Alma Mater Studiorum · Università di Bologna

Facoltà di scienze matematiche, fisiche e naturali

Corso di Laurea Specialistica in Informatica

Miglioramento della fase d'acquisizione in un sistema di fast reverse engineering

Relatore:
Dott. Giulio Casciola

Candidato:
Andrea Zoli

Co-relatore:
Dott. Serena Morigi

Sessione II
Anno Accademico 2009-2010

Ai miei familiari
e a chi ha creduto in me.

Indice

Elenco delle figure

Capitolo 1

Introduzione

Verranno introdotti i concetti e le tecnologie che si occupano attualmente del processo di reverse engineering assistiti dal calcolatore. Il processo di reverse engineering ha lo scopo, in questo ambiente, di ricavare un modello tridimensionale al computer partendo da un oggetto reale. Il modello potrà essere successivamente utilizzato da applicazioni di CAD (Computer-Aided Design), come oggetto virtuale per promuovere prodotti online o all'interno di giochi. Verrà successivamente introdotto F.I.R.E.S. che è l'acronimo di Fast Interactive Reverse Engineering System. Questo sistema si basa su stereovisione ibrida, ovvero, due camere ad infrarossi e una smart-pen wireless. Quest'ultima viene utilizzata interattivamente dall'utente per acquisire le linee di stile toccando con la punta della penna un oggetto reale. Il sistema genererà automaticamente una mesh a bassa risoluzione refinita attraverso superfici di suddivsione.

Con questa tesi si cerca di migliorare F.I.R.E.S, lavorando principalmente sulla fase d'acquisizione. Si vuole intervenire sul software, rivedendo ed ottimizzando il codice e cercando di migliorare il supporto ai led infrarossi, ma anche sull'hardware del sistema con l'intenzione di creare un nuovo prototipo che abbia caratteristiche migliori. Si vorrebbe introdurre, inoltre, un dispositivo inerziale in aggiunta a quello visivo che ci consente di effettuare la fusione dei sensori al fine di migliorare l'accuratezza del sistema. La fusione

dei sensori può essere effettuate utilizzando un filtro di Kalman che è uno stimatore ottimo per questo tipo di problemi. Verranno effettuati, infine, dei test di precisione per comprendere se, grazie alla fusione dei sensori, si otterrebbe un miglioramento effettivo valutando, quindi, l'utilià del dispositivo inerziale in questo tipo di sistema.

Capitolo 2

Reverse Engineering

In questo capitolo introdurremo concetti e tecnologie che si occupano attualmente del processo di reverse engineering assistiti dal calcolatore, vedi [1]. Il processo di reverse engineering in questo ambiente ha lo scopo, come vedremo, di ricavare un modello tridimensionale al computer a partire da un oggetto reale. Questo processo verrà osservato più nel dettaglio spiegando le due fasi di scansione e ricostruzione. Verrano anche mostrate le proprietà e i parametri con cui questi sistemi vengono valutati.

2.1 Computer-Aided Engineering (CAE)

Per ingegneria assistita dall'elaboratore (CAE) si intende, in generale, l'utilizzo di sistemi informatici di supporto nel processo di realizzazione di un prodotto come, ad esempio, CAD+CAM. I sistemi di progettazione assistita dall'elaboratore (CAD, dall'inglese Computer Aided Design) rivoluzionarono la disciplina ingegneristica negli anni '80-'90; il boing 777, ad esempio, è stato progettato e preassemblato utilizzando soltanto un sistema di questo tipo. I più recenti processi di produzione automatizzati con macchine a controllo numerico (CAM, dall'inglese Computer-Aided Manufacturing)permettono la produzione di oggetti direttamente da descrizioni CAD.

2.2 Computer-Aided Reverse Engineering (CARE)

L'ingegneria inversa (Reverse Engineering) viene definita generalmente come il processo di analisi e ricerca di principi tecnologici di un dispositivo, oggetto o sistema. Tramite le informazioni ottenute con l'ingegneria inversa è possibile, dunque, creare un nuovo dispositivo, oggetto o sistema con caratteristiche simili o identiche all'originale. Nel caso specifico dell'ingegneria inversa con assistenza dell'elaboratore (CARE) si intende il processo inverso a CAE ovvero la creazione di un modello CAD al computer a partire dall'oggetto reale.

In molti ambiti è possibile trarre vantaggio dal modello virtuale 3D ricostruito.

- Collaborative design: Mentre i sistemi CAD sono utili per la progettazione di parti, in qualche caso il metodo di progettazione più intuitivo è l'interazione fisica con un modello. Spesso le compagnie pagano degli scultori per la progettazione di questi modelli con materiali come la creta. Una volta che la scultura è pronta, questa viene digitalizzata e ricostruita al computer.

- Manufattura: Molti dei prodotti vengono attualmente progettati usando sistemi CAD; tuttavia può anche capitare che una parte meccanica esista ma non si trovi più il modello al computer per riprodurlo (ad esempio:non esiste più il produttore originale o la documentazione del prodotto è andata persa). È possibile, dunque, rimuovere la parte dal sistema in funzione e digitalizzarla. Un altro utilizzo diffuso è quello dello spionaggio industriale; è possibile, infatti, disassemblare un prodotto della concorrenza e tramite reverse engineering analizzarlo per comprenderne le funzionalità.

- Controllo qualità: Esistono figure professionali, interne o esterne alle aziende interessate, addette all'ispezione e/o controllo della qualità;

tramite i sistemi di reverse engineering è possibile confrontare i prodotti fabbricati con i modelli CAD o con gli standard.

- Medicina: Anche in questo settore è ampio l'utilizzo di sistemi di reverse engineering; si possono creare, infatti, protesi su misura, con un'ottima precisione partendo dalle dimensioni originali. I chirughi plastici, ad esempio, possono usare la forma del viso di un paziente per modellare e visualizzare i risultati di un possibile intervento.

- Musei: gli artefatti di un museo rappresentano un classico esempio di oggetti che attraggono l'interesse di molte persone. Per visionare opere d'arte, normalmente, si viagga anche in musei molto distanti e con le foto si ottengono immagini che non sono interattive. Digitalizzando le opere d'arte i curatori dei musei possono rendere disponibile una visualizzazione interattiva dell'opera stessa (ad esempio il progetto digitale Michelangelo).

- Mondo virtuale, giochi ed effetti speciali: Le immagini computerizzate sono sempre più utilizzate come componente importante di film, giochi e realtà virtuale. Tutti queste applicazioni richiedono modelli 3D che possono essere presi dalla realtà o create appositamente da artisti.

- Web commerce: Il World Wide Web fornisce un'infrastruttura per l'interazione su internet; le aziende produttrici usano sempre più internet come canale di comunicazione per vendere prodotti. Grazie a modelli 3D è possibile visualizzare interattivamente i prodotti.

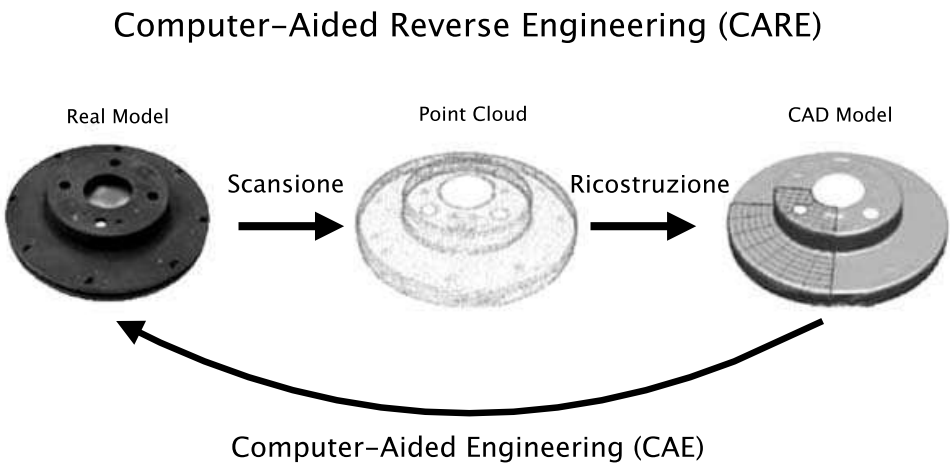


Figura 2.1: Computer-Aided Reverse Engineering (CARE)

Il processo di CARE tradizionale è suddivisibile in due passi fondamentali (vedi fig. 2.1): la scansione (o misurazione) di un oggetto esisitente e la ricostruzione di tale oggetto in un modello geometrico tridimensionale. L'output del processo di acquisizione è, in genere, una nuvola di punti (Point Cloud), cioè un insieme di punti 3D a cui non è associata alcuna informazione topologica, oppure, in dipendenza del sistema preso in considerazione, una serie di scansioni multiple allineate. Per poter utilizzare le informazioni acquisite nei software di progettazione occorre quindi effettuare una ricostruzione in un modello geometrico come una mesh triangolare o un insieme di NURBS. I sistemi che utilizzano scansioni multiple necessitano di effettuare un passo prima della ricostruzione, ovvero, l'allineamento. Questo significa definire un sistema di riferimento unico per le varie acquisizione fatte in modo da consentirne una successiva ricostruzione topologica.

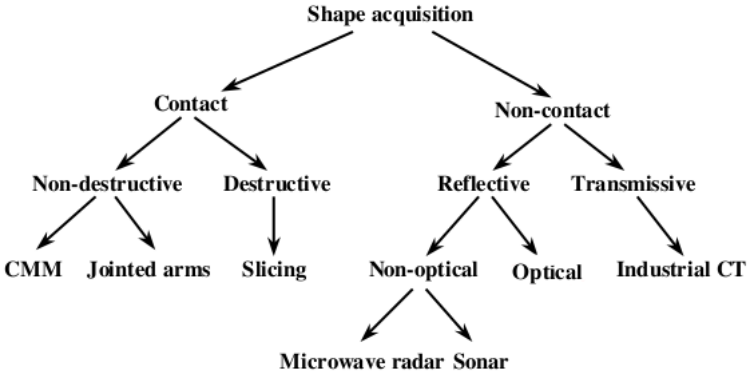


Figura 2.2: Tassonomia delle tecnologie di scansione

2.3 Dispositivi di scansione

I dispositivi di scansione 3D sono molti ed utilizzano differenti tecnologie ma, tuttavia, possono essere catalogati (vedi fig. 2.2) in due grossi gruppi in base al loro contatto o meno. Avremo modo di trattare in questa dissertazione soprattutto i sistemi a contatto ottico che rappresentano il contesto in cui stiamo operando.

Per i processi di scansione vengono generalmente considerate le seguenti metriche di valutazione:

- Precisione e risoluzione
- Ripetibilità
- Velocità
- Costo

2.3.1 Scanner a contatto

Questi dispositivi sfruttano sonde a contatto che seguono automaticamente i contorni di una superficie, sono basati su tecnologie CMM ed hanno una precisione dai +0.01 ai +0.02 mm. Se la sezione scansionata è molto grande, tuttavia, questi sistemi possono risultare lenti poichè ogni punto

viene generato sequenzialmente a partire dalla sonda. Essendo dispositivi a contatto necessario che conservino un certo grado di pressione per registrare i punti e risulta, per questo, impreciso, oltre che problematico, nel caso in cui l'oggetto da scansionare è sia di un materiale soffice (come ad esempio gomma o pelle).

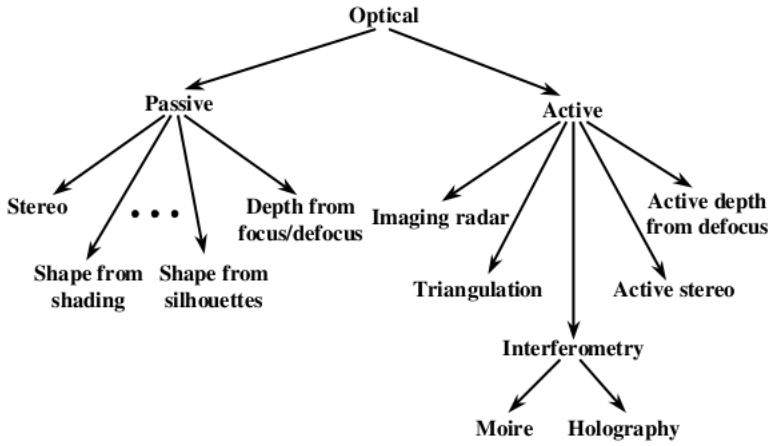


Figura 2.3: Sistemi ottici

2.3.2 Scanner non a contatto

Come è possibile vedere in fig. 2.3 le tecnologie di scansione che non sfruttano il contatto sono i sistemi ottici, come i laser, ad onde e sensori CDD. Sebbene questi sistemi consentono di catturare un largo quantitativo di dati in un tempo relativamente breve, essi hanno problematiche relative alla loro tecnologia. La tolleranza tipica di un sistema di scansionamento non a contatto è tra ± 0.025 e 0.2 mm; alcuni sistemi hanno problemi a generare dati che descrivano superfici parallele all'asse del laser. I dispositivi ottici, basandosi sulla luce, hanno difficolt nel riconoscimento di superfici riflettenti e quindi occorre rivestire temporaneamente i prodotti da scansionare. Queste problematiche limitano l'uso di questi dispositivi in aree dell'ingegneria dove la precisione delle informazioni è secondaria alla rapidità dell'acquisizione;

con la ricerca nell'ambito delle tecnologie ottiche in continuo aggiornamento, tuttavia, se ne trovano in commercio di sempre più precisi.

