“ *Alla mia famiglia”*

“L'impossibile è un limite della mente.
Non dell'anima e nemmeno del cuore.”

M.Fiorillo

Indice

Abstract

La tesi in oggetto è stata svolta presso l'azienda BTS S.r.l di Padova ed ha avuto come obiettivo lo sviluppo di un metodo di calibrazione che permetta di integrare e confrontare i dati provenienti da due sensori D-RGB quali *Microsoft Kinect* e *Asus Xtion Pro Live.* Ci si è avvalsi, quindi,del linguaggio di programmazione Qt e delle librerie disponibili in rete OpenNI, NITE e OpenCV. Per quanto riguarda l'algoritmo di calibrazione, il metodo sviluppato si basa sull'algoritmo di *Horn* e permette il calcolo dei parametri estrinseci di rototraslazione tra due dispositivi, sulla base di un data set composto da N>600 punti di controllo riconoscibili dai sensori in esame. E' stato inoltre necessario sviluppare il tool da utilizzare per la calibrazione dei due sensori. Per stimare l'accuratezza del metodo adottato ci si è avvalsi di uno strumento di confronto quale la stereofotogrammetria (sistema SMART-DX/100, BTS). Si sono svolti pertanto dei test in modo da replicare la prova di *spot-checks per* la misura di distanza tra marcatori che hanno permesso di stimare un errore massimo (in termini di RMSD) di 6,42 mm (1 sensore) e minimo di 3,56 mm (2 sensori).

Capitolo 1

Introduzione

Il presente lavoro di tesi è stato svolto in collaborazione con la sezione *R&D* dell'azienda BTS di Padova, leader nella progettazione e produzione di sistemi di *motion capture* e analisi del movimento.
L'analisi del movimento umano ha lo scopo di raccogliere informazioni quantitative relative alla meccanica del sistema muscolo-scheletrico durante l'esecuzione di un atto motorio.
Le grandezze che forniscono queste informazioni possono essere misurate oppure stimate mediante modelli matematici morfo-funzionali dei tessuti, degli organi, degli apparati o dei sistemi coinvolti nell'analisi.
Così facendo, possono essere ottenute descrizioni quantitative delle funzioni a carico dell'apparato locomotore in condizioni definite normali, nonché delle loro variazioni (potenziamento o riduzione della funzione).Tutto questo per ragioni euristiche ed applicative che possono spaziare dal campo della robotica, alla *computer vision*, dalla medicina al *gaming*.

Ad oggi le tecniche maggiormente utilizzate per ricavare queste informazioni si basano sulla ricostruzione della posizione e orientamento dei vari segmenti corporei a partire dalla posizione rilevata attraverso un sistema stereofotogrammetrico di alcuni marcatori aderenti alla superficie del segmento stesso.

I limiti di questa tecnica, legati in particolare alla deformabilità dei tessuti molli ai quali sono fatti aderire i marcatori, hanno spinto i ricercatori a sviluppare nuove metodiche che non prevedano l'utilizzo di *markers*.
Le tecniche *markerless* per l'analisi del movimento sono quindi oggetto di forte interesse ma presentano tutt'oggi forti limiti che ne impediscono l'adozione a livello clinico.

Il recente sviluppo di telecamere D-RGB (Depth-RGB) a basso prezzo in grado di generare mappe di profondità ha spalancato le porte a nuove possibilità, fornendo strumenti utilizzabili in un'innumerevole quantità di ambiti tra cui appunto l'analisi del movimento attraverso un approccio *markerless*.
La disponibilità delle librerie *OpenNI* e *NITE* per l'utilizzo su PC di tali sensori, ha suscitato un grande movimento nella comunità di sviluppo libero di software dando pieno sfogo alla fantasia degli sviluppatori nella creazione delle applicazioni più disparate.

In soli venticinque giorni dal suo lancio sul mercato il 4 novembre 2010, Microsoft ha venduto due milioni e cinquecentomila unità Kinect fino a raggiungere il record di otto milioni di unità vendute nei suoi primi sessanta giorni di commercializzazione.
Con un curriculum simile, anche BTS, in quanto azienda aperta alle nuove tecnologie nel continuo tentativo di miglioramento, ha ben pensato di indagare le reali potenzialità di tali dispositivi e la loro possibile applicazione nel campo della *motion capture*.
In questa tesi si è cercato dunque di acquisire le capacità per l'utilizzo degli strumenti hardware e software forniti da questa nuova tecnologia in forte espansione e indagarne le potenzialità con lo scopo di adottarle come miglioria nei prodotti BTS in fase di sviluppo e in particolare all'interno del progetto "*Nirvana*".

La primissima fase del periodo di tesi è stata caratterizzata dall'apprendere le basi della programmazione in *C#/C++*, linguaggi con cui è sviluppato il software per la gestione dei dispositivi, e dell'ambiente di lavoro *QT* nel quale è stato implementato l'intero progetto di tesi.

In una fase successiva si è iniziato a lavorare con le *OpenNI*. Queste, come verrà ampiamente discusso, sono le librerie sviluppate e rilasciate dalla società israeliana *Primesense* che permettono di interagire con i sensori in questione. Si tratta di librerie estremamente ampie ed è stato quindi necessario un primo periodo di studio per comprenderne ed apprenderne il funzionamento, capire come consentono l'interazione con il dispositivo, che tipo di dati si possono ottenere e come ottenerli, al fine di iniziare a scrivere alcune semplici applicazioni.

Apprese le basi indispensabili per interfacciarsi con tali dispositivi, come primo *step* è stata valutata la bontà dei dati ricavabili dalle primitive di "*skeleton tracking*" messe a disposizione dalle *OpenNI* e *NITE* mediante lo sviluppo di una sorte di avatar, un modello 3D costruito utilizzando le librerie *OpenGL*, in grado di riproporre in tempo reale posizione e gesti effettuati dal soggetto all'interno dell'area di lavoro.

In questo modo, fornendo la posizione 3D istantanea di 15 *key points* che descrivono le principali articolazioni del soggetto in input al modello è possibile valutare, da un punto di vista qualitativo, la bontà
con cui il dispositivo identifica tali punti in esame e valutare quindi la bontà del *tracking*.

Determinata la scarsa precisione dei dati forniti lavorando ad alto livello con le *OpenNI* e le problematiche sorte nell'utilizzare un dispositivo D-RGB nel progetto Nirvana si è indagata la possibilità di utilizzare contemporaneamente più sensori in modo da poter coadiuvare ed integrare i dati forniti da punti di vista multipli nel tentativo di aumentare l'accuratezza della mappa di profondità ricostruita.

L'obiettivo della seconda parte della tesi è stato dunque quello di implementare una nuova tecnica di calibrazione che, a differenza di quelle presenti in letteratura, si basa sul dato per cui tali sensori sono stati progettati ossia la terza dimensione ricavabile dalla *depth map.*

Viene dunque qui proposta e implementata una nuova tecnica per procedere alla calibrazione di due o più sensori di profondità basata sulla miglior stima dei parametri di rototraslazione in grado di mappare punti visti da una camera in punti corrispondenti visti da un'altra.

È stato quindi sviluppato un software che, mediante l'utilizzo di alcuni *widget* (elementi grafici dell'ambiente *QT*), permette l'acquisizione e l'elaborazione dei frame al fine di ricavare il *dataset* su cui lavora l'algoritmo, e infine di calcolare il valore dei parametri estrinseci, scopo della calibrazione.
L'intero processo di elaborazione è completamente automatico e sono necessari non più di quattro minuti per procedere alla completa calibrazione del sistema.

A validazione del metodo sviluppato sono riportati i risultati dei test svolti per quantificare la bontà di un sistema stereo costituito da due dispositivi *Asus Xtion Pro Live*, nel ricostruire l'interdistanza tra due punti, confrontando i risultati ottenuti con quelli acquisiti dal sistema stereo fotogrammetrico optoelettronico *SMART-DX\100* sviluppato da BTS che rappresenta il *gold standard* per quanto riguarda il *marker-based motion capture*.

Capitolo 2

L’azienda e il progetto Nirvana

2.1 Presentazione dell’azienda

Fig. 1 Marchio dell’azienda BTS Bioengineering [41]

BTS Spa [41] viene fondata nel 1986 in seguito ad uno *Spin-off* del Centro di Bioingegneria della Fondazione Don Gnocchi e del Politecnico di Milano. Scopo della start-up era lo sfruttamento industriale delle innovative metodologie per l’analisi del movimento sviluppate dai ricercatori del Centro di Bioingegneria.
Nel 2000 nasce a Padova, come *Spin-off* dell'Università di Padova e con l'aiuto di un paio di imprenditori locali, un’azienda chiamata *eMotion*; anch’essa si occupa di innovazioni tecnologiche per l’analisi del movimento.
Nel 1999 la BTS viene acquistata da *TC Sistema* Spa e nel 2000 *TC Sistema* acquista pure *eMotion*.
Nel 2004, attraverso un’operazione di *MBO* la società BTS è stata acquistata da dodici dipendenti con la partecipazione di tre imprenditori esterni ed incorpora, nel 2005, la padovana *eMotion* che, nella sede di Padova, costituisce il gruppo di ricerca e sviluppo dell’azienda.
Con oltre trecentoquaranta clienti in quaranta paesi del mondo, ventotto dipendenti (di cui ventuno ingegneri, undici dei quali dedicati alla ricerca e sviluppo) e due sedi (Padova e Milano), oggi BTS è uno dei leader mondiali delle tecnologie per l’analisi del movimento.

