Università degli Studi di Milano

FACOLTÀ DI SCIENZE MATEMATICHE FISICHE E NATURALI

Corso di Laurea in Comunicazione Digitale

COMUNICAZIONE DIGITALE APPLICATA

INTERAZIONE NATURALE IN AMBIENTI DI REALTÀ VIRTUALE: TECNICHE STRUMENTI ED ESEMPI

Tesi di Laurea di:

MATHIEU BOSI

Relatore:

PROF. DANIELE MARINI

Correlatore:

ING. CINZIA ZANNONI

Sessione

Anno Accademico 2004/2005

Parole chiave:

Realtà virtuale

Interazione Naturale

Motion capture

Wireless Sensors Network

MEMS

Dipartimento di Scienze Matematiche, Fisiche e Naturali
Università degli Studi di Milano.

Alla mia famiglia e ai miei amici

Indice

Introduzione **1**

1 L'interazione naturale **3**

1.1 Tecnologie per la *natural interaction* 5

1.1.1 Spazializzazione del suono 5

1.1.2 Speech processing 6

1.1.3 Visualizzazione Immersiva 8

1.1.4 *Computer Vision* 9

1.1.5 Tecnologia aptica 11

1.1.6 Olfatto 12

1.1.7 *Ambient Intelligence* 13

1.2 Interazione naturale: alcuni casi di esempio 14

1.2.1 *Bricks* 14

1.2.2 *reacTable* 14

1.2.3 *Point At* 17

1.2.4 Multi-touch 18

2 Overview dell'hardware **21**

2.1 *Wireless Body Area Sensor Network* 21

2.2 Il guanto 22

2.2.1 Il nodo sensore *WiMoCA* 22

2.2.2 *Bend-sensor* 25

2.3 Il *Gateway* 26

3 Algoritmi **29**

3.1 Stabilimento della connessione con il sistema 29

3.1.1 Gestione dell'I/O da porta seriale 29
3.1.2 Controllo della sessione di comunicazione 30
3.1.3 Lettura dei dati provenienti dalla periferica 31
3.1.4 Decodifica dei pacchetti di dati 31
3.2 Filtraggio del segnale di accelerazione 34
3.2.1 Media viaggiante esponenziale (*EWMA*) 34
3.2.2 Un possibile miglioramento: tecniche di predizione della posa 35
3.3 Premessa sui sistemi di riferimento utilizzati 38
3.4 Uso dell'informazione di accelerazione 40
3.4.1 Rilevamento gravimetrico dell'inclinazione 40
3.4.2 Calcolo della posizione nello spazio 44
3.4.3 Gestione alternativa dello spostamento 45
3.4.4 *Threading* e interpolazione lineare per un'animazione fluida 48
3.5 Paradigmi di interazione adottati 50
3.5.1 Interpretazione delle posizioni delle dita 51
3.5.2 Stato di *idle* 52
3.5.3 Puntamento 52

4 Casi applicativi 59
4.1 *Glow*: un ambiente per il testing immediato 59
4.1.1 Generazione di un *avatar* 60
4.1.2 *Testing* iniziale con sequenze pre-registrate di dati 60
4.1.3 Scene di prova 61
4.2 *Il Muro della Memoria* 65
4.3 Lo schema di interazione 66
4.3.1 Vista globale 66
4.3.2 Vista ravvicinata e selezione della foto 66
4.3.3 Visualizzazione dei contenuti multimediali 67

Bibliografia 71

Elenco delle figure

1.1 Sistemi di visualizzazione immersivi 10
1.2 Un *graspable object* (a), un esempio di applicazione per il *floor planning* (b), più *brick* per definire una curva *spline*(c) 15
1.3 Una vista del *reacTable* 16
1.4 Immagini del sistema *Point At* 17
1.5 Il sistema *Multi-touch* 18
1.6 Lo schema di funzionamento della *FTIR* 18

2.1 Il guanto ad accelerometro 22
2.2 Strati del nodo sensore WiMoCA 22
2.3 Schema delle funzionalità del nodo sensore 23
2.4 Vista ravvicinata della batteria (a), di un nodo sensore (b) e del modulo *transceiver* (c) 23
2.5 *Bend-sensors* 26
2.6 Vista dall'alto del *gateway* USB 27
2.7 Architettura di funzionamento del *gateway* 27

3.1 Il controllo di una sessione di comunicazione 30
3.2 Formato dei pacchetti di dati 31
3.3 Il comportamento della EWMA sui valori di accelerazione rilevati durante un movimento "su-giù" rapido (a) e lento (b) per i valori α_{min} e α_{max} 36
3.4 Il sistema di riferimento utilizzato 38
3.5 Disposizione degli assi di rilevamento dell'accelerometro triassiale rispetto al *WCS* 39
3.6 La risultante della somma dei vettori $\mathbf{a}^s_{dyn}$ e $\mathbf{g}^s$ 42

3.7 l'andamento del valore di reattività ρ durante un periodo di interazione caratterizzato da movimenti più o meno bruschi della mano . . 44
3.8 Misuramento inerziale e doppia integrazione 45
3.9 Comportamento del filtraggio geometrico adottato sul valore di accelerazione complessiva $\mathbf{a}^s$ 47
3.10 Comportamento del rilevatore di inviluppo sui valori stimati di accelerazione dinamica 48
3.11 Interpolazione lineare dei valori 50
3.12 Gesture delle dita scelte per distinguere le fasi di puntamento, selezione, *dragging* e *idle* 51
3.13 Il cono di *idle* 52
3.14 Geometrie del puntamento 57

4.1 L'*avatar* della mano utilizzato in *Glow* 60
4.2 Scene di *testing* in *Glow* 63
4.3 *Glow* - alcune delle schermate di *debugging* e di visualizzazione dei valori di animazione 64
4.4 Bologna: monumento dei Caduti della Resistenza 65
4.5 La griglia planare di selezione di un sotto-blocco 66
4.6 I *widget* di spostamento ed il ciclo dello schema di interazione . . . 68
4.7 *Il Muro della Memoria* - vista generale e vista ravvicinata 69
4.8 Una delle pagine HTML contenenti i dati di un partigiano 70

Elenco delle tabelle

2.1 Caratteristiche del nodo WiMoCA 26

3.1 Disposizione dei dati rilevati dal sensore all'interno del corpo del pacchetto 31
3.2 Valori di *Id sensing* e corrispondenti *device* 32
3.3 Soglie adottate per le varie dita 51

Introduzione

La *Human computer interaction* (HCI) si trova agli incroci di molte aree scientifiche fra cui l'intelligienza artificiale, la *computer vision*, il riconoscimento di volti ed il *motion tracking*. La continua introduzione di nuovi dispositivi interattivi informatici nelle nostre case, uffici, auto, luoghi di commercio e turismo, implica la necessità di progettare una usabilità pervasiva nei vari contesti di utilizzo ponendo così la necessità di una *human-computer intelligent interaction* (HCII), la quale per essere efficace richiede che il computer sia in grado di interagire in modo *naturale* con l'utente. Le tecnologie per questi nuovi dispositivi di interazione sono perciò nettamente orientate a favorire scambi di informazione attraverso i canali sensoriali naturali della vista, del suono e del tatto. Questo tipo di interazione uomo-computer che mira essere simile, se non uguale, all'interazione che avviene quotidianamente tra esseri umani e tra esseri umani e mondo fisico conduce all'affermazione del paradigma dell'***interazione naturale***.

Il presente lavoro di tesi è stato volto all'esplorazione delle potenzialità offerte in questo ambito da un particolare dispositivo *wireless* di input: si tratta di un guanto che dispone di sensori di flessione per ogni dito e di un misuratore di accelerazioni incorporato sul dorso. Il sistema *embedded* che consente il funzionamento del guanto è chiamato **WiMoCA** [1] ed è sviluppato presso il *Micrel Lab* del DEIS di Bologna.

In particolare ciò che si è cercato di ottenere è un dispositivo che consentisse l'interazione di puntamento con una generica scena virtuale partendo dall'analisi dei dati di accelerazione e di posizione delle dita restituiti dalla periferica. In una prima fase del lavoro è stato realizzato un'apposito *software* chiamato ***Glow*** che ha consentito di sperimentare gli algoritmi necessari al funzionamento dell'interfaccia. Sono così state esplorate varie problematiche come l'interfacciamento con la

[1] acronimo di *Wireless Motion Capture with Accelerometers*

periferica, le tecniche di elaborazione dei dati e gli aspetti determinanti la qualità di interazione e di resa visiva.

In una seconda fase è poi stata reingegnerizzata un'applicazione esistente chiamata ***Il Muro della Memoria*** la quale propone una versione virtuale del sacrario dei Caduti della Resistenza che si trova a Bologna in piazza Nettuno. L'applicazione è stata sviluppata in ambiente *OpenMAF* [2] partendo da alcune librerie già esistenti, migliorandole o creandone di nuove dove necessario.

I risultati ottenuti con il dispositivo non sono ancora all'altezza di un'interfaccia per l'interazione naturale, tuttavia con il progredire dello sviluppo dell'hardware e tramite l'ausilio di algoritmi di classificazione per i gesti sarà possibile ricercare un miglioramento della qualità dell'interazione.

Nel primo capitolo verrà offerta una panoramica sull'interazione naturale e verranno presi in considerazione alcuni casi concreti di utilizzo di questo paradigma di interazione; si procederà quindi all'illustrazione del lavoro eseguito durante il periodo di tirocinio.

[2] acronimo di *Multimod Application Framework* (Framework per Applicazioni Multimodali), software *Open Source* sviluppato presso il consorzio interuniversitario CINECA di Casalecchio di Reno (BO).

Capitolo 1

L'interazione naturale

Spesso capita di avere a che fare con dispositivi di interazione dall'utilizzo non intuitivo o poco efficace, oppure con sistemi che necessitano di un tempo di apprendimento lungo. La necessità dell'attuazione di un paradigma di interazione evoluto che renda l'interazione fra uomo e macchina equiparabile in spontaneità alla quotidiana interazione fra uomo ed oggetti fisici viene discussa da Edwin L. Hutchins *et al.* nell'articolo "*Direct Manipulation Interfaces*" [29] sul primo volume del giornale *Human-Computer Interaction* sottolineando l'importanza del fattore del ***direct engagement***:

> (...) Per ottenere una sensazione di *direct engagement* l'interfaccia deve presentare all'utente un mondo in cui interagire. Gli oggetti di quel mondo devono essere percepiti come oggetti coinvolgenti, con i quali si possano fare cose e vedere come essi reagiscono. Per far sì che questo avvenga il linguaggio di *output* deve fornire rappresentazioni degli oggetti che si comportino secondo una forma che rispecchi il modo in cui l'utente pensa del comportamento di questi. Qualsiasi cambiamento venga causato negli oggetti dall'insieme di operazioni deve essere rispecchiato dalla loro rappresentazione. Questo uso dello stesso oggetto come entità sia di *input* che di *output* è essenziale per disporre di oggetti che si comportino come qualcosa che è *reale*. É proprio perché una espressione di *input* può contenere una precedente espressione di *output* che l'utente sente che l'espressione di *output* stessa è la cosa in sé e che l'operazione viene direttamente applicata ad essa. (...)

Proprio il paradigma dell'***interazione naturale*** mira ad offrire la possibilità di interagire attraverso interfacce che entrino il più possibile in simbiosi con le nostre

azioni quotidiane (come ad esempio i gesti, le parole, le espressioni del viso, i movimenti) e che si confondano con l'ambiente circostante (*ubiquitous computing* [45]). In questo ambito assumono importanza i sistemi *embedded*[1] non invasivi, invisibili, economici, affidabili, e allo stesso tempo *intelligenti*, ossia in grado di interpretare e soddisfare o addirittura di anticipare le necessità dell'utente adattandosi ad esso (*proactive computing* [46]) senza interferire nelle sue normali azioni.

L'interazione naturale riduce il divario fra i mezzi informatici ed il mondo fisico della vita di tutti i giorni e richiede che le interfacce siano diverse dalle tradizionali interfacce uomo-macchina basate su menù, icone, *mouse* e tastiera. In sintesi un'interfaccia per l'interazione naturale presenta le seguenti caratteristiche:

- gli schemi di interazione sono basati sui modelli di interazione persona-persona e persona-oggetto fisico
- è in grado tramite apposito *hardware* e *software* di comprendere azioni ed intenzioni degli utenti
- consente una elevata mobilità all'utente per una piena naturalezza nei movimenti
- scompare il più possibile nell'ambiente e negli oggetti creandone un *potenziamento* nelle loro funzionalità
- presta particolare attenzione ad aspetti estetici e cognitivi

Una interfaccia naturale consente di avere a che fare con forme di dati comprensibili e realistiche, includendo tipi misti di dato (es. dati provenienti da diverse modalità di ingresso come l'immagine e l'audio) e dovrebbe infine consentire a qualsiasi utente l'accesso a qualsiasi tipo di dato da qualsiasi tipo di terminale.

La digitalizzazione consente a tutti i tipi di modalità[2] (es. testo, suono, video) di essere trasformate in numeri e processate dal computer. Per questo computer con

[1] in informatica con il termine sistema *embedded* (sistema *incapsulato*) si identificano genericamente dei sistemi elettronici a microprocessore progettati appositamente per una determinata applicazione, spesso con una piattaforma hardware *ad-hoc*

[2] Nell'interazione uomo-macchina per *modalità* (*modality*) si intende una via di comunicazione fra l'umano ed il computer, come ad esempio la vista[30]

una percezione sensoriale intelligente possono fornire vie naturali di interazione che consentano una miglior integrazione nelle reciproche attività fra umani e computer.

La maggior limitazione esistente nei media attuali sta nel processo necessario all'utente per l'interpretazione completa della scena che viene rappresentata nell'immagine. Una visualizzazione *completamente realistica* può essere ottenuta con l'ausilio di una simulazione della percezione multimodale e dinamica del mondo reale, in modo da ottenere una rappresentazione che possa duplicare la fisicità che esiste nella realtà passando dal semplice foto-realismo alla completa immersione dell'osservatore in un ambiente di visualizzazione multisensoriale ed interattivo.

1.1 Tecnologie per la *natural interaction*

Ognuno dei cinque sensi umani rappresenta un possibile canale di comunicazione fra uomo e macchina. Grazie all'avanzare della ricerca scientifica e ai progressi nel campo della tecnologia il divario comunicativo fra uomo e macchina ha subito una notevole riduzione.

1.1.1 Spazializzazione del suono

Il fatto che il suono nel mondo reale venga percepito spazialmente fornisce lo spunto iniziale per la sua inclusione in uno scenario di simulazione. Di fatto una delle soluzioni per migliorare l'immersività e la qualità percepite durante una simulazione visiva virtuale è per l'appunto quella di utilizzare il suono come canale aggiuntivo verso l'utente [3]. La *spazializzazione del suono* è il processo che consente di creare un ambiente sonoro tridimensionale fedele ad un ambiente acustico o reale, o completamente virtuale o anche ibrido.

Il vantaggio del suono rispetto alla vista è che attraverso l'uso di queste tecniche di spazializzazione del suono possono essere simulate più fonti sonore provenienti da una qualsiasi direzione attorno all'utente fornendogli così un importante canale di *feedback* per renderlo partecipe di azioni o situazioni che avvengano al di fuori del suo campo visivo. Ad esempio sarebbe costoso sia in termini tecnici che economici fornire un *feedback* tattile all'utente (par. 1.1.5), tuttavia è altresì possibile utilizzare un suono per evocare una sensazione tattile, come nel caso di un tonfo o

di un suono di vetri infranti che venga riprodotto ogniqualvolta vengano oltrepassati determinati limiti di un mondo virtuale.

Come la tecnica del *texture-mapping* è in grado di migliorare la percezione visiva della profondità in una scena tridimensionale così un suono che sembri provenire da una particolare direzione (ad esempio un suono scrosciante proveniente da una cascata) può determinare un maggior grado di consistenza della simulazione.

1.1.2 Speech processing

Lo *speech processing*, ovvero lo studio del segnale del parlato e dei metodi necessari all'elaborazione di questo segnale, rende possibile l'interazione verbale con il calcolatore. L'elaborazione del segnale avviene in digitale, per questo lo *speech processing* può essere visto come l'intersezione delle tecniche di elaborazione digitale del segnale (*digital signal processing*, DSP) e delle tecniche di elaborazione del linguaggio naturale (*Natural Language Processing*, NLP).

I campi di ricerca di questa disciplina sono distinguibili in:

Speech synthesis (sintesi del parlato) ovvero la riproduzione artificiale del parlato attraverso una varietà di algoritmi di sintesi.

Speech recognition (riconoscimento del parlato) analisi del contenuto linguistico del segnale

Speaker recognition (riconoscimento di chi parla) in cui l'obbiettivo è riconoscere l'identità di chi sta parlando

Speech coding (codifica del parlato) si occupa di ricercare le soluzioni ottimali per la trasmissione dell'informazione del parlato, come ad esempio le tecniche di compressione

Miglioramento del segnale parlato ossia le tecniche di elaborazione del segnale necessarie a rimuovere interferenze nel segnale come i rumori ambientali per consentirne una elaborazione efficace.

Il riconoscimento del parlato automatico (*Automatic Speech Recognition*, ASR) insieme alla sintesi del parlato rappresenta un primo passo nella direzione dell'interazione naturale verbale fra uomo e macchina [4].

Speech Synthesis

I primi sistemi elettronici per la sintesi del parlato avevano un suono estremamente artificiale ed erano spesso poco intellegibili. Tuttavia la qualità del parlato sintetico è nettamente migliorata tanto che talvolta una voce generata da un sistema di sintesi vocale odierno è difficilmente distinguibile dal parlato umano.

I progressi nel campo della sintesi del parlato e in generale del segnale vocale consentono oggi ad un computer non solo di parlare ma anche di cantare in maniera realistica, cosa che avviene ad esempio con il sistema *Vocaloid* sviluppato dalla *Yamaha* [5].

Speech Recognition e *Speaker Recognition*

Le applicazioni per il riconoscimento automatico del parlato trovano già applicazione ad esempio nel campo della refertazione clinica automatica [6] consentendo di sostituire buona parte dei metodi di dettatura basati sulla trascrizione umana. Un altro utilizzo notevole del riconoscimento del parlato è quello riguardante l'accessibilità, dato che si tratta di un efficace mezzo per l'utilizzo del computer per le persone che per esempio hanno limitazioni nell'uso delle mani.

Speech Interaction

Lo studio della *speech interaction* (interazione verbale) con il computer [7] è il passo successivo nell'attuazione dell'interazione naturale verbale con il computer. La *speech interaction* studia il processo che controlla le funzioni del parlato e l'interazione con l'utente. Le regole della *speech interaction* stabiliscono quando il sistema parla, ciò che esso dice e quando viene dato ascolto all'utente e che cosa viene ascoltato. Inoltre è anche responsabile della configurazione dell'input e dell'output del parlato, nonché del riconoscimento di chi sta parlando. Questa disciplina è strettamente correlata sia con le discipline legate all'intelligenza artificiale che alle discipline di studio delle relazioni tra esseri umani.

1.1.3 Visualizzazione Immersiva

Head Mounted Display (HMD)

Il sistema HMD [11] (fig. (a)) viene indossato sul capo ed è provvisto di una coppia stereoscopica di schermi miniaturizzati che forniscono all'utente l'immagine che ricevono da un computer, isolando efficacemente l'utentedal mondo esterno ed immergendolo in ciò che viene visualizzato. Un sistema di *tracking* fornisce i dati di posizione e di orientazione della testa in modo che il computer possa modificare l'immagine visualizzata in accordo con la posizione della testa creando l'illusione di essere presenti fisicamente nell'ambiente virtuale.

PowerWall

Lo scopo primario di un *PowerWall* [12] (fig. (b)) è quello di consentire la visualizzazione a risoluzioni molto elevate di immagini provenienti o da simulazioni scientifiche o da applicazioni di resa delle immagini in alta risoluzione. Oltre a possedere una risoluzione elevata questo sistema fornisce un'ampia area di visualizzazione per facilitare la collaborazione di più utenti sulla stessa immagine, in modo che ognuno di questi possa ad esempio camminare fino allo schermo per poter indicare manualmente regioni di interesse come avverrebbe su di una lavagna. Alcuni modelli recenti di *PowerWall* consentono anche la visione stereoscopica attraverso l'utilizzo di sistema di stereoscopia attiva[3].

VisionDome

Il *VisionDome* [14] (fig. (c)) fornisce la proiezione di un'immagine su di un grande schermo emisferico inclinato che riesce a coprire l'intero campo visivo dello spettatore creando un senso di piena immersione.

[3] ovvero vengono indossati occhiali dotati di otturatori a cristalli liquidi opportunamente sincronizzati con la coppia stereoscopica di proiettori

CAVE

Il sistema CAVE[4] [15] (fig. (d)) basa il suo funzionamento sulla proiezione delle immagini su due o più schermi disposti in maniera tale da circondare lo spettatore. Alternativamente le immagini possono essere proiettate direttamente sulle pareti e sul pavimento di una stanza. Nelle versioni più avanzate di questo tipo di sistema viene tenuta traccia dei movimenti dello spettatore per poter ricalcolare la giusta prospettiva per ognuna delle superfici proiettate consentendo la libertà di muoversi fisicamente nello spazio. Esistono anche tipi di CAVE dotati di sistemi di proiezione stereoscopica.