Prendendiamo in esame gli scanner riflessivi basati sulla rilevazione di riflessi di un qualche tipo di radiazione per sondare un oggetto o un ambiente. Gli scanner riflessivi ottici, nello specifico, possono essere attivi o passivi a seconda se il sistema ha un emettitore di luce o sfrutta quella ambiente; eccone alcuni del primo tipo:

- I *time-of-flight laser scanner* cercano la distanza di una superficie misurando il round-trip-time di un impulso luminoso. La precisione di questi sistemi dipende da quanto è precisa la misurazione del tempo.

- I *triangulation laser scanner* emette un fascio di luce su di un soggetto utilizzando una camera per comprendere la posizione della luce proiettata. Possono essere utilizzate le stesse tecniche dei sistemi di 3D stereo vision.

- I *structured light scanner* proiettano un pattern luminoso su un soggetto osservandone la deformazione del pattern.

- I *dispositivi a stereovisione attiva* hanno un proiettore o un laser che proietta uno o più punti o fasci di luce e le immagini dei punti o fasci sulle due camere viene analizzato per generare la struttura 3D dell'oggetto osservato. In generale è preferibile usare sistemi a stereovisione attiva in quanto questa gestisce oggetti smooth (che non contengono, ovvero, angoli o spigoli) e non ha bisogno di confronto tra le caratteristiche comuni delle immagini delle due camere (come angoli o spigoli) poichè la corrispondenza non è ambigua; i sistemi attivi tendono ad essere più costosi, lenti e generalmente intrusivi della controparte passiva e necessitano di un ambiente interno con luce controllata.

2.4 Ricostruzione

La point cloud generalmente definisce numerosi punti di una superficie di un oggetto secondo le coordinate 3D x, y e z più un evenutale valore per il colore. In genere i dati acquisiti sono meno e e meno chiari, in quanto rovinati da rumore; senza ulteriori processi il dato non è in una forma da poter essere utilizzata da sistemi CAD/CAM o di protipizzazione rapida. Le tecniche di ricostruzione vengono usate per modificare i dati della point cloud stabilendo una topologia di punti e combiandoli in modo da ottenere un modello 3D. Nei sistemi di scansione 3D tradizionali la fase di misurazione non richiede molto tempo a differenza del processo di ricostruzione che può impiegare anche diversi giorni.

Il processo di ricostruzione consiste generalmente nella ricerca di una mesh che interpola o approssima una superficie a partire dalla point cloud ed include l'integrazione di scansioni multiple in un singolo modello 3D. Quest'ultimo processo viene chiamato generalmente registrazione o allineamento. In genere è necessario, infatti, effettuare più di una singola scansione per coprire l'intera superficie di un oggetto. I dati vanno registrati o allineati secondo un unico sistema di coordinate in modo da ottenere un singolo modello 3D. Esistono differenti metodi:

- Mechanical tracking: lo scanner o l'oggetto viene attaccatoo ad una macchina che misura le coordinate tenendo traccia sia della posizione che dell'orientamento.

- Optical tracking: caratteristiche dell'oggetto o markers vengono usati per l'allineamento.

- Interactive alignment: l'utente fornisce due o tre punti di matching che vengono utilizzati per calcolare la trasformazione rigida.

- Automatic alignment: si basa sulla ricerca automatica di caratteristiche dell'oggetto per effettuare l'allineamento (è un'area di ricerca attiva).

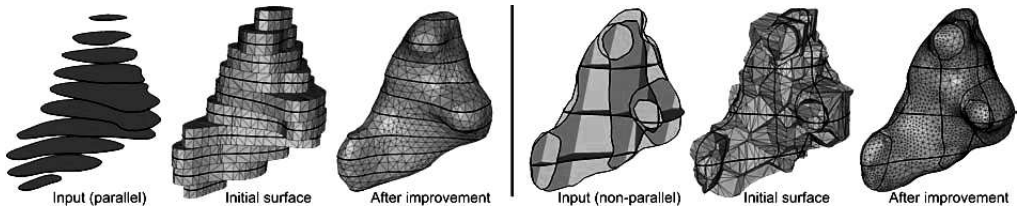


Figura 2.4: Ricostruzione basata su scansioni parallele ed incrociate

Altre tecniche differenti per la ricostruzione 3D vengono applicate se l'output non è nella forma di point cloud; la fig. 2.4 ad esempio, mostra una ricostruzione tipica per sistemi di scanning medico basata su scansioni a sezioni incrociate non parallele.

Per il processo di ricostruzione vengono generalmente considerate le seguenti metriche di valutazione:

- Automaticità
- Nessuna restrizione sulla topologia
- Efficenza spazio-temporale
- Robustezza rispetto al rumore nei dati

Capitolo 3

F.I.R.E.S.

Abbiamo visto come i processi di reverse engineering si differenzino e lavorino in scenari molto diversi con requisiti di precisione tempo e costo differenti; si possono considerare differenti soluzione in base alle diverse circostanze.

F.I.R.E.S è l'acronimo di Fast Inverse Reverse Engineering System [2] e si colloca tra i sistemi di reverse engineering che forniscono, a basso costo, rapidità nell'acquisizione lasciando la precisione come metrica di valutazione secondaria. Queste caratteristiche rendono Fires un candidato per applicazioni home user il cui processo di sviluppo è orientato a questo nuovo mercato. La misurazione 3D viene effettuata con un sistema ibrido di stereovisione attiva a contatto che impiega due camere infrarossi e una penna guidata dall'utente che è in realtà un semplice emettitore di raggi infrarossi. Le camere si occupano di gestire la stereovisione attiva come un normale sistema ottico non a contatto, mentre la penna viene usata come una sonda per effettuare una scansione a contatto.

Considerano i costi di un sistema simile, questi sono molto bassi, l'equipaggiamento è formato da:

- Due Wiimotes Remote Controllers usati come camere ad infrarossi
- Un emettitore led ad infrarossi che viene rilevato dalle camere

- Un personal computer con supporto bluetooth

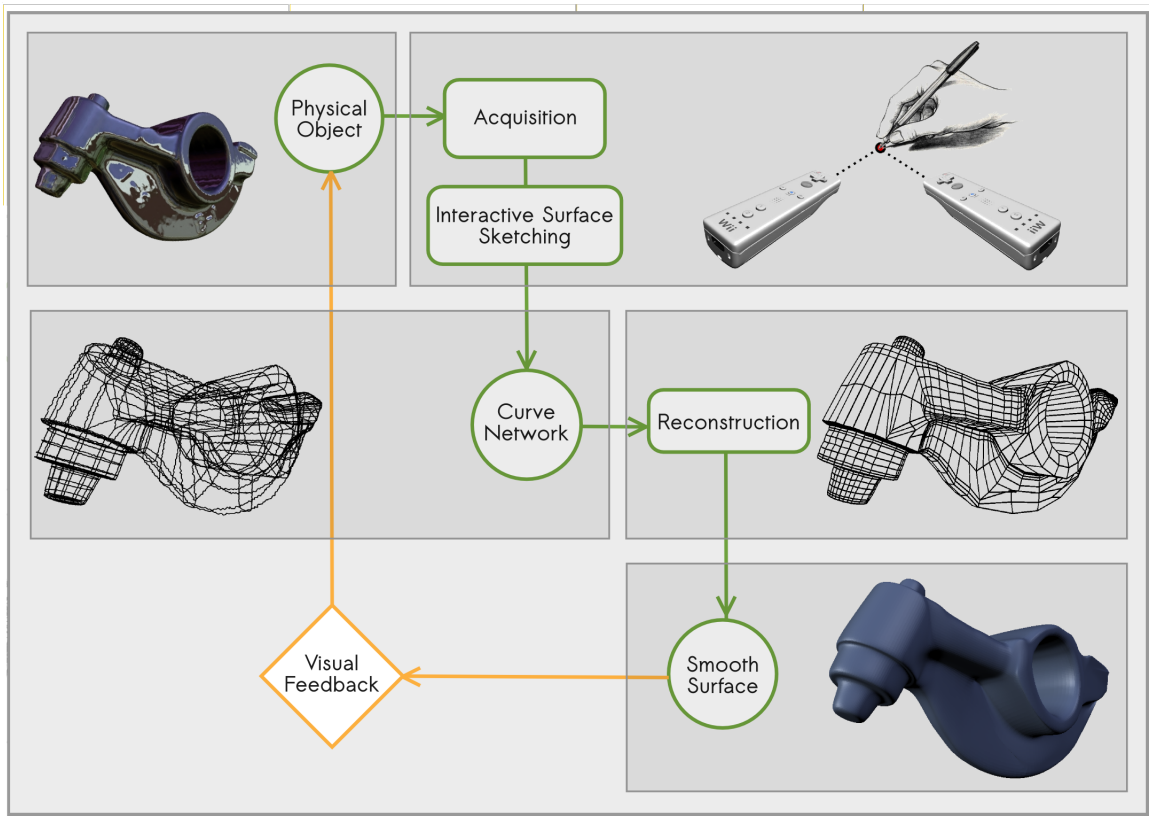


Figura 3.1: Architettura di Fires

Questi dispositivi sono la base necessaria per la creazione di un sistema di scansione 3D che permetta l'interazione uomo-macchina, ma se ne possono usare anche di differenti o aggiuntivi, come ad esempio camere non ad infrarossi e markers, ottenendo così un sistema di stereovisione passiva, oppure passare a dispositivi inerziali. Nei sistemi classici di stereovisione la precisione dipende principalmente dalla risoluzione delle camere utilizzate e nella dimensione e qualità dell'emettitore di luce. In Fires, invece, la precisione è limitata dall'utilizzo umano che sostituisce il movimento meccanico di proiettori di luce. Il coinvolgimento dell'utente nel processo di misura, tuttavia, ha i suoi vantaggi:

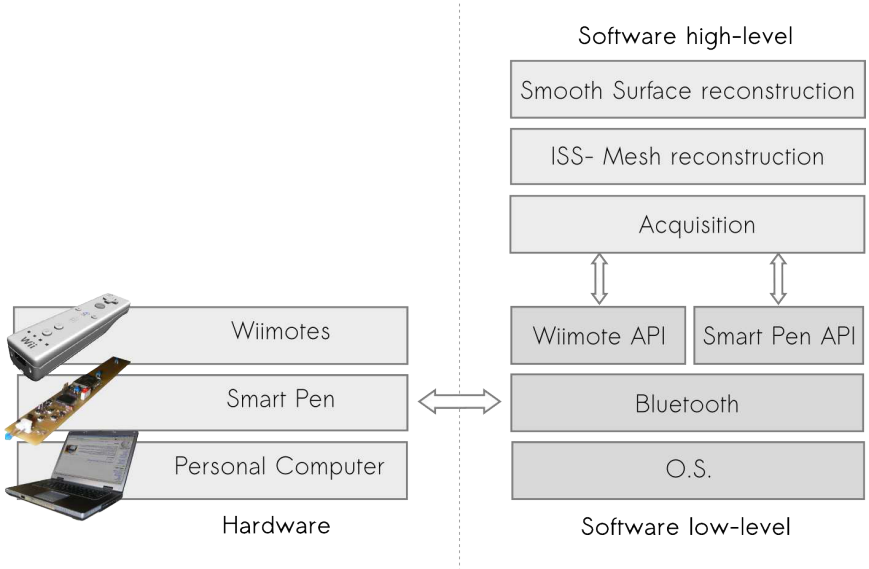


Figura 3.2: Fast interactive reverse engineering using low-cost hardware

- Semplifica i casi di ambiguità e riduce le risorse necessarie per la ricostruzione, delegando all'utente la responsabilità di definire la topologia dell'oggetto interagendo con il sistema.
- Permette di intervenire aggiungendo modificando o eliminando delle misurazione durante il processo di acquisizione.
- Consente di rilevare le caratteristiche degli oggetti come gli angoli e gli spigoli.