Giorno dopo giorno l'azienda mira a diventare il riferimento industriale per le tecnologie e le applicazioni dell'analisi del movimento nel mondo clinico.

BTS Bioengineering si occupa quindi di misurare il movimento del corpo umano in ambito clinico e di fornire quest'informazione a ricercatori e medici interessati a capirne a fondo le dinamiche. Pioniera in questo campo, ha inventato il laboratorio integrato di analisi del cammino (*BTS Elite*) ed aiutato migliaia di pazienti affetti da patologie neuromuscolari a migliorare la loro condizione attraverso una valutazione strumentale della loro disabilità.
I prodotti sviluppati da BTS comprendono:

- Sistemi integrati per l'analisi clinica del movimento.
- Sistemi per l'acquisizione e l'elaborazione di segnali elettromiografici.
- Sistemi dinamici per l'analisi della postura.
- Sistemi optoelettronici per la misura della ventilazione polmonare.
- Sistemi per il controllo degli stati vegetativi.
- Sistemi per la riabilitazione neuromotoria in ambienti immersivi.

Ospedali, centri di cura e cliniche riabilitative utilizzano i sistemi di analisi del movimento BTS per valutare le migliori cure per le disfunzioni del movimento e l'efficacia dei percorsi di riabilitazione. Il personale medico ottiene in modo rapido, accurato e non invasivo una dettagliata analisi quantitativa integrata dei parametri biomeccanici e neuromuscolari del paziente.
Attualmente BTS offre una soluzione completa per l'intero processo riabilitativo, che comprende sia i sistemi per la terapia sia i sistemi per la valutazione funzionale del movimento. Alcuni esempi includono:

- robot per la riabilitazione di arti superiori ed inferiori e per il paziente costretto a letto per patologie varie
- la realtà virtuale, che grazie alla computer grafica e alla creazione di ambienti interattivi, motiva il paziente alla partecipazione attiva al percorso riabilitativo
- cinematica, dinamica ed elettromiografia di superficie integrate e comprendenti analisi specifiche per disordini motori: *template* predisposti per la valutazione di cammino, *grasping*, *tapping*, *hand to mouth*, mobilità cervicale
- analisi di biosegnali, (come impedenza cutanea, variabilità cardiaca, temperatura, frequenza respiratoria) per il *biofeedback* terapeutico e valutativo.

Recentemente, BTS ha fondato BTS *Biomedical* [41*]*, una nuova divisione impegnata nello sviluppo di soluzioni tecnologiche per migliorare l'analisi, supportare la diagnosi e rendere più efficace la terapia in ambito clinico. L'obiettivo è quello di fornire al medico e allo specialista strumenti potenti e facili da usare per la valutazione funzionale e la riabilitazione, in grado di anticipare la diagnosi di molte malattie degenerative, di migliorare la terapia e di favorire il recupero di pazienti affetti da disabilità conseguenti a traumi.
I medici possono ottenere soluzioni integrate complete, pronte all'uso, con un'opportuna formazione e con assistenza continua affinché nel tempo le soluzioni offerte possano diventare sempre più utili.

BTS *Biomedical* ha il compito di concretizzare gli investimenti di tutti questi anni affinché le persone vittime di eventi traumatici possano recuperare al meglio la loro capacità di muoversi e quelle affette da problemi meno gravi possano mantenere le loro capacità motorie per periodi sempre più lunghi.

2.1.1 Settori in cui opera BTS

L'analisi computerizzata del movimento trova impiego nei più svariati ambiti di lavoro. Ad esempio nella ricerca clinica per valutare le limitazioni motorie conseguenti ad una patologia o per testare l'efficienza di ausili ed protesi, nel campo musicale o negli studi ergonomici per valutare la postura conseguente all'uso di uno strumento musicale o alla postazione di lavoro, negli studi aeronautici e spaziali per studiare il movimento in assenza di gravità.
Non meno importanti sono le applicazioni sugli oggetti, come il monitoraggio del movimento effettuato da robot o da oggetti manovrati dall'uomo (bracci meccanici, attrezzi da lavoro, *crash test*).

Ospedali, Centri di Cura e Cliniche Riabilitative utilizzano i sistemi di analisi del movimento BTS per analizzare le disfunzioni del movimento, prescrivere il trattamento riabilitativo più adatto e valutarne l'efficacia. Il personale medico ottiene in modo rapido, accurato e non invasivo una dettagliata analisi quantitativa integrata dei parametri biomeccanici e neuromuscolari del paziente esaminato. Oggi, oltre 10.000 pazienti all'anno vengono assistiti nel loro percorso clinico dai sistemi di analisi del movimento BTS.

I più prestigiosi istituti scientifici e Universitari utilizzano i sistemi BTS per compiere ricerche nei campi della biomeccanica, biometria, diagnostica e scienze motorie. La tecnologia BTS è stata impiegata nei più importanti programmi spaziali; sistemi BTS sono stati installati sulle stazioni *Spacelab e MIR*. Sistemi BTS sono impiegati da *NASA, ESA, CNRS* nei programmi di volo parabolico e per una serie di esperimenti per l'*International Space Station Program*.

I sistemi BTS sono infine impiegati da numerose organizzazioni di medicina sportiva con lo specifico obiettivo di massimizzare le prestazioni degli atleti, prevenire infortuni e migliorare l'efficacia della riabilitazione post-traumatica. I sistemi BTS sono anche utilizzati da prestigiose squadre di calcio quali l'A.C. Milan, F.C. Juventus e F.C. Real Madrid per le valutazioni di routine dei loro giocatori. Tutte queste variabili possono essere sincronizzate e integrate in un'unica e completa analisi multifattoriale.
Il monitoraggio dello stato biomeccanico dell'atleta permette di:

- definire nuove tecniche di allenamento
- ridurre nel lungo periodo le probabilità di infortuni sportivi
- individuare e correggere con anticipo eventuali carenze nella preparazione atletica
- in caso di infortunio, identificare il miglior trattamento da applicare (programmi riabilitativi o interventi chirurgici)
- monitorare nel tempo gli effetti del trattamento applicato

- individuare il raggiungimento del pieno recupero in modo da evitare ricadute a causa di una precoce ripresa delle attività

2.2 Il progetto Nirvana

BTS *Nirvana* [34] è una soluzione terapeutica innovativa per la riabilitazione di pazienti affetti da patologie neuromotorie. Il sistema include esercizi specifici per le diverse problematiche attentive e motorie associate alle disabilità: il terapeuta può quindi utilizzare soluzioni riabilitative predefinite come definirne di nuove, mirate alle criticità del paziente. L' efficacia di BTS *Nirvana* è nel suo approccio motivazionale: la natura ludica, *target oriented*, delle attività e la ricchezza dei feedback sensoriali, stimolano proattivamente il paziente.

Fig. 2 Esempio di proiezioni create da Nirvana con la quale il paziente può interagire [34]

Nirvana è il primo sistema markerless che consente una completa immersione visiva ed uditiva in un ambiente virtuale, senza l'ausilio di alcun dispositivo invasivo che limiti o alteri la libertà d'interazione. Basato su un dispositivo optoelettronico a raggi infrarossi, *Nirvana* crea ambienti virtuali, proiettabili su superfici orizzontali e verticali, con i quali il paziente è in grado di interagire attraverso il semplice movimento. Gli specifici esercizi riabilitativi presentano diverse modalità e livelli di difficoltà crescenti. Sono inoltre caratterizzati da numerosi feedback sensoriali: rispetto a un approccio terapeutico convenzionale, il paziente riceve maggiori stimoli cognitivi e motori che agiscono sul profilo motivazionale.