CyberSphere

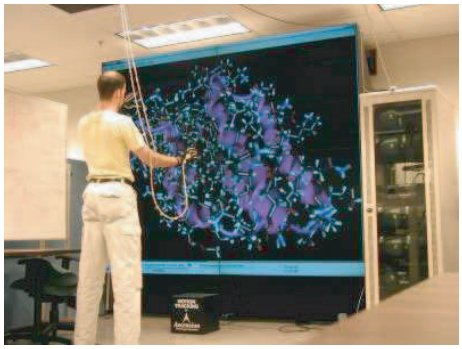
Il sistema di visualizzazione *Cybersphere* [13] (fig. (e)) supera la principale limitazione dei sistemi fino ad ora citati, ovvero la possibilità per l'utente di deambulare nell'ambiente virtuale come farebbe naturalmente, questo perché egli è vincolato o dai limiti spaziali come nel sistema CAVE o dalla portata limitata dei sistemi di tracking per l'HMD. La *Cybersphere* racchiude completamente l'utente nel mondo virtuale. Il sistema di proiezione è costituito da una sfera traslucida cava del diametro di 3.5 metri in cui l'utente entra passando da un apertura serrabile. La sfera è sostenuta da un cuscino di aria a bassa pressione ed i movimenti in qualsiasi direzione della camminata dell'osservatore ne causano la rotazione. La sfera a sua volta è interfacciata con sensori di rotazione, in questo modo il computer può conoscere i movimenti dell'utente consentendogli di esplorare il mondo virtuale in maniera naturale simulando la corretta vista che viene poi proiettata sulle pareti della sfera da appositi proiettori ad alta intensità.

1.1.4 *Computer Vision*

La *computer vision* comprende lo studio e l'applicazione dei metodi che consentono ai computer di *comprendere* il contenuto di un'immagine, ovvero di estrarre da essa un'informazione specifica per un dato scopo. Solitamente i sistemi di *computer vision* partono dall'elaborazione di immagini digitali in scala di grigi o a colori che

[4] acronimo ricorsivo di *Cave Automatic Virtual Environment*. *cave* (caverna) questo nome è anche un riferimento all'allegoria della Caverna ne La Repubblica di Platone dove il filosofo valuta percezione, realtà e illusione.

(a) *Head Mounted Display*



(b) *PowerWall*

(c) *VisionDome*

(d) Sistema CAVE

(e) *Cybersphere*

Figura 1.1: Sistemi di visualizzazione immersivi

possono provenire ad esempio da una o più telecamere di vario tipo (es. ad infrarossi [36]). Nella maggior parte delle applicazioni di *computer vision* il calcolatore viene programmato appositamente per la risoluzione di un compito determinato, tuttavia è attiva la ricerca di metodi che consentano l'apprendimento da parte della macchina.

La computer vision può essere usata efficacemente nell'interazione con il computer [9] ed è l'ideale nell'ottica dell'interazione naturale poiché consente il rilevamento ottico dinamico delle geometrie fisiche in una maniera completamente non invasiva per l'utente [39] permettendo l'attuazione di tecniche quali il *motion-capture* dei movimenti umani [8] ed il riconoscimento di volti [16].

1.1.5 Tecnologia aptica

La tecnologia aptica[5] consente di avere un interfaccia con l'utente di tipo tattile. Si tratta di una tecnologia emergente che offre vasti campi di applicazione. Le interfacce aptiche (*haptic interfaces*) forniscono all'utente il *feedback* della forza che verrebbe percepita durante l'interazione con un dato oggetto concreto attraverso attuatori elettromeccanici controllati da un calcolatore. Attraverso questa tecnologia i dati che rappresentano le sensazioni tattili come la ruvidità [24] possono ad esempio essere registrati e riprodotti.

Una delle principali applicazioni di questa tecnologia è la simulazione virtuale di interventi chirurgici. Tramite la simulazione fisica del comportamento dei vari tessuti del corpo umano [26] è possibile restituire la sensazione tattile che si avrebbe ad esempio durante il taglio [27] e con opportune tecniche è anche possibile interagire direttamente con dati medici tridimensionali [25].

La tecnologia aptica è ormai ampiamente accettata come componente chiave dei sistemi di realtà virtuale per aggiungere il senso del tatto a quelle che prima erano solo soluzioni visuali. Alcune di queste interfacce aptiche utilizzano penne impugnabili o altri strumenti collegati ad un braccio meccanico come nel caso del dispositivo chiamato *Phantom* [22]. Dispositivi del genere sono in grado di fornire

[5] dal greco *Hapthai*, significa "riguardante il senso del tatto"

un *feedback* tattile molto realistico su di un computer odierno.

Già da tempo esistono altre forme più economiche di dispositivi aptici per l'uso con i videogiochi, come ad esempio i cosiddetti *rumbe pack* presenti in alcuni *joypad* o volanti che restituiscono ad esempio una simulazione della forza che normalmente si genera durante lo sterzo delle ruote.

Anche in campo artistico i dispositivi aptici sono usati per consentire all'artista di avere contatto diretto con applicazioni di sintesi sonora, di animazione e di modellazione. Ad esempio è stato costruito un particolare dispositivo che simula un violino restituendo le vibrazioni delle varie corde virtuali a partire dalla pressione ed espressività dell'archetto tenuto dall'artista [23].

(a) Il sistema aptico *Phantom*

(b) Un volante dotato di *force-feedback*

1.1.6 Olfatto

Nella HCI l'olfatto rappresenta ancora un canale di comunicazione pressoché inesplorato per il fatto che non esiste ancora una conoscenza dettagliata di come avvenga nell'essere umano la percezione degli odori e di come si possano sintetizzare odori complessi a partire da componenti semplici, oltre a ciò non esistono ancora schemi validi di classificazione o di descrizione degli odori.

Al giorno d'oggi è comunque possibile acquistare a poco prezzo hardware facilmente controllabile per l'emissione di essenze [19], inoltre incorporare la gene-

razione di odori in una applicazione è oggigiorno relativamente semplice.

Sono stati condotti esperimenti che dimostrano come la modalità dell'olfatto possa essere utilizzata per notificare ad un utente un evento pur non distraendolo dall'esecuzione di un task principale [20].

Altri studi [21], basandosi sul fatto che il senso dell'olfatto presenta un forte collegamento con la memoria e le emozioni, hanno investigato la possibilità di usare l'olfatto per la ricerca in una collezione di foto digitali ottenendo risultati incoraggianti.

L'olfatto è anche stato proposto in diversi sistemi per di realtà virtuale per migliorarne la resa di fedeltà [18]. Ad esempio uno di questi sistemi [17] ha previsto l'utilizzo di un sistema di pompaggio dell'aria per presentare aromi in un ambiente virtuale. Sono stati forniti un aroma per l'ambiente virtuale visualizzato e altri aromi per i vari oggetti in esso presenti registrando così un maggior gradimento dell'esperienza da parte degli utenti coinvolti.

1.1.7 *Ambient Intelligence*

L'***Ambient Intelligence*** (*AmI* [47]) è un nuovo modo di intendere lo sviluppo dei sistemi integrati, basato sull'idea di rendere i dispositivi non più semplicemente *embedded*, ma completamente inseriti nell'ambiente circostante, disponibili all'occorrenza e attivabili con semplici operazioni. Ciascun dispositivo deve contenere al suo interno uno o più sensori (quali ad esempio sensori di posizione, temperatura, accelerazione) le cui informazioni opportunamente rielaborate siano in grado di fornire indicazioni riguardo un particolare evento.
L'*Ambient Intelligence* è il punto di incontro di tre *trend* tecnologici:

ubiquitous computing presenza dei microprocessori negli oggetti con cui interagiamo ogni giorno

ubiquitous communication strumenti per la comunicazione *wireless* dei dispositivi fra loro

intelligence interfaces interfacce che si adattano all'utente

Il concetto di *AmI* racchiude al suo interno tutto ciò che riguarda l'intelligenza dei dispositivi *embedded* nell'intuire la situazione da monitorare, l'invisibilità del dispositivo stesso, il suo grado di adattamento e la sua personalizzazione. Sono proprio queste caratteristiche a creare un forte nesso con la *natural interaction* per quanto riguarda la capacità dei dispositivi informatici di entrare in simbiosi con la realtà fisica quotidiana.

1.2 Interazione naturale: alcuni casi di esempio

1.2.1 *Bricks*

Nel sistema *Bricks* [32] si introduce il concetto di *Graspable User Interface*[6], cioè una UI che consente il controllo diretto di oggetti elettronici virtuali attraverso oggetti fisici impugnabili (*graspable objects*). Questi oggetti fisici sono di fatto un nuovo dispositivo di *input* che può essere strettamente correlato a oggetti virtuali per la loro manipolazione o per esprimere un'azione. Gli oggetti fisici, detti *bricks*, (mattoncini) sono stati utilizzati su di un ampio *display* orizzontale ed un'applicazione chiamata *GraspDraw* al fine di esplorare le idee alla base delle *Graspable UI* e per progettare tecniche di interazione a una e a due mani.

1.2.2 *reacTable*

Il *reacTable* è uno strumento musicale elettronico dotato di una *Tangible User Interface* sviluppato presso il *Music Technology Group* dell'università di *Pompeu Fabra* di Barcellona. Gli oggetti (*tangibles*) sul reacTable rappresentanoi componenti di un classico sintetizzatore modulare.
Le caratteristiche salienti del *reacTable* sono:

- i *tangibles* sono passivi, cioè non dispongono né di sensori né di attuatori
- sonoramente avvincente ed interessante
- uso intuitivo: nessun manuale, nessuna istruzione, apprendibile anche da un bambino

[6] interfaccia utente afferrabile

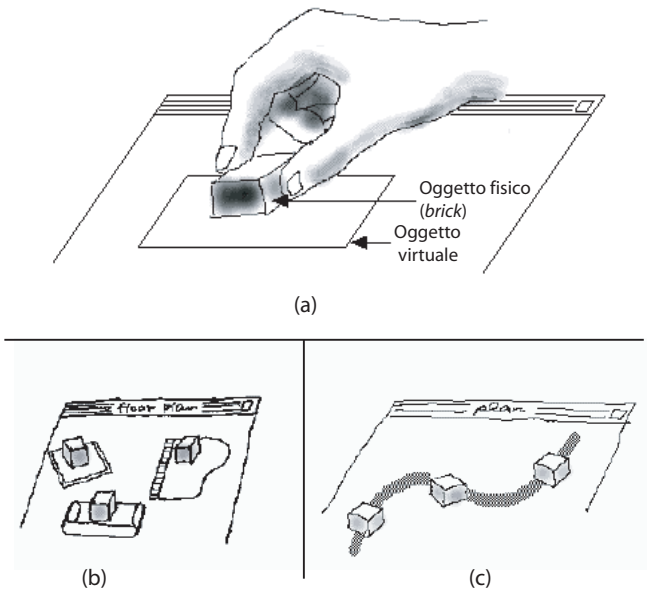


Figura 1.2: Un *graspable object* (a), un esempio di applicazione per il *floor planning* (b), più *brick* per definire una curva *spline*(c)

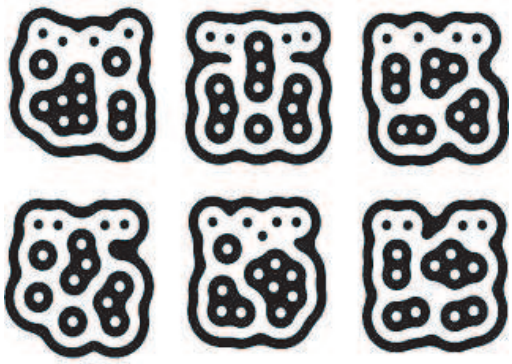
- uso collaborativo: più esecutori (anche remoti) possono interagire contemporaneamente

L'*hardware* del *reacTable* si basa su di un tavolo rotondo traslucido e su di una videocamera posizionata al di sotto di esso che ne analizza continuamente la superficie per ricavare il tipo, la posizione e l'orientazione degli oggetti posti sulla sua superficie. Questi oggetti rappresentano i componenti di un classico sintetizzatore modulare: gli utenti interagiscono cambiandone posizione ed orientazione controllando direttamente la struttura topologica e i parametri del sintetizzatore. Inoltre, sempre da sotto il tavolo, un proiettore traccia animazioni dinamiche sulla superficie fornendo un *feedback* visuale dello stato, dell'attività e delle caratteristiche principali del suono prodotto dal sintetizzatore.

Un sistema chiamato *reacTIVision* [34] tiene traccia degli appositi *fiducial markers* presenti sui *tangibles* nel flusso video *real-time*. Tramite particolari algoritmi di classificazione l'immagine di ogni fotogramma viene analizzata per risalire all'informazione che identifica l'oggetto, la sua posizione e la sua orientazione. I dati così raccolti vengono codificati secondo il protocollo *TUIO* [35] e trasmessi tramite una connessione *OSC*[7] all'applicazione *client*.

[7] *OSC* [38] è l'acronimo di *OpenSound Control*. Si tratta di un protocollo di comunicazione per computer, sintetizzatori e altre periferiche multimediali. Ottimizzato per le moderne tecnologie di

(a) Una vista del *reacTable*



(b) Alcuni *fiducial markers*

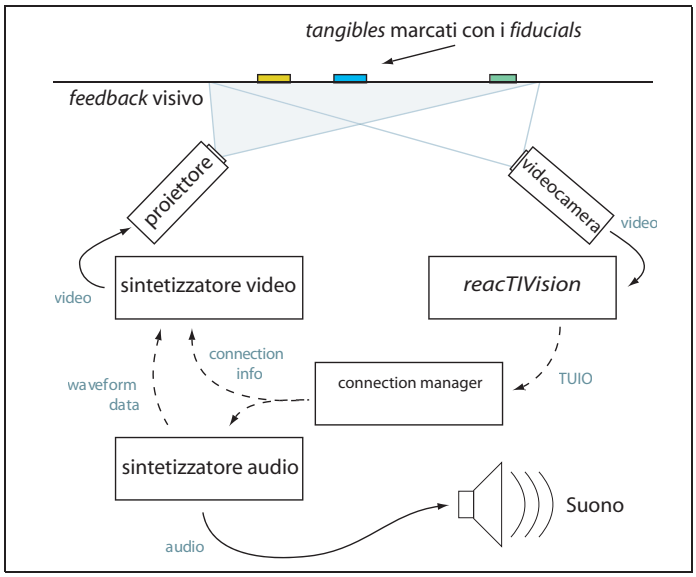


Figura 1.3: Una vista del *reacTable*

1.2.3 *Point At*

Il sistema *Point At* [36] è stato sviluppato in Italia e si trova presso il museo di Palazzo Medici Riccardi a Firenze ed è stato sviluppato per consentire ai visitatori di esplorare in una maniera semplice e intuitiva una versione digitale dell'affresco "*Il viaggio dei Magi*". I visitatori sostano dinnanzi ad un grande schermo retro-proiettato e puntano con le loro mani verso le parti dell'affresco a cui sono interessati. Due videocamere digitali catturano l'azione di puntamento del visitatore ed un algoritmo di *computer vision* calcola l'esatta parte di dipinto da egli indicata. Lo schermo mostra solo una porzione dell'affresco ma il visitatore può spostare la vista a destra o a sinistra puntando verso due apposite icone dalla forma di freccia. Sulla vista dell'affresco sono evidenziate con una leggera sagoma (fig. 1.4(a)) una quantità di aree di interesse selezionabili che includono oggetti o persone. L'immagine ingrandita (fig. 1.4(b)) presenta anche un apposito bottone che quando viene puntato riporta la vista allo stato iniziale. Quando il visitatore sosta per più di un secondo su una di queste aree selezionabili questa viene ingrandita, appare un breve testo di contestualizzazione e una voce fornisce informazioni concise sulle persone o sull'oggetto di interesse. Il grande vantaggio nell'utilizzo di questo sistema di *computer vision* è che i visitatori non devono indossare nessun tipo di dispositivo.

(a) una sagoma selezionabile (b) vista ravvicinata

Figura 1.4: Immagini del sistema *Point At*

networking, viene comunenente utilizzato con il protocollo UDP.

1.2.4 Multi-touch

Jeff Han, consulente del Dipartimento di *Computer Science* della *New York University*, ha presentato all *ETech* 2006 (*ÒReilly Emerging Technology Conference*) il suo ultimo progetto, denominato *Multi-touch interaction* [37].

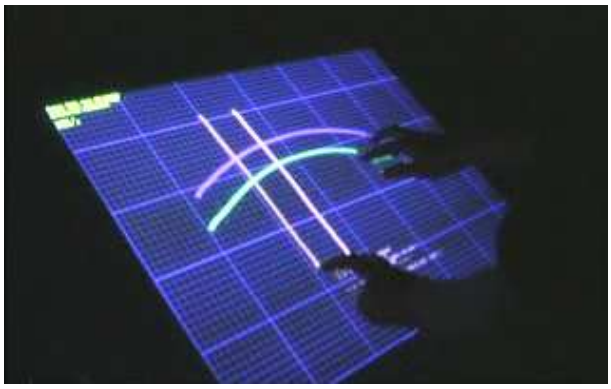


Figura 1.5: Il sistema *Multi-touch*

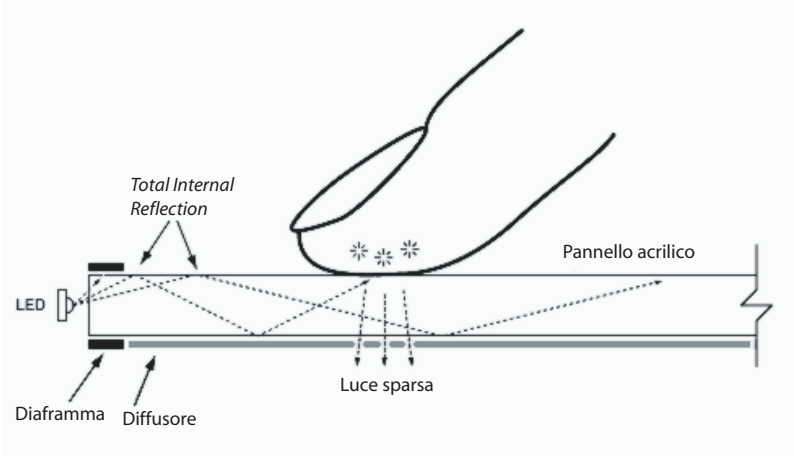


Figura 1.6: Lo schema di funzionamento della *FTIR*

Mentre i comuni *touch-screen* elaborano un singolo punto di contatto alla volta, il *Multi-touch* rende possibile un'interazione più completa ed elaborata grazie al riconoscimento di più punti di pressione. La tecnica adottata (fig. 1.6) si basa sulla *frustrated total internal reflection* (*FTIR*), una tecnica usata nel campo della biometria per l'acquisizione delle impronte digitali. Questa tecnica è sensibile alla forza di pressione e provvede una risoluzione ed una scalabilità senza precedenti. É possibile creare sofisticati *multi-point widgets* per applicazioni sufficientemente

ampie da accogliere entrambe le mani e più utenti. Le modalità di interazione con l'oggetto o gli oggetti visivi presentati a schermo possono essere amplificate a dismisura. Ad esempio con due dita o con le due mani grazie a particolari sequenze di contatto è possibile spostare immagini sullo schermo, ruotare e traslare fotografie, dimensionarle a piacimento, eseguire editing, gestire archivi e console musicali. Per ora è stato utilizzato un display a retro-proiezione che misura $91,44$ x $68,58$ *cm* e che ha una risoluzione tattile di $0,25$ *cm* ad una frequenza di campionamento di 50 *Hz*.

Le informazioni sugli eventi di tocco sono inviate all'applicazione utilizzando il protocollo *OSC* su di una trasmissione UDP. La ricerca verte ora sull'implementazione di display con dimensioni diverse e di un sistema in grado di riconoscere la digitazione di ogni singolo dito, in modo da poter attribuire una funzione a ciascuno di essi.

Capitolo 2

Overview dell'hardware

2.1 *Wireless Body Area Sensor Network*

L'attuale diminuzione delle dimensioni dei computer e dei sensori consente alle persone di indossare questi dispositivi in una maniera simile ai vestiti, da qui il termine *wearable computing*. Un *wearable computer* è di fatto un piccolo *computer* progettato per essere indossato sul corpo durante l'uso e solitamente viene o integrato negli indumenti oppure fissato al corpo ad esempio attraverso un braccialetto o un guanto. Più *wearable computer* che dispongano di sensori e che comunichino tra di loro attraverso una connessione *wireless* danno vita a quella che viene detta *Wireless Body Area Sensor Network* (WBASN [42]). Usando una metafora biologica una WBASN può essere vista come il *sistema sensoriale* di un'oggetto intelligente, in particolare di un *indumento intelligente*.