Fires, essendo un sistema ibrido a contatto, non si basa su point cloud ma su di un processo di disegno incrementale dove vengono acquisite delle linee di stile della superficie dell'oggetto; questo processo di acquisizione, chiamato interactive surface sketching, produce un network di curve gestito internamente come una mesh polyline: un insieme di facce, vertici, lati e polyline. Il processo di ricostruzione trasforma una mesh polyline in una superficie morbida tramite tre passi:

- ricostruzione di una tri-quad mesh per via di un processo di triquadrificazione

- ricostruzione di una mesh base attraverso bilinearly blending coons patches

- ricostruzione della superficie smooth tramite subdivision

Le caratteristiche realtime e di interattività di Fires permettono all'utente di rifinire il modello in maniera incrementale e di visualizzare il processo in corso grazie ad un feedback visivo. Queste caratteristiche identificano una precisa tipologia di sistema chiamato Fast Interactive Reverse Engineering System, ovvero F.i.r.e.s.. L'hardware e il software del sistema Fires è mostrato in fig. 3.2.

3.1 Scansione

Il sistema di scansione/acquisizione di Fires utilizza le camere ad infrarossi di due Wiimotes Controller di una comune console Nintendo Wii. Le due camere, fisse, sono disposte parallele e la penna, composta da quattro led infrarossi, opera nell'area di visibilità definita dal posizionamento delle due camere [4]. Tramite i punti 2D rilevati da queste è possibile effettuare una triangolazione ed ottenere così punti tridimensionali; una volta ottenuti (ne avremo quattro) viene calcolato il centroide e, grazie alla stima della retta passante per i punti, viene calcolata la posizione della punta della penna. Il movimento di quest'ultima definisce una linea di stile che viene filtrata per eliminare il rumore e, quindi, aggiunta in una struttura dati polyline network sulla quale viene applicato successivamente il processo di ricostruzione realtime.

3.1.1 Wii Remote Controller

Il Wii remote controller, abbreviato in wiimote, è il dispostivio di input principale della console Nintendo Wii, usato come interfaccia principale per l'interazione tra l'uomo e il sistema. Questo avviene grazie alla capacità di

Figura 3.3: Un wiimote da diverse angolazioni

rilevare punti e movimenti attraverso sensori ottici e accelerometrici. I wiimotes comunicano con la console Wii via Bluetooth attraverso l'interfaccia HID (Human Interface Device); i dati inviati non sono criptati e non è necessario effettuare un'autenticazione. Grazie a queste caratteristiche è possibile sfruttare a pieno le funzionalità dei wiimotes su ogni computer che disponga dell'interfaccia bluetooth. Da notare, inoltre, che i wiimotes non utilizzano tecniche innovative; il loro successo dato dalla contenuta spesa per l'acquisto.

I Wiimotes sono composti da:

- una camera che permette di rilevare fino a 4 punti infrarossi con frequenza d'onda tra gli 850 e i 950 nm alla velocità di 100 Hz. La risoluzione della camera è di soli 128 x 96 pixel ed utilizza l'analisi subpixel 8x per ottenere una risoluzione virtuale di 1024x768. È possibile, inoltre, rimuovere il fitro IR per tracciare qualsiasi oggetto luminoso.

- un accelerometro 3-axis

- 11 bottoni e una porta esterna per collegare dispositivi aggiuntivi.

Non esistendo delle specifiche tecniche per la camera occorre, quindi, effettuare una calibrazione dei wiimotes che è il processo con cui vengono trovati i parametri che caratterizzano le camere. Si può dire che questo processo non offra risultati precisi, per cui necessario lavorare con valori approssimativi; diverse fonti riportano caratteristiche come il campo di vista orizzontale (FoV) di circa 41 gradi mentre quello verticale di circa 31. Conoscendo questi valori e la risoluzione del piano immagine (di 1024x768) si possono approssimare i parametri intrinsechi delle due pin-hole camera [link to marco rucci].

Per l'accesso ai dati delle due camere il sistema Fires utilizza la libreria wiiuse [3] con la quale è possibile gestire in modo semplice i due wiimotes tramite connessione bluetooth standard.

3.1.2 Smart-pen

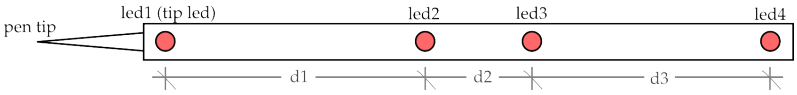


Figura 3.4: Smart-pen layout

La smart pen è un dispositivo wireless di piccole dimensioni a basso peso, per cui di facile utilizzo, ed formata da 4 emettitori ad infrarossi, due accelerometri 3-axis e un dispositivo bluetooth. I sensori inerziali raccolgono informazioni sull'accelerazione della penna e le inviano in maniera grezza tramite il dispositivo bluetooth (in maniera analoga al funzionamento dei wiimotes per i dati visivi). La frequenza di campionamento dei dati inerziali è di 40 Hz, differentemte del sistema video che lavora, invece, a 100 Hz.

I quattro led IR collineari sono disposti a distanze differenti per ogni coppia di led (vedi fig. 3.4). La ridondanza del numero di led è stata scelta necessaria per garantire una maggiore precisione e permettere la stima della punta della penna anche nei casi di occultamento o disturbi video. Per calcolare la punta della penna bastano, infatti, una coppia di led che consentono di stimare l'orientamento della penna e il centroide.

Fires utilizza, per ora, solo i dati video e non quelli inerziali (nello specifico accelerometrici). Uno dei cambiamenti che verranno apportati sarà, appunto, il supporto di un dispositivo inerziale oltre a quello video (sensor fusion).

3.1.3 Triangolazione e calibrazione

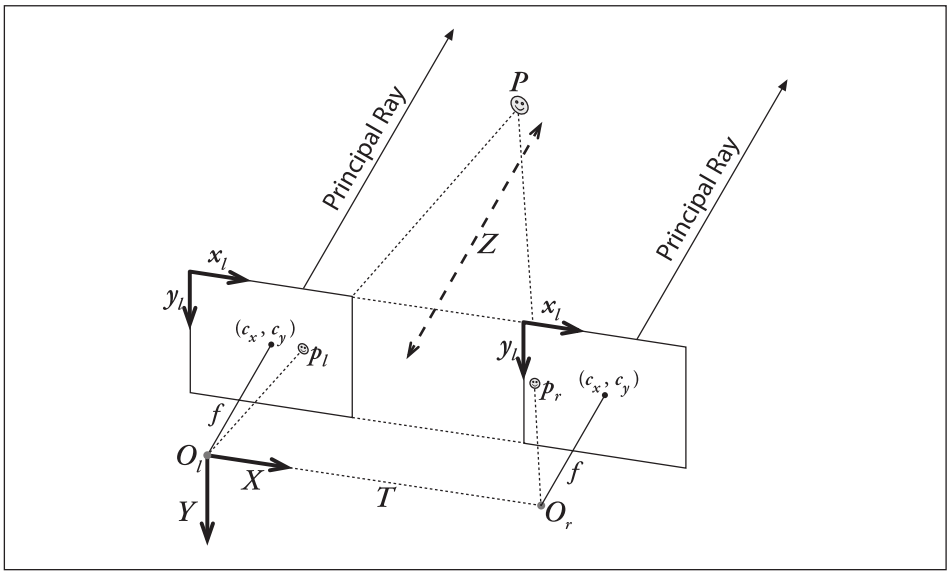


Figura 3.5: Triangolazione

La libreria Wiiuse mette a disposizione direttamente le coordinate dei punti 2d (u,v,1) visibili con ciascuna camera. Si può effettuare, dunque, una triangolazione stereoscopica che trasforma due punti 2d in un unico punto 3d [8]. Nel caso di fires è possibile scegliere tra triangolazione lineare o mid-point. La fig 3.5 mostra come viene calcolato il punto tridimensionale a partire dalle immagini 2d. Ol e Ov sono le origini delle due pin-hole camera (i wiimotes). Qui le due camere sono posizionate parallelamente sull'asse Z. pl e pr sono i punti 2d rilevati da ciascuna camera mentre P' è il punto nel 3d. Per effettuare la triangolazione , però, bisogna conoscere sia le proprietà

intrinseche delle due camere (fov, distanza dal piano immagine, etc.) che la loro disposizione (quanto sono distanti e l'angolo di rotazione tra di esse). Le proprietà intrinseche sono fisse (calcolate per approssimazione, come già mostrato), mentre per la disposizione delle camere va misurata, per ogni spostamennto, sia la traslazione che la rotazione; tale misurazione può essere fatta sia fisicamente che tramite calibrazione, un meccanismo che calcola con accuratezza la traslazione e la rotazione di due camere basandosi su un pattern visivo rimandabile ad una scacchiera.

Nel caso di Fires la calibrazione viene effettuata con un pattern di 4 punti (il massimo numero di punti che si può rilevare), disposti a rettangolo, mosso davanti alle camere da cui vengono acquisite diverse immagini; con queste viene, poi, calcolata la rotazione e la traslazione tra le due camere. Fires utilizza OpenCV [7] per ottenere la matrice di rotazione (3x3) e il vettore di traslazione (x,y,z) tra le due camere.

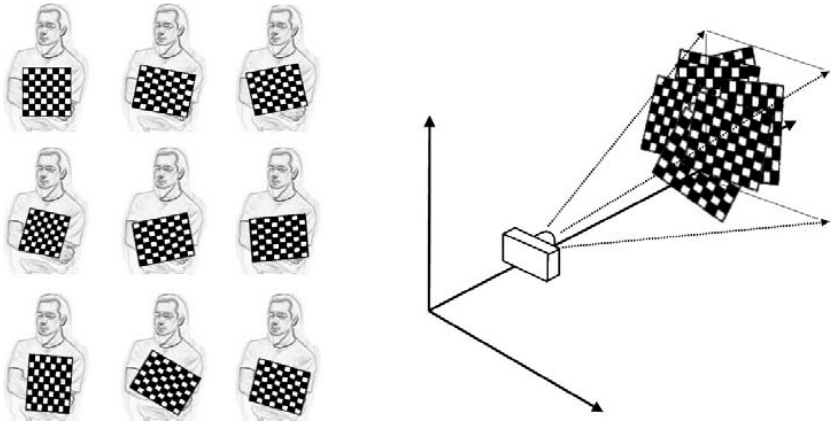


Figura 3.6: Immagini campione di una camera con le quali viene effettuata la calibrazione

3.1.4 Stima della punta

La triangolazione fornisce le coordinate dei punti 3D $\overline{p}_i, i \leq 4$ ed ad ogni istante di campionamento viene stimata, dunque, la posizione della punta

della penna tramite la formula:

$$p = C' + V_{pen}(C + T) \tag{3.1}$$

dove v_{pen} è l'inclinazione della penna, $C' \in \mathbb{R}^3$ è il centroide dei punti 3D $\overline{p}_i$, $C \in \mathbb{R}^2$ è il centroide reale tra i led calcolato per mezzo delle distanze fisiche e T è la distanza reale tra il centroide reale e la punta della penna.

L'inclinazione della penna può essere vista come una retta di regressione e viene calcolata tramite principal component analysis a partire dai 4 punti tridimensionali [5].

Una volta calcolata la posizione della punta della penna, viene applicato un filtro di smoothing gaussiano per ridurre gli errori, dovuti all'imprecisione delle camere, ottenendo dei buoni risultati nel bilanciamento tra reattività e filtraggio del rumore.

L'ultimo passo da considerare prima di parlare di ricostruzione è l'acquisizione di una linea di stile (chiamato Interactive Surface Sketching). Fires è un sistema interattivo ed effettua la ricostruzione della superficie in tempo reale basandosi sull'inserimento di linee di sketch tracciate per mezzo della penna sull'oggetto. La ricostruzione parte non appena l'utente ha terminato di acquisire una linea di stile.