Nirvana trova la sua collocazione all'interno di qualunque centro che si occupi di riabilitazione di pazienti con deficit sensori- motori agli arti inferiori e superiori e cognitivi, siano essi dovuti all'effetto di lesioni centrali, come negli *stroke* e nei traumi cranici, o causati da malattie neurologiche a decorso cronico e progressivo, quali la malattia di *Parkinson* e la sclerosi multipla. Il sistema, caratterizzato da un' elevata versatilità, mette a disposizione del fisioterapista un set di

esercizi predefiniti per l'arto superiore, per l'arto inferiore e per il controllo del tronco. Alcuni esercizi, che mirano al recupero del controllo motorio e alla riduzione dell'*impairment*, hanno carattere generale e possono essere utilizzati sia nella riabilitazione del paziente emiplegico, sia in pazienti affetti da malattia di *Parkinson* o da sclerosi multipla.
Altri esercizi, per la presenza di particolari feedback uditivi e visivi, piuttosto che per una disposizione studiata ad hoc degli oggetti nello spazio, sono specifici per il trattamento di una patologia. È possibile ad esempio utilizzare stimoli visivi nella riabilitazione del paziente parkinsoniano, *rhythmic cueing* per l'emiplegico, o esercizi per pazienti con *neglect* che, durante il trattamento riabilitativo degli arti inferiori e superiori, lo spingano all'esplorazione spaziale.
Ogni esercizio proposto è corredato da una metrica che permette al fisioterapista di monitorare la performance del particolare esercizio e di decidere quando variarne il livello di difficoltà. Inoltre, lo score, visualizzato anche dal paziente durante l'esecuzione dell'esercizio, costituisce un' ulteriore spinta motivazionale. L'acquisizione via webcam di ogni sessione, unitamente all'opportunità di valutare secondo parametri quantitativi il lavoro svolto, rendono possibile un'analisi nel tempo dell'attività del paziente, i cui dati anagrafici e medici sono contenuti nella scheda *RFID*. Alla fine di ogni sessione di lavoro, è possibile generare un report con l' elenco degli esercizi svolti e i punteggi ottenuti per ognuno. È possibile, inoltre, avere una rappresentazione temporale dei risultati del ciclo riabilitativo per poter evidenziare i progressi del paziente e i benefici ottenuti con il trattamento. In opzione, *Nirvana* può essere integrato con il sistema di elettromiografia di superficie BTS per una valutazione funzionale del soggetto, permettendo l'analisi ulteriore delle strategie di esecuzione del movimento.

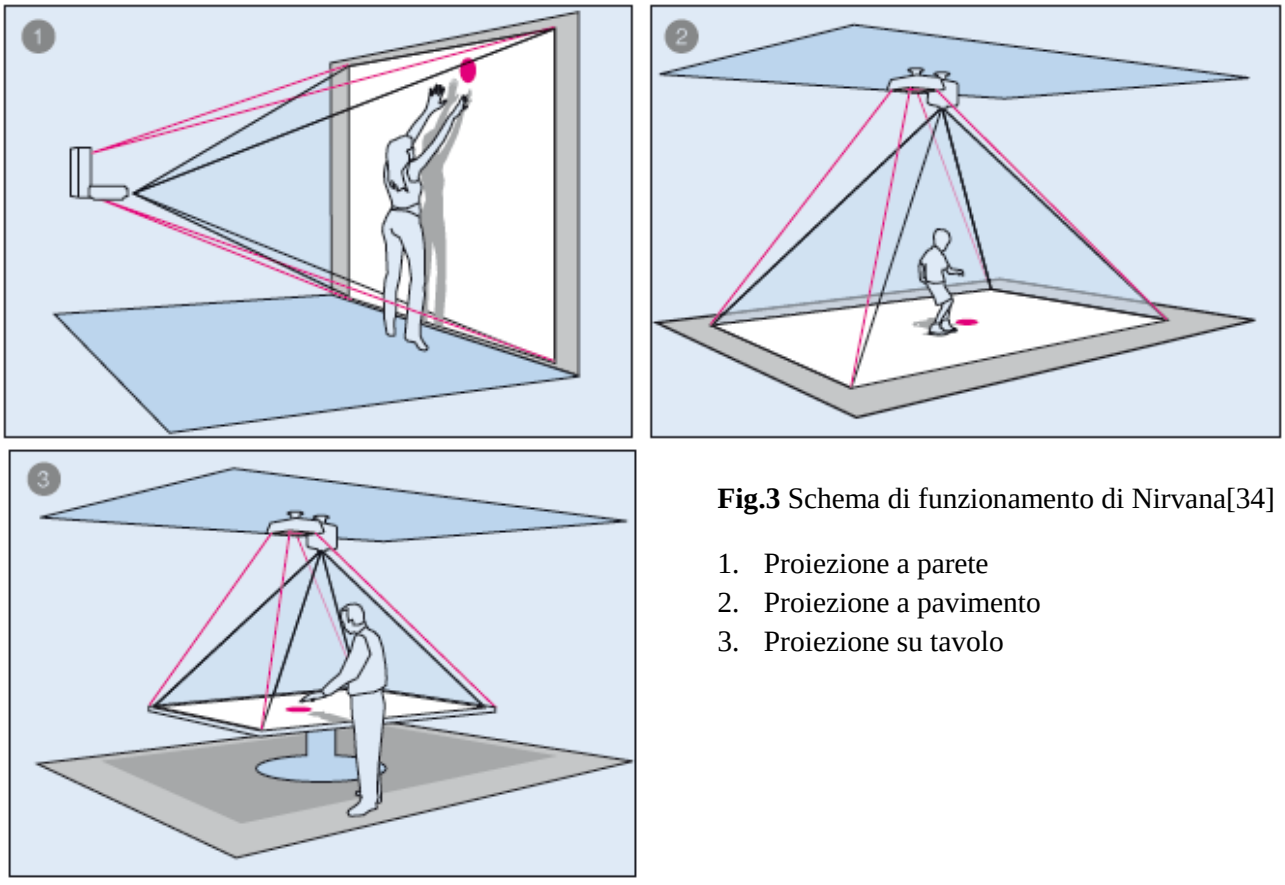


Fig.3 Schema di funzionamento di Nirvana[34]

1. Proiezione a parete
2. Proiezione a pavimento
3. Proiezione su tavolo

2.2.1 Protocolli e tipologie di esercizi

I protocolli riabilitativi inclusi in BTS Nirvana, sviluppati dall'Ospedale Valduce - Villa Beretta, di Costamasnaga (LC), definiscono ambienti virtuali per programmi di esercizio terapeutico

di rieducazione neuromotoria.
BTS Nirvana è particolarmente indicato per l'impiego in programmi di riabilitazione di pazienti con problematiche attentive e problematiche motorie degli arti superiori, inferiori e del tronco conseguenti a:

- Esiti da ictus in fase subacuta
- Esiti da traumi cranio-encefalici
- Morbo di Parkinson
- Sclerosi Multipla
- Paraparesi di varia natura
- Lesioni nervose periferiche

I protocolli riabilitativi inclusi in BTS Nirvana comprendono esercizi di:

- *Sprites*
- *Follow me*
- *Motion*
- *Hunts*
- *Games*

Gli esercizi riabilitativi sono spesso ripetitivi e poco stimolanti per il paziente che si annoia rapidamente, diminuendo l'impegno e la partecipazione e, di conseguenza, l'efficacia del trattamento stesso.
Con BTS Nirvana l'esercizio motorio diventa lo strumento di interazione con la scena virtuale.
Tutti gli esercizi proposti sono caratterizzati da un'elevata stimolazione sensoriale visiva e uditiva e la grafica accattivante esercita una forte spinta motivazionale sul paziente.

Rispetto ad altri approcci di interazione virtuale, Nirvana presenta due significativi vantaggi:
1) L'interazione attraverso il semplice movimento, non mediata da un avatar, ovvero da una rappresentazione grafica ricreata, ne permette l'utilizzo anche in presenza di difficoltà cognitive.
2) L'assenza di dispositivi invasivi, di tipo visivo (caschi, occhiali) o di movimento (guanti e *marker*) permette, oltre ad una completa tollerabilità per tutti i pazienti, di ridurre al minimo o eliminare i rischi di caduta.
Gli esercizi sono concepiti per essere eseguiti anche in presenza di ausili o in presenza dello stesso terapista a sostegno del paziente.

2.2.2 Controllo dei risultati

Il sistema di *scoring* degli esercizi permette, in maniera semplice e intuitiva, di valutare i progressi del paziente e di modificare, se necessario, in tempo reale lo stesso programma riabilitativo incrementando o riducendo il livello di difficoltà degli esercizi selezionati.

BTS Nirvana viene fornito con un'applicazione che semplifica la gestione del paziente. L'interazione è affidata a un'interfaccia *touch screen* funzionale e intuitiva.