Il concetto di WBASN è una specializzazione del più comune concetto di *Wireless Sensor Network* (WSN) dove più *nodi sensore* (nodi in quanto appartenenti ad una rete) partecipano nel compito di rilevamento di qualche particolare fenomeno fisico. Le WSN trovano già da tempo applicazioni che spaziano dalla sicurezza [54, 55] alla già citata *Ambient Intelligence*.

Il nodo sensore *WiMoCA* [43] è un nodo sensore *wireless* composto da un accelerometro triassiale integrato, un microcontrollore ed un *transceiver* wireless che è consente di creare una WBASN per un sistema distribuito di riconoscimento dei movimenti in cui i nodi sono fissati su varie parti del corpo umano. Per questo la-

voro di tesi è stato utilizzato un singolo nodo sensore fissato sul dorso di un guanto provvisto anche di *bend-sensors*.

2.2 Il guanto

Figura 2.1: Il guanto ad accelerometro

2.2.1 Il nodo sensore *WiMoCA*

Gli attuali nodi-sensore utilizzati (figura 2.2) sono denominati *WiMoCA*[1] [42]. Sono stati progettati per essere indossabili e a basso consumo energetico, con un'architettura modulare per facilitare la sostituzione o l'aggiornamento di ogni componente.

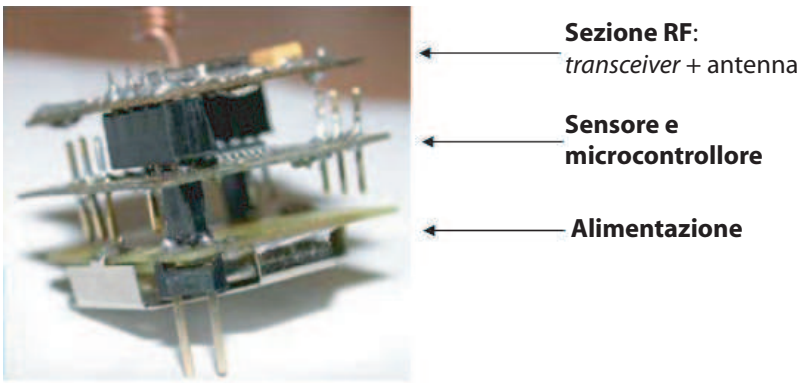


Figura 2.2: Strati del nodo sensore WiMoCA

[1] *WiMoCA*, acronimo di *Wireless Motion Capture*

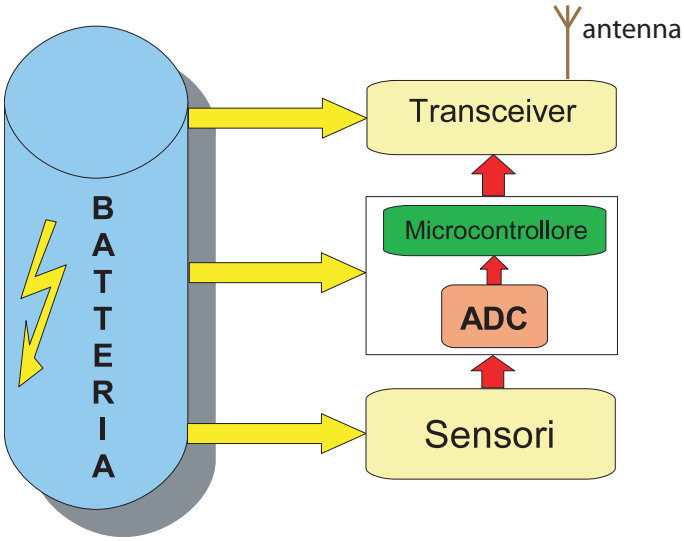
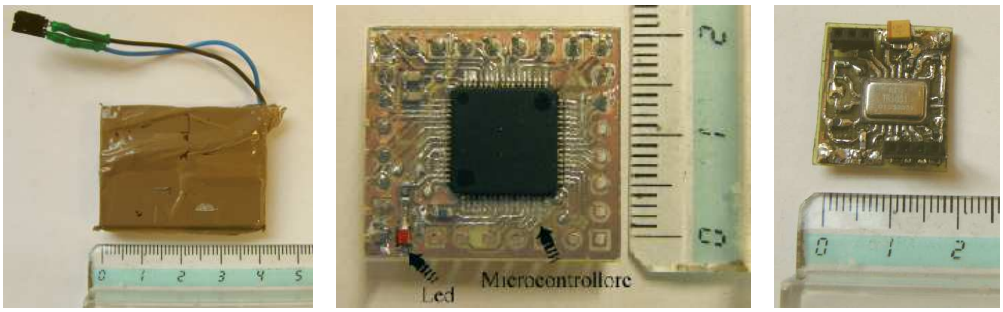


Figura 2.3: Schema delle funzionalità del nodo sensore



(a) (b) (c)

Figura 2.4: Vista ravvicinata della batteria (a), di un nodo sensore (b) e del modulo *transceiver* (c)

Il microcontrollore è suddiviso a strati, corrispondenti ai vari blocchi funzionali come presentato in figura 2.3. I livelli costitutivi il nodo sensore sono 3:

- L'alimentazione
- Il microcontrollore e il sensore inerziale
- Il transceiver radio

Il cuore del nodo sensore è un microcontrollore[2] interfacciato con un *transceiver* [3] la cui massima distanza di trasmissione è attorno ai 50 metri in ambiente aperto. Come *alimentazione* viene usata una apposita batteria compatta al litio ricaricabile da 3.6 V. Grazie alla modularità offerta da questo tipo di nodo possono essere implementare nuove applicazioni semplicemente inserendo un nuovo *layer* nel nodo o risviluppando un *layer* già esistente.

Accelerometro triassiale MEMS

Un accelerometro triassiale, da qui in poi chiamato semplicemente *accelerometro*, è un trasduttore elettromeccanico che misura le forze di accelerazione in tre direzioni. Queste forze possono essere:

- *statiche* come ad esempio la semplice forza di gravità
- *dinamiche* ovvero causate da una variazione di velocità o comunque dalla vibrazione dell'accelerometro.

Gli accelerometri sono costruiti sfruttando varie tecnologie. Alcuni accelerometri sfruttano l'*effetto piezoelettrico*[4] alimentato dalle forze di accelerazione. Altre tecniche prevedono il rilevamento dei cambiamenti della *capacità elettrica* che si genera fra due micro-strutture poste in stretta prossimità l'una dell'altra. Quando una forza di accelerazione muove una delle due strutture, la capacità elettrica cambia. Grazie alla presenza di un apposito circuito elettronico questa capacità elettrica può essere convertita in un voltaggio proporzionale all'accelerazione. Esistono anche altre tecnologie che consentono la realizzazione di accelerometri, il tipo più

[2] ATMEL ATmega8 ad 8-bit [76], lo stesso utilizzato nel *gateway*

[3] si tratta di un TR1001 [74] che opera nella banda libera Europea degli 868 MHz con un *bit-rate* massimo di 115.2 kbps.

[4] La *piezoelettricità* (la parola deriva dal greco *piezein*, pressione, compressione) è la proprietà di alcuni cristalli di produrre un voltaggio quando sono soggetti ad una deformazione meccanica [1].

elementare di accelerometro è comunque costituito da una *massa di prova* vincolata al movimento lungo un unico asse e che viene riportata in una posizione centrale da una molla. Accelerometri ad alta precisione chiamati *closed-loop* utilizzano campi elettromagnetici al posto delle molle per riportare la massa di prova verso la sua posizione di riposo. Questo ultimo tipo di accelerometri viene oggi realizzato con l'uso della tecnologia MEMS[5] [50, 51] basando la rilevazione su una quantità di fenomeni fisici. Accelerometri MEMS con sensibilità di $1 - -10$ mg vengono oggi prodotti in massa, possono essere acquistati a basso prezzo e sono utilizzati estensivamente in applicazioni commerciali; esistono anche accelerometri che possiedono un'accuratezza di 10 μg. Nel caso in esame l'accelerometro utilizzato dal nodo sensore [73] possiede una sensibilità di circa 1 mg.

Vi sono alcuni usi già da tempo consolidati per gli accelerometri, ad esempio vengono usati in alcuni computer per proteggere gli *hard-disk* da danni dovuti a cadute accidentali [48], inoltre accelerometri con un range dinamico elevato (detti anche *high g accelerometers*) sono lo standard industriale per il rilevamento dinamico negli autoveicoli al fine di attivare per tempo i dispositivi di sicurezza come gli *air-bag* o i sistemi di frenata intelligenti e di controllo della stabilità [49]. Misurando l'accelerazione statica causata dalla gravità è possibile ricavare l'orientazione del dispositivo rispetto al piano terrestre. Misurando invece il valore dell'accelerazione dinamica è possibile analizzare il modo in cui il dispositivo si sta muovendo.

2.2.2 *Bend-sensor*

I *bend-sensor* (detti anche *flex-sensors*) sono particolari dispositivi piezoresistivi, che cambiano cioè la propria resistenza elettrica in base alle sollecitazioni meccaniche che ricevono. Sono posizionati al di sopra di ogni dito in modo da fornire un valore di flessione per ognuno di essi. Questa versione del nodo *WiMoCA* è progettata *ex-novo* per gestire anche questa informazione per ogni dito [40]. Un problema dei *bend-sensor* è che col tempo essi vanno in contro ad usura meccanica e per questo è necessario ricalibrare via *software* il *range* dei valori letti all'inizio di ogni sessione di utilizzo.

[5] acronimo di *Micro Electro Mechanical Systems*

Figura 2.5: *Bend-sensors*

Caratteristiche principali del nodo	
Sensibilità sensore inerziale	0.001 g
Range massimo di rilevamento	$\pm 2g$
Risoluzione	12 bit
Frequenza di *sampling*	20 Hz

Tabella 2.1: Caratteristiche del nodo WiMoCA

2.3 Il *Gateway*

Il *gateway* è un nodo particolare della rete: deve assolvere le funzioni di *bridge* (ponte) fra i nodi sensore applicati sul corpo (*Body Sensor Network*, BSN) e il terminale di elaborazione. Altro compito del *gateway* in questo tipo di rete è di gestire i protocolli di comunicazione all'interno della rete. Concettualmente ogni nodo della rete può lavorare come *gateway* se viene equipaggiato con un transceiver Bluetooth [44] o con un qualsiasi altro tipo di interfaccia con un dispositivo *general purpose*.

In questo scenario, è possibile delegare al terminale centralizzato tutti i compiti legati all'elaborazione dei segnali; ciascun nodo sensore ha quindi il solo compito di collezionare e trasmettere i dati acquisiti, limitando la sua complessità e quindi la sua potenza dissipata. La minor potenza di calcolo richiesta ai sensori *wireless* viene tuttavia compensata dai maggiori requisiti di connettività e di *performance* sulla progettazione dell'intero sistema.

Il *gateway* è in grado da un lato di interrogare in qualsiasi momento ogni singolo nodo della rete e dall'altro di comunicare con il sistema remoto, facendo dialogare

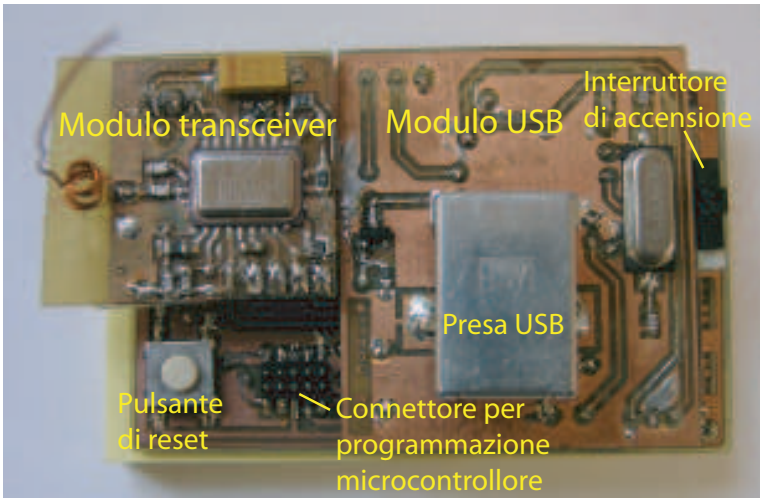


Figura 2.6: Vista dall'alto del *gateway* USB

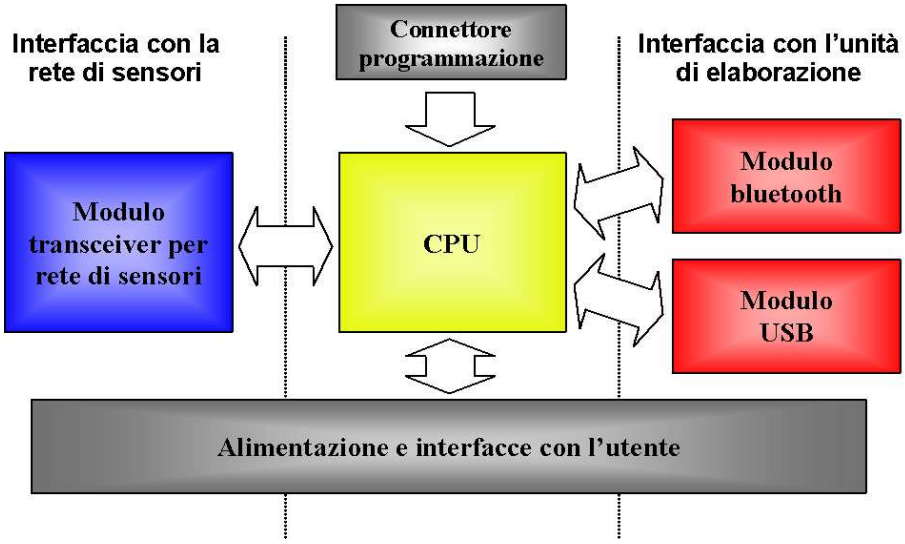


Figura 2.7: Architettura di funzionamento del *gateway*

in modo trasparente i nodi della rete con quest'ultimo. In altre parole, il *gateway* ha il compito di svincolare la rete di sensori dal terminale centralizzato, occupandosi di problematiche di interfacciamento di basso livello (temporizzazioni, vincoli elettrici, gestione di segnali interferenti, riduzione del consumo di potenza,..) e della gestione e sincronizzazione della rete. A tal fine, il *gateway* dispone di un nucleo computazionale in grado di memorizzare ed elaborare i dati per fornirli al sistema remoto, al quale viene affidato il solo compito di lettura ed uso del dato. Grazie al *gateway*, il terminale remoto accede alla rete di sensori in modo astratto tramite l'accesso ad una porta seriale (virtuale).

La realizzazione complessiva del *gateway* è orientata alla riduzione delle dimensioni e dei consumi di potenza, mediante un'accurata selezione dei componenti utilizzati e una gestione delle modalità di basso consumo dei moduli periferici inutilizzati. Come interfacce, il *gateway* presenta da una parte il *transceiver* per la rete di sensori e dall'altra un modulo per la connessione USB. L'alimentazione viene prelevata direttamente dal dispositivo *general purpose* a cui è collegata.

Capitolo 3

Algoritmi

Verranno ora illustrati gli algoritmi sono stati implementati per l'utilizzo del guanto ad accelerometri. Questi algoritmi sono stati principalmente creati durante le sessioni di sperimentazione con il *software* di testing *Glow* (cap. 4).

3.1 Stabilimento della connessione con il sistema

3.1.1 Gestione dell'I/O da porta seriale

Innanzitutto per poter stabilire il collegamento con il sistema è stato necessario interfacciarsi con le apposite *API*[1] di lettura dei dati da porta seriale fornite dal sistema operativo Microsoft Windows®.

L'accesso alla porta seriale (virtuale) avviene attraverso funzioni di astrazione dell'accesso ai *file* con l'ausilio di apposite funzioni di configurazione specifiche della porta seriale. A causa della complessità implicata dalla gestione corretta delle comunicazioni con questo dispositivo si sono cercate soluzioni che fossero già state implementate da terzi. Dopo diversi tentativi con varie librerie di pubblico dominio è stata trovata una libreria scritta per il compilatore *LCC*[2] che ha dovuto

[1] *API* è l'acronimo di *Application Program(ming) Interface*, indica ogni insieme di procedure disponibili al programmatore, di solito raggruppate a formare un *set* di strumenti specifici per un determinato compito. È un metodo per ottenere un'astrazione, di solito tra l'hardware e il programmatore, o tra software a basso ed alto livello. Le API permettono di evitare ai programmatori di scrivere tutte le funzioni dal nulla. Le API stesse sono un'astrazione: il software che fornisce una certa *API* è detto *implementazione dell'API*[1].

[2] *Local C Compiler* o *Little C Compiler*, si tratta di un compilatore ANSI C gratuito.

essere adattata all'ambiente di sviluppo utilizzato. La libreria fornisce le funzioni essenziali per l'accesso in lettura e scrittura alla porta seriale e per la gestione degli errori di connessione.

3.1.2 Controllo della sessione di comunicazione

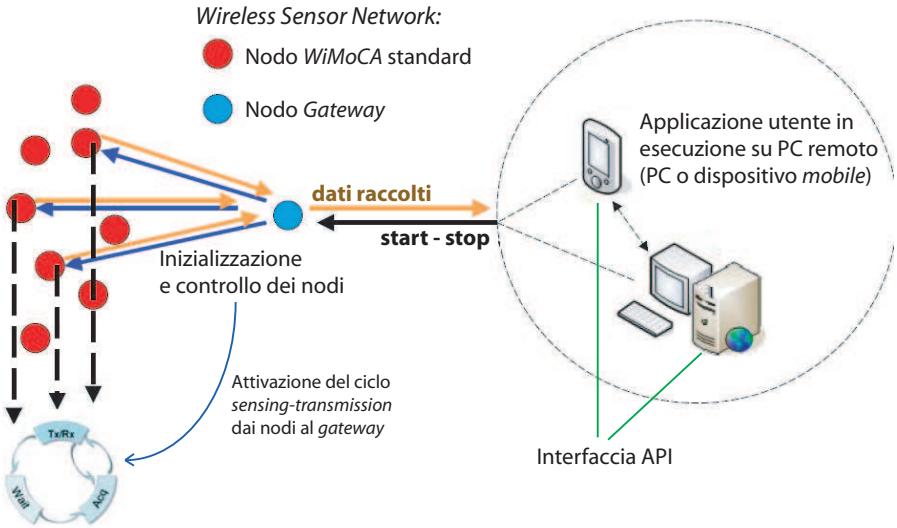


Figura 3.1: Il controllo di una sessione di comunicazione

Per motivi di efficienza nel consumo energetico sia i nodi sensore che il *gateway* appena vengono accesi si trovano in stato di *stand-by*. Il processo di rilevamento e trasmissione dei dati viene avviato e arrestato dall'applicazione utente in modo che il sistema consumi le risorse energetiche solo quando è necessario. Il controllo dall'applicazione utente avviene tramite appositi byte di controllo che vengono trasmessi al *gateway*. In particolare quando il *gateway* riceve dall'applicazione un apposito carattere *start* (il carattere ASCII "*s*") questo esce dallo *stand-by*, attiva il ciclo *sensing-transmission* nei nodi sensore e avvia la comunicazione con essi e con l'applicazione utente. Allo stesso modo quando il *gateway* riceve un'altro apposito carattere di *stop* (il carattere ASCII "*e*") dall'applicazione utente questo riporta i nodi-sensore nella condizione di *stand-by* iniziale per poi tornare esso stesso in *stand-by* cessando la trasmissione di dati verso l'applicazione utente. Questo processo è schematizzato in figura 3.1.

3.1.3 Lettura dei dati provenienti dalla periferica

Una volta attivata la trasmissione dei dati verso l'applicazione utente è possibile procedere alla loro lettura. I dati sono organizzati in pacchetti che una volta smistati e decodificati forniscono l'informazione originariamente registrata dai vari nodi-sensore.

3.1.4 Decodifica dei pacchetti di dati

La decodifica delle informazioni trasmesse dal *gateway* al sistema *client* richiede innanzitutto una fase preliminare di ricostruzione del pacchetto valido seguita dalla vera e propria conversione dei valori ricostruiti verso il formato numerico in virgola mobile che viene poi calibrato in *range* predefiniti.