3.2 Ricostruzione

Subito dopo viene aggiornata in tempo reale la struttura dati di base polyline network che consiste in una serie di linee guida più la mesh costruita a partire dalle linee guida. Il processo di ricostruzione è un processo iterativo e, quindi, maggiore è il numero di linee acquisite maggiore sarà la fedeltà all'oggetto che si sta acquisendo. La mesh all'interno della polyline network conterrà facce non planari non convesse e con N lati. Occorre dunque effettuare una suddivisione delle singole facce in modo da ottenere soltanto triangoli e quadrilateri. La mesh risultante viene chiamata Tri-Quad Mesh che resta comunque molto grezza, specialmente se si sono acquisite poche

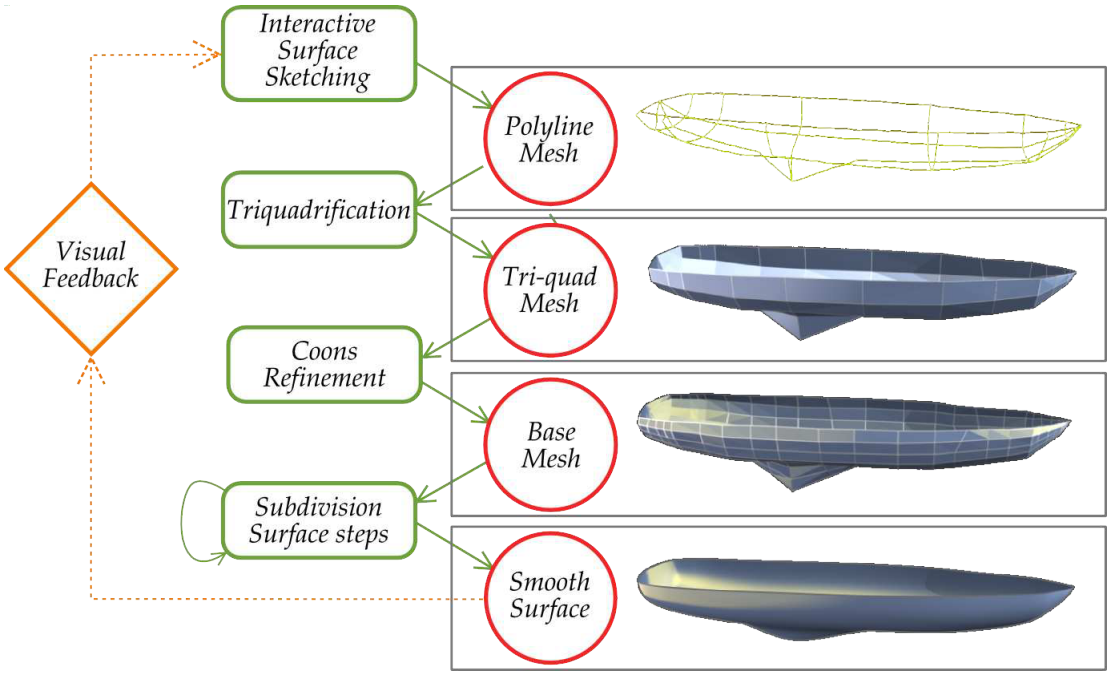


Figura 3.7: I vari passi di ricostruzione di una superficie

linee di stile e occorre effettuare, quindi, un passo di raffinamento chiamato Coons Refinement generando, quindi, una mesh di base per l'operazione finale di subdivision.

Capitolo 4

Obiettivi

L'idea generale è quella di migliorare il sistema Fires agendo sul sistema di acquisizione, cercando dove possibile di rimuovere restrizioni e vincoli e migliorare la precisione del sistema. Si può lavorare su molti aspetti, di seguito vengono mostrati quelli scelti come obiettivi per questa tesi:

1. Fires è un sistema real-time; si può, dunque, ottimizzare il codice e le strutture dati per ridurre il costo computazionale.

2. Il sistema, nonostante sia progettato per funzionare con un numero variabile di led, attualmente, funziona soltanto se sono visibili 4 led su ogni camera o nel caso vi ‘ un occlusione sul led di punta. Questa limitazione può essere superata tramite un opportuno algoritmo di riconoscimento.

3. Fires lavora su un'area piuttosto ristretta, si può estendere l'area di visiblità del sistema visivo migliorando l'hardware.

4. Migliorare il supporto da riga di comando e gestire un file per la configurazione del sistema.

5. Creare un'infrastruttura di debug per il sistema d'acquisizione basato sulla simulazione di una traccia.

6. Effettuare la fusione di sensori introducendo un dispositivo inerziale in aggiunta al sistema visivo atto al miglioramento della stabilità e precisione del sistema.

Per quanto riguarda l'ultimo punto, occorrerà fare dei test qualitativi per valutare se il supporto ad un dispositivo inerziale in aggiunta al sistema visivo (sensor fusion) porta ad un guadagno in termini di precisione.

Capitolo 5

F.I.R.E.S. Miglioramenti

5.1 Hardware

L'hardware di Fires, escludendo i wiimotes e il personal computer, è composto da due prototipi: una penna e un pattern di calibrazione; nonostante siano progettati molto bene, consentono di lavorare soltanto in un'area piuttosto ristretta (20x20 cm). Agli estremi dell'area di visibilità i punti 2d cominciano ad essere intermittenti fino a scomparire del tutto, cosa che accade anche inclinando la penna oltre i 30 gradi.

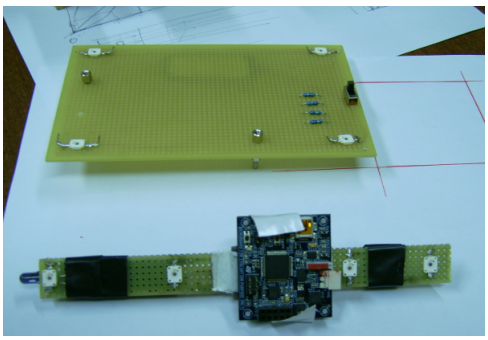


Figura 5.1: il nuovo prototipo di smart-pen e la basetta di calibrazione

Sono stati costruiti, quindi, due nuovi prototipi che consentono di lavorare in maniera agevole in un'area di due metri; sia la penna che il pattern di calibrazione sono composti da 4 led ad infrarossi disposti in parallelo. Succes-

sivamente è stato attaccato, manualmente, alla penna il dispositivo inerziale in modo da catturare contemporaneamente le informazioni video ed inerziali.

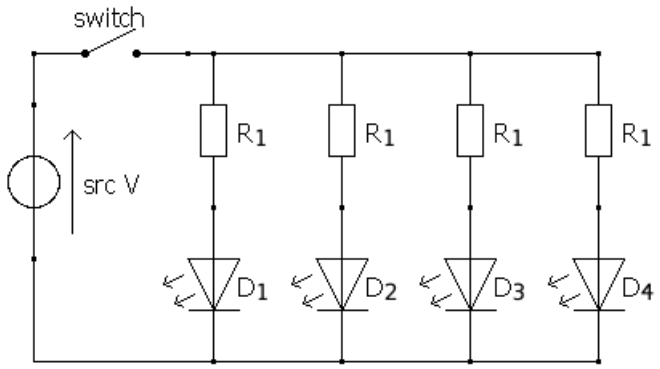


Figura 5.2: Circuito elettrico della penna e della basetta di calibrazione

Quando si vuole utilizzare un emettitore IR bisogna tenere conto dell'intensità di corrente in quanto bisogna evitare di bruciarlo oltre che di sprecare troppa batteria. L'intensità di corrente, inoltre, e' direttamente proporzionale alla luminosità del raggio infrarosso; e' meglio, quindi, tenerla abbastanza alta quando possibile. I prototipi della smart-pen e della basetta di calibrazione utilizzavano la stessa soluzione circuitale scelta per i nuovi prototipi, venivano utilizzate due batterie ricaricabili da 1.2 Volt in serie in modo da ottenere una differenza di potenziale di 2.4V; le resistenze garantivano, poi, il passaggio di 20 mA di corrente su ogni emettitore IR. Volendo migliorare l'area di visibilità e i problemi ad essa legati, dovuti all'inclinazione, sono stati scelti dei led ad alta intensità della Osram, con un angolo di visuale di 120Â° e si è considerata una corrente di 100mA su ogni singolo di essi. Sono state calcolate, quindi, opportunamente le resistenze tramite la legge di ohm ($V = RI$): 25Ω.

L'area di lavoro è migliorata notevolmente ma non supera tutt'ora i 2 metri quadrati. Per quanto riguarda l'inclinazione si è notato che il problema dato dai led è scomparso, anche se rimane, comunque, un problema di fondo costituito dall bassa risoluzione delle camere che non consente di riconoscere

due led troppo vicini. Il prototipo di penna non supera, dunque, i 60 gradi sull'asse verticale mentre su quello orizzontale, ovviamente, il problema non si verifica. Per consentire una maggiore inclinazione bisogna scegliere opportunamente le distanze tra i 4 led, cercando di tenerle più alte possibile.

Il dispositivo inerziale o IMU (Inertial Measurement Unit) [6] posizionato sulla penna è un STMicroelectronics iNEMO (iNErtial MOdule) che ha le seguenti caratteristiche:

- MCU 32-bit (ARM) con 256 kB Flash
- giroscopio 3-axis
- accelerometro 3-axis
- sensore di pressione
- sensore di temperatura
- connettore COM, USB

5.2 Riprogettazione del software

È stato subito problematico comprendere il codice sorgente che presentava, sin da subito, una cattiva strutturazione sia dal punto di vista dei dati che dell'organizzazione delle cartelle. È stato molto difficile, inoltre, comprendere l'ordine delle chiamate a funzione; questo è dovuto al fatto che il codice è stato gestito aggiungendo ogni volta funzionalità senza aver tempo per tenerlo ben strutturato e commentato. Il programma conteneva circa 45.000 linee di codice, molte delle quali matematiche o di trasformazione geometrica spesso senza commenti, rendendo il lavoro di comprensione e modifica piuttosto difficile. I cambiamenti apportati al sistema Fires riguardano soltanto il sistema di acquisizione composto da 5.000 righe di codice; i sorgenti, per evitare di perdere le modifiche apportate e per facilitare lo scambio di informazioni con gli altri sviluppatori del sistema, sono stati inseriti subito in un

nuovo repository git, mentre le modifiche sono state salvate in maniera incrementale cercando di rispettare il più possibile le buone norme di commit. L'idea di partenza è stata di studiare il codice e fare refactoring gestendo il tutto in piccole commit. Dopo non molto si è compreso che il problema era più grande di quanto stimato e, data la mancanza di una struttura coerente e la disorganizzazione dei sorgenti oltre che delle directory, non era possibile lavorare in maniera facile sul codice.

È stato necessario, dunque, reingegnerizzare il codice eliminando l'uso di variabili globali e contestualizzando le strutture dati e le funzioni in sorgenti differenti. Durante le operazioni di refactoring, in caso di mancanza, si è anche commentato il codice e si è lavorato a fondo sulla parte di questo dedicata all'acquisizione ottenendo un codice strutturato che supera di poco le 3000 righe. Per il resto del programma si è cercato di riordinare le funzioni e commentare il più possibile la dove era necessario. Occorrerebbe, in futuro, effettuare sul resto del codice un refactoring che riguardi sia la GUI che il processo di ricostruzione.

Come è possibile vedere in fig. 5.3 l'architettura di Fires non è molto cambiata; sono state aggiunte alcune librerie e il supporto alla seriale e al modem tramite i moduli kernel; questi ultimi sono necessari per poter utilizzare alcuni strumenti inerziali di test che si interfacciano al computer e a Fires tramite porta COM. Sono state apportate delle modifiche ai sorgenti di fires e di ssgl (evidenziate in rosso) che riguardano principalmente l'acquisizione, la GUI oltre che, solo parzialmente, i sorgenti di ssgl.