Fig. 4 Screenshot dell'interfaccia di lavoro di Nirvana [34]

2.2.3 Componenti e accessori

Componenti Hardware	Dotazione standard:	Opzioni:
BTS NIRVANA Markerless Motion Analysis Unit	•	
Videoproiettore 4000 ansilumens	•	
Workstation All-in-one con touchscreen		•
Webcam	•	
Videoproiettore 6000 ansilumens	•	
Lettore Schede RFID		•

Software precaricato:	Dotazione standard	Opzioni:
Gestione anagrafica pazienti	•	
Progettazione e gestione terapia	•	
Pacchetto esercizi standard	•	

Esercizi aggiuntivi		•

Accessori:	Dotazione standard	Opzioni:
Supporto parete o soffitto	•	
Schermo LCD o plasma		•
KIT Trigger per analisi EMG		•
KIT Trigger per analisi VIDEO		•

Supporto e training:	Dotazione standard	Opzioni:
Installazione, prima formazione e start-up delle attività	•	
Helpdesk - 3 mesi	•	
Supporto e assistenza all inclusive		•

2.2.4 Area di lavoro

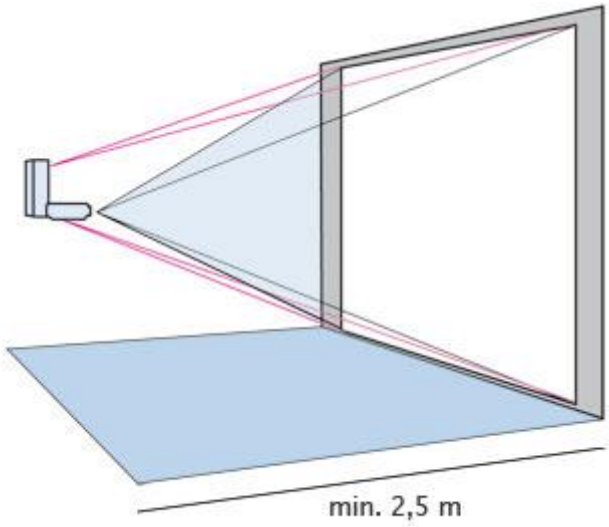


Fig. 5 Area di lavoro del sistema Nirvana [34]

La distanza tra l'unità BTS Nirvana e la proiezione deve essere almeno di 2,5 m per una proiezione a parete e di 2,8 m per una proiezione a pavimento.
Proiezioni su tavolo o su altre superfici possono essere ottenute anche da distanze inferiori.

Capitolo 3

Motion capture

3.1 Introduzione

Lo studio del movimento umano prevede la misura di variabili che descrivono la cinematica e la dinamica dei segmenti anatomici. La branchia della scienza che si occupa di indagare queste grandezze è l'analisi del movimento, in particolare la disciplina nota come *motion capture*, spesso indicata con l'abbreviazione *MoCap*,studia i meccanismi per l'acquisizione e il processo stesso di acquisizione del movimento.
L'analisi rappresenta uno strumento cardine per gli studi nel campo della biomeccanica e viene usato tradizionalmente come supporto diagnostico nelle patologie muscolo scheletriche.

Essa riveste una considerevole importanza in quanto permette di:
- definire il livello di limitazione funzionale e di disabilità conseguente alla patologia e il suo evolversi con la
crescita/invecchiamento dell'individuo;
- valutare e quantificare gli effetti dei diversi trattamenti e monitorare tali effetti nel tempo;
- contribuire alla pianificazione di un trattamento riabilitativo permettendo la stesura di un programma personalizzato che consenta di verificare oggettivamente, lo stato clinico del paziente prima, durante e alla fine del trattamento stesso.

In accordo con la definizione appena data, scopo della *MoCap* è dunque quello di "digitalizzare" il movimento di un soggetto,fornendone una rappresentazione matematica quantitativa che renda il movimento stesso facilmente fruibile come input per successivi studi e successive elaborazioni.

Le variabili cinematiche, su cui si basa il *MoCap* (posizione, velocità, accelerazione), sono ottenute tramite i sistemi di analisi del movimento, mentre le variabili dinamiche si ottengono indirettamente tramite la misura della forze esterne agenti sul soggetto. Nell'analisi del cammino, ad esempio, le variabili dinamiche sono ricavate integrando la cinematica con le misure ottenute con piattaforme di forza.

Allo stato dell'arte, i sistemi di *MoCap* esistenti per l'analisi della cinematica di un soggetto si possono concettualmente dividere in due grandi categorie: le tecniche che sfruttano principi di tipo ottico e quelle invece che affrontano il tema con approcci diversi da quest'ultimo [3].
Per le tecniche "non ottiche", possiamo individuare diverse soluzioni: elettromeccanica, elettromagnetismo , approcci di tipo inerziale e acustico. A questa categoria appartengono elettrogoniometri e accelerometri che consentono di ottenere misure dirette di alcune delle variabili di interesse, presentando tuttavia lo svantaggio di richiedere il contatto tra la superficie del corpo e dispositivi elettrici. Inoltre, essendo questi dispositivi ingombranti, possono rappresentare un ostacolo al movimento stesso riducendone la naturalità.
I sistemi basati su sensori elettromagnetici sfruttano un generatore esterno di campo magnetico per fornire misure dirette della cinematica: dal segnale di un singolo sensore elettromagnetico aderente alla superficie di un segmento anatomico, si possono ottenere, infatti,i sei gradi di libertà (tre rotazioni e tre traslazioni) che ne descrivono posizione ed orientamento. Tuttavia l'elevata sensibilità a interferenze da oggetti ferromagnetici presenti nell'ambiente rende tali sistemi poco adatti e affidabili.
I sistemi basati su sensori acustici, invece,sono costituiti da ricevitori di onde sonore solidali con il laboratorio e sorgenti poste sul soggetto. Essi forniscono solo una stima indiretta delle variabili cinematiche e sono caratterizzati da problemi di interferenza legati all'incostanza della velocità del suono in aria e a fenomeni di eco difficilmente eliminabili.
Per questa serie di motivi i sistemi optoelettronici operanti nella gamma del visibile o del vicino infrarosso rappresentano tutt'oggi la soluzione tecnologica più diffusa e affidabile per la stima del movimento umano.

I sistemi *MoCap* che utilizzano quest'ultimo tipo di approccio sono basati sull'utilizzo di una o di un set di telecamere sincronizzate al fine di ricostruire in ogni istante la posa 3D di un soggetto in movimento [3].
Concettualmente, possiamo classificare i sistemi *MoCap* ottici a seconda del numero di punti di vista assunti durante l'acquisizione (sistemi monoculari o *multi-view*) e,a seconda dell'impiego o meno di
markers (siano questi attivi o passivi), distinguendo dunque due tipi di approccio: il *marker-based* e il *markerless (MMC markerless motion capture).*
Nei sistemi *marker-based* l'obiettivo è ricostruire la cinematica dei segmenti ossei attraverso l'individuazione della traiettoria 3D dei *markers*(piccoli oggetti sferici applicati alla superficie corporea del soggetto in movimento), mentre le tecniche markerless mirano a estrarre i parametri relativi al movimento a partire da *silhouettes* o altre caratteristiche del soggetto (come ad esempio i contorni) ricavabili dalle immagini [6] .
I principali motivi del successo di tali sistemi riguardano innanzitutto il ridotto ingombro sul soggetto [3], e la possibilità di acquisire, con opportuni setup, anche movimenti ampi come il cammino, la corsa o il ballo. Un altro importante vantaggio riguarda la flessibilità nella scelta della posizione e del numero dei *markers* a seconda del tipo di acquisizione, anche se sono stati sviluppati numerosi protocolli per definire il loro posizionamento (*Davis-Helen Hayes, CAST,VCM, SAFLo)* [3].
Infine, i sistemi ottici possono raggiungere frequenze di campionamento tendenzialmente più elevate rispetto alle altre tecniche citate, permettendo l'analisi anche di movimenti rapidi con una buona accuratezza: nei sistemi *marker-based* a *marker* passivi, ad esempio, l'errore di ricostruzione

della posizione dei *markers* con la tecnologia "*Smart*" sviluppata da *BTS* raggiunge l'ordine del decimo di mm.

3.2 Sistemi optoelettronici

3.2.1 Triangolazione

Il funzionamento delle tecniche *marker-based* si basa fondamentalmente sul principio di triangolazione: quando almeno due telecamere riprendono contemporaneamente un punto dello spazio 3D allora è possibile, a partire dalle immagini formatesi sui piani principali delle telecamere, ricostruire la posizione di quel punto tramite semplici retroproiezioni (Fig. 1).