Formato dei dati in ingresso dell'applicazione utente

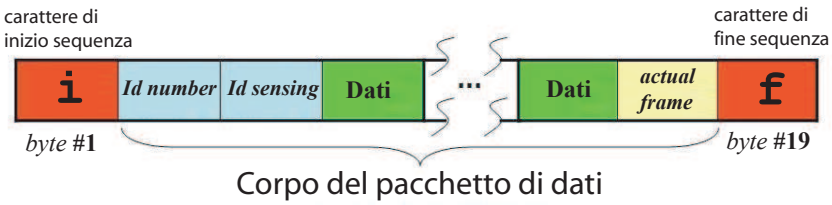


Figura 3.2: Formato dei pacchetti di dati

Byte #	4	5	6	7	8	9	10	11	12	13	14	15	16	17
valore	a_X		a_Y		a_Z		gyr_x	gyr_y	gyr_z	pollice	indice	medio	anulare	mignolo
note	B_h	B_l	B_h	B_l	B_h	B_l	N.D.	N.D.	N.D.	/	/	/	/	/
Tipo di dato	16-bit *signed*		16-bit *signed*		16-bit *signed*		8-bit *signed*	8-bit *signed*	8-bit *signed*	8-bit *signed*	8-bit *signed*	8-bit *signed*	8-bit *signed*	8-bit *signed*

Tabella 3.1: Disposizione dei dati rilevati dal sensore all'interno del corpo del pacchetto. Con B_l e B_h si indicano rispettivamente il byte meno significativo e quello più significativo (sistema *Big endian*). I valori gyr_x, gyr_y e gyr_z sono destinati a contenere i dati provenienti da un giroscopio che attualmente deve ancora essere aggiunto.

I pacchetti di dati (descritti in figura 3.2 e nella tabella 3.1) sono costituiti da sequenze di 19 byte il cui inizio e fine sono rispettivamente delimitati da un carattere di inizio sequenza (il carattere ASCII "*i*") e da un carattere di fine sequenza (il

carattere ASCII *"f"*).

Nel pacchetto tre *byte* contengono rispettivamente:

- un identificativo del nodo trasmittente detto *Id number*
- un valore detto *Id sensing* che indica il tipo di sensori presenti sul tale nodo (v. tabella 3.2)
- il numero del pacchetto di dati (*actual frame*).

sensori presenti / ID Sensing	**accelerometri**	***bend-sensors***	**giroscopi**	**frequenza cardiaca**	***galvanic skin response* (GSR)**
50	•				
4	•		•		
6		•			
7	•	•	•		
10				•	•
12	•	•	•	•	•

Tabella 3.2: Valori di *Id sensing* e corrispondenti *device*

Filtraggio logico dei pacchetti

Il flusso di pacchetti di dati restituito dal *gateway* è soggetto alla presenza di errori, per questo prima di poter procedere alla decodifica dei valori è necessario operare una discriminazione sulla validità della stringa di byte letti. Per questo nell'applicazione utente sviluppata viene eseguito un ciclo di allineamento che attende l'arrivo del byte contenente il carattere di *start*. Vengono poi sequenzialmente memorizzati gli altri valori fino a raggiungere la lunghezza complessiva del pacchetto pari a byte. A questo punto si controlla la coerenza del valore dei byte di controllo. Nel caso venga riscontrata una incongruenza il pacchetto viene scartato e la lettura del flusso di byte ricomincia dal ciclo iniziale di allineamento. Negli algoritmi implementati non viene fatto uso del valore *actual frame*.

Conversione di formato

I valori presenti nei pacchetti di dati sono caratterizzati per gli accelerometri da una codifica a valori interi a 16 bit in complemento a 2 aventi un range di 12 bit, mentre per ogni *bend-sensor* viene usato un valore ad 8-bit con segno. Per la converzione dei valori di accelerazione viene utilizzato il seguente algoritmo:

```
float convert_acceleration(BYTE high, BYTE low)
{
union {
    struct {BYTE high, low;} data;
    short val;
} a;

a.data.high = high;
a.data.low  = low;

// Accelerometro: ADC a 12 bit
// valore di accelerazione restituito
// come valore con segno a 16 bit in complemento a 2
//————————————————————————————————————————————
// Range valori:
// complemento a 2 a 16 bit:
// . max valore positivo = 2^15 – 1
// . min valore negativo = – 2^15
float max_mag =  (1 << 15) - 1.0;
float min_mag =  (1 << 15);

// Valore scalato per il rispettivo range in base al segno
// e moltiplicato per 2.0 (range accelerometri: +/– 2g)
if(a.val > 0)
 return  2.0 * (float)a.val / max_mag;
else
 return  2.0 * (float)a.val / min_mag;
}
```

Per i *bend-sensor* è invece sufficiente convertire il valore in un intero con segno per la successiva calibrazione del valore.

Calibrazione dei valori

Le specifiche dell'accelerometro fanno sì che i valori restituiti siano già automaticamente calibrati per occupare il range massimo di 12 bit, per questo gli unici valori che necessitano di calibrazione sono quelli provenienti dai *bend-sensor*. Questa calibrazione viene sempre richiesta poiché come già accennato in precedenza (cap. 2 par. 2.2.2) i *bend-sensor* offrono una risposta dinamica che varia nel medio e lungo periodo a causa di un processo di usura meccanica.

Per il processo di calibrazione è sufficiente che l'utente compia all'inizio dell'interazione il semplice gesto di stringere il pugno. In questa breve fase vengono calcolati gli estremi del *range* massimo dei valori per ogni dito al fine di operare una

normalizzazione dei valori nell'intervallo $[0.0; 1.0]$ che verranno utilizzati durante l'interazione.

3.2 Filtraggio del segnale di accelerazione

Avendo a che fare con un sistema di rilevamento delle accelerazioni bisogna considerare la natura del segnale che deve essere processato. Nel caso di un accelerometro il segnale è caratterizzato da

- ***rumore bianco*** generato dal sensore
- ***trembling*** dovuto alle piccole oscillazioni proprie del braccio e della mano

Si è scelto di utilizzare un filtro passa-basso dinamico in grado di modificare la propria frequenza di taglio in funzione della rapidità di cambiamento nei movimenti dell'utente. L'idea alla base di questa scelta è che quando l'utente non si muove molto stia puntando oppure sia immobile e quindi è necessario ridurre al massimo gli effetti di oscillazione prodotti dal *trembling* poiché anche se al costo di una maggior latenza nell'interazione con il mondo virtuale questo consente una interazione più precisa in aree di selezione ristrette rendendo "fluido" il movimento. Al contrario quando l'utente si muove in maniera più accentuata, viene innalzata la frequenza di taglio riducendo in questo modo la latenza e rendendo il sistema più sensibile ai rapidi cambiamenti nei valori di accelerazione. I valori utilizzati per il filtro fanno sì che la componente di rumore bianco venga comunque sempre attenuata abbastanza da non interferire con le successive fasi di elaborazione del segnale.

3.2.1 Media viaggiante esponenziale (*EWMA*)

La media viaggiante esponenziale (*Exponential Weighted Moving Average* o EWMA [67]) approssima una media viaggiante moltiplicando il valore in ingresso per un fattore α e sommandolo al precedente valore filtrato calcolato moltiplicato per un fattore $1-\alpha$ secondo la seguente formula:

$$y_i = \alpha x_i + (1-\alpha) y_{i-1} \tag{3.1}$$

Dove x_i e y_i sono rispettivamente il valore in ingresso del filtro e il risultante valore filtrato nell'istante i-esimo. In particolare la letteratura indica che la *EWMA*

approssima molto bene una media viaggiante (*Moving Average, MA*) usando come valore per α il valore α_{MA} qui definito:

$$\alpha_{MA} = \frac{2}{n+1} \tag{3.2}$$

dove n è il numero di campioni che si sarebbero usati nella media viaggiante[3]. Si noti che ogni valore passato x nella serie temporale è in parte contenuto in ogni nuovo risultato y. Tuttavia i valori più vecchi di x ricevono un peso che decresce esponenzialmente fino ad assumere un valore trascurabile con il progredire della serie temporale. In termini di DSP la EWMA rappresenta un filtro IIR[4] passa-basso del primo ordine. Il principale vantaggio rispetto ad una semplice *MA* è la capacità di ottenere gli stessi livelli di *smoothing* del segnale con un minor ritardo temporale, questo perché viene dato maggior peso ai valori recenti rispetto a quelli più vecchi. In questo modo diminuisce la latenza fra il rilevamento dei valori del *tracker* ed il *feedback* visivo su schermo.

3.2.2 Un possibile miglioramento: tecniche di predizione della posa

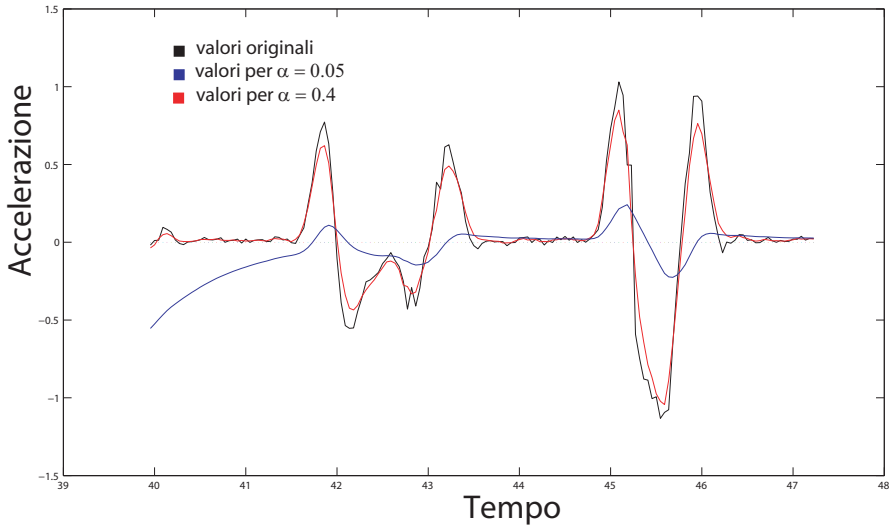
Un buon dispositivo di *motion tracking* dovrebbe essere in grado non solo di seguire accuratamente la posa dell'utente nel corrente istante di tempo ma anche di *predire* il movimento abbastanza in anticipo da compensare il *ritardo* che intercorre fra il movimento dell'utente ed il *feedback visuale*. Questo ritardo è causato dai seguenti fattori:

- latenza del *tracker*
- ritardi nei canali comunicativi hardware
- tempi di *rendering* e di visualizzazione su *display*

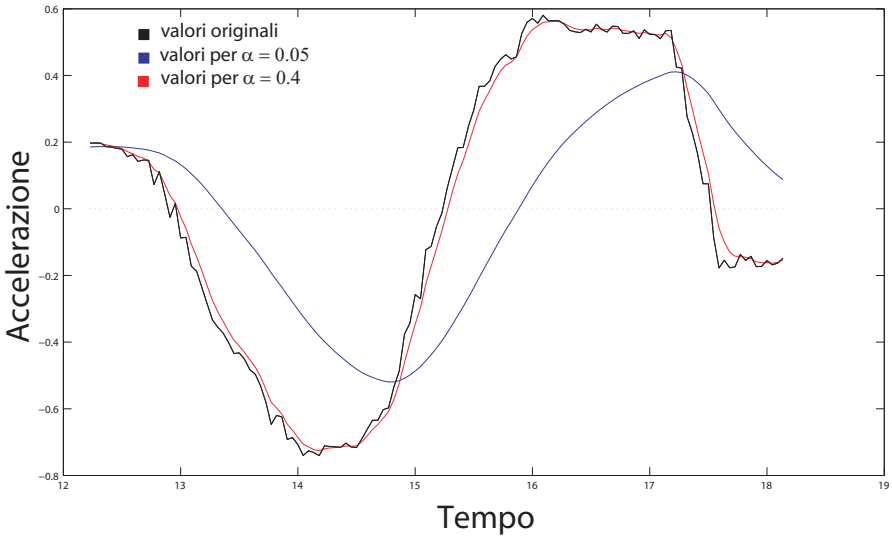
Questo ritardo tipicamente spazia da un minimo di 25 millisecondi a un massimo di 150 o più millisecondi. Ovviamente più aumenta l'anticipo nella predizione, più questa tenderà ad essere inesatta, per questo la predizione non può fare nulla nel

[3] questa formula verrà utilizzata più avanti nella definizione del procedimento di calcolo del fattore di *reattività* nella (3.5).

[4] *Infinite Impulse Response*: con risposta all'impulso infinita



(a) movimento rapido



(b) movimento lento

Figura 3.3: Il comportamento della EWMA sui valori di accelerazione rilevati durante un movimento "su-giù" rapido (a) e lento (b) per i valori α_{min} e α_{max}

caso di un sistema di generazione di ambienti virtuali lento. Tuttavia se la latenza totale è prossima ai 60–80 millisecondi le tecniche di predizione possono portare ad un miglioramento drastico della qualità dell'interazione. Queste tecniche non sono state implementate, tuttavia se ne citeranno brevemente alcune che sono parse interessanti per una possibile implementazione.

Filtraggio di Kalman

Nel 1960 Rudolph E. Kalman pubblicò un articolo descrivente una soluzione ricorsiva al problema del filtraggio di dati discreti. Da allora grazie soprattutto ai progressi in campo computazionale il filtro di Kalman è stato il soggetto di estensive ricerche ed applicazioni, in particolare nel campo della navigazione autonoma o assistita [65, 66]. Per una trattazione approfondita della tecnica originale si rimanda a [62], riguardo invece alle importanti evoluzioni di questa tecnica si rimanda a [63, 64].

Exponential Smoothing Prediction

L'*exponential smoothing prediction*(levigazione/smorzamento esponenziale) (ESP) è un modello statistico-matematico di predizione basato sull'assunto che i futuri valori siano una media pesata dei valori passati: è una serie matematica nella quale viene dato maggior peso ai valori recenti. Esiste una varietà di questi modelli i quali hanno in comune le seguenti proprietà:

- le previsioni sono medie pesate delle passate osservazioni con pesi che decrescono esponenzialmente con l'invecchiare del dato osservato (da qui il nome *exponential smoothing*)
- sono basati su equazioni di aggiornamento, ognuna delle quali possiede una costante che varia fra 0 e 1
- alle osservazioni recenti viene dato relativamente più peso che a quelle vecchie

Il nome di questa tecnica è legato al fatto che i pesi decrescono esponenzialmente con l'invecchiare del dato osservato.

La ESP lungo gli anni si è dimostrata una tecnica molto utile in varie situazioni di predizione. Il metodo fu per la prima volta suggerito da C.C. Holt nel 1957 [67].

Oggigiorno è uno schema molto diffuso nella produzione di serie temporali *smoothed* (levigate) e in alcune sue varianti [68] rappresenta un'invitante alternativa [69] al filtraggio di Kalman.

3.3 Premessa sui sistemi di riferimento utilizzati

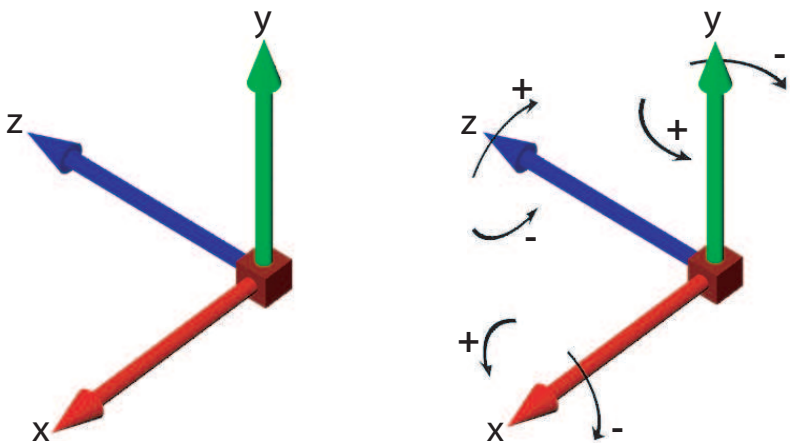
Prima di procedere alla descrizione degli algoritmi adottati è bene chiarire quali sono i sistemi di coordinate a cui si farà riferimento. Principalmente i sistemi necessari saranno due:

- il sistema di riferimento solidale al sensore
- il sistema di riferimento del mondo virtuale

Questi sistemi di riferimento verranno ora dettagliati.

Sistema di riferimento del mondo virtuale (*WCS*)

Il sistema di riferimento che si è deciso di utilizzare è un sistema di riferimento cartesiano destrorso (fig. 3.4). D'ora in poi questo particolare sistema di riferimento verrà chiamato con l'abbreviazione *WCS* (*World Coordinate System*).



(a) Disposizione degli assi

(b) Sensi di rotazione attorno agli assi

Figura 3.4: Il sistema di riferimento utilizzato

Sistema di riferimento del sensore

Gli assi di rilevamento dell'accelerometro hanno una disposizione differente rispetto al sistema di riferimento *WCS* (fig. 3.5). Per questo motivo le grandezze rilevate sono state trasposte in maniera tale da essere conformi ad esso. La trasposizione dei valori di accelerazione avviene da subito nella fase di decodifica dei pacchetti nella seguente maniera:

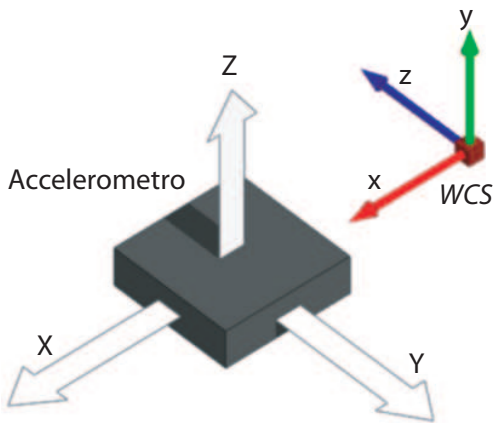


Figura 3.5: Disposizione degli assi di rilevamento dell'accelerometro triassiale rispetto al *WCS*

$$\begin{cases} a_x^s = & a_X \\ a_y^s = & a_Z \\ a_z^s = & -a_Y \end{cases}$$

$$\mathbf{a}^s = \begin{bmatrix} a_x^s & a_y^s & a_z^s \end{bmatrix}$$

a_X, a_Y e a_Z sono i tre valori di accelerazione come provenienti dal sistema di riferimento originario del sensore. L'apice s che appare nelle variabili indica che si tratta del valore della forza di accelerazione espressa all'interno del nuovo sistema di riferimento stabilito per il sensore. D'ora in poi per "sistema di riferimento del sensore" si intenderà quello appena sopra descritto.

3.4 Uso dell'informazione di accelerazione

L'accelerazione di una massa richiede una forza proporzionale all'accelerazione. Inoltre, in presenza della gravità, sulla massa agisce una forza gravitazionale aggiuntiva. In un accelerometro viene rilevata la forza risultante che agisce su di una massa di prova contenuta nel sensore. Per questo motivo in termini di accelerazione nel sistema di riferimento del sensore la forza risultante $\mathbf{a}_s$ è pari a:

$$\mathbf{a}^s = \mathbf{a}^s_{dyn} + \mathbf{g}^s \tag{3.3}$$

dove $\mathbf{a}^s_{dyn}$ e $\mathbf{g}^s$ sono rispettivamente l'accelerazione dinamica e la forza di gravità espresse nel sistema di riferimento del sensore.

La componente gravitazionale del segnale può essere usata per determinare l'inclinazione del sensore. Tuttavia poiché il segnale è composto sia dall'accelerazione dinamica che da quella gravitazionale la stima dell'inclinazione è accurata solo nei periodi in cui la magnitudine dell'accelerazione dinamica è piccola rispetto a g e il segnale ha una magnitudine prossima a $1\,g$ [61].

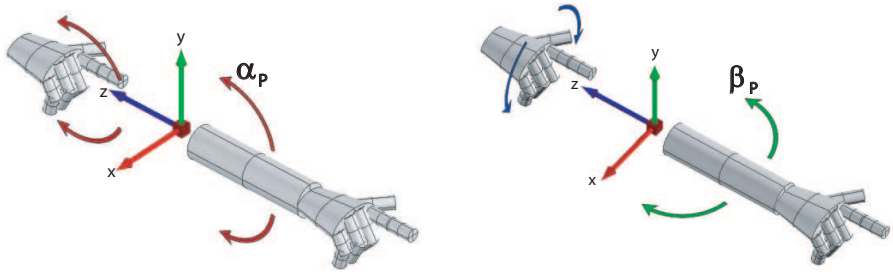
3.4.1 Rilevamento gravimetrico dell'inclinazione

Negli algoritmi che sono qui stati implementati le informazioni provenienti dagli accelerometri sono state utilizzate per risalire ai dati sia di inclinazione che di accelerazione dinamica. Un accelerometro sottoposto al campo gravitazionale fornisce una lettura molto accurata della componente gravitazionale lungo il proprio asse di rilevamento. Persino i più economici accelerometri MEMS possono misurare inclinazioni con la precisione di circa 1°. Per calcolare l'inclinazione a partire dai dati forniti dall'accelerometro triassiale sono state utilizzate le seguenti funzioni che identificano gli angoli φ_{YZ} e φ_{XZ} formati dalla accelerazione $\mathbf{a}_s$ con il piano verticale YZ (fig. 3.6(a)) e con il piano orizzontale XZ (fig. 3.6(b)) nel sistema di

riferimento del sensore:

$$\varphi_{YZ} = atan2\left(-a_x^s,\ a_z^s\right) \tag{3.4a}$$

$$\varphi_{XZ} = atan2\left(-a_y^s,\ a_z^s\right) \tag{3.4b}$$



(a) L'angolo φ_{YZ} (b) L'angolo φ_{XZ}

La funzione *atan2*[5] restituisce la misura dell'angolo nel piano cartesiano fra un vettore bidimensionale (x,y) a partire dall'asse delle ascisse in senso antiorario; questo angolo è espresso in radianti e varia nell'intervallo $[-\pi;\ \pi]$

Limitazioni nella misurazione dell'inclinazione

Come già accennato esiste una limitazione alla capacità di un'accelerometro di misurare le inclinazioni poiché qualsiasi inclinometro deve poter rilevare la direzione della componente gravitazionale del vettore di accelerazione risultante perchè si possa determinare l'inclinazione rispetto al campo gravitazionale (fig. 3.6) [70]. Questa è una limitazione fondamentale imposta dalla fisica legata al rilevamento gravimetrico dell'inclinazione che può essere solo parzialmente arginata da un sensore con un *design* migliore. Sensori con un accurata regolazione delle inerzie di smorzamento, oscillazione e rotazione della massa di prova possono diminuire la sensibilità alle traslazioni orizzontali, tuttavia movimenti caratterizzati da una bassa frequenza si manifesteranno sempre come se il dispositivo fosse sottoposto a cambiamenti di inclinazione.