Occorre specificare, a questo punto, le varie librerie utilizzate da Fires e le motivazioni per cui sono state aggiunte altre due librerie. Le *GLUT* [link] permettono la creazione della finestra e del context OpenGL del programma oltre che la gestione dell'input da mouse e tastiera. Questa libreria è famosa per essere semplice ma rappresenta anche un problema nel caso il programma sia piuttosto complesso; con la sua struttura a callback senza meccanismi per lo scambio di dati definiti dall'utente rende necessario l'utilizzo di variabili globali. Viene incentivato l'utilizzo delle variabili globali e qualsiasi

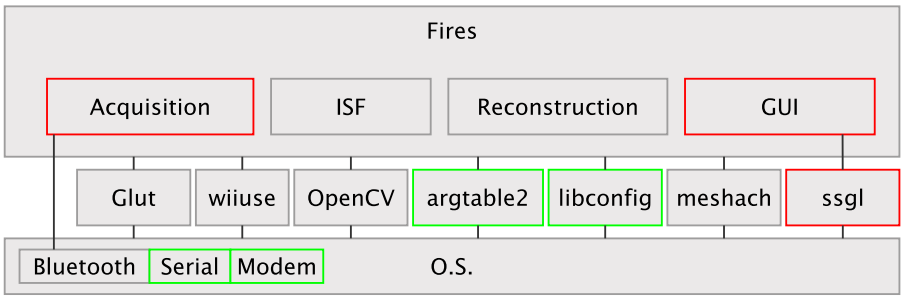


Figura 5.3: Aggiornamenti all'architettura di fires, in rosso le modifiche e in verde le aggiunte

funzione definita dall'utente non necessita più all'utilizzo di parametri locali. Esistono altre librerie strutturate a callback che permettono il passaggio dei dati per mezzo di un puntatore nullo $void * data$ oppure utilizzano funzioni del tipo $setPrivateData()$ e $getPrivateData()$. Per questa serie di motivi è consigliabile la migrazione ad un'altra libreria come ad esempio SDL.

La libreria *wiiuse* permette di connettersi a diversi Nintendo Wii remotes, supporta il rilevamento di movimenti e il tracking ad infrarossi, il collegamento di controller nunchuk, classic e Guitar Hero 3. Resta una buona libreria di base leggera e semplice.

OpenCV è una libreria molto ampia che copre molti degli aspetti legati alla computer vision; nel sistema Fires viene utilizzata solo per la calibrazione e, come vedremo successivamente, per effettuare la fusione tra i sensori tramite il filtro di Kalman. Se si vuole evitare la dipendenza da questa libreria bisognerebbe riscrivere queste due funzionalità.

La libreria *meshach* è usata di routine per il calcolo matriciale, supporta

dinamicamente strutture dati come permutazioni, vettori, matrici anche di numeri complessi. È molto utilizzata anche all'interno del sistema Fires e risulta essere piuttosto semplice . Anche OpenCV gestisce vettori e matrici in maniera molto intuitiva; un motivo in più per considerare necessario o meno l'uso di OpenCV e definire tutte le funzionalità legate all'algebra matriciale con una delle due librerie.

La libreria *ssgl* è stata sviluppata dagli autori di Fires ed è di base per la gestione della geometria 3D, buffer ed astrazioni di utility come i Timer, i Workspace, la Camera ed altro; permette, inoltre, di definire un'interfaccia grafica modale in cui l'utente può definire sia i comandi che gli stati di un menù. Questa funzionalità è necessaria in un sistema in cui le operazioni da svolgere sono molte e sono legate ad uno stato specifico del sistema e sono perfette per un sistema come Fires.

5.2.1 argtable2

È stata introdotta la libreria **argtable2** per il parsing command-line in quanto senza risultava abbastanza difficile comprendere e ricordare i parametri d'avvio. Questa libreria semplifica il lavoro permettendo al programmatore di definire le opzioni della linea di comando direttamente nel codice sorgente come un array statico di strutture che viene passato alle funzioni della libreria di argtable la quale provvederà al parsing. I valori estrapolati dalla linea di comando sono depositati in variabili definite dal programmatore (float, int o stringhe) e se ci sono errori di sintassi, o tipologia di dato, durante la fase di parsing viene visualizzato da argtable un messaggio d'errore. La libreria è stata scelta perchè open-source e platform-indipendent. Di seguito viene mostrato il nuovo help da riga di comando di Fires.

./fires −−help
Usage: fires [−hvn] [−−simulate=<trace>] [−f <file.obj>|<file.sss>] [−s <factor>]
−h, −−help Print this help and exit
−v, −−version Print version information and exit
−n, −−no−acquisition Not use acquisition system

−−simulate=<trace> Simulate smartPen movement from trace file
−f <file.obj>|<file.sss> Load an object file (.obj, .pln) or a session file (.sss)
−s <factor> Object scaling factor (default 1.0)

Report bugs to <myname@firessytem.com>.

5.2.2 libconfig

È stata introdotta, inoltre, un' altra libreria molto utile: **libconfig**. Libconfig è una libreria semplice per la processazione di file di configurazione strutturati. Questo formato file è più compatto e leggibile dell'XML; a differenza di quest'ultimo, infatti, è type-aware per cui l'applicazione non necessita di fare parsing. Si tratta di una libreria molto compatta (una frazione della grandezza della libreria di parsing XML expat); anche questa, come argtable2, è stata scelta perchè open-source e platform-indipendent. Viene mostrata, di seguito, una parte del file di configurazione di Fires che permette la sua configurazione senza bisogno di ricompilare i sorgenti o utilizzare parametri da riga di comando.

```
#−−−−−−−−−−−−−−−−−−−−−−−−−−−
# Fires configuration files
#−−−−−−−−−−−−−−−−−−−−−−−−−−−
#

fires:
{
    window:
    {
        fullscreen = false;
        width = 1440;
        height = 900;
        widthmm = 288.0;    // width in mm
        heightmm = 216.0;    // height in mm
    };
```

```
    acquisition:
    {
        pen:
        {
            tipdist = 0.0;    // distance from pen tip and led1 in mm
            dist1 = 52.0;     // distance from led1 and led2 in mm
            dist2 = 74.0;     // distance from led2 and led3 in mm
            dist3 = 35.0;     // distance from led3 and led4 in mm
        };

        pattern:
        {
            xd = 131.0;      // pattern x-axis offset between leds in mm
            yd = 81.0;       // pattern y-axis offset between leds in mm
        };

        ...
    };
};

### eof
```

5.2.3 Il componente Acquisizione

Questo componente di Fires è stato totalmente reingegnerizzato e migliorato. Le tre fasi necessarie per l'acquisizione di un punto 3D sono: il sampling dei dati, la triangolazione e la stima della punta della penna, vedi fig. 5.4. Queste unità (in figura sono rettangoli con gli angoli smussati) sono state separate logicamente facilitando la lettura e la comprensione del codice, il riuso e l'aggiunta e sostituzione di funzionaltà. Il sistema d'acquisizione effettua un ciclo (sampling, triangulation, tpEstimation) ad una frequenza stabilita dall'utente (per esempio 100 Hz) ottenendo come risultato la punta della

penna che viene utilizzata nel processo successivo di interactive sketching per aggiornare una polyline network.

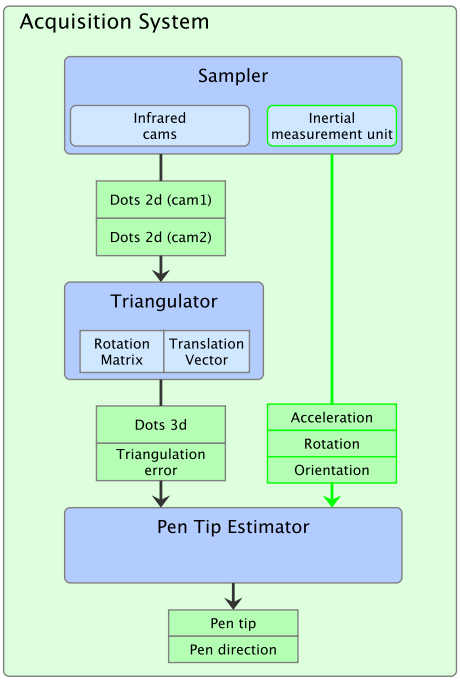


Figura 5.4: Struttura e flusso del sistema di acquisizione

Campionamento

Il componente di campionamento si basa su due unità di base: irCams, che è l'astrazione di una camera ad infrarossi, e imuDevice che astrae un dispositivo di misura inerziale. Attualmente irCams è poco più che un wrapper di wiiuse ma l'intenzione è permettere una facile sostituzione nel caso si cambi tipo di camera IR. I dati vengono campionati ad una frequenza stabilita su file di configurazione ma sotto viene gestita in maniera asincrona per entrambi i dispositivi. Questa scelta è derivata dal fatto che i wiimotes inviano pacchetti di aggiornamento in maniera continua e senza timestamp. Wiiuse ha un meccanismo di polling con frequenza molto maggiore di quella di campionamento e viene gestito tramite aggiornamento asincrono (si è scelto lo stesso anche per il dispositivo inerziale). Per permettere al sistema di

funzionare in real-time è necessario, quindi, tenere sotto controllo tale rapporto tra le frequenze (freq di polling e di campionamento). I dati in output sono le n coordinate (u,v) dei punti 2D ordinati lungo l'asse verticale con $2 \leq n \leq 4$ variabile per ciascuna camera.

Triangolazione

Nel Triangulator vengono mantenute e gestite tutte le informazioni necessarie per effettuare la triangolazione dei punti 2D in punti 3D. Sono incluse, quindi, le routine per la calibrazione e il loro output viene mantenuto internamente. L'interazione dell'utente durante la calibrazione è stata modificata. Prima veniva premuto un tasto per avviare la calibrazione e mosso il pattern in un breve tempo, qualche secondo, nel quale venivano acquisite le immagini necessarie per la calibrazione. Ora, invece, l'utente deve premere un tasto per avviare la calibrazione e premerne un altro ogni volta che intende aggiungere un immagine fino al raggiungimento di 50 campioni immagine. Questo consente all'utente di scegliere esattamente quali immagini inserire e, quindi, ottenere un campione di immagini disomogeneo migliorando notevolmente i risultati della calibrazione.

Stima della punta della penna

Il Pen Tip Estimator serve per ottenere la punta della penna dai 4 punti tridimensionali. Il filtro di smoothing che veniva applicato viene fatto internamente al Pen Tip Estimator, sempre attivabile. Questa astrazione è stata in modo da conservare lo schema lineare e chiaro.

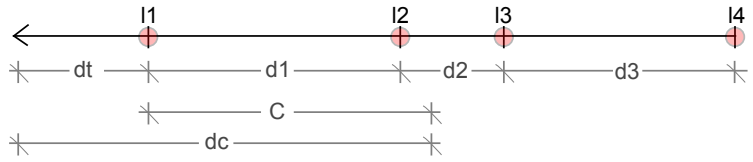
5.3 Simulazione

La prima funzionalità aggiunta al sistema di acquisizione è stata la simulazione della penna. In pratica si è trattato di permettere il salvataggio di una traccia dati grezza con la quale è poi possibile rilanciare il programma senza bisogno di usare la penna. È possibile simulare, quindi, più volte la

stessa acquisizione facilitando il debug del sistema d'acquisizione. La traccia tiene conto di diverse cose:

- i parametri fisici del sistema visivo (rotazione e traslazione tra le camere, distanze fisiche tra i led).
- per ogni led di ciascuna camera: coordinate 2d e visibilità.
- i dati inerziali.

5.4 Supporto led dinamico



pen scheme		pen scheme	
1 2 3 4	4	X X 3 4	2
X 2 3 4	3	X 2 X 4	2
1 X 3 4	3	X 2 3 X	2
1 2 X 4	3	1 X X 4	2
1 2 3 X	3	1 X 3 X	2
		1 2 X X	2

Figura 5.5: Circuito elettrico della penna e della basetta di calibrazione

Fires è stato progettato per gestire un numero variabile da 2 a 4 led secondo lo schema (x.y). In realtà non è stato implementato un algorimo generico che riconoscesse un numero variabile di led. Durante lo sviluppo di questa tesi sono state usate per molto tempo diverse penne con diverse configurazioni di led non riuscendo mai a visualizzare 4 led. Si è deciso,

dunque, di supportare un numero dinamico di led per entrambe le camere. Per poter stabilire se due di loro sono visibili e quale sia la loro posizione 3D occorre effettuare la triangolazione di due coppie di punti 2d ed effettuare un test sulla distanza dei due punti 3D ottenuti con la distanza reale tra i led. Come possibile vedere dalla fig. 5.6, i led 2d sono stati ordinati dal basso verso l'alto ma senza sapere in quale configurazione di led tra le due camere ci si trovi. Si possono raggruppare le configurazioni possibili in base al numero di led 2D visibili da ciascuna camera (con $2 \leq ledcam_1, ledcam_2 \leq 4$) ed effettuare tutti i test necessari ogni volta che si ha un nuovo campione. Per ciascuna categoria bisogna effettuare un numero variabile di triangolazioni e di confronti e stabilire dunque quali sono i led 3D visibili. La triangolazione è un'operazione che richiede un discreto tempo di calcolo e la sequenza di test può, quindi, essere ottimizzata scegliendo per primi quelli più probabili. Nell'immagine viene omesso il caso in cui in entrambe le camere siano visibili 2 led perchè in quel caso è sufficiente fare la triangolazione del led0 e led1 di entrambe le camere ed effettuare 3 test (quelli del caso 4 led visibili in entrambe le camere).