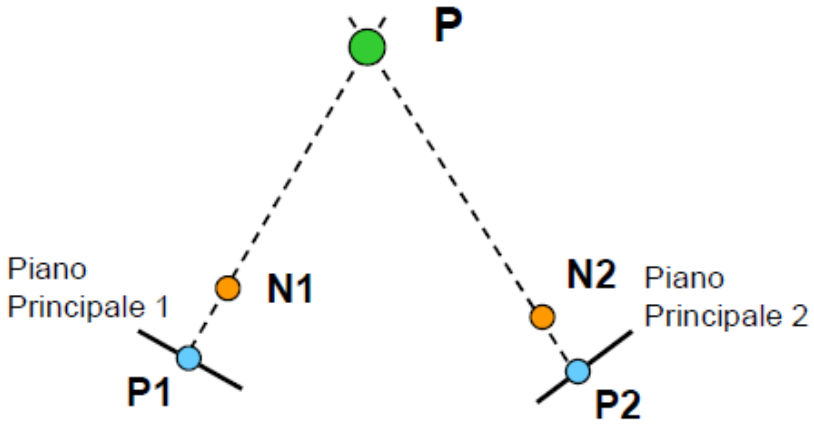


Fig. 1 Ricostruzione del la posizione del punto P, note le sue proiezioni P1 e P2 sui piani principal i del le telecamere e le posizioni N1 e N2 dei punt i nodal i del le telecamere [44]

Per ricostruire la posizione di un punto tramite triangolazione è necessario aver definito a priori un sistema di riferimento globale e che siano noti i parametri dell'intero sistema stereofotogrammetrico, ossia si conoscano i parametri esterni (posizione e orientamento dei piani principali delle telecamere) e i parametri interni (come lunghezza focale, coordinate del punto nodale, coefficienti di distorsione) delle telecamere. In altri termini, prima di poter effettuare le acquisizioni è necessario calibrare lo spazio all'interno del quale verrà eseguito il movimento.

La tecnica *MoCap* ottica *marker-based* è detta anche stereofotogrammetria, dove il prefisso stereo discende proprio dal principio appena esposto.
In questo caso a dover essere stimata è la posizione, rispetto al sistema di riferimento globale,di alcuni marcatori che vengono fatti aderire al corpo del paziente e dalla quale è possibile, nota la disposizione degli stessi, risalire alle grandezze cinematiche e anatomiche di interesse.

I marcatori utilizzati possono essere di due tipologie: attivi, ossia dotati di luce propria, o passivi cioè piccole sfere ricoperte di materiale rifrangente in grado di riflettere la luce proveniente da degli illuminatori esterni.
I sensori optoelettronici utilizzati per l'acquisizione sono telecamere che lavorano nello spettro infrarosso e integrano dispositivi per la rilevazione e trasduzione del segnale luminoso in un segnale elettrico, quali ad
esempio sensori *CCD* (*charged coupled device*) e *CMOS* (*complementary metal oxide semiconductor).*

Fig.2 esempio di acquisizione mediante stereofotogrammetria [41].

3.2.2 Elaborazione dell'immagine

Nel caso più comune in cui il sistema lavori con *markers* passivi, le telecamere sono costruite in modo da emettere esse stesse fasci di luce infrarossa proiettata nell'area di lavoro che viene riflessa dai *markers* passivi ricoperti da materiale riflettente e acquisita dalle camere sensibili a questa stessa lunghezza d'onda. Il segnale luminoso di ritorno proveniente dai *markers* sarà molto più intenso rispetto a un'eventuale riflessione di luce da parte del background, permettendo così una facile localizzazione dei *markers*.
La struttura di acquisizione nel caso di un sistema optoelettronico *marker-based* può essere quindi riassunto secondo lo schema seguente:

Fig. 3 Schematizzazione della struttura di acquisizione di un sistema stereo fotogrammetrico.

Dallo schema si nota come oltre alla struttura di acquisizione (composta da telecamere e illuminatori IR) il sistema di analisi prevede la presenza di un software atto a elaborare le informazioni rilevate dalle telecamere stesse al fine di fornire all'operatore la posizione istante per istante di ciascun *markers* presente nella scena.

In particolare ogni frame acquisito viene sottoposto ad una prima fase di segmentazione con lo scopo di estrarre dall'immagine la posizione 2D dei *markers* che servirà poi per effettuare il processo di triangolazione.
L'immagine viene inizialmente filtrata al fine di ridurre i gradienti di illuminazione presenti per poi procedere con la costruzione dell'istogramma delle frequenze nelle diverse classi di tonalità di grigio. L'istogramma assume un tipico andamento a doppia gaussiana dove il primo picco, molto vicino allo zero, indica tutta l'informazione proveniente dallo sfondo mentre il secondo rappresenta l'oggetto della nostra elaborazione ossia l'informazione proveniente dai *markers*.

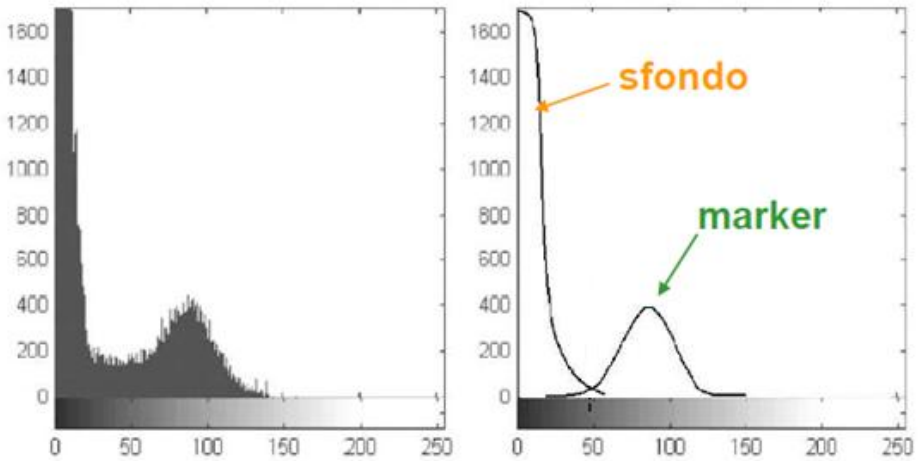


Fig. 4 Istogramma dell'immagine e sua approssimazione con due gaussiane [44]

Utilizzando una tecnica a mistura di gaussiane è possibile approssimare l'istogramma con due curve e calcolare quindi il valore della soglia ottima, come quel valore in corrispondenza dell'intersezione delle curve stesse, che permette di minimizzare gli errori di tipo I (falso positivo) e di tipo II (falso negativo).

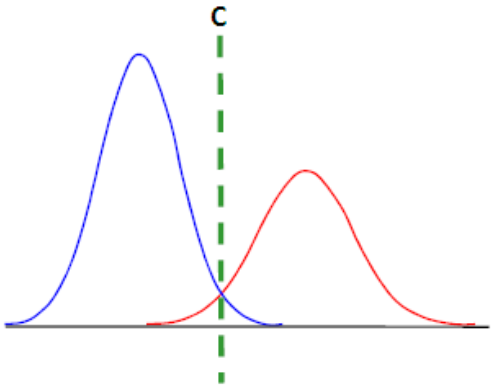


Fig. 5 Calcolo del valore ottimo della soglia come intersezione tra le due curve [44]

A questo punto andando a sogliare l'immagine di partenza è possibile ottenere quelli che vengono definiti *blobs* ossia la proiezione 2D dei *markers* sul piano immagine di cui si dovranno stimare le coordinate del centro.

Per far ciò è possibile procedere secondo diversi approcci.
In primis il centro del marcatore può essere stimato come baricentro dei pixel sopra la soglia (indipendentemente dai livelli di grigio). In particolare si procede al calcolo delle grandezze b_x e b_y, rispettivamente coordinate nel piano immagine lungo x e lungo y del centro del *blob*, secondo:

$$b_x = \frac{1}{N}\sum_{i=1}^{N} p_{xi} \qquad b_y = \frac{1}{N}\sum_{i=1}^{N} p_{yi}$$

Dove N è il numero di pixel sopra la soglia e p_{xi} è la coordinata lungo l'asse x del pixel i-esimo.
Si può procedere alla stime del centro del *marker* anche mediante una tecnica denominata "*circle fitting*" andando cioè ad evidenziare il bordo dell'area illuminata (sopra soglia) e approssimandolo con una circonferenza.
In genere, prima di procedere al calcolo del centro dei diversi *markers*, è buona norma effettuare un processo cosiddetto di *blob-analysis*. Questa tecnica ha come obiettivo quello di estrarre dall'immagine solo le zone di effettivo interesse, eliminando false misure dovute principalmente ai riflessi.
Si procede quindi ad una prima verifica sulle dimensione del *blob* andando ad eliminare tutte quelle proiezioni che o perché troppo grandi o perché troppo piccole risultano incompatibili con la reale dimensione del *blob*. In un secondo momento viene effettuato un controllo sulla forma assunta dal *blob*. In tal caso, l'ellissoide che approssima il marcatore può essere stimato a partire dalla matrice

di covarianza sperimentale delle posizioni dei *pixel* occupati dell'immagine del marcatore rispetto al baricentro

$B=(b_x, b_y)^T$: dall'analisi delle componenti principali della matrice così ottenuta è possibile risalire al fattore di forma del *blob* che deve essere compatibile con la proiezione di una sfera.