[5] La funzione *atan2* è definita in diversi linguaggi di programmazione fra i quali anche il C e ha un comportamento simile alla funzione $\arctan\left(\frac{y}{x}\right)$, tranne che per il fatto che i segni di entrambi gli argomenti vengono utilizzati per stabilire il corretto quadrante del risultato. Inoltre il valore restituito è corretto anche nel caso in cui x sia nullo, ovvero per gli angoli $\pm\frac{\pi}{2}$.

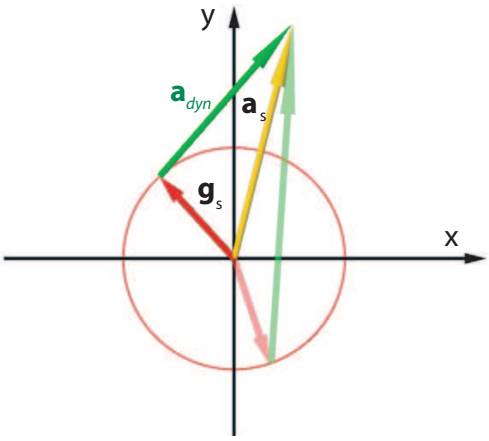


Figura 3.6: Il vettore $\mathbf{a}^s$ è la risultante della somma dei vettori $\mathbf{a}^s_{dyn}$ e $\mathbf{g}^s$, tuttavia il vettore $\mathbf{g}^s$ potrebbe giacere ovunque sulla sfera centrata nell'origine e avente raggio pari al modulo di $\mathbf{g}$ (in questa figura per chiarezza visiva si è considerata la situazione solo sul piano XY del sistema di riferimento dell'accelerometro)

Controllo dinamico del filtraggio dei valori di accelerazione

Nella *EWMA* si può controllare agevolmente la velocità a cui i vecchi valori vengono smorzati influendo direttamente sul parametro α: quando questo valore è prossimo a 1 lo smorzamento è rapido mentre quando α, è prossimo a 0 lo smorzamento avviene lentamente (fig. 3.3).
Il parametro α della *EWMA* è controllato da una approssimazione γ della *varianza* che viene usata per calcolare un valore che verrà chiamato *reattività* (indicato con il simbolo ρ). Il calcolo di ρ è basto su una serie di *EWMA* dei valori di magnitudine $m(n)$ dell'accelerazione dinamica non filtrata $\mathbf{a}^s_{dyn}$ considerati su di una finestra temporale equivalente a circa 20 *samples*. Questo valore è in grado di fornire un'indice di quanto il sensore si stia muovendo (fig. 3.7) ed è calcolato secondo il seguente procedimento:

Innanzitutto calcoliamo secondo la (3.2) il valore di *smoothing* che utilizzeremo nelle due successive *EWMA* in modo da ottenere un comportamento analogo ad una *MA* avente una finestra di 20 campioni:

$$\alpha_{MA} = \frac{2}{20+1}$$

viene quindi prelevato il valore del modulo del vettore di accelerazione complessiva $\mathbf{a}^s$:

$$m_n = |\mathbf{a}^s|$$

a questo punto nel valore m_n^{avg} calcoliamo una media dei valori di magnitudine dell'accelerazione:

$$m_n^{avg} = \alpha_{MA}\, m_n + (1 - \alpha_{MA})\, m_{n-1}$$

possiamo ora ricavare il quadrato del discostamento dal valore medio precedentemente calcolato

$$\Delta_n^m = (m_n - m_n^{avg})^2$$

procediamo quindi al calcolo finale della media dei valori di discostamento per ricavare la stima della varianza:

$$\gamma = \alpha_{MA}\, \Delta_n^m + (1 - \alpha_{MA})\, \Delta_{n-1}^m$$

infine per calcolare il valore di reattività ricorriamo ad una apposita scalatura del valore stimato di varianza secondo la seguente funzione razionale:

$$\rho = \frac{\gamma}{\gamma + 1}$$

Il valore così calcolato di ρ viene utilizzato per controllare il parametro α della EWMA che filtra i valori di accelerazione provenienti dal sensore secondo la seguente funzione:

$$\alpha = (1 - \rho)\, \alpha_{min} + \rho\, \alpha_{max} \qquad \text{con} \qquad \alpha_{min} = 0.05, \quad \alpha_{max} = 0.4 \tag{3.6}$$

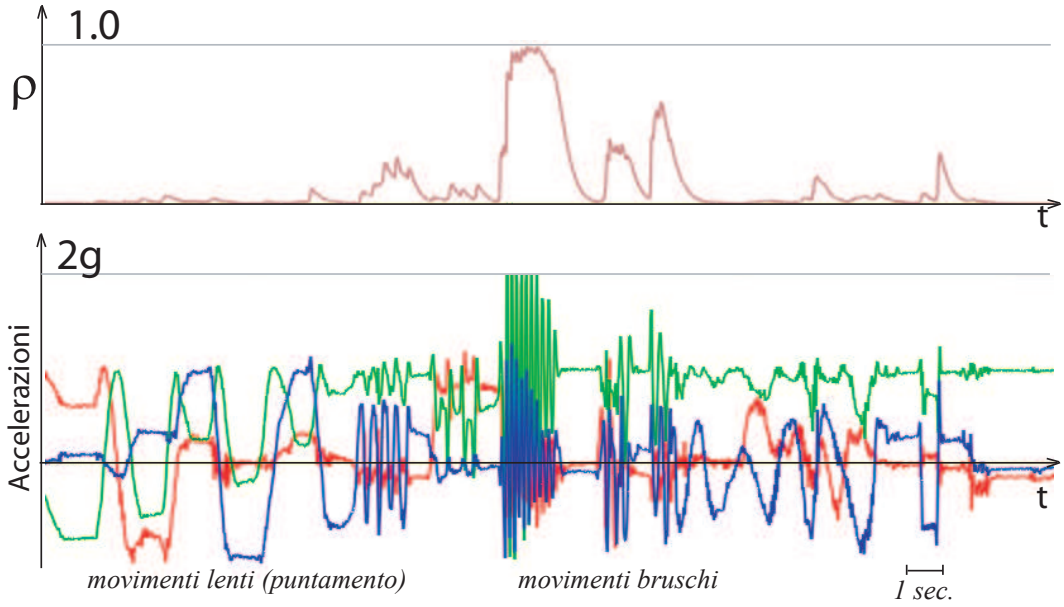


Figura 3.7: l'andamento del valore di reattività ρ durante un periodo di interazione caratterizzato da movimenti più o meno bruschi della mano

Gli estremi dei valori α_{min} e α_{max} usati per calcolare α sono stati determinati sperimentalmente in modo da ottenere una sufficiente attenuazione dei movimenti durante le fasi di puntamento e una reattività più elevata durante l'esecuzione movimenti pronunciati come quelli usati per il calcolo dello spostamento (trattato al paragrafo 3.4.2).

3.4.2 Calcolo della posizione nello spazio

Avendo a disposizione dei sensori inerziali una delle aspettative principali è quella di poter calcolare con appositi algoritmi la posizione nello spazio dell'oggetto tracciato. Tuttavia la cosa è più complessa di quanto si potrebbe credere in un primo momento.

Nel caso in esame disponendo di un accelerometro triassiale si potrebbe essere portati a pensare che sia sufficiente integrare due volte il dato dell'accelerazione (fig. 3.8) in modo da poter ricavare il dato della posizione. In realtà come già detto in condizioni dinamiche l'accelerometro è incapace di distinguere quale sia la direzione della forza di gravità. Ciò è possibile solo disponendo di un ulteriore componente hardware: un *giroscopio* al fine di poter escludere dal vettore di accelerazione rilevato dal sensore la componente (costante) dovuta alla forza di gravità

risalendo così alla pura componente di accelerazione dinamica [60]. Sarebbe anche possibile effettuare la doppia integrazione ponendo come vincolo una orientazione fissa del sensore, tuttavia questo oltre che a rendere il dispositivo di *tracking* meno *user-friendly* risulta essere di difficile attuazione a causa della attualmente limitata frequenza di *sampling* del dato di accelerazione ($20\,Hz$). Un approccio del genere ha invece avuto risultati accettabili ricorrendo a sensori con una frequenza di *sampling* di $50\,Hz$ [59].

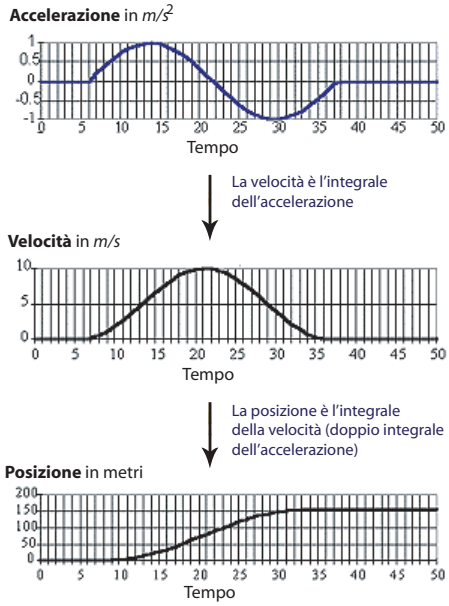


Figura 3.8: Misuramento inerziale e doppia integrazione

In alcuni test preliminari è stato anche considerato l'uso del valore di flessione delle dita combinato con il dato di orientazione per la gestione dell'ampiezza degli spostamenti dell'*avatar* nel mondo virtuale, tuttavia si è rivelata una via non particolarmente intuitiva e quindi è stata da subito scartata.

3.4.3 Gestione alternativa dello spostamento

Alla luce dell'impossibilità di eseguire un *tracking* efficace senza ricorrere all'uso di un giroscopio è stato scelto un metodo alternativo per gestire le traslazioni a partire da una stima dei valori di accelerazione dinamica ricavati tramite un filtraggio geometrico della componente gravitazionale. In seguito un rivelatore di inviluppo viene applicato ai valori stimati di accelerazione dinamica di ogni asse. In questo

modo si ottiene un valore che dipende sia dalla direzione del movimento che dalla sua intensità e che può essere utilizzato come dato di velocità per l'interazione di spostamento.

Filtraggio geometrico della componente gravitazionale

Per rimuovere l'apporto della forza di gravità dall'accelerazione totale $\mathbf{a}^s$ si è utilizzato un metodo empirico che consiste nel sottrarre a tale vettore un'altro vettore che abbia direzione opposta e modulo pari a g che nel sistema di riferimento del sensore ha direzione ignota ma possiede comunque modulo costante. In questo modo viene filtrata la componente continua del segnale a discapito però di un cambiamento nelle proporzioni dei valori di accelerazione ricavati che risultano essere influenzati dall'orientazione rispetto alla direzione di $\mathbf{g}$. Per ovviare a questo inconveniente si è ricorsi ad una scalatura dei valori proporzionale alle componenti di $\mathbf{g}^s$ ottenendo così le seguenti funzioni per il ricavo dell'accelerazione dinamica stimata $\mathbf{a}_{dyn}$:

$$\begin{cases} \tilde{\mathbf{a}} = \mathbf{a}^s - \dfrac{\mathbf{a}^s}{|\mathbf{a}^s|}|\mathbf{g}| \\ \mathbf{a}_{dyn} = \begin{bmatrix} \tilde{a}_x\left((1-|g_x^s|)k_{dyn}+1\right) \\ \tilde{a}_y\left((1-|g_y^s|)k_{dyn}+1\right) \\ \tilde{a}_z\left((1-|g_z^s|)k_{dyn}+1\right) \end{bmatrix} \end{cases}$$

$k_{dyn} = 9$ è una costante il cui valore è stato fissato a 9 dopo varie prove sperimentali. Il risultato di questa approssimazione dell'accelerazione dinamica può essere osservato in figura 3.9.

Rilevatore di inviluppo

Il rilevatore di inviluppo implementato opera sul segnale in ingresso seguendone rapidamente la magnitudine quando questa cresce e seguendola invece più lentamente quando essa decresce. Il rilevatore di inviluppo opera sul valore assoluto del segnale, quindi viene anche memorizzato il segno del primo valore al di sopra di una data soglia S_{env} al fine di tenere in conto la direzione del moto.

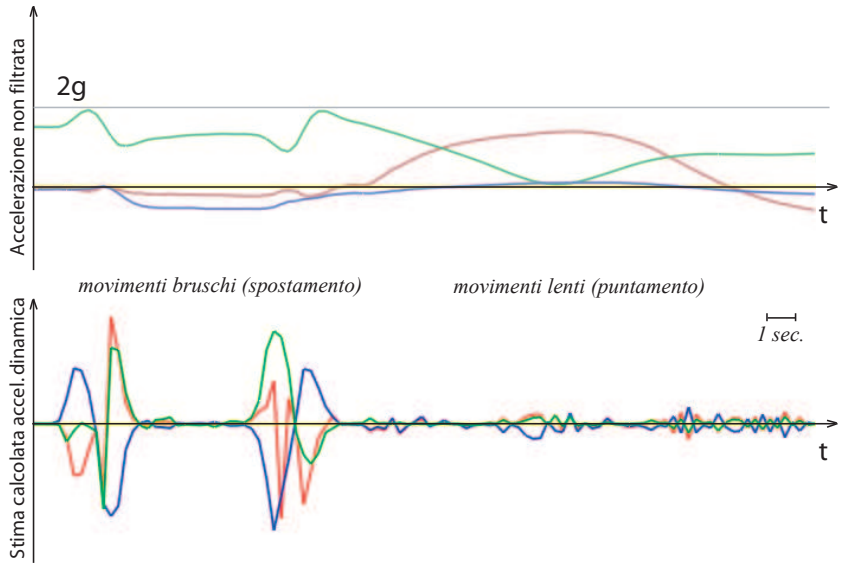


Figura 3.9: Comportamento del filtraggio geometrico adottato sul valore di accelerazione complessiva $\mathbf{a}^s$

Le fasi di funzionamento dell'algoritmo utilizzato sono le seguenti:

1) Attesa del "silenzio"

Attendi un numero N_{env} di campioni aventi un valore al di sotto di una soglia S_{env}

2) Inizio dell' "ascolto"

Entra in ascolto dei valori di accelerazione che superano la soglia S_{env}

3) Rilevamento della direzione

Se un valore ha superato la soglia memorizzane il segno in k_{env},altrimenti torna a **2)**

4) Rilevamento del valore di inviluppo

Rileva l'inviluppo e restituisci il valore A_{env} così calcolato moltiplicato per k_{env}

Il dato ricavato dal rivelatore di inviluppo viene utilizzato come valore di magnitudine per la velocità dello spostamento. Dopo aver esplorato diverse possibilità di interazione si è deciso di separare l'interazione di spostamento nel mondo virtuale in due fasi distinte:

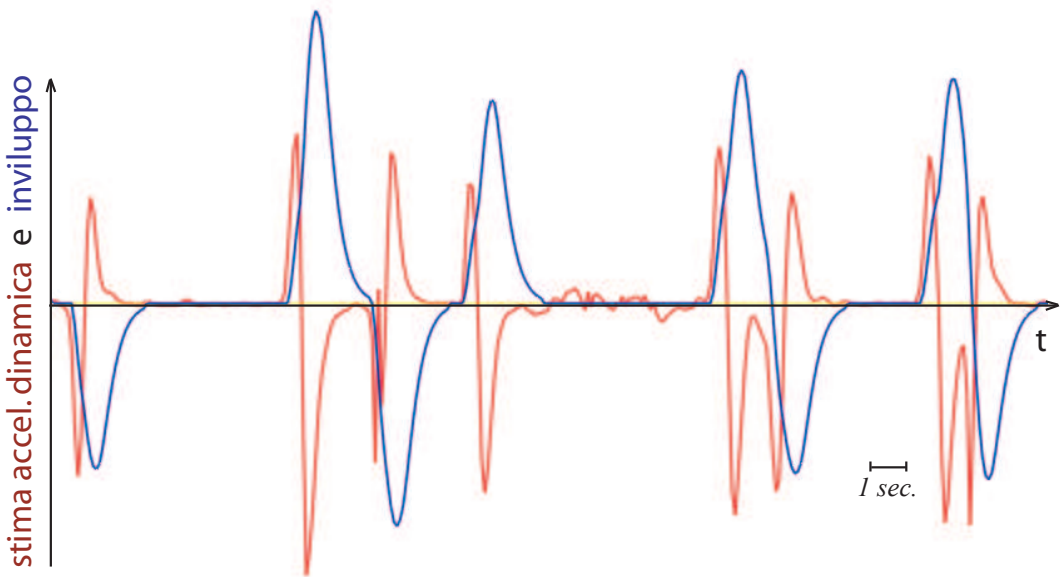


Figura 3.10: Comportamento del rilevatore di inviluppo sui valori stimati di accelerazione dinamica

A) spostamento lungo il piano XY del mondo virtuale

B) spostamento lungo l'asse Z del mondo virtuale

Nel primo caso per decidere il verso della velocità si parte dalla proiezione del vettore di accelerazione gravitazionale sul piano XY del sistema di riferimento dell'accelerometro. Nel secondo caso invece alla velocità viene assegnato il verso dell'asse z del *WCS*.

3.4.4 *Threading* e interpolazione lineare per un'animazione fluida

Come già detto la lettura dei dati dal dispositivo avviene ad una frequenza di approssimativamente 20 Hz. D'altra parte è vero che il *rendering* su schermo potrebbe avvenire ad una frequenza di 60 o più fotogrammi per secondo migliorando la percezione di continuità dell'animazione da parte dell'utente [10]. É quindi auspicabile l'implementazione di un disaccoppiamento fra la lettura dei dati dal dispositivo ed il *rendering* su schermo al fine di migliorare l'esperienza di interazione. Per ottenere questo risultato è stato necessario affidare ad un apposito *thread* la lettura dei

dati dalla porta seriale. Per stabilire la comunicazione fra il *thread* e la parte di programma delegata ai calcoli ed al *rendering* si è ricorso all'uso di un *buffer FIFO*[6] software. Il funzionamento è il seguente: il *thread* delegato legge i pacchetti di dati dalla porta seriale e li memorizza nell'elemento del buffer che si trova nella prima posizione libera assieme ad una precisa informazione riguardo l'istante di arrivo del dato. Appena il nuovo dato è stato memorizzato questo può essere prelevato dalle funzioni di elaborazione del segnale, processato e reso disponibile per il ciclo di *rendering*. A questo punto la chiave per ottenere un'animazione fluida è quella di effettuare una interpolazione dei vari valori dai quali vengono ricavati i parametri di calcolo legati alla posa dell'*avatar* e alle geometrie di interazione. Più precisamente i valori filtrati dell'accelerazione e dei *bend-sensor* degli ultimi due *frame* di dati disponibili nel buffer FIFO vengono utilizzati come estremi per un'*interpolazione lineare* (fig. 3.11) che avviene proprio nel ciclo di *rendering*, quindi alla massima frequenza possibile.

Definiamo $\mathbf{V}_{lerp}(t)$ come il il vettore che conterrà al tempo t_a dell'animazione l'interpolazione lineare dei vettori dei valori degli ultimi due frame immagazzinati $\mathbf{V}_{F_{n-1}}$ e $\mathbf{V}_{F_n}$. Tale *frame* di valori interpolati viene calcolato a partire dal vettore dei valori $\mathbf{V}_{F_n}$ e dal relativo dato temporale di arrivo $t(F_n)$ presente in ogni *frame* F_n secondo il seguente procedimento:

per prima cosa calcoliamo il tempo trascorso fra l'arrivo degli ultimi due pacchetti di dati:

$$\Delta_{FIFO} = t(F_n) - t(F_{n-1})$$

ora calcoliamo all'interno del ciclo di *rendering* calcoliamo il fattore k_t proporzionale al tempo t trascorso dall'arrivo del penultimo pacchetto di dati:

$$k_{t_a} = \frac{t_a - t(F_{n-1})}{\Delta_{FIFO}}$$
$$\mathbf{V}_{lerp}(t) = (1 - k_{t_a})\,\mathbf{V}_{F_{n-1}} + k_{t_a}\,\mathbf{V}_{F_n}$$

[6]acronimo di *First In, First Out*, ovvero i dati vengono accodati in ordine di arrivo e processati appena possibile con lo stesso ordine.