5.5 Supporto dei dispositivi inerziali

Abbiamo già introdotto i dispositivi inerziali utilizzati. Da lato software vengono gestiti in una struttura imuDevice in base al tipo di connessione e al tipo di protocollo dati scambiati. Bisogna tener conto che non è facile separare i dati dei diversi dispositivi inerziali in strutture separate dato che il loro utilizzo è verticale, ovvero legato sia alla componente di Acquisition Sytem che all'unità di Pen Tip Estimation. Di seguito vengono mostrati i diversi tipi di connessione ed i protocolli che è possibile utilizzare.

```
enum imuConnection
{
    BLUETOOTH,
    SERIAL,
```

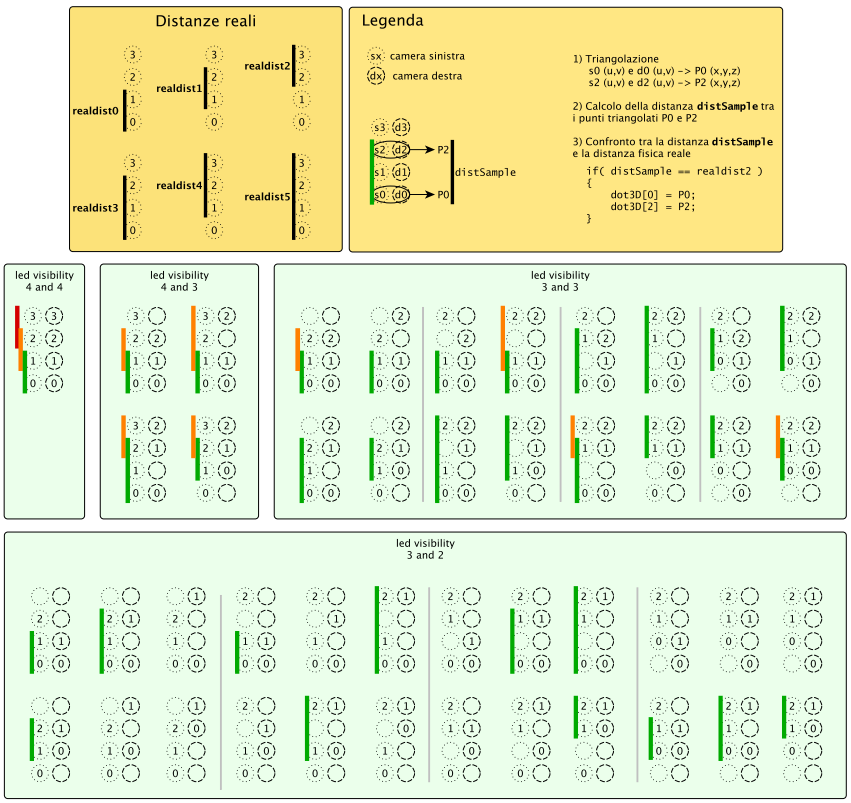


Figura 5.6: Circuito elettrico della penna e della basetta di calibrazione

```
    MODEM,
};

enum imuProtocol
{
    TWO_ACCS,   // using 2 accelerometers, 4 buttons, ledOn and counter
    IMU,        // using accelerometer, magnetometer and gyroscope
    INEMO,      // using accelerometer, magnetometer, gyroscope pressure and temperature..
};
```

Capitolo 6

F.I.R.E.S. Fusione di sensori

La fusione di sensori è la combinazione di dati sensoriali provenienti da sorgenti differenti e serve ad ottenere informazioni migliori rispetto a quelle che si otterrebbero utilizzando i singoli sensori. Esistono differenti algoritmi di sensor fusion che ci permettono di migliorare la stima della posizione e dell'inclinazione della penna. Il filtro di Kalman è, tuttavia, lo stimatore ottimo per il nostro caso.

In questo capitolo introdurremo, dunque, il filtro di Kalman [9] che ci consente di stimare lo stato di un modello a spazio-stato in maniera ottimale. Definiremo due possibili modelli della penna a stato-spazio che sfruttano le informazioni accelerometriche. Verrà mostrato, poi, come sia relativamente semplice definire i filtri di Kalman grazie alla libreria OpenCV ed infine verrà fatta una piccola parentesi sui sistemi di riferimento utilizzati per la fusione dei sensori.

6.1 Filtro di Kalman

Nonostante il filtro di Kalman abbia più di 40 anni soltanto nel 20-esimo secolo ha iniziato ad essere ampiamente utilizzato nelle applicazioni di computer graphics. Queste applicazioni spaziano dalla simulazione di strumenti musicali nella realtà virtuale, fino all'head tracking, dall'estrazione del mo-

vimento labbiale da sequenze video all'adattamento di superfici spline su collezioni di punti. Il filtro di Kalman è lo stimatore ottimo per un'ampia classe di problemi e risulta molto efficace ed utile per una classe di problemi ancora più grande. Esistono molti altri metodi di stima specifici per calcolare (stimare) uno stato sconosciuto a partire da un insieme di misurazioni di processo, molti dei quali non tengono in considerazione il tipico rumore naturale delle misurazioni. Consideriamo i sistemi di scansione o acquisizione. Mentre i requisiti e le informazioni di base possono variare, la sorgente delle informazioni rimane la stessa: gli stimatori vengono derivati da misurazioni elettriche rumorose di sensori meccanici, inerziali, ottici, acustici o magnetici. Il rumore è tipicamente statistico per natura (o può essere modellato come tale) e ci consente di stabilire dei metodi stocastici per risolvere certi problemi.

6.1.1 Modelli a spazio-stato

I modelli di spazio-stato sono essenzialmente una notazione di convenienza per stimatori e problemi di controllo. Sviluppati per rendere facilmente analizzabili problemi che avrebbero altrimenti una notazione illeggibile. Consideriamo un processo dimanico descritto da un n^{th} ordine di equazioni alle differenze (sono analoghe alle equazioni differenziali ma nel caso discreto) della forma

$$y_{i+1} = a_{0,i}y_i + ... + a_{n-1,i}y_{i-n+1} + u_i, i \geq 0, \tag{6.1}$$

dove u_i è il random noise di processo bianco (nello spetto) con media nulla (statisticamente) con autocorrelazione

$$E(u_i, u_j) = R_u = Q_i\delta_{ij}, \tag{6.2}$$

e valori inziali $y_0, y_{-1}, ..., y_{-n+1}$ sono variabili casuali con media nulla e matrice di covarianza

$$P_0 = E(y_{-j}, y_{-k}), j, k \in 0, n-1. \tag{6.3}$$

Si assume inoltre la media nulla, ovvero

$$E(u_i, y_i) = 0 \text{ per } -n+1 \leq j \leq 0 \text{ e } i \geq 0, \tag{6.4}$$

che equivale a dire:

$$E(u_i, y_i) = 0, i \geq j \geq 0, \tag{6.5}$$

In altre parole il rumore è statisticamente indipendente dal processo da stimare.

Questa equzione alle differenze può essere riscritta come

$$\hat{x}_{i+1} \equiv \begin{bmatrix} y_{i+1} \\ y_i \\ y_{i-1} \\ \vdots \\ y_{i-n+2} \end{bmatrix} = \underbrace{\begin{bmatrix} a_0 & a_1 & \cdots & a_{n-2} & a_{n-1} \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} y_i \\ y_{i-1} \\ y_{i-2} \\ \vdots \\ y_{i-n+1} \end{bmatrix}}_{\hat{x}_i} + \underbrace{\begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{G} u_i \tag{6.6}$$

che ci porta al modello spazio-stato

$$\begin{aligned} \hat{x}_{i+1} &= A\hat{x}_i + Gu_i \\ \hat{y}_i &= \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} \hat{x}_i \end{aligned} \tag{6.7}$$

o in forma generale

$$\hat{x}_{i+1} = A\hat{x}_i + Gu_i \tag{6.8}$$

$$\hat{y}_i = H_i\hat{x}_i \tag{6.9}$$

L'equazione 6.7 rappresenta il modo in cui il nuovo stato $\hat{x}_{i+1}$ viene modellato come combinazione lineare dello stato precedente $\hat{x}_i$ unito al *rumore*

di processo u_i. L'equazione 6.9 descrive la maniera in cui le misure di processo o *osservazioni* $\hat{y}_i$ sono derivate dallo stato interno $\hat{x}_i$. Ci si riferisce spesso a queste due equazioni parlando rispettivamente di *modello di processo* e *modello di misurazione* e sono la base di tutti i metodi di stima lineari, come il filtro di Kalman.

6.1.2 Il problema dell'osservatore

C'è un problema generale che riguarda la teoria dei sistemi lineari, chiamato *observer design problem*. Il problema principale è determinare (stimare) lo stato interno del sistema lineare avendo accesso soltanto agli output del sistema. È molto simile al problema black box dove si hanno accesso a qualche segnale del box (gli output) ma non è possibile guardare cosa c'è al suo interno.

La maggior parte degli approcci a questo problema base sono basati tipicamente sul modello spazio-stato presentato nella sezione precedente. Esiste un modello di processo tipico per la trasformazione dello stato di processo. Viene spesso rappresentato come equazione alle differenze lineare stocastica simile all'equazione 6.7

$$x_k = Ax_{k-1} + Bu_k + w_{k-1} \tag{6.10}$$

In aggiunta, c'è una forma di *modello di misurazione* che descrive la relazione tra lo stato di processo e le misurazioni. Viene rappresentato in maniera simile all'equazione 6.9

$$z_k = Hx_k + v_k \tag{6.11}$$

I termini w_k e v_k sono variabili casuali che rappresentano rispettivamente il rumore di processo e di misurazione.

6.1.3 Il filtro discreto di Kalman

Il filtro di Kalman è essenzialmente un insieme di equazioni matematiche che implementano uno stimatore del tipo predittore-correttore che è *ottimale*

nel senso che minimizza gli *errori*di covarianza stimati, quando si verificano cerce condizioni.

Il processo da stimare

Il filtro di Kalman si occupa del problema generale di ricercare lo stato $x \in \mathbb{R}^n$ di un processo controllato a tempo discreto che è gestito tramite l'equazione alle differenze lineare stocastica già introdotte

$$x_k = Ax_{k-1} + Bu_k + w_{k-1} \tag{6.12}$$

con una misura $z \in \mathbb{R}^m$ che è

$$z_k = Hx_k + v_k \tag{6.13}$$

Le variabili w_k e v_k sono il rumore di processo e di misurazione. Si assume che siano indipendenti, a media nulla e con distribuzione normale

$$p(w)\ N(0, Q) \tag{6.14}$$

$$p(v)\ N(0, R) \tag{6.15}$$

In pratica, la *covarianza del rumore di processo* Q e la *covarianza del rumore di misurazione* possono cambiare ad ogni step o misurazione, tuttavia vengono assunte come costanti.