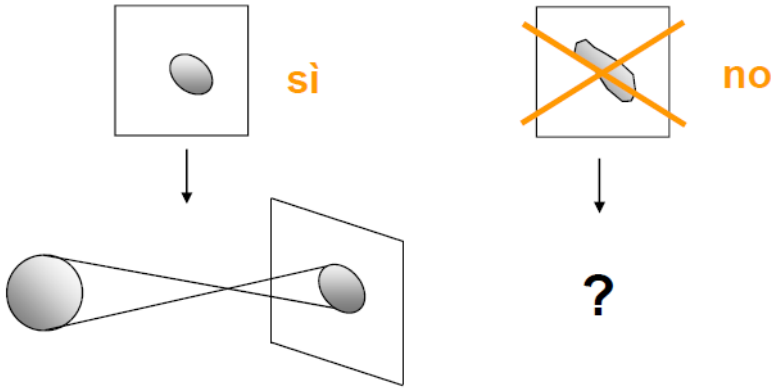


Fig. 6 Analisi del blob in base alla forma dello stesso. A sinistra immagine compatibile con la proiezione di una sfera. A destra proiezione non compatibile con quella di una sfera. [44]

In questo modo è possibile eliminare tutta quell'informazione che attraverso la semplice sogliatura verrebbe invece mantenuta e processata. Un esempio è presentato dal fenomeno dell'occlusione parziale: in tal caso l'immagine del marker non è compatibile con la proiezione di una sfera e il dato viene quindi correttamente scartato in quanto condurrebbe ad un calcolo errato del baricentro.
Un metodo alternativo all'identificazione del blob consiste nel procedere mediante un processo di correlazione.
In tal caso viene creata una maschera (*kernel*), di forma e dimensioni pari a quelle attese dell'immagine del *marker*, con la quale si va a spazzolare l'intera immagine alla ricerca di quelle aree che presentano un fattore di correlazione superiore ad una determinata soglia.
A questo punto per ogni *marker* sono note le coordinate 2D in ciascun frame acquisito dalle diverse telecamere dai quali sarà possibile risalire alla posizione 3D previa calibrazione del sistema adottato.

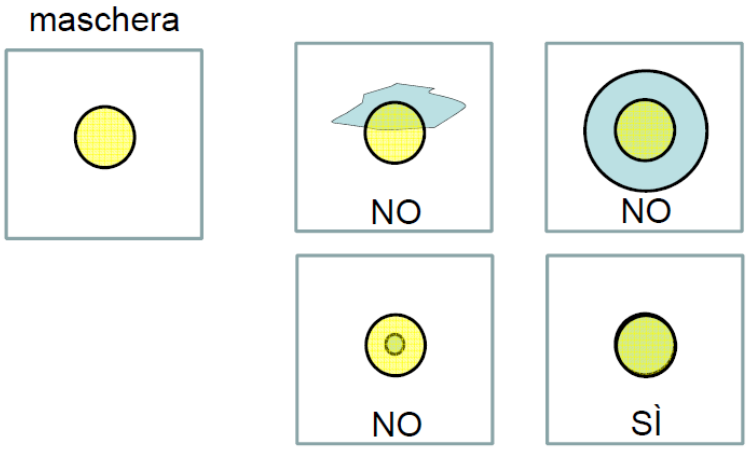


Fig. 7 Esempio di filtraggio con kernel di correlazione.[44]

3.2.3 Calibrazione

3.2.3.1 Basi della calibrazione

Come introdotto in precedenza,per ricostruire la posizione 3D di un punto tramite triangolazione è necessario aver definito a priori un sistema di riferimento globale e aver effettuato la calibrazione delle telecamere che compongono il sistema di acquisizione.

Fig. 8 Esempio di setup stereofotogrammetrico per l'analisi del movimento. [41]

Si procede dunque alla determinazione dei valori assunti dai parametri geometrici intrinseci ed estrinseci necessari per la successiva fase di ricostruzione tridimensionale. Tali parametri intrinseci sono definiti in letteratura (*Abdel-Aziz e Karara, 1971; Tsai, 1987; Weng et all., 1992*) come: lunghezza focale, coordinate del punto principale e coefficienti di distorsione; mentre quelli esterni sono atti a descrivere la posizione del sistema di riferimento della telecamera rispetto al sistema di riferimento assoluto.

Queste grandezze permettono di descrivere tre diverse trasformazioni: una prima operazione permette di trasformare le coordinate del sistema assoluto in coordinate relative al sistema della telecamera localizzato nel suo centro di prospettiva, una seconda trasformazione prospettica restituisce coordinate 2D a partire dalle coordinate 3D acquisite, e infine una terza trasformazione 2D dal sistema di coordinate riferite al piano immagine al sistema di riferimento del sensore (vedi figura 9).

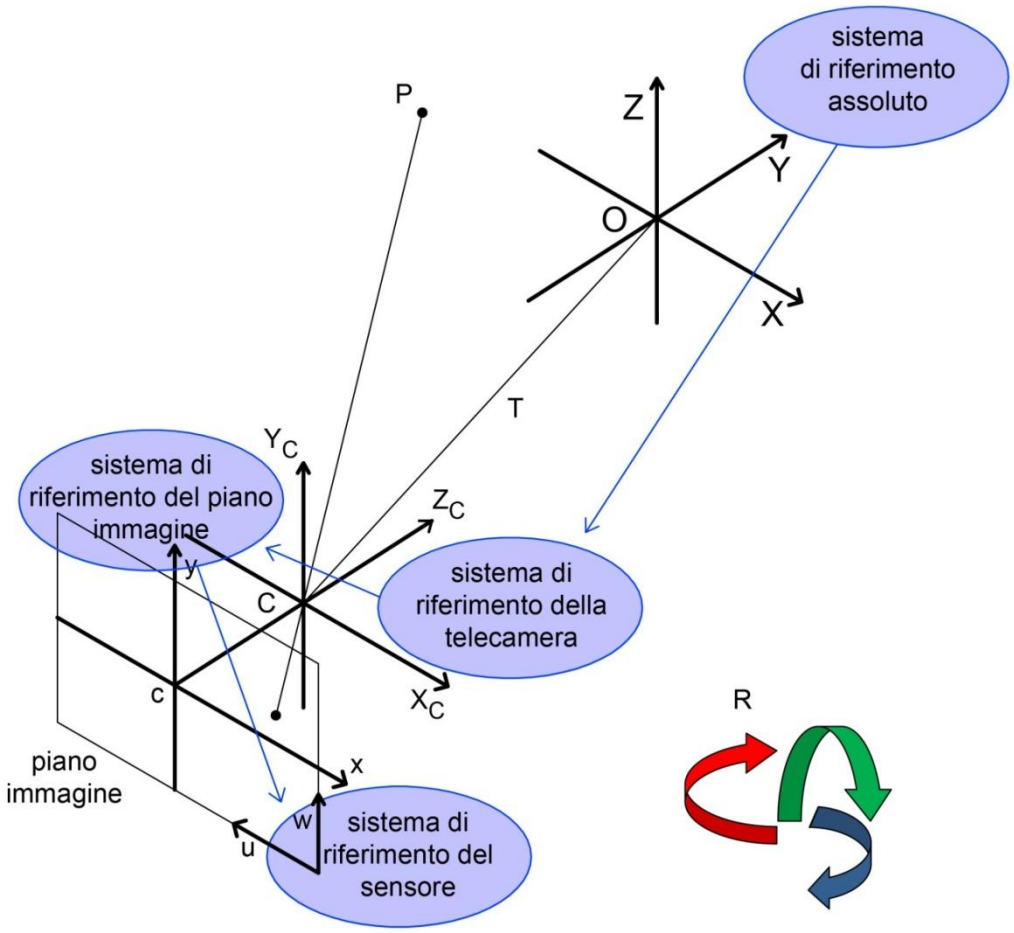


Fig. 9 La proiezione sul piano immagine di un punto P nello spazio 3D: 1) trasformazione rigida (T,R) dal riferimento assoluto di coordinate *(O,X,Y,Z)* al sistema di coordinate della telecamera *(C,X_c,Y_c,Z_c);* 2) trasformazione prospettica dal sistema di riferimento della telecamera al sistema di riferimento del piano immagine (*c,x,y);* 3) trasformazione affine 2D dal sistema di riferimento del piano immagine al sistema di riferimento *(u,v)* del sensore. Il vettore T e la matrice di rotazione R esprimono rispettivamente la traslazione e l'orientamento del sistema di riferimento della camera rispetto al sistema di riferimento assoluto.

Il primo passo verso la calibrazione del sistema è la definizione di un modello matematico dello stesso.
A tale scopo, le tecniche di calibrazione proposte in letteratura fanno generalmente riferimento al modello della telecamera a foro stenopeico (*pin-hole*) la quale specificità sta nel trascurare la presenza dell'ottica che, in questo caso, viene modellata come un foro di diametro infinitesimo posto nel centro della prospettiva.