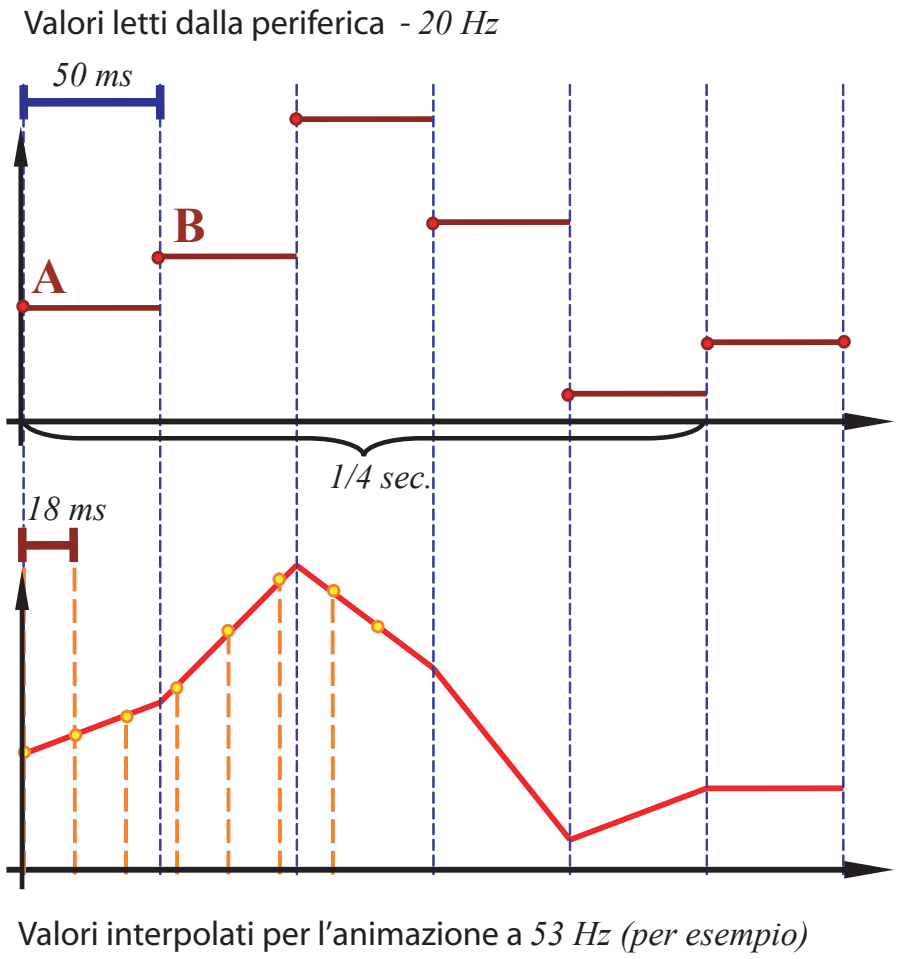
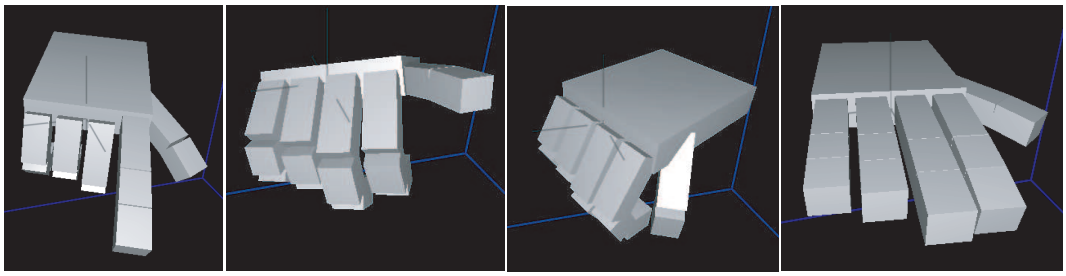


Figura 3.11: Interpolazione lineare dei valori

I valori vengono interpolati unicamente ai fini del calcolo dell'animazione dell'*avatar* e delle interazioni di puntamento e di movimento delle dita mentre il filtraggio ed i calcoli numerici dei valori provenienti dal dispositivo avvengono a monte di questa operazione, quindi solo quando un nuovo pacchetto di dati viene letto dal buffer FIFO.

3.5 Paradigmi di interazione adottati

I paradigmi di interazione scelti fanno uso sia dei dati dell'accelerometro elaborati che, come già specificato, dei dati provenienti dai *bend-sensor*. Per attivare i vari stati dell'interazione come puntamento e selezione si è deciso di ricorrere a particolari configurazioni di posizioni delle dita.



(a) puntamento (b) selezione (c) *dragging* (c) *idle*

Figura 3.12: Gesture delle dita scelte per distinguere le fasi di puntamento, selezione, *dragging* e *idle*

3.5.1 Interpretazione delle posizioni delle dita

Si è scelto di utilizzare il dato di flessione di ogni dito per riconoscere alcune particolari gesti successivamente utilizzabili per passare a differenti stati di interazione (fig. 3.12).

Per stabilire se un dito deva essere considerato o no serrato si è ritenuto sufficiente discriminare i valori letti calibrati nell'intervallo $[0; 1]$ rispetto ad una soglia S_{dito_i} appartenente a questo intervallo, determinata sperimentalmente e comunque dipendente dalla diversa mobilità di ogni dito, ad esempio l'indice ha una mobilità nettamente superiore a quella del mignolo o dell'anulare. In parte il valore di soglia adottato è anche legato all'ergonomia del guanto ed alla sua fedeltà nel seguire tramite i *bend-sensor* i movimenti delle varie dita.

Dito	Numerazione dito	Soglia adottata
Pollice	0	0.35
Indice	1	0.4
Medio	2	0.4
Anulare	3	0.3
Mignolo	4	0.3

Tabella 3.3: Soglie adottate per le varie dita

Viene utilizzato un valore numerico distinto per ogni dito per codificare in maniera compatta una qualsiasi disposizione delle dita.

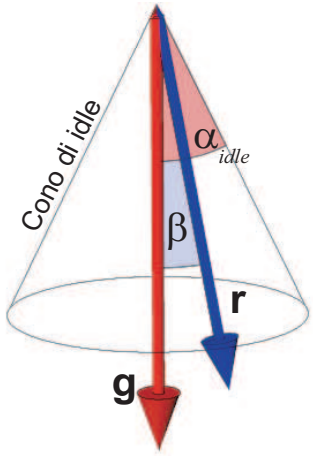


Figura 3.13: Il cono di *idle*

3.5.2 Stato di *idle*

Lo stato di *idle* (inattività) viene causato sia dalla posizione completamente aperta della mano che da una direzione di puntamento che rientri in un determinato *cono di idle* (fig. 3.13). Tale cono ha l'asse parallelo al vettore di forza di gravità **g** (asse verticale) ed ha un'ampiezza variabile definita dall'angolo α_{idle}. Per determinare se la direzione di puntamento $\mathcal{R}_d$ si trova all'interno di questo cono è sufficiente controllare se il valore dell'angolo β formato con il vettore **g** è inferiore in valore assoluto al valore di α_{idle}. Il valore scelto per α_{idle} è stato scelto per non interferire con la normale interazione di puntamento ed è pari a 35 °.

3.5.3 Puntamento

Una volta attivata la modalità di puntamento tramite l'opportuna disposizione delle dita diventa necessario risalire ad una stima della direzione di puntamento della mano dell'utente, la quale viene espressa come una retta orientata (raggio) nello spazio tridimensionale. Per rendere possibile il calcolo di tale raggio la miglior soluzione apparsa possibile è stata quella di partire dall'inclinazione rilevata dall'accelerometro. Più precisamente si è ricorso agli angoli φ_{XY} e φ_{XZ} che il vettore $\mathbf{g}^s$ di accelerazione gravitazionale forma sia con il piano XY che con il piano XZ nel sistema di riferimento dell'accelerometro. Una volta calcolati questi angoli vengono trasformati e utilizzati per ricavare una matrice di rotazione che servirà per costruire il raggio di puntamento.

Una volta che si dispone del raggio di puntamento questo può essere utilizzato per generare la selezione di elementi della scena virtuale secondo vari approcci. La tecnica generale adottata vede il raggio originarsi da una posizione (eventualmente fissa) all'interno della scena virtuale.

Per gli scopi di selezione posti dall'applicazione "*Il Muro della Memoria*" (cap. 4, par. 4.3) è stato necessario poter calcolare l'intersezione del raggio con un piano per poter in seguito risalire ad una coppia di indici interi rappresentanti le coordinate di una cella selezionabile all'interno di una *griglia di selezione* (si vedano fig. 3.14(b) e fig. 4.3.2 al capitolo 4).

In un'altro tipo di approccio sperimentato che prevede l'utilizzo delle funzionalità di *picking* (selezione) delle *API OpenGL* (cap. 4, par. 4.1) il raggio viene considerato come avente un'origine fissa al di fuori dello schermo calcolandone poi l'intersezione con il piano dello schermo, in modo da poter disporre delle coordinate bidimensionali necessarie al successivo processo di *picking*.

Il calcolo del punto di intersezione sul piano a partire dal raggio di puntamento

Gli angoli φ_{YZ} e φ_{XZ} vengono calcolati attraverso le funzioni già definite nelle equazioni (3.4a) e (3.4b). Successivamente viene applicata la seguente trasformazione per ottenere gli *angoli di puntamento* α_{P} e β_{P}

$$\alpha_{\mathrm{P}} = (\varphi_{YZ} + \phi_{YZ})\, A_{YZ} \tag{3.7a}$$

$$\beta_{\mathrm{P}} = (\varphi_{XZ} + \phi_{XZ})\, A_{XZ} \tag{3.7b}$$

ϕ_{YZ} e ϕ_{XZ} servono per poter impostare uno sfasamento nella misura delle inclinazioni in funzione di una posizione confortevole della mano mentre i valori A_{YZ} e A_{XZ} servono ad amplificare gli angoli di inclinazione per consentire all'utente di poter variare anche con movimenti contenuti la direzione del raggio di puntamento. Nei test eseguiti si sono alla fine scelti questi valori:

$$\phi_{YZ} = 5^\circ \qquad A_{YZ} = 3$$

$$\phi_{XZ} = 10^\circ \qquad A_{XZ} = 4$$

$$\mathbf{R}_{\mathrm{P}} = \begin{bmatrix} C_\beta & 0 & S_\beta \\ S_\alpha S_\beta & C_\alpha & -S_\alpha C_\beta \\ -C_\alpha S_\beta & S_\alpha & C_\alpha C_\beta \end{bmatrix} \tag{3.8}$$

dove

$$S_\alpha = \sin\alpha_{\mathrm{P}} \qquad C_\alpha = \cos\alpha_{\mathrm{P}}$$
$$S_\beta = \sin\beta_{\mathrm{P}} \qquad C_\beta = \cos\beta_{\mathrm{P}}$$

Il raggio di puntamento $\mathcal{R}$ viene definito da un'origine e da una direzione:

$$\mathcal{R}_0 = \left[x_0, y_0, z_0\right]^{\mathrm{T}}$$
$$\mathcal{R}_d = \left[x_d, y_d, z_d\right]^{\mathrm{T}}$$

un punto appartenente al raggio $\mathcal{R}$ avrà perciò coordinate del tipo

$$\mathcal{R}(t) = \mathcal{R}_0 + t\,\mathcal{R}_d \quad \text{dove} \quad t > 0 \tag{3.9}$$

la direzione $\mathcal{R}_d$ del raggio di puntamento può quindi essere calcolata con questa formula:

$$\mathcal{R}_d = \mathbf{R}_P \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} -S_\beta & S_\alpha C_\beta & -C_\alpha C_\beta \end{bmatrix}^{\mathrm{T}} \tag{3.10}$$

Il piano $\mathcal{P}$ viene definito a partire dal vettore $\mathbf{w}$ a esso normale e dall'origine $\mathbf{O}_{\mathcal{P}}$, l'equazione del piano nelle tre dimensioni diviene quindi:

$$\left(\mathbf{P}_{\mathcal{P}} - \mathbf{O}_{\mathcal{P}}\right) \cdot \mathbf{w} = 0 \tag{3.11}$$

dove $\mathbf{P}_{\mathcal{P}}$ è un generico punto appartenente al piano. A questo punto l'intersezione $\mathbf{P}$ fra il raggio $\mathcal{R}$ ed il piano $\mathcal{P}$ può essere trovata attraverso il seguente calcolo:

$$\begin{cases} t = -\dfrac{\mathbf{w} \cdot \left(\mathcal{R}_0 - \mathbf{O}_{\mathcal{P}}\right)}{\mathbf{w} \cdot \mathcal{R}_0} \\ \mathbf{P} = \mathcal{R}(t) \quad \text{per} \quad t > 0 \end{cases} \tag{3.12}$$

nel caso che il valore t sia negativo significa che l'intersezione è antecedente l'origine del raggio mentre se il denominatore della (3.12) è nullo significa che o il raggio è parallelo e distinto rispetto al piano (nessuna intersezione) oppure che il raggio appartiene alla superficie piano (infinite intersezioni).

Il calcolo delle coordinate della cella puntata

Disponendo ora del punto di intersezione **P** espresso nel sistema di coordinate *WCS* siamo in grado di risalire alle sue coordinate nel sistema di riferimento planare formato dai due vettori **u** e **v** avente origine sempre in $\mathbf{O}_{\mathcal{P}}$.

I due vettori **u** e **v** vengono calcolati a partire dal vettore **y** del *WCS* attraverso il seguente procedimento:

- In un primo passo calcoliamo il vettore delle ascisse **u** come prodotto vettoriale di **y** e **w**

$$\mathbf{u} = \mathbf{y} \times \mathbf{w}$$

- In un secondo passo calcoliamo il vettore delle ordinate **v** come prodotto vettoriale di **w** e **u**

$$\mathbf{v} = \mathbf{w} \times \mathbf{u}$$

- In un terzo e ultimo passo procediamo alla normalizzazione dei vettori **u** e **v** così ottenuti.

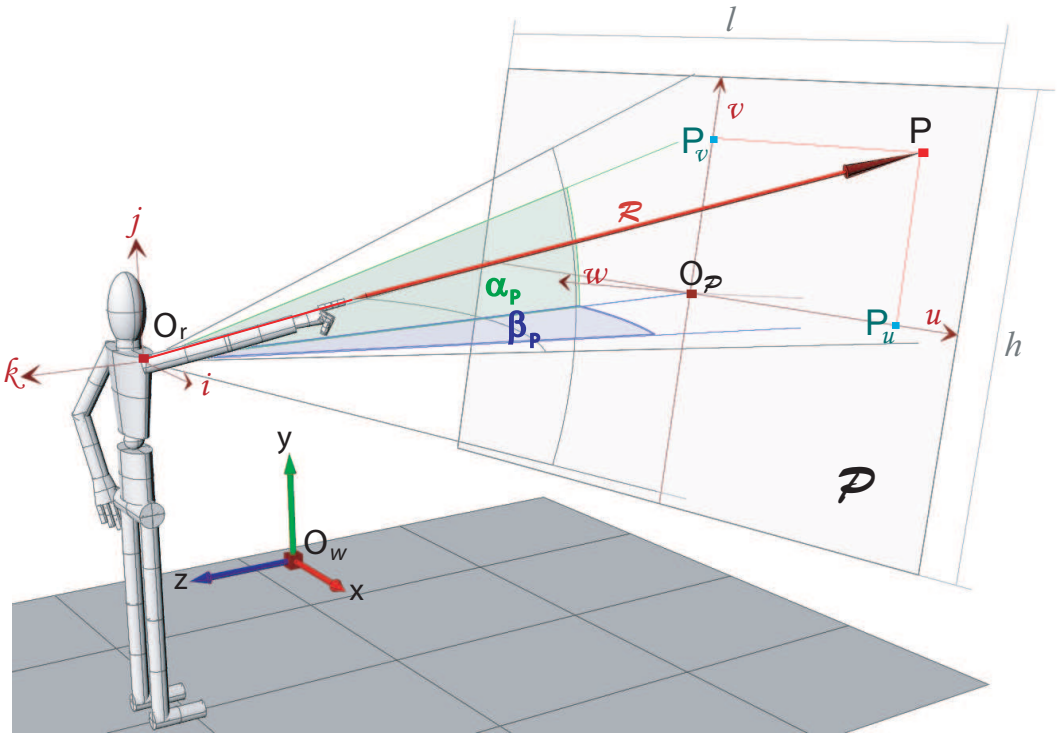
A questo punto siamo in grado di ricavare le coordinate P_u e P_v calcolando i valori delle proiezioni del vettore che origina in $\mathbf{O}_{\mathcal{P}}$ e termina in **P** rispettivamente lungo i vettori **u** e **v** precedentemente calcolati:

$$P_u = (\mathbf{P} - \mathbf{O}_{\mathcal{P}}) \cdot \mathbf{u}$$
$$P_v = (\mathbf{P} - \mathbf{O}_{\mathcal{P}}) \cdot \mathbf{v}$$

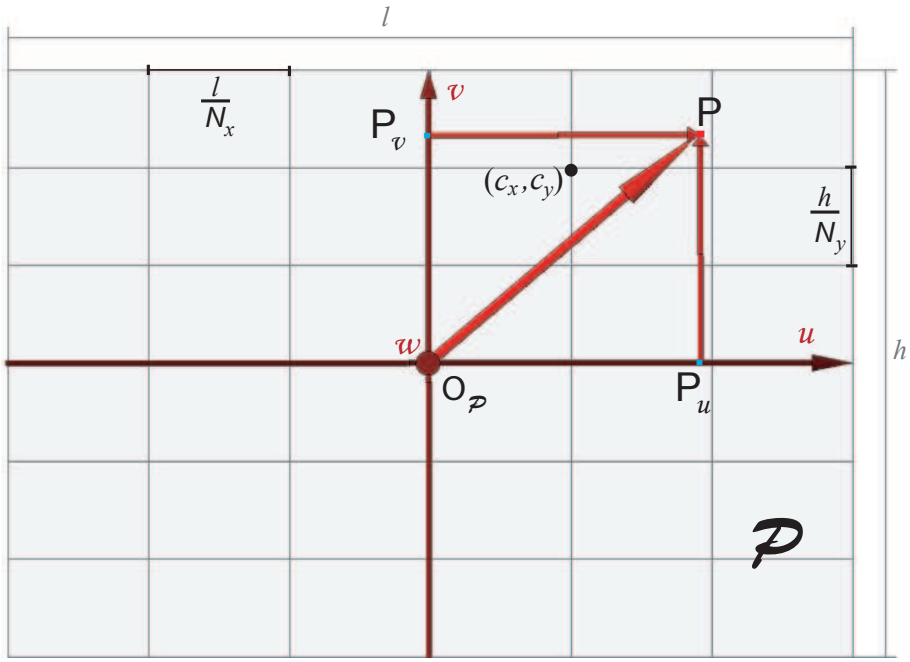
Queste coordinate vengono utilizzate per costruire una coppia di indici interi c_x e c_y che determinano le coordinate di una cella in una *griglia di selezione* (fig. 3.14(b)) disposta sul piano:

$$c_x = \left\lfloor \frac{P_u N_x}{l} \right\rfloor$$
$$c_y = \left\lfloor \frac{P_v N_y}{h} \right\rfloor$$

Dove l e h sono la larghezza e l'altezza della griglia nel sistema di riferimento formato dai vettori $\mathbf{u}$ e $\mathbf{v}$ e N_x e N_y sono il numero di celle orizzontali e verticali della griglia.