L'origine del filtro

Definiamo $\hat{x}_k^- \in \mathbb{R}^n$ (notare il meno come apice) come la stima dello stato *a priori* al tempo k data la conoscenza del processo precedente a k, e $\hat{x}_k \in \mathbb{R}^n$ come la stima dello stato *a posteriori* al tempo k data la misurazione z_k. Possiamo definire gli errori di stima come

$$\begin{aligned} e_k^- &\equiv x_k - \hat{x}_k^- \\ e_k &\equiv x_k - \hat{x}_k \end{aligned} \tag{6.16}$$

La covarianza dell'errore di stima *a priori* è dunque

$$P_k^- = E[e_k^- e_k^{-T}] \tag{6.17}$$

e la covarianza dell'errore di stima *a posteriori* è

$$P_k = E[e_k e_k^T] \tag{6.18}$$

Nel derivare le equazioni per il filtro di Kalman, si compincia con l'obiettivo di trovare l'equazione che calcola la stima *a posteriori* $\hat{x}_k$ come combinazione lineare di una stima *a priori* $\hat{x}_k^-$ e una differenza pesata tra la misura attuale z_k e la misura stimata $H\hat{x}_k^-$ come mostrato nell'equazione seguente

$$\hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k^-) \tag{6.19}$$

la differenza $(z_k - H\hat{x}_k^-)$ viene chiamata *innovazione*della misurazione o *residuo*. Il residuo rispecchia la discrepanza tra misura stimata $H\hat{x}_k^-$ e l'attuale misurazione z_k. Un residuo nullo significa che le due misure sono uguali. La matrice K viene scelte per essere il *guadagno* o *blending factor* che minimizza la covarianza dell'errore *a posteriori*. Una forma risultante di K è data da

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \tag{6.20}$$

Osservando l'equazione **??** si può vedere che come la covarianza dell'errore di misurazione R si avvicina allo zero, il guadagno K pesa maggiormente l'innovatore. Nel dettaglio,

$$\lim_{R_k \to 0} K_k = H^{-1} \tag{6.21}$$

D'altro canto, come la covarianza dell'errore di stima *a priori* P_k^- si avvicina a zero, il guadagno K pesa meno l'innovatore. Nel dettaglio,

$$\lim_{P_k^- \to 0} K_k = 0 \tag{6.22}$$

Un'altro modo di pensare al peso K è: come la covarianza dell'errore di misura R si avvicina a zero, la misurazione attuale z_k è affidabile sempre più, mentre la misurazione in previsione Hx_k^- e' meno affidabile. In maniera analoga, se la matrice di covarianza dell'errore della stima *a priori* P_k^- si avvicina a zero, la misurazione z_k è poco affidabile mentre la misurazione in previsione Hx_k^- viene considerata più affidabile.

L'algoritmo del filtro di Kalman

Il filtro di Kalman stima un processo usando una specie di controllo di feedback: il filtro stima lo stato del processo in un certo periodo e quindi ottiene un feedback nella forma di misurazione (con rumore). Le equazioni del filtro di Kalman finiscono in uno dei seguenti due gruppi: equazioni di aggiornamento del tempo ed equazioni di aggiornamento della misurazione. Le equazioni di aggiornamento del tempo si occuppano di proiettare lo stato corrente avanti nel tempo e le equazioni di covarianza dell'errore di ottenere le stime *a priori* per il prossimo time step. Le equazioni di aggiornamento della misurazione si occupano di fornire feedback, ad esempio, inserendo una misurazione nella stima *a priori* per ottenere una migliore stima *a posteriori*.

Le equazioni di aggiornamento del tempo possono essere considerate equazioni di predizione, mentre le equazioni di aggiornamento di misura di correzione.

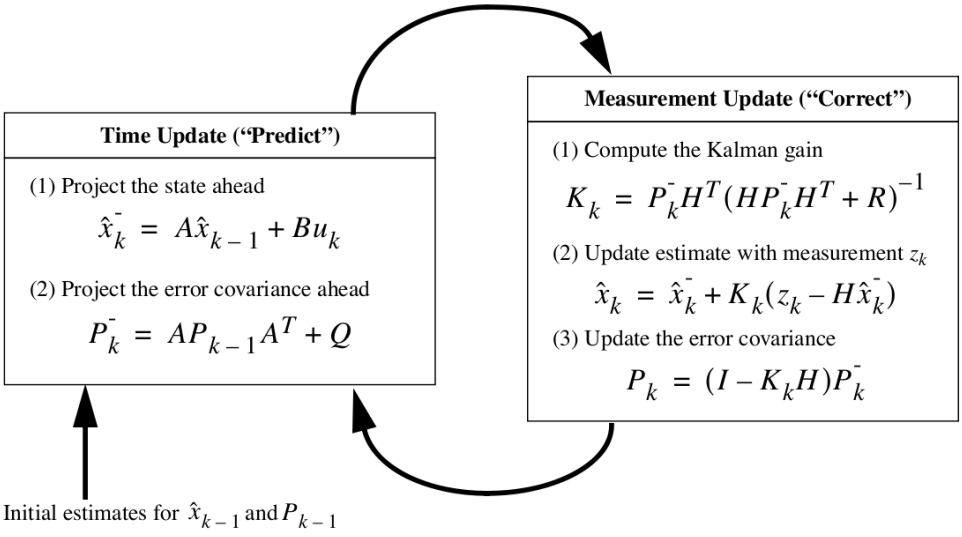


Figura 6.1: Le due fasi dell'algoritmo di Kalman

6.2 Modello n.1

Il filtro di Kalman ci permette, dunque, di effettuare la fusione dei sensori. Abbiamo bisogno, quindi, di definire un modello spazio-tempo che rappresenti la penna e che tenga conto delle informazioni date dal sistema visivo e da quello inerziale. Il sistema visivo ci fornisce la posizione della punta della penna e la sua inclinazione, mentre per il sistema inerziale consideriamo solo l'inclinazione della retta ricavata dai dati accelerometrici.

Possiamo definire un modello di sistema ad evoluzione libera che tenga conto della posizione, velocità ed accelerazione della penna come

$$\begin{cases} p(t) = p(t-1) + v(t-1)dt + Q_p \\ v(t) = v(t-1) + a(t-1)dt + Q_v \\ a(t) = a(t-1) + Q_a \end{cases} \tag{6.23}$$

$$m(t) = p(t) + R_p \tag{6.24}$$

Tutte le variabili del sistema sono vettori $[x, y, z] \in \mathbb{R}^3$. La posizione $p(t)$ si basa sulla posizione precedente più una certa velocità ed un rumore Q_p. La velocità è definita in maniera analoga mentre l'accelerazione è semplicemente quella dello stato precedente più un certo rumore. L'equazione 6.24 ci mostra la corrispondenza tra la misurazione $m(t)$ e la posizione $p(t)$.

Il modello dell'inclinazione della penna tiene conto dell'inclinazione e della velocità angolare della penna e può essere definito come

$$\begin{cases} \alpha(t) = \alpha(t-1) + \omega(t-1)dt + Q_\alpha \\ \omega(t) = \omega(t-1) + Q_\omega \end{cases} \tag{6.25}$$

$$\begin{cases} \alpha_v(t) = \alpha(t) + R_v \\ \alpha_a(t) = \alpha(t) + R_a \end{cases} \tag{6.26}$$

$\alpha \in \mathbb{R}^2$ è l'orientazione della retta definita come angoli di eulero $[\theta, \gamma]$ (pitch e roll) mentre ω rappresenta la velocità angolare. L'equazione di misurazione 6.26 mostra come la misurazione del sistema visivo α_v e del sistema inerziale α_a vengono legate con con lo stato di α.

6.3 Modello n.2

Possiamo costruire un modello molto simile al precedente che utilizzi l'accelerazione data dall'accelerometro, oltre che per calcolare l'inclinazione della penna, anche per aggiornare la posizione della penna. Possiamo ridefinire dunque le equazioni 6.23 e 6.24 come

$$\begin{cases} p(t) = p(t-1) + v(t-1)dt + Q_p \\ v(t) = v(t-1) + a(t-1)dt + Q_v \\ a(t) = a(t-1) + Q_a \end{cases} \tag{6.27}$$

$$\begin{cases} m_p(t) = p(t) + R_p \\ m_a(t) = a(t) + R_a \end{cases} \tag{6.28}$$

6.4 Implementazione

Come abbiamo visto, la punta della penna e l'inclinazione hanno equazioni di spazio-stato differenti ed occorreranno, dunque, due stime differenti e di conseguenza due filtri di Kalman. Il primo ci permetterà di stimare in maniera ottima la punta della penna e il secondo l'orientamento.

OpenCV implementa il filtro di Kalman in maniera piuttosto semplice [10] ma è necessario cambiare notazione e considerare le equazioni in forma matriciale. Possiamo rappresentare il nostro modello n.1, ad esempio, come moltiplicazioni di matrici. La rappresentazione della punta della penna è

data da

$$\begin{bmatrix} p_x(t) \\ p_y(t) \\ p_z(t) \\ v_x(t) \\ v_y(t) \\ v_z(t) \\ a_x(t) \\ a_y(t) \\ a_z(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & dt & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & dt & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & dt & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & dt & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & dt & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & dt \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{A} \begin{bmatrix} p_x(t-1) \\ p_y(t-1) \\ p_z(t-1) \\ v_x(t-1) \\ v_y(t-1) \\ v_z(t-1) \\ a_x(t-1) \\ a_y(t-1) \\ a_z(t-1) \end{bmatrix} + Q \tag{6.29}$$

$$\underbrace{\begin{bmatrix} m_x(t) \\ m_y(t) \\ m_z(t) \end{bmatrix}}_{m} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{H} \begin{bmatrix} p_x(t) \\ p_y(t) \\ p_z(t) \end{bmatrix} + R \tag{6.30}$$

Mentre per rappresentare l'orientamento della retta è necessario calcolare

$$\begin{bmatrix} \alpha_\theta(t) \\ \alpha_\gamma(t) \\ \omega_\theta(t) \\ \omega_\gamma(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & dt & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{A} \begin{bmatrix} \alpha_\theta(t-1) \\ \alpha_\gamma(t-1) \\ \omega_\theta(t-1) \\ \omega_\gamma(t-1) \end{bmatrix} + Q \tag{6.31}$$

$$\underbrace{\begin{bmatrix} \alpha^v_\theta(t) \\ \alpha^v_\gamma(t) \\ \alpha^a_\theta(t) \\ \alpha^a_\gamma(t) \end{bmatrix}}_{m} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}}_{H} \begin{bmatrix} \alpha^v_\theta(t-1) \\ \alpha^v_\gamma(t-1) \\ \alpha^a_\theta(t-1) \\ \alpha^a_\gamma(t-1) \end{bmatrix} + R \tag{6.32}$$

Abbiamo definito, dunque, la matrice di transazione di stato A, la matrice di misurazione H e le due matrici Q e R che sono, nel nostro caso, le uni-

che variabili da impostare per utilizzare il filtro di Kalman di OpenCV. La struttura dati $CVKalman$ risulta, a questo punto, piuttosto chiara

```
typedef struct CvKalman
{
    ...

    CvMat* state_pre;              /* predicted state (x'(k)):
                                      x(k)=A*x(k-1)+B*u(k) */
    CvMat* state_post;             /* corrected state (x(k)):
                                      x(k)=x'(k)+K(k)*(z(k)-H*x'(k)) */
     CvMat* transition_matrix;     /* state transition matrix (A) */
     CvMat* control_matrix;        /* control matrix (B)
                                      (it is not used if there is no control)*/
    CvMat* measurement_matrix;     /* measurement matrix (H) */
    CvMat* process_noise_cov;      /* process noise covariance matrix (Q) */
    CvMat* measurement_noise_cov;  /* measurement noise covariance matrix (R) */
    CvMat* error_cov_pre;          /* priori error estimate covariance matrix (P'(k)):
                                      P'(k)=A*P(k-1)*At + Q*/
    CvMat* gain;                   /* Kalman gain matrix (K(k)):
                                      K(k)=P'(k)*Ht*inv(H*P'(k)*Ht+R)*/
    CvMat* error_cov_post;         /* posteriori error estimate covariance matrix (P(k)):
                                      P(k)=(I-K(k)*H)*P'(k) */
     ...
}
CvKalman;
```

Le funzioni di aggiornamento del tempo (predict) e della misurazione (correct) che vengono fornite da OpenCV sono: $cvKalmanPredict()$ e $cvKalmanCorrect()$.

6.5 Sistemi di riferimento

Per effettuare la fusione dei sensori occorre che le informazioni grezze date dai dispositivi siano rappresentate sullo stesso sistema di riferimento

(vedi fig.6.2). Il sistema di riferimento globale è posizionato in uno dei due Wiimotes, per la penna invece è il centroide che, per essere visibile, deve stare nel semipiano $-Z$ ad una certa distanza dai Wiimotes. Il sistema di riferimento del dispositivo inerziale è solidale con quello della penna.

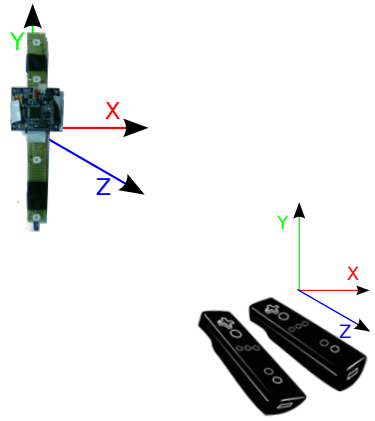


Figura 6.2: Sistemi di riferimento

Capitolo 7

Test

Sono stati svolti tre test per valutare le differenze introdotte dalla sensor fusion. I primi due servono per valutare l'accuratezza della punta della penna a differenti velocità di acquisizione, il terzo per testare la precisione dell'inclinazione della penna.