Un raggio di luce proveniente dal punto P dello spazio attraversa il foro e incide sul piano immagine in un punto *p*.

Considerando una terna cartesiana con l'origine nel centro di prospettiva della camera e l'asse *Z* coincidente con l'asse ottico, per similitudine si ottiene la seguente relazione tra le coordinate (x_p,y_p), espresse nel sistema di riferimento della camera (C, X_c, Y_c, Z_c), dei punti *p* e *P* rispettivamente:

$$\begin{pmatrix} x_p \\ y_p \end{pmatrix} = -f \begin{pmatrix} \frac{1}{Z_p} & 0 \\ 0 & \frac{1}{Z_p} \end{pmatrix} \begin{pmatrix} X_p \\ Y_p \end{pmatrix} \quad [1]$$

Dove *f* è la lunghezza focale, ovvero la distanza del centro di prospettiva della camera dal piano immagine.

Ricorrendo all'uso delle coordinate omogenee, per cui il punto di coordinate (x_p,y_p) corrisponde alla retta $(\lambda x_p, \lambda y_p, \lambda)$, la formula può essere riscritta come:

$$\lambda \begin{pmatrix} x_p \\ y_p \\ 1 \end{pmatrix} = \begin{pmatrix} -f & 0 & 0 & 0 \\ 0 & -f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_p \\ Y_p \\ Z_p \\ 1 \end{pmatrix} = MP_p \quad [2]$$

Noto che, per convenzione, il sensore della telecamera possiede un proprio sistema di riferimento centrato sull'angolo inferiore destro del piano dell'immagine, nel quale sono espresse le coordinate 2D in uscita dalla camera, si può affermare che, date le dimensioni orizzontale e verticale del pixel (k_u, k_v) e le coordinate del punto principale (u_0, v_0), le coordinate (u, v) del punto p nel sistema di riferimento del sensore sono:

$$p = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{k_u} & 0 & u_0 \\ 0 & \frac{1}{k_v} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y_p \\ 1 \end{pmatrix} = H\overline{p} \quad [3]$$

Infine, dato che il punto *P* è originariamente espresso rispetto al sistema di riferimento assoluto (O,X,Y,Z), si ha:

$$P_p = \begin{pmatrix} X_p \\ Y_p \\ Z_p \\ 1 \end{pmatrix} = \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix} \begin{pmatrix} X_a \\ Y_a \\ Z_a \\ 1 \end{pmatrix} = DP \quad [4]$$

dove il vettore T e la matrice di rotazione R esprimono rispettivamente la traslazione e l'orientamento del sistema di riferimento della camera rispetto al sistema di riferimento assoluto e P_p è il punto P espresso nel sistema di riferimento della telecamera (C, X_c, Y_c, Z_c).

In conclusione il modello della telecamera *pin-hole* può essere rappresentato come una matrice A di dimensioni 3x4 ottenuta come composizione delle tre precedenti trasformazioni:

$$A = HMD = \begin{pmatrix} \frac{1}{k_u} & 0 & u_0 \\ 0 & \frac{1}{k_v} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -f & 0 & 0 & 0 \\ 0 & -f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix} \quad [5]$$

con $\lambda p = AP$ che rappresenta la formulazione in coordinate omogenee delle equazioni di col linearità.

Come intuibile, quello a *pin-hole* è un modello ideale non realizzabile in quanto l'intensità della luce che attraversa il foro e che giunge sugli elementi fotosensibili del piano immagine è troppo piccola per essere rilevata, per cui è necessario allargare il foro (diaframma) e ricorrere ad un'ottica per mettere a fuoco l'immagine.

Tuttavia tale soluzione, ricorrente al gruppo ottico, introduce una serie di deformazioni dell'immagine che è possibile classificare in distorsioni cromatiche, che degradano la qualità o il dettaglio dell'immagine, e in distorsioni geometriche, che causano lo spostamento di ogni punto dell'immagine dalla sua posizione nominale predetta dal modello ideale di proiezione prospettica secondo una specifica funzione non nota a priori. La stima delle distorsioni prevede la definizione di un modello per questa funzione.

In generale, ci sono due possibili modi per rappresentare le distorsioni geometriche: il primo, attraverso un semplice modello radiale che assume che i punti vengano allontanati o avvicinati al centro dell'immagine in accordo ad una funzione polinomiale della distanza fra il centro di distorsione e il punto, la seconda attraverso un modello tangenziale secondo il quale i punti subiscono uno spostamento in direzione tangente alla direzione radiale.

Per la stima delle distorsioni si possono utilizzare due approcci differenti:

- stima a priori dei parametri di distorsione, indipendente dalla calibrazione dei parametri esterni della camera, utilizzando un oggetto di geometria nota (struttura planare);
- calcolo delle distorsioni in parallelo con i parametri geometrici durante la fase di calibrazione delle telecamere.

Il primo approccio implica la stima di un operatore di trasformazione G tale che:

$$m_c = G(m_d) \quad [6]$$

Dove $m_c = \{(x_c, y_c)\}$ e $m_d = \{(x_d, y_d)\}$ sono rispettivamente le coordinate dei punti (in 2D) corretti e distorti.

La relazione in eq. 6 è detta modello diretto poiché mappa coordinate distorte in corrette e non può essere definita a meno della conoscenza di un numero minimo di coppie di punti (punti distorti della griglia di calibrazione acquisita $\{(x_d, y_d)\}$ e nominali $\{(x_n, y_n)\}$), qualsiasi sia il numero dei parametri utilizzati nel calcolo di G. Dato un punto di coordinate distorte, tale modello, fornisce la stima del corrispondente punto corretto, quindi data la regione di pixel distorta che rappresenta il marcatore, se ne calcola prima il baricentro per poi correggerlo tramite la funzione *G*.
Il modello diretto però non può essere considerato adatto alla correzione dell'immagine in caso di forte distorsione ai bordi: fornisce infatti, in suddetto caso, risultati poco accurati.

Una soluzione a questo tipo di problema, sebbene più lenta, è rappresentata dalla correzione dell'intera immagine anziché dei soli baricentri dei marcatori: si tratta dunque di correggere le distorsioni sull'immagine originale prima dell'identificazione.

Per il calcolo della funzione di correzione,l'approccio e funzione globale (*Gronenschild, 1997*) non necessita di un modello predefinito di distorsione ma prevede l'utilizzo di un polinomio come il seguente:

$$G(p, m_p) = \begin{bmatrix} a_{0x} + a_{1x}x_d + a_{2x}y_d + a_{3x}x_d y_d + a_{4x}x_d^2 + a_{5x}y_d^2 + \ldots \\ a_{0y} + a_{1y}x_d + a_{2y}y_d + a_{3y}x_d y_d + a_{4y}x_d^2 + a_{5y}y_d^2 + \ldots \end{bmatrix} \quad [7]$$

Dove il vettore p contiene i due insiemi di parametri $\{a_x\}$ e $\{a_y\}$.

In alternativa sono state proposte come funzioni di correzione (*Cerveri et all., 2002*) funzioni locali e reti neurali. Esse garantiscono la possibilità di correggere le distorsioni localizzate in particolari zone dell'immagine. Una considerazione importante riguarda il numero di gradi di libertà del problema, ovvero la differenza tra il numero di coppie di punti impiegati nella stima (equazioni) e il numero di parametri stimati (incognite). Ovviamente il numero di equazioni deve essere maggiore o uguale al numero di incognite ma, per evitare overfitting sui punti di controllo, e quindi ottenere un migliore comportamento tra i punti di controllo stessi, è bene che la differenza sia consistente (ad es. 200 punti (equazioni) per stimare 40 parametri). Il secondo approccio alla stima dei parametri di distorsione prevede che essi siano determinati insieme ai parametri geometrici delle telecamere durante la fase di calibrazione 3D.

3.2.3.2 Determinazione dei parametri di calibrazione

La calibrazione del sistema, ovvero la determinazione dei parametri geometrici delle telecamere, implica in genere l'utilizzo di un insieme di punti di controllo distribuiti internamente al volume di calibrazione.

Normalmente, per la determinazione dei parametri, vengono utilizzati metodi di soluzione in forma chiusa delle equazioni di collinearità come la DLT (*direct linear transformation*), basata sulla soluzione di una sistema lineare assumendo note le coordinate dei punti di controllo. Recentemente

si sono studiati metodi basati sulla geometria *epipolare* (*Luong e Faugeras, 1996*) che utilizzano parimenti soluzioni lineari ricavabili, però, a prescindere dalla conoscenza delle coordinate dei punti di controllo nello spazio tridimensionale (*Hartley, 1992; Hartley, 1997; Cerveri et al., 1998*). Quest'ultimo metodo è stato utilizzato per risolvere il problema della calibrazione di due telecamere.