(a) Geometria del puntamento nello spazio



(b) La griglia planare nel sistema di riferimento formato da **u**, **v** e $\mathbf{O}_{\mathcal{P}}$ (in questa immagine $N_x = N_y = 6$)

Figura 3.14: Geometrie del puntamento

Capitolo 4

Casi applicativi

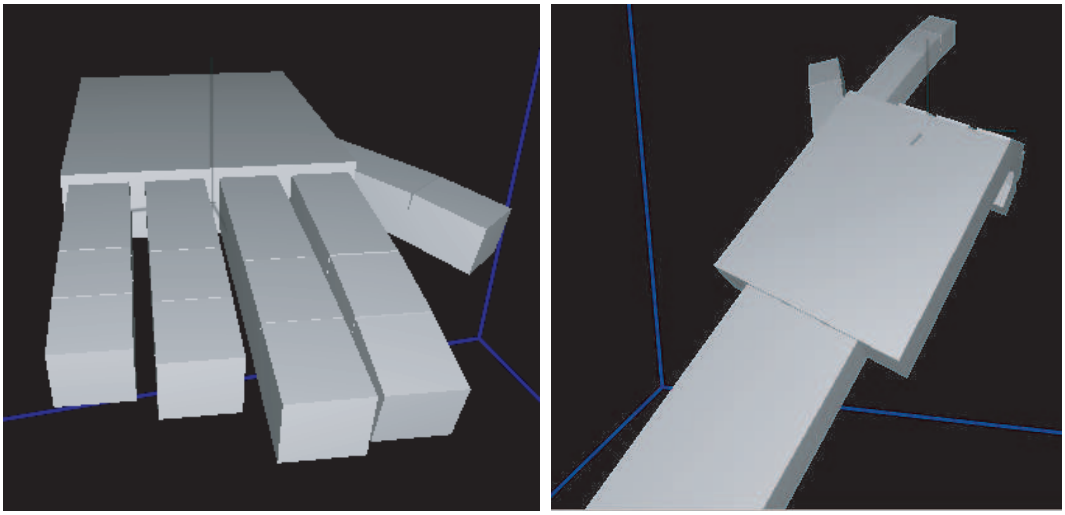
Sono stati sviluppati due *software* indipendenti che fanno uso del guanto *wireless* basato su *WiMoCA*. Il primo è stato chiamato *Glow* ed è stato ideato principalmente con lo scopo di prendere familiarità con gli algoritmi sia di interfacciamento verso la periferica che di decodifica ed elaborazione dei segnali. Questo passo è stato fondamentale in particolare per poter ideare un metodo che consentisse di sfruttare i dati ricavati dal dispositivo per fornire una interazione il più possibile immediata. In seguito gran parte delle soluzioni ideate in questo primo *software* è stata esportata in una seconda applicazione sviluppata chiamata "*Il Muro della Memoria*" e concepita per l'esplorazione virtuale di un monumento storico realmente esistente costituito da un insieme di lapidi riportanti le foto dei Caduti della Resistenza bolognesi.

4.1 *Glow*: un ambiente per il testing immediato

Per disporre un ambiente di sperimentazione per il device si è scelto di realizzare un'applicazione OpenGL [2] che è stata sviluppata con *Microsoft Visual Studio 6.0*®C/C++. Il framework realizzato comprende diverse funzioni base per la gestione del *rendering*, la gestione degli eventi, la temporizzazione, il *threading*, l'accesso in lettura e scrittura alla porta seriale e funzioni per la decodifica ed il *signal processing* dei segnali raccolti. Man mano sono state previste diverse scene interattive per testare le differenti funzionalità implementate e per verificare l'efficacia sull'interazione dei diversi algoritmi adottati.

4.1.1 Generazione di un *avatar*

Una volta approntate le funzionalità di *rendering* dell'applicazione è stato realizzato l'*avatar* di una mano (fig. 4.1(a)) tramite l'uso di primitive *OpenGL* in modo da poter rappresentare visivamente l'inclinazione calcolata della mano dell'utente e la posizione di ogni singolo dito. Per generare il modello della mano si è scelta una semplice implementazione ricorsiva definendo nel codice le varie proporzioni dell'arto e delle dita nonché le disposizioni e gli angoli assunti da ogni giunto.
Durante le azioni di puntamento alla mano viene aggiunto anche un'avambraccio per accentuare la direzione di puntamento.



(a) Il semplice avatar della mano

(b) l'avatar con l'aggiunta dell'avambraccio per le azioni di puntamento

Figura 4.1: L'*avatar* della mano utilizzato in *Glow*

4.1.2 *Testing* iniziale con sequenze pre-registrate di dati

Il passo successivo è stato quello di procedere alla lettura di alcune sequenze di dati pre-registrate su *file* binari per verificare il funzionamento degli algoritmi di decodifica e le funzioni di resa visiva dell'*avatar*. Funzioni di *timing* sono state utilizzate per ottenere una corretta velocità dell'animazione. In questa fase è stato possibile consolidare gli algoritmi di decodifica dei dati che sono poi stati applicati ai dati letti in tempo reale dalla porta seriale virtuale. Da subito si è vista la necessità

di effettuare un filtraggio dei dati per attenuare il rumore ad alta frequenza presente nei valori restituiti dal dispositivo, optando in fine per l'uso della *EWMA* come algoritmo di *smoothing* sia per la sua semplicità implementativa e computazionale che per le sue caratteristiche di risposta al segnale.

4.1.3 Scene di prova

Una volta assodate le funzionalità di resa visiva e di lettura dei dati si è potuto procedere all'esplorazione ed all'affinamento degli algoritmi necessari per poter utilizzare i dati provenienti dal sensore. Le scene di prova sono state create per osservare i vari approcci possibili all'elaborazione dei dati cercando di scegliere ogni volta la soluzione migliore in termini di efficacia dell'interazione.

Uso come inclinometro

Per valutare in maniera immediata l'uso dei dati di inclinazione ricavati dall'accelerometro è stata realizzata una semplice simulazione fisica di un piano inclinato sul quale si muove liberamente una pallina sferica (fig. 4.2c). Cambiando l'inclinazione del sensore è possibile influenzare l'inclinazione del piano controllando così i movimenti della pallina. Quando viene raggiunta una delle quattro pareti la pallina rimbalza contro esse con un urto anaelastico fino ad arrivare a fermarsi. Viene anche emesso un suono ogni qualvolta la pallina urti una delle pareti con una certa intensità.

Puntamento e selezione di una griglia planare

Questa scena di prova ha consentito di sviluppare gli algoritmi alla base del puntamento nella scena tridimensionale. In particolare l'obbiettivo qui perseguito è stato quello di poter selezionare una cella arbitraria appartenente ad una griglia definita da:

- un piano tridimensionale di appartenenza
- un vettore necessario a stabilire l'orientamento della griglia sul piano
- dai valori di larghezza e altezza nel sistema di riferimento definito sul piano
- dal numero di celle presenti in questa area

Per poter sperimentare oltre al puntamento anche la selezione è stata realizzata una scena che ripropone il famoso *gioco del 15* sulla griglia definita come sopra spiegato (fig. 4.2a). L'utente può puntare uno qualsiasi degli elementi: nel momento in cui poi avviene la selezione la disposizione dei pezzi del gioco cambia a seconda che il pezzo selezionato possa o no muoversi. Fin dal primo momento è stato ritenuto necessario fornire all'utente un *feedback* sia ottico che acustico dell'interazione con la scena. Il *feedback* visivo consiste in un rettangolo di *highlight* del pezzo correntemente selezionato mentre il *feedback* acustico è costituito da particolari suoni evocativi legati al fatto che il pezzo possa o no essere mosso e che il gioco venga ultimato.

Puntamento e selezione di una scena 3D arbitraria

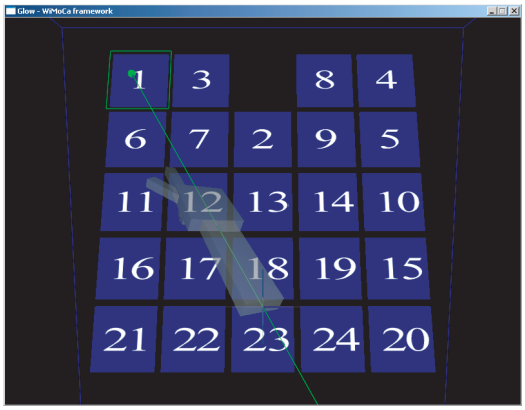
Con scopi analoghi a quelli della scena precedente è stata realizzata un'altra scena in cui è possibile selezionare una arbitraria geometria tridimensionale che sia visibile sullo schermo. Per raggiungere questo obbiettivo si è fatto ricorso alle funzionalità di *picking* (selezione) fornite nativamente dalle *API* di *OpenGL*. La scena (fig. 4.2b) rappresenta una serie di pupazzi di neve che possono essere puntati e selezionati individualmente. Una volta selezionati i pupazzi vengono animati a partire dal dato di inclinazione del sensore.

Si è fatto uso dello stesso algoritmo di calcolo dell'intersezione con il piano di selezione sviluppato per la scena precedente, tuttavia il piano di selezione viene in questo caso fatto coincidere con lo schermo ed il raggio di puntamento viene originato dalla mano del'utente che è considerata come avente una posizione determinata e fissa. Anche in questo caso è stato fornito un *feedback* sia acustico che visivo dell'interazione modificando il colore e la forma dei pupazzi puntati o selezionati e causando l'emissione di un suono ogni volta che un nuovo pupazzo viene selezionato.

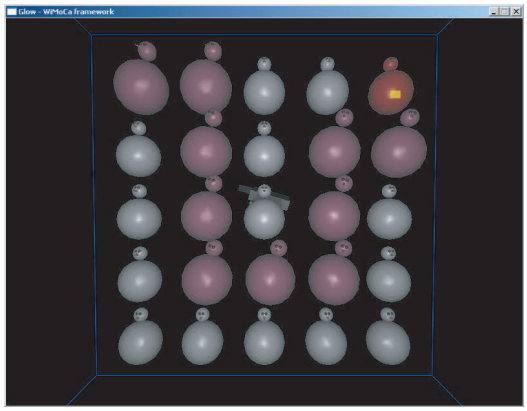
Spostamento tridimensionale

In questa scena si è voluta sperimentare la qualità dell'interazione legata ad uno spostamento in uno spazio tridimensionale. Per ottenere un riferimento visivo del movimento vengono visualizzati dei cubi in una disposizione regolare e virtualmente infinita. Quando l'utente effettua un azione di spostamento i cubi si spostano in

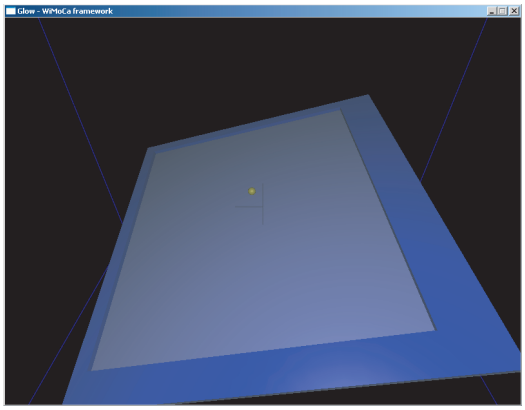
accordo con il dato di movimento calcolato. La velocità e la direzione di questo spostamento variano con l'intensità e la direzione del movimento compiuto simulando una interazione di spostamento nella scena. In particolare in questa fase di sperimentazione si è notato come per l'utente sia risultata più intuitiva una interazione analoga al movimento in prima persona nella scena piuttosto che di una relazione causa-effetto fisica come il trascinamento, probabilmente anche per il fatto che per la natura del dispositivo utilizzato non può esistere una relazione uno a uno con gli spostamenti compiuti dall'utente.



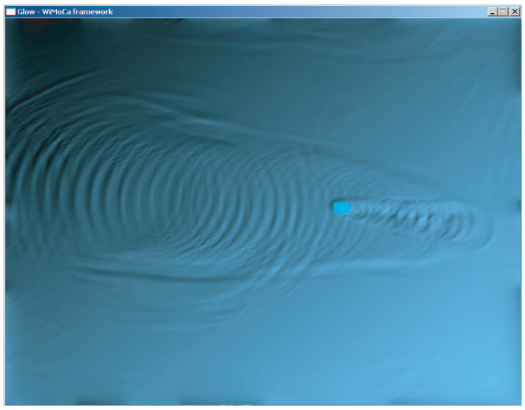
(a) puntamento e selezione su di una griglia planare nello spazio 3D



(b) puntamento in una generica scena 3D utilizzando le funzionalità di *picking* di OpenGL

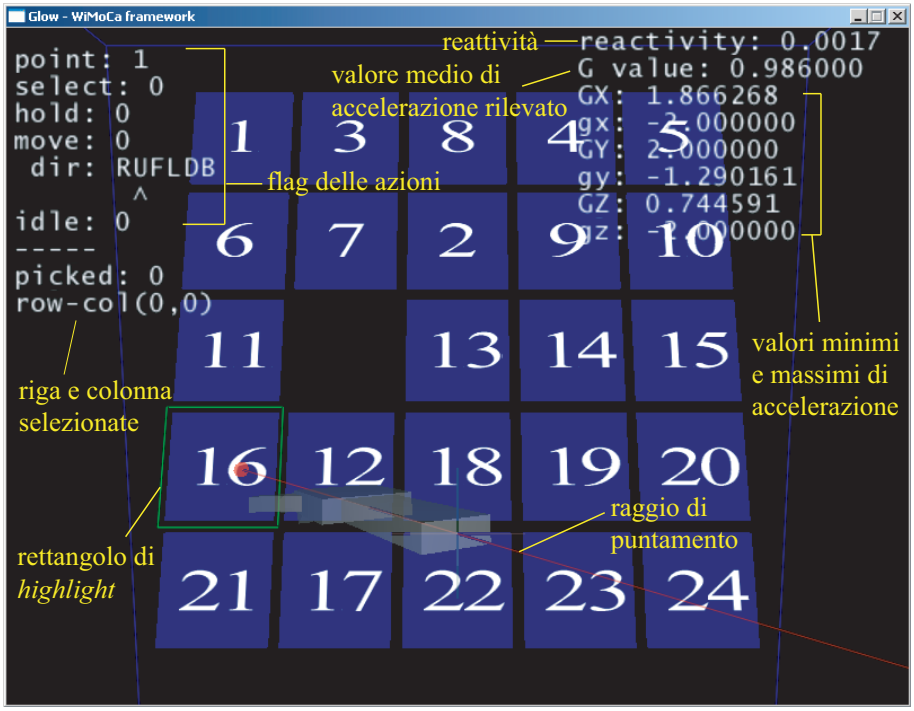


(c) simulazione fisica di una pallina su di un piano inclinato a partire dal rilevamento dell'inclinazione

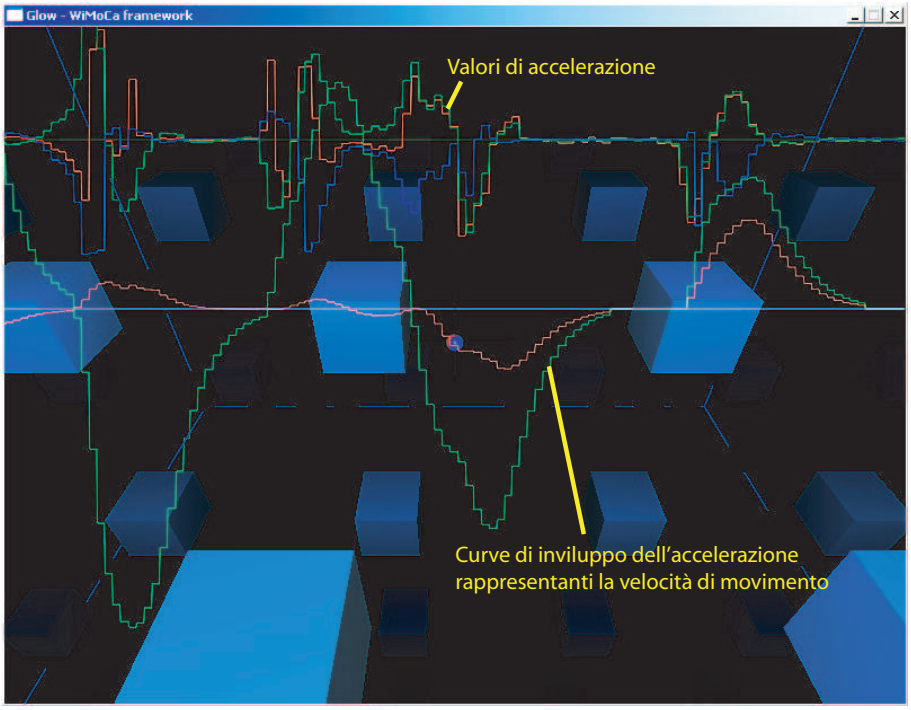


(d) simulazione di increspature dell'acqua a partire dal movimento del punto di selezione e dalla flessione dell'indice

Figura 4.2: Scene di *testing* in *Glow*



(a) *Glow* - schermata di *debug*



(b) *Glow* - schermata grafici

Figura 4.3: *Glow* - alcune delle schermate di *debugging* e di visualizzazione dei valori di animazione

4.2 *Il Muro della Memoria*

Figura 4.4: Bologna: monumento dei Caduti della Resistenza

Il monumento dei Caduti della Resistenza è situato sulle mura di palazzo D'Accursio in piazza Nettuno. Nato spontaneamente dalla deposizione di icone sacre e fiori da parte dei parenti dei partigiani delle brigate bolognesi uccisi, si è poi trasformato nell'attuale sacrario dei Caduti della Resistenza. La versione virtuale di questo monumento vuole fornire la possibilità attraverso semplici interazioni di puntamento di poterne vedere da vicino ogni parte e di consentire la selezione di ognuna delle foto presenti per la visualizzazione dei contenuti a queste collegati.

4.3 Lo schema di interazione

4.3.1 Vista globale

All'avvio dell'applicazione la visuale si trova al centro del muro (fig. 4.7(a)). Lo schermo mostra solo una porzione del muro e all'utente viene data la possibilità di spostarsi verso due altre viste che inquadrano il muro rispettivamente fino al bordo del lato destro e al bordo del lato sinistro puntando verso un'area dello schermo dove sono presenti due icone laterali dalla forma di freccia. Allo stesso tempo viene mostrato un rettangolo rosso che evidenzia quale degli 8 sotto-blocchi nel muro si sta puntando.

4.3.2 Vista ravvicinata e selezione della foto

Figura 4.5: La griglia planare di selezione di un sotto-blocco

Quando l'utente esegue una selezione attraverso la relativa disposizione delle dita si viene portati ad una vista ravvicinata di gruppi di 14 x 10 foto (fig. 4.7(b)). A questo punto l'utente può scegliere ogni singola foto (fig. 4.6) in maniera analoga alla precedente selezione dei sotto-blocchi per ottenere i contenuti multimediali ad

essa annessi (fig. 4.3.3).

Sia i rettangoli di selezione che le icone sono state animate per attrarre in maniera discreta l'attenzione dell'utente sulle azioni a esse collegate [39]. Ad esempio le icone *destra* e *sinistra* compiono un movimento che mette in risalto la direzione che verrà presa nel caso queste vengano selezionate. Allo stesso modo i rettangoli di selezione dei sotto-blocchi e delle singole foto compiono un movimento di pulsazione che rende l'idea di avvicinamento.

Quando l'utente genera un evento di selezione all'interno dell'applicazione vengono calcolate le coordinate della singola foto selezionata attraverso gli algoritmi presentati nel capitolo 3 per la selezione di una cella in una griglia planare. Le coordinate globali del sistema di riferimento delle celle del muro sono due e sono riferite una alla riga (caratteri da A a V dell'alfabeto italiano) ed una alla colonna (numeri interi da 1 a 112).

Per il sotto-blocco selezionato vengono perciò calcolate le coordinate relative al sistema di indici (i;j) che successivamente vengono combinate con i dati di posizionamento del sotto-blocco all'interno del muro per risalire alle coordinate di riga e colonna globali.

4.3.3 Visualizzazione dei contenuti multimediali

Per ogni foto selezionata sono presenti contenuti multimediali come testi e foto. Per esempio la scheda di un partigiano può contenere una breve storia della sua vita, foto o anche lettere scritte (fig. 4.3.3).