7.1 Configurazione

L'hardware utilizzato per i test è composto da:

- due Nintendo Wiimotes.
- Il nuovo prototipo smart-pen con led Osram SFH 4233 e IMU iNemo.
- Basetta di calibrazione con led Osram SFH 4233.
- Un pc con processore Intel(R) Core(TM)2 Duo CPU T9400 @ 2.53GHz e scheda video Nvidia.

L'area di acquisizione scelta per il testing è di $0.5m^2$ incentrata a circa $0.5m$ dai wiimotes. La scelta dell'area è stata fatta in base alle limitazioni attuali del prototipo della penna il cui dispositivo inerziale è stato collegato al computer via USB. Le camere sono state disposte parallelamente a 25 centimetri di distanza tra di loro ed è stata fatta la calibrazione del sistema

visivo e di quello inerziale. Ci sono vari problemi che non consentono di ottenere una traccia fedele e precisa: per primo bisogna contare il cavo USB che rende più rigido il movimento, secondo la punta della penna, nel prototipo costruito, è composta da un led che non è del tutto fisso.

7.2 Test 1: Traccia veloce

Lo scopo del primo test è quello di stimare in maniera più accurata possibile la posizione della punta della penna in uno scenario d'acquisizione limitato dalla penna in posizione sempre verticale. È stato preso un lettore mp3 e si è cercato di tracciare i bordi e le linee di stile sempre considerando la penna verticale ma muovendosi in maniera piuttosto rapida. In questo test è stata acquisita una traccia simulativa cercando di stare il piu' possibile a contatto con l'oggetto con errore massimo di 3 millimetri. Sono stati fatti varie prove confrontando il risultato ottenuto variando i parametri del filtro di Kalman. Come già visto nel capitolo precedente la matrice di covarianza dell'errore di processo Q con valori bassi porta ad una traccia più smooth. Lo stesso vale per valori alti nella matrice di covarianza dell'errore di misurazione R. Sono state dunque ricavati i valori di variazione delle due matrici del filtro di Kalman applicato alla posizione che sono:

- per $Q : 7e^{-3}$
- per $R : 1e^{0}$

La fig. 7.1 mostra la posizione della punta della penna secondo i tre assi assi [x,y,z].

7.3 Test 2: Inclinazione

Il secondo test è un test generale su entrambi i filtri di Kalman. Lo scopo principale è controllare che l'inclinazione sia migliorata grazie alla fusione dei sensori. Abbiamo calcolato, dunque, i valori di Q ed R per il filtro di

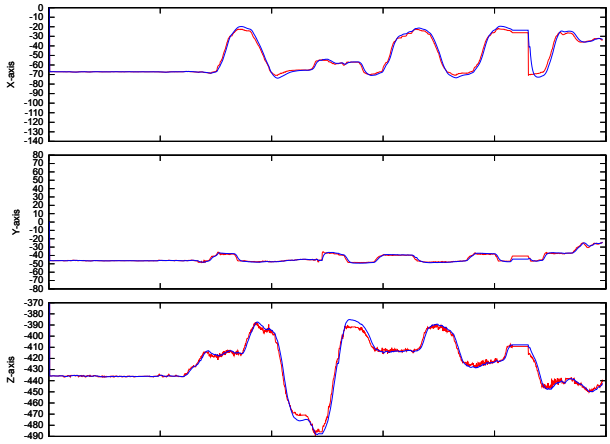


Figura 7.1: L'acquisizione veloce della traccia grezza (rosso) è abbastanza precisa, migliorata con il filtro di Kalman (blu)

Kalman applicato all'inclinazione della penna, mentre abbiamo mantenuto i valori precedenti per quanto riguarda la stima ottimale della posizione. La traccia simulativa è stata acquisita muovendo volontariamente la penna lungo gli assi di rotazione ad un inclinazione massima di 45 gradi. I valori ottimali per le matrici di correlazione del filtro di Kalman applicato sull'inclinazione della penna sono:

- per $Q : 5e^{-2}$
- per $R : 5e^{+1}$

La fig 7.2 mostra la posizione della punta della penna ottimale stimata tenendo conto della fusione di sensori.

7.4 Test 3: Quadrato

Lo scopo di questo ultimo test è quello di stimare l'accuratezza in base ad un pattern di riferimento. È stata acquisita una traccia simulativa seguendo il bordo di un quadrato di $7x7cm$ disegnato su un foglio posizionato, quindi,

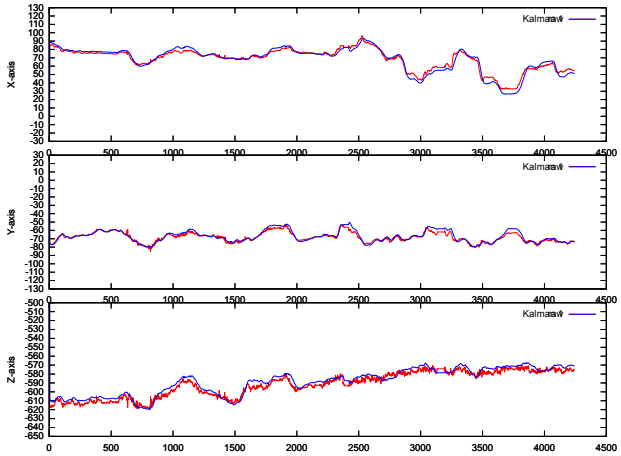


Figura 7.2: Fusione di sensori. dalla traccia grezza (rosso) viene applicato il filtro di Kalman (blu)

sul piano di lavoro. Sono stati mantenuti i valori ricavati dal Test1 e dal Test2 e confrontati con la traccia grezza ed il filtro gaussiano con una finestra di 150 valori già presente. Come è possibile vedere dalla figura **??** e dalla figura 7.4 i due filtri applicati approssimano molto bene il quadrato. L'errore di base è dato da una cattiva acquisizione sulla traccia grezza. Inoltre c'è un paio di millimetri di traslazione per colpa del dispositivo inerziale che non è posizionato in maniera del tutto precisa e quindi c'e' una traslazione rispetto al sistema di riferimento della penna.

7.5 Valutazioni sulla fusione di sensori

Abbiamo definito un modello spazio-stato semplice in cui la fusione dei sensori viene gestita solo per stimare l'inclinazione della penna. Stimando l'inclinazione dal sistema visivo e dai valori accelerometrici si ottengono, secondo quanto visto dai test, dei buoni risultati di poco migliori di un filtro gaussiano. L'utilizzo dei soli dati inerziali accelerometrici, però, peggiorano la stima della punta e l'acquisizione. Questo perchè i valori grezzi accelerometrici sono imprecisi a causa di errori numerici e senza considerare le

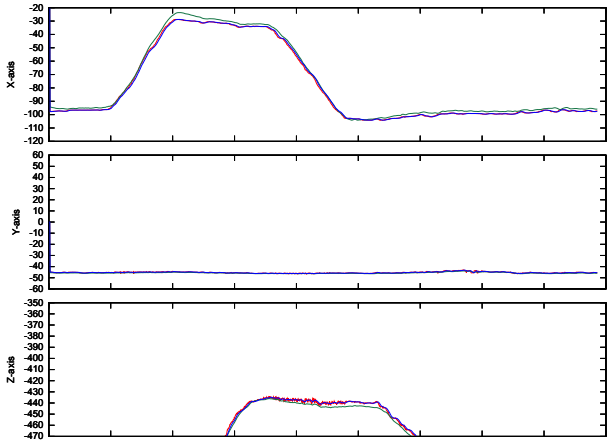


Figura 7.3: Confronto con il filtro gaussiano (in blu) e il filtro di Kalman (verde) di un quadrato 7x7 (traccia rossa).

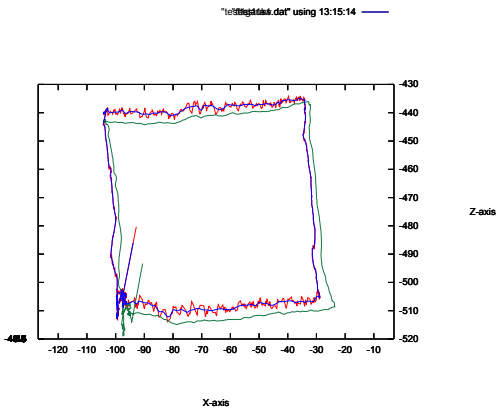


Figura 7.4: Proiezione della posizione stimata sul piano XZ. Il rosso è la traccia grezza, blu con il filtro gaussiano e verde con il filtro di Kalman

informazioni fornite dal giroscopio e magnetometro non riescono a dare un buon contributo all'accuratezza del sistema.

Capitolo 8

Conclusioni

Si voleva migliorare F.I.R.E.S. agendo su diversi aspetti legati sia al software che all'hardware. Prima di introdurre nuove funzionalità si è pensato di migliorare l'architettura del software tramite una reingegnerizzazione riuscendo così ad ottenere un accoppiamento lasco del componente d'acquisizione. Il codice è stato ristrutturato e commentato adeguatamente consentendo una più semplice lettura del codice e una diminuzione del costo computazionale. La reingegnerizzazione favorisce, inoltre, il riuso del componente che può essere quindi utilizzato anche in altri sistemi applicativi che necessitano di un sistema di stereovisione attiva o ibrida. Sono state introdotte delle librerie per il supporto dei file di configurazione e della gestione dei parametri da riga di comando rendendo il sistema più flessibile e configurabile. È stato migliorato Il supporto ai led, introducendo un algoritmo efficente che tiene conto di un numero dinamico di led visibili per ciascuna camera (da 2 a 4) ottimizzando i confronti in base ai casi più probabili. È stata ampliata l'area di lavoro cambiando la tipologia dei led ed aumentandone l'intensità di corrente arrivando così a creare un'altra basetta di calibrazione ed un altro prototipo di smart-pen. Questo hardware è da considerarsi ottimo per il nostro caso, in cui le camere infrarosse hanno bassa sensibilità, dovrebbero essere rivalutati però nel caso in cui si decida di utilizzare delle camere migliori. È stata aggiunta un'infrastruttura di debug basata sulla simulazione di una

traccia, che tramite la replica della stessa acquisizione permette la valutazione di differenti configurazioni del sistema. È stato utilizzato, infine, un filtro di Kalman per effettuare la fusione tra il sensore inerziale (accelerometro) e quello visivo già presente. Come si è visto nei test, il filtro di Kalman utilizzato su un modello fornisce dei risultati ottimi, di poco migliori rispetto al filtro gaussiano utilizzato precedentemente. L'utilizzo dei soli dati inerziali accelerometrici, però, peggiorano la stima della punta e l'acquisizione.

Per migliorare la precisione del sistema si potrebbe, in futuro, agire sul sistema visivo utilizzando delle camere ad infrarossi più sensibili oppure gestire anche le informazioni del giroscopio e del magnetometro per ottenere dei dati inerziali precisi e affidabili. Quest'ultima soluzione permetterebbe, inoltre, di gestire i casi in cui per molto tempo non siano disponibili i dati visivi. Occorrerebbe inoltre reingegnerizzare il resto dell'applicazione separando concettualmente l'architettura in più componenti software.

Bibliografia

[1] R. Vinesh, Fernandes, J. Kiran, *Reverse Engineering. An Industrial Perspective.* Springer Series in Advanced Manufacturing, 2008

[2] A. Beccari, E. Farella, A. Liverani, S. Morigi, M. Rucci, *A fast interactive reverse-engineering system*, Computer-Aided Design 42 (p. 860-873), 2010

[3] M. Laforest *Wiiuse - the wiimote C library.* Disponibile qui

[4] T. Cuypers, T.V. Eede, S. Ligot, Y. Francken, C. Hermans, F. Arickx, P. Bekaert, *StereoWiision: Stereo Vision with Wiimotes.* International 3D Stereo Film and Technology Festival (3D Stereo MEDIA), 2009

[5] *CGAL User and Reference Manual: Principal Component Analysis* Disponibile qui

[6] *Inertial Measurement Unit*, Wikipedia. Disponibile qui

[7] G. Bradski, A. Kaehler, *Learning OpenCV. Computer Vision with the OpenCV Library.* O'Reilly Media, 2008

[8] R. Hartley, A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge University Press (p. 672), 2004

[9] G. Welch, G. Bishop, *An Introduction to the Kalman Filter.* SIGGRAPH Course 8, 2001

[10] *OpenCV 2.1 documentation: Kalman filter.* Disponibile qui