlMotionInitGrande(****void****)* per l'inizializzazione della motion detection per immagini 640x480 pixel.

- Modifica della funzione ***int*** *vlMotionRGBImage(****vlImage *pic, vlWindow *window, vlImage *dest)*** in modo tale da poter mostrare a video il valore di correlazione tra le immagini confrontate.

2.2. Modifiche in /src/display.c

Con riferimento a quanto spiegato in appendice nel paragrafo 5.3 si è solamente inserito un *case* per far funzionare correttamente la webcam per dispositivi video che supportano una profondità anche di 32 bit.

2.3. Modifiche in /src/common.c

Definizione della funzione ***int*** *vlInitGrande(****void)*** in modo tale da poter inizializzare correttamente il grabber, il display e la motion detection per immagini 640x480 pixel.

2.4. Modifiche in /include/vislib.h

In questo file sono state definite le opportune macro per le immagini 640x480 pixel.

2.5. Modifiche in /include/vlMotion.h

In questo file sono stati inclusi i prototipi delle funzioni definite in /src/motion.c

2.6. Modifiche in /include/vlCommon.h

In questo file sono state aggiunte le macro per la creazione delle immagini e per l'inizializzazione del display in formato 640x480 pixel.

2.7. Programmi creati in /examples/

In questa cartella sono stati inseriti quattro programmi:

- ***acquisizionePiccola.c*** che consente di acquisire un unico frame di grandezza 320x240 pixel e salvarlo su Hard-Disk;
- ***acquisizioneGrande.c*** che consente di acquisire un unico frame di grandezza 640x480 pixel e salvarlo su Hard-Disk;
- ***correlazionePiccola.c*** che consente di calcolare in modo continuativo la correlazione tra l'immagine (320x240 pixel) di partenza, precedentemente salvata o meno, e quella corrente (320x240 pixel), mostrando sia un output testuale, contenente il valore della correlazione percentuale, sia uno visivo, che permette in modo intuitivo di spostare la webcam in modo tale da riportarla nelle posizione corretta;
- ***correlazioneGrande.c*** che consente di calcolare in modo continuativo la correlazione tra l'immagine (640x480 pixel) di partenza, precedentemente salvata o meno, e quella corrente (640x480 pixel), mostrando sia un output testuale, contenente il valore della correlazione percentuale, sia uno visivo, che permette in modo intuitivo di spostare la webcam in modo tale da riportarla nelle posizione corretta.

3. Modalità operative

Per far funzionare correttamente il software realizzato controllare di essere dotati di quanto riportato in seguito:

- Una webcam
- Sistema Operativo Unix/Linux
- Compilatore C (gcc), qualsiasi versione. L'ultima testata è la 4.1
- Vislib versione 1.9.2 o superiore

Procedere quindi con i seguenti passi:

- Decomprimere il file vislib-1.9.2.tar.gz in qualsiasi posizione
- Lanciare il comando make nella cartella principale in modo tale da ricompilare l'intera libreria
- Per ogni modifica che viene apportata successivamente ai file basta ricompilare il file stesso con il comando *make <nome_file>* (nel caso si tratti di un file di libreria è necessario ricompilare l'intera libreria come spiegato nel passo precedente).

> ➢ **Il software prodotto è stato testato sia per la telecamera PHILIPS messa a disposizione sia per una della CREATIVE e tutto funzionava correttamente. Non è possibile garantire il corretto funzionamento per qualsiasi telecamera esistente.**

4. Conclusioni

Al termine del lavoro si è potuto osservare il corretto funzionamento dei programmi realizzati, la sensibilità e la velocità di esecuzione degli stessi anche su architetture non potenti.

5. Appendice - Installazione web-cam PHILIPS

5.1. Introduzione

Le webcam, sotto Linux, vengono gestite da un modulo del kernel che si chiama Video For Linux, abbreviato con v4l[1]. Questo modulo acquisisce il flusso delle immagini provenienti dalla webcam e le rende disponibili, attraverso delle API, in un dispositivo a caratteri che di solito corrisponde al file */dev/video0* quindi per il corretto funzionamento di una qualsiasi webcam controllare di avere il modulo correttamente installato e funzionante utilizzando il comando *lsmod | grep v4l*, altrimenti sarà necessario ricompilare il kernel includendo tale modulo.

Per poter acquisire le immagini dalla webcam si è utilizzata la libreria vislib[2] disponibile gratuitamente per linguaggio C in ambiente Linux.

5.2. Problema affrontato

Il problema affrontato riguarda il corretto funzionamento della webcam per quei sistemi dotati di scheda video che supportano una depth (profondità di visualizzazione dell'immagine) anche uguale a 32 bit. Tale problema si è manifestato la prima volta quando si è eseguito il programma di acquisizione video di esempio, incluso nella vislib, */exaples/videoin.c*. L'output del programma è stato un messaggio di errore (_vlDisplayImage: depth=32 is not supported).

Il problema si presenta in quanto nel file */src/display.c* non è contemplato l'utilizzo di profondità maggiori di 24 bit e quindi, dato che la funzione di libreria del serverX (*XGetVisualInfo*), utilizzata per reperire le informazioni sull'Hardware, restituiva il valore 32, veniva generato un errore.

5.3. Soluzioni adottate

Prima di trovare una soluzione al problema, si è controllato se l'Hardware supportasse realmente una profondità pari a 32 bit, come restituito in errore, utilizzando il comando *xdpyinfo* che esegue una chiamata di sistema e recupera direttamente dall'Hardware le informazioni necessarie.

Una volta verificato che la scheda video supportasse effettivamente la profondità di 32 bit, si sono adottate due strategie differenti per risolvere il problema.

- Si è forzato l'hardware a lavorare a 24 bit, modificando quindi il file *display.c* e impostando manualmente una *depth=24*, evitando di utilizzare il valore restituito dalla funzione *XGetVisualInfo*.
- Si è inserito nello *switch/case* del medesimo file l'opzione per l'hardware a 32 bit.

Tra le 2 soluzioni si è adottata la seconda in quanto si può adattare dinamicamente a diversi tipi di schede video e non solo così a quelle con depth uguale a 24 bit.

6. Bibliografia

[1] Sito v4l - http://www.exploits.org/v4l

[2] Sito ActiveMedia Robotics – http://www.activerobots.com
http://robots.activemedia.com

[3] Linguaggio C – http://it.wikipedia.org/wiki/C_(linguaggio)