Viene avviato un *browser* che apre l'*URL* corrispondente alle coordinate di riga e di colonna selezionate. L'*URL* è composta dall'indirizzo della pagina principale di accesso al database e da una *query string* contenente le coordinate scelte. Ad esempio per la prima foto in alto a sinistra la *query string* corrispondente è

`http://certosa.cineca.it/2/partigiano.php?COORD=A1`

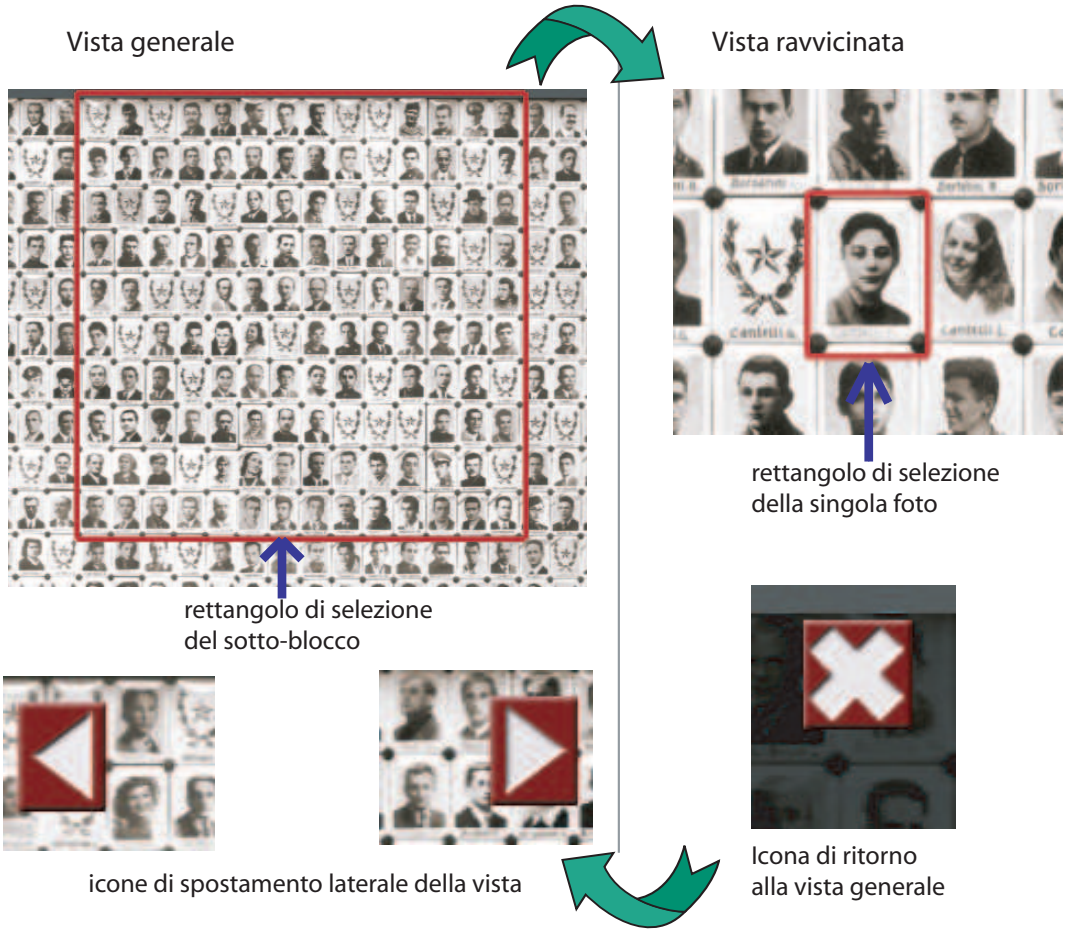
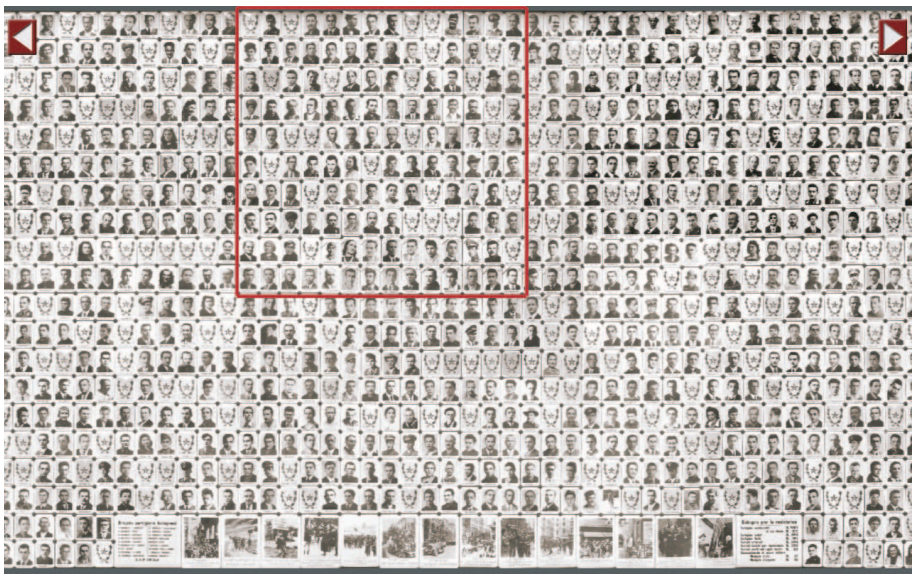
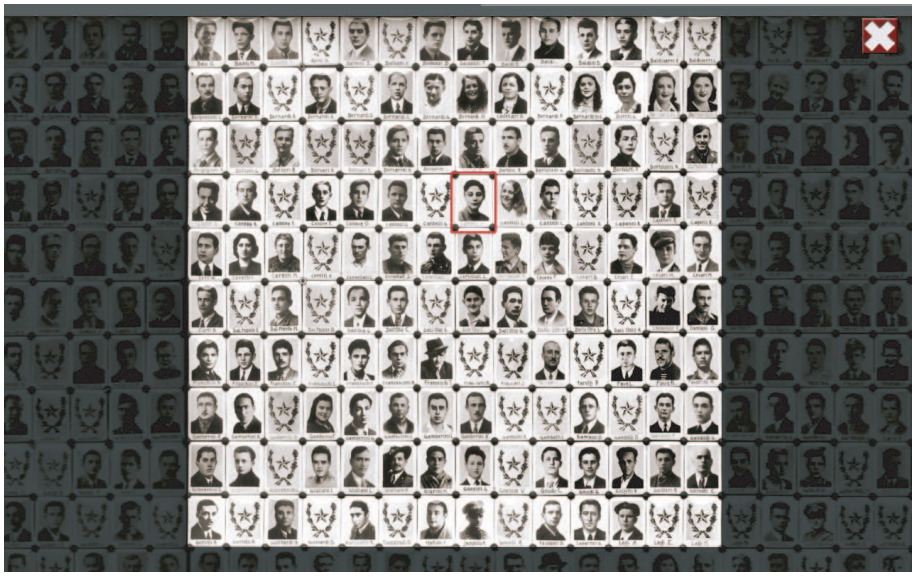


Figura 4.6: I *widget* di spostamento ed il ciclo dello schema di interazione



(a) vista esterna



(b) vista ravvicinata

Figura 4.7: *Il Muro della Memoria* - vista generale e vista ravvicinata

Figura 4.8: Una delle pagine HTML contenenti i dati di un partigiano

Bibliografia

[1] "*Wikipedia, l'enciclopedia libera*", http://wikipedia.org

[2] "*OpenGL - The Industry Standard for High Performance Graphics*" - http://www.opengl.org/

[3] D.R. Begault, "*3D Sound for Virtual Reality and Multimedia*" Academic Press, 1994 - macs.hw.ac.uk

[4] B.H. Juang, S. Furui, "*Automatic Recognition and Understanding of Spoken Language - A First Step Toward Natural Human-Machine Communication*", IEEE Proceedings vol. 88(8), pp. 1142–1165, 2000.

[5] "*Yamaha - Vocaloid*" - vocaloid.com

[6] A. Zafar, J.M. Overhage, C.J. McDonald, "*Continuous speech recognition for clinicians*", Journal of the Medical Informatics Association, vol. 6(3): pp. 195–204, 1999, j-amia.org

[7] "*spokendialogue.dk - SpeechInteractionTheory*" - spokendialogue.dk

[8] T.B. Moeslund, E. Granum, "*A survey of computer vision-based human motion capture*", Computer Vision and Image Understanding, 2001, cs.ucsb.edu

[9] W. Freeman et al. "*Computer vision for computer interaction*", ACM SIGGRAPH Computer Graphics, 33(4). pp. 65–68, 2000.

[10] M. Deering, "*Data Complexity for Virtual Reality: Where do all the Triangles Go?*", Proceedings of VRAIS'93, pp. 357–363, 1993.

[11] I.E. Sutherland, "*A head-mounted three dimensional display*", SPIE Milestone Series MS, 2001 - aec.at

[12] "*Power Wall^{TM}*", http://www.lcse.umn.edu/research/powerwall/powerwall.html

[13] K.J. Fernandes, V. Raja, J. Eyre, "*Cybersphere: The Fully Immersive Spherical Projection System*", Communications of the ACM, 2003.

[14] F. Evans, A. Narayanan, "*Immersive data visualization with the VisionDome*", Proceedings of SPIE, 2003 - adsabs.harvard.edu

[15] C. Cruz-Neira, J. Leigh, M. Papka, C. Barnes, S.M. Cohen, "*Scientists in Wonderland: A Report on Visualization Applications in the CAVE Virtual Reality*", IEEE 1993 Symposium on Research Frontiers in Virtual Reality, 1993.

[16] S. Arca, P. Campadelli, R. Lanzarotti, "*A Face Recognition System Based on Local Feature Analysis*", Lecture notes in Computer Science, 2003 - Springer

[17] K. Tominaga, S. Honda, T. Ohsawa, H. Shigeno, K. Okada & Y. Matsushita, "*Friend Park - expression of the wind and the scent on virtual space*", In Proceedings of Virtual Systems and Multimedia '01, pp. 507–515, IEEE, 2001.

[18] D. Washburn, & L.M. Jones, "*Could olfactory displays improve data visualization?*", Computing in Science & Engineering, 6(6), pp. 80–83, 2004.

[19] "*Dale Air*" http://daleair.com

[20] A. Bodnar, R. Corbett, D. Nekrasovski, "*AROMA: Ambient awaReness through Olfaction in a Messaging Application*", In Proceedings of ACM International Conference on Multimodal Interfaces (ICMI'04), (State College, PA, USA), ACM Press 2004, pp. 183–190.

[21] S.A. Brewster, D.K. McGookin, C.A. Miller, "*Olfoto: Designing a Smell-Based Interaction*", dcs.gla.ac.uk

[22] T.H. Massie, J.K. Salisbury, "*The PHANToM Haptic Interface: A Device for Probing Virtual Objects*", In Proceedings of the ASME Winter Annual Meeting, Symposium on Haptic Interfaces for Virtual Environment and Teleoperator System, 1994.

[23] C. Nichols, "*The vBow: Development of a Virtual Violin Bow*", Proceedings of NIME, NIME-02, Dublin, Ireland, 24–26 Maggio, 2002.

[24] M.R. McGee, P. Gray, S. Brewster, "*Haptic perception of virtual roughness*", Extended Abstracts of ACM CHI 2001.

[25] A. Mor, S. Gibson, J. Samosky, "*Interacting with 3-dimensional medical data: Haptic feedback for surgical simulation*", Proceedings of Phantom User Group Workshop, 1996 - ri.cmu.edu

[26] S. Cotin, H. Delingette, "*Real-Time Surgery Simulation with Haptic Feedback using Finite Elements*", ICRA, 1998.

[27] M. Mahvash, V. Hayward, "*Haptic Rendering of Cutting: A Fracture Mechanics Approach*", Haptics-e, the Electronic Journal of Haptics Research, 2000 - haptics-e.org

[28] S. Greenberg and C. Fitchett, "*Phidgets: Easy development of physical interfaces through physical widgets*", In Proceedings of the UIST 2001.

[29] Edwin L. Hutchins, James D. Hollan, & Donald A. Norman, "*Direct Manipulation Interfaces*", Human-Computer Interaction, Volume 1, pp. 311–338, 1985.

[30] L. Nigay, J. Courtaz, "*A design space for multimodal systems, concurrent processing and data fusion*", INTERCHI'93 Conference on Human Factors in Computing Systems Amsterdam, Addison Wesley, pp. 172–178, 1993.

[31] , P. Wellner, "*Interacting with Paper on the DigitalDesk*", Communications of the ACM, vol. 36, Number 7, pp. 86–97, 1993.

[32] G.W. Fitzmaurice & H. Ishii & W. Buxton, "*Bricks: Laying the Foundations for Graspable User Interfaces*", CHI, pp. 442–449, 1995.

inproceedingsrauterberg97build, M. Rauterberg, M. Bichsel, Ulf Leonhardt, & M. Meier, "*BUILD-IT: a computer vision-based interaction technique of a planning tool for construction and design*", INTERACT, pp. 587–588, 1997.

[33] S. Jorda, M. Kaltenbrunner, G. Geiger, R. Bencina, "*The reacTable*" - iua.upf.edu

[34] R. Bencina, & M. Kaltenbrunner, "*The Design and Evolution of Fiducials for the reacTIVision System*", Proceedings of the 3rd International Conference on Generative Systems in the Electronic Arts, 2005.

[35] M. Kaltenbrunner, T. Bovermann, R. Bencina, E. Costanza, "*TUIO - A Protocol for Table Based Tangible User Interfaces*", Proceedings of the 6th International Workshop on Gesture in Human-Computer Interaction and Simulation (GW 2005).

[36] T.M. Alisi, A. Del Bimbo, A. Valli, "*Natural interfaces to enhance visitors' experiences*", IEEE, Multimedia vol. 12, Issue 3, pp. 80–85, Luglio-Settembre 2005.

[37] J.Y. Han, "*Low-Cost Multi-Touch Sensing through Frustrated Total Internal Reflection*", In Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology, 2005 - http://mrl.nyu.edu/ jhan/ftirtouch/

[38] "*OpenSound Control*" - http://www.cnmat.berkeley.edu/OpenSoundControl/

[39] A. Valli, "*Notes on natural interaction*", 2005 - http://www.naturalinteraction.org/

[40] M. Sama, V. Pacella, E. Farella, L. Benini, B. Riccò, "*3dID: a Low-power, Low-cost Hand Motion Capture Device*", accepted for publication (DATE06 - Designers' Forum), 2006.

[41] M. Sama, V. Pacella, E. Farella, L. Benini, B. Riccò, "*3dID: a Low-power, Low-cost Hand Motion Capture Device*", accepted for publication (DATE06 - Designers' Forum), 2006.

[42] E. Farella, A. Pieracci, D. Brunelli, A. Acquaviva, L. Benini, B. Riccò, "*Design and Implementation of WiMoCA Node for a Body Area Wireless Sensor Network*", Proceeding of International Conference on Sensor Networks, IEEE 2005.

[43] E. Farella, A. Acquaviva, L. Benini and B. Riccò, "*A wearable gesture recognition system for natural navigation interfaces*", Proceedings of EUROMEDIA2005, Toulouse, 11-13 April 2005, pp. 110-115.

[44] "*Bluetooth Core Specification version 1.1*", http://www.bluetooth.com

[45] Mark Weiser, "*The Computer for the 21st Century*", Scientific American, pp. 94–104, Settembre 1991.

[46] Shekhar Y. Borkar, Hans Mulder, Pradeep Dubey, Stephen S. Pawlowski, Justin R. Rattner, Kevin C. Kahn, David J. Kuck, R. M. Ramanathan, Vince Thomas, "*Platform 2015: Intel® Processor and Platform Evolution for the Next Decade - White Paper*", Intel Corporation, www.intel.com/go/platform2015, 2005.

[47] K. Ducatel et al. "*ISTAG, Scenarios for Ambient Intelligence in 2010*" - Technical report, ISTAG, Feb. 2001 - http://www.cordis.lu/ist/

[48] "*IBM Active Protection System Whitepaper*", Ottobre 2003.

[49] C.Y. Chan, "*A treatise on crash sensing for automotive air bag systems*", IEEE/ASME Transactions on Mechatronics, 2002.

[50] http://www.memsnet.org/mems/what-is.html "*What is MEMS technology?*"

[51] T.P. Burg, J. Fritz, S.R. Manalis, "*Microfabricated Mechanical Biosensor with Inherently Differential Readout*", MTL annual research report 2005, p.165.

[52] J. M. Kahn, R. H. Katz, K. Poster, "*Proc. 5th International Conference on Mobile Computing and Networking (Mobicom 1999)*", p.271, 1999.

[53] Jung Kwang Ryul, "*Wireless Sensor Networks*", CCLAB.

[54] P. Bajcsy, "*Toward hazard aware spaces: knowing where, when and what hazard occur*" - NCSA Private Sector Program May, 2005 - algdocs.ncsa.uiuc.edu

[55] P.S. Greenberg and G.W. Hunter, "*MEMS particulate sensors for spacecraft fire safety and environmental monitoring*" - Conference and Exhibit on International Space Station Utilization - 2001, Cape Canaveral, FL - http://www.aiaa.org/

[56] Benny P.L. Lo, Surapa Thiemjarus, Rachel King, Guamg-Zhong Yang, "*Body Sensor Network — a wireless sensor platform for pervasive healthcare monitoring*", Adjunct Proceedings of the 3rd International Conference on Pervasive Computing (PERVASIVE 2005), pp. 77–80, Maggio 2005.

[57] Davide Brunelli, Elisabetta Farella, Laura Rocchi, Marco Dozza, Lorenzo Chiari, Luca Benini, "*Bio-feedback System for Rehabilitation Based on Wireless Body Area Network*", Journal of NeuroEngineering and Rehabilitation, 2:6, 2005.

[58] J.P. Rolland, L. Davis, Y. Baillot, "*A survey of tracking technology for virtual environments*", Fundamentals of Wearable Computers and Augmented Reality, 2001 - pbl.cc.gatech.edu

[59] G.A. Podkolinska, "*Exploring Bimanual Spatial Input Devices in Static and Manipulable Environments*", University of Calgary, Aprile 2003 - mfadt.parsons.edu

[60] A. Pezeshk, "*Designand Implementationofa 3D Computer Game Controller Using Inertial MEMS Sensors*", Michigan Technological University, 2004 - sunshine.lib.mtu.edu

[61] H.J. Luinge, P.H. Veltink, "*Measuring orientation of human body segments using miniature gyroscopes and accelerometers*", Medical & Biological Engineering & Computing, 2005 - xsens.com

[62] R.E. Kalman, "*A New Approach to Linear Filtering and Prediction Problems*", Transactions of the ASME–Journal of Basic Engineering, vol. 82, Series D, pp. 35–45, 1960.

[63] G. Welch and G. Bishop, "*An introduction to the Kalman filter*", Technical Report TR 95-041, University of North Carolina, Department of Computer Science, 1995.

[64] Julier, Simon J. and Jeffery K. Uhlmann, "*A New Extension of the Kalman Filter to nonlinear Systems*", Proceedings of AeroSense: The 11th International Symposium on Aerospace/Defense Sensing,Simulation and Controls, Multi Sensor Fusion, Tracking and Resource Management II, SPIE, 1997.

[65] N. Funk, "*A Study of the Kalman Filter Applied to Visual Tracking*", CMPUT 652, University of Alberta, Dicembre 7, 2003.

[66] P. Gutman, M. Velger, "*Tracking Targets Using Adaptive Kalman Filtering*", IEEE Transactions on Aerospace and Electronic Systems Vol. 26, No. 5, pp. 691–699 (1990).

[67] C.C. Holt, "*Forecasting seasonal and trends by exponentially weighted moving averages*", Office of Naval Research, Research Memorandum No. 52, 1957.

[68] J. Deisinger, A. Kunz, "*Double Exponential Smoothing: An Alternative to Kalman Filter-Based Predictive Tracking*",

[69] J.J. LaViola Jr. "*An Experiment Comparing Double Exponential Smoothing and Kalman Filter-Based Predictive Tracking Algorithms*", Proceedings of the IEEE Virtual Reality 2003.

[70] Eric Foxlin, "*Motion Tracking Requirements and Technologies*", Extended draft version of Chapter 8 in Handbook of Virtual Environment Technology, Ed. Lawrence Erlbaum Associates, 2002.

[71] Merryn Mathie, Nigel H. Lovell, Branko G. Celler, A.C.F. Coster, "*Determining activity using a triaxial accelerometer*", Proceedings of the second joint EMBS/BMES Conference, Houston, TX, USA, 23-26 Ottobre 2002.

[72] Dean M. Karantonis, Michael R. Narayanan, Merryn Mathie, Nigel H. Lovell, Branko G. Celler, "*Implementation of a Real-Time Human Movement Classifier Using a Triaxial Accelerometer for Ambulatory Monitoring*", IEEE transactions on information technology in biomedicine, vol. 10, n. 1, Gennaio 2006.

[73] STMicroelectronics, "*LIS3L02DQ MEMS INERTIAL SENSOR - 3 Axis, $\pm 2g$, Digital Output Low Voltage Linear Accelerometer datasheet*", http://www.st.com/stonline/products/literature/ds/10175/lis3l02dq.pdf

[74] RF Monolithics Inc, "*TR1001 - 868.35*MHz *Hybrid Transceiver datasheet*", 2003, http://www.rfm.com/products/data/tr1001.pdf

[75] Texas Instruments, "*MSP430x13x, MSP430x14x, MSP430x14x1 Mixed Signal Microcontroller datasheet*", Giugno 2004.

[76] ATMEL, "*ATmega8 ATmega8L, 8-bit AVR with 8K Bytes In-System Programmable Flash datasheet*", Ottobre 2004, http://www.atmel.com/dyn/resources/prod_documents/doc2486.pdf

Ringraziamenti

Questo periodo di tirocinio a Bologna mi ha permesso di imparare nuove cose e di condividere momenti piacevoli insieme a persone interessanti. Colgo l'occasione per ringraziare tutto il personale di ricerca di *Microidea*. In particolare sono grato ad Omar per i consigli e le discussioni riguardanti i vari algoritmi e per la sua grande gentilezza e disponibilità. Lo stesso vale per Augusto, Elisabetta, Piero, Luciano, Sergio, Simone, Michele e Vincenzo.
Voglio anche ringraziare per la compagnia durante buona parte del periodo di tirocinio Fabio e Francesco con i quali ho trascorso momenti sereni.
Voglio qui salutare anche Paolo Quadrani, Silvano Imboden e Marco Petrone del laboratorio del CINECA, specialmente Paolo e Silvano il cui aiuto è stato essenziale per poter realizzare alcune parti fondamentali dell'applicazione de "Il Muro della Memoria".
Un ringraziamento sentito va anche a Cinzia Zannoni e al mio relatore prof. Daniele Marini dei quali ho apprezzato la solarità e disponibilità.

Ringrazio in fine mia sorella Claire ed i miei genitori, grazie per essermi sempre stati di supporto nei momenti difficili.

Grazie a tutti

Mathieu