Considerata, infatti, la rotazione *R* e le traslazione *T* tra i due sistemi di coordinate delle telecamere è possibile scrivere, per il punto *P*, espresso nel sistema di riferimento della prima camera (vedi Fig. 10) la seguente relazione:

$$P' = R\,(P - T) \qquad [8]$$

Dove *P'* è lo stesso punto *P* espresso nel sistema di riferimento della camera 2.
Poiché i tre vettori *PC, PC'* e *CC'* giacciono sullo stesso piano (piano epipolare), vale la seguente relazione:

$$\overline{PC'} \cdot \overline{CC'} \Lambda \overline{PC} = P' \cdot (T\Lambda)P = 0 = R(P - T)(T\Lambda)P = RP(T\Lambda)P \qquad [9]$$

Con:

$$(T\wedge) = \begin{pmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{pmatrix} \qquad [10]$$

Dalle equazioni 3, 4 e 5 si ottiene:

$$(R^{-1}K'^{-1}\,p') \cdot (T\wedge)K^{-1}p = p'^{T}\,(K'^{T})^{-1}R(T\wedge)K^{-1}p = 0 \qquad [11]$$

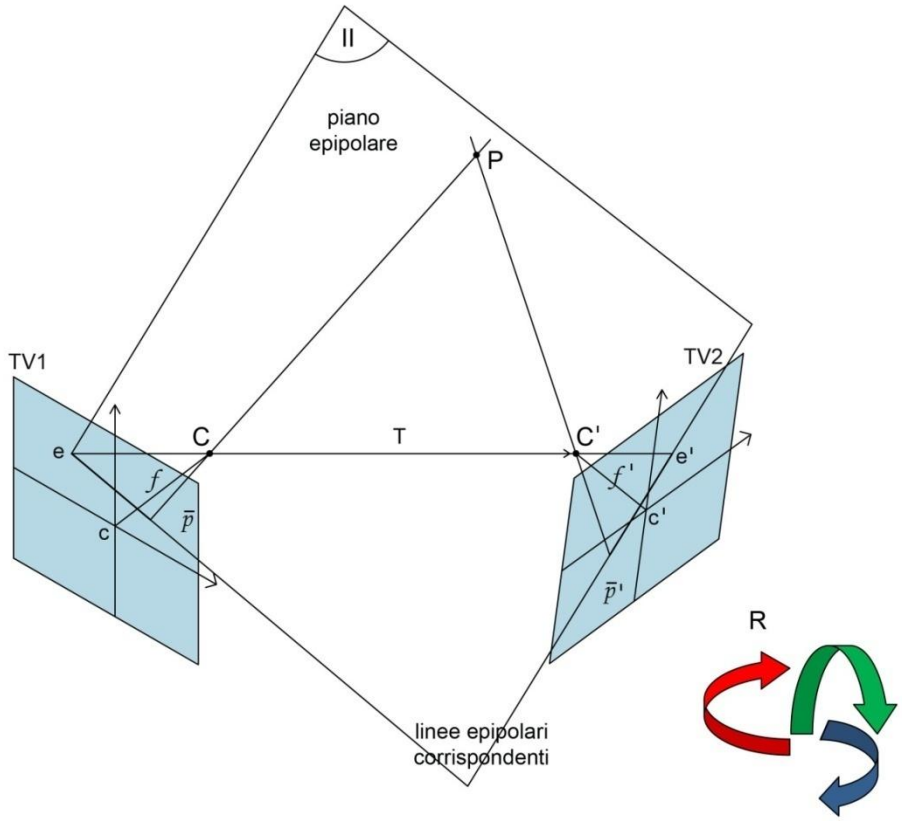


Fig. 10 Geometria epipolare per due telecamere. Il vettore T e la matrice di rotazione R rappresentano la traslazione e l'orientamento relativo tra il sistema di riferimento della camera 2 e quello della camera 1. I punti e e e' rappresentano gli epipoli delle due telecamere.

Dove p' e p sono punti corrispondenti nella trasformazione epipolare

$$K = \begin{pmatrix} \frac{-f}{k_u} & 0 & u_0 \\ 0 & \frac{-f}{k_v} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \qquad [12]$$

L'eq. 11 può essere sinteticamente espressa come segue:

$$p'^{T} Fp = 0 \qquad [13]$$

Con $F=(K'^{T})^{-1}\ R(T\hat{})K^{-1}$matrice fondamentale che contiene i parametri interni delle due telecamere e i parametri esterni relativi tra le due camere.
Assumendo noti i parametri interni delle due telecamere l'eq. 13 viene trasformata in

$$\overline{p}'^{T} E\bar{p} = 0 \quad con \quad E = K'^{T} FK \qquad [14]$$

Dove E è la matrice essenziale e $\bar{p}'$ e $\bar{p}$ sono i punti corrispondenti le cui coordinate sono espresse nel sistema di riferimento dei rispettivi piani immagine (coordinate linearizzate). La matrice di rotazione R e la traslazione T sono ricavate dalla matrice essenziale tramite la soluzione di un sistema lineare omogeneo e fattorizzazioni matriciali (*Hartley, 1997*).

Dato un insieme di punti corrispondenti (almeno 8), la matrice F può essere determinata risolvendo un sistema lineare omogeneo. Tuttavia, la geometria *epipolare* intrinsecamente non consente di risolvere completamente il problema della calibrazione di un sistema di telecamere: infatti, oltre ai 6 parametri esterni di rototraslazione relativa, essa consente di calcolare al massimo altri due parametri (ad es. le due lunghezze focali : f ed f').

Esistono diversi procedimenti atti al calcolo dei parametri necessari:i due parametri appena citati, le lunghezze focali, possono essere calcolate a partire dalla matrice fondamentale mentre, in linea di principio, tutti gli altri parametri (u_0, v_0, u'_0, v'_0, k_u, k_v, k_u', k_v') devono essere determinati a priori con altre procedure.

Più recentemente Cerveriat al. Hanno provato che i punti principali possono essere stimati dalla matrice fondamentale con una valutazione a posteriori dell'accuratezza della calibrazione. In particolare, data una coppia di punti principali e risolta la geometria epipolare, l'accuratezza di ricostruzione dà un indice della bontà della scelta.

In *Cerveri*(2001) tale metodo è stato integrato in una procedura di ottimizzazione basata su strategie evolutive che sfruttano le potenzialità degli algoritmi genetici. In questo approccio, a partire da una scelta plausibile della posizione dei punti principali, sui due piani immagine, l'algoritmo di calibrazione effettua una ricerca guidata, nello spazio delle 4 coordinate, della quaterna che fornisce la migliore accuratezza 3D, misurata come errore sulla distanza di due punti ricostruiti e posti alle estremità di una barra.

3.2.3.3 Calibrazione operativa nel sistema BTS

Il sistema BTS per procedere operativamente alla calibrazione del sistema stereo prevede una procedura di calibrazione (denominata BTS *THOR2*) che si articola in due fasi successive e distinte.

Nel primo step denominato "*Axes*" si procede all'acquisizione di un particolare tool costituito da tre barre in fibra di carbonio disposte a formare una terna ortogonale. Su ognuna di tali barre vengono posizionati dei *markers* posti secondo una disposizione nota ben definita: in particolare lungo un'asse vengono applicati 4 marker, 3 lungo un'altra e 2 lungo la rimanente. In questo modo una

volta acquisite ed elaborate le immagini secondo le tecniche prima esposte, sarà possibile procedere all'identificazione dei *markers* risalendo quindi alla posa del tool (e quindi alla direzione dei suoi assi) permettendo la definizione di un sistema di riferimento globale assoluto. La costruzione del tool di calibrazione da parte di BTS viene effettuato basandosi su di una dima costruita mediante controllo numerico in modo da poter garantire un certo grado di accuratezza nella forma del tool e sulla disposizione dei marker.

Nella seconda fase "*Wand*", una bacchetta contenente due marker posizionati a distanza nota l'uno dall'altro, viene mossa dall'operatore all'interno del volume di interesse in modo da utilizzare l'informazione sull'interdistanza tra i marker per raffinare la stima dei parametri calcolati in precedenza e correggere i fenomeni di distorsione. Inoltre questa fase permette al software di individuare il volume totale di acquisizione in cui il paziente effettivamente si muoverà andando quindi ad ignorare tentativi di calibrazione su dati provenienti dal di fuori di tale area.

Procedendo secondo quanto appena esposto per ogni camera vengono calcolati posizione e orientamento, nonché la lunghezza focale, la posizione del centro ottico e i parametri di distorsione che possono quindi essere corretti. I parametri di calibrazione appena stimati per ogni telecamera, permettono di definire una trasformazione di coordinate dal sistema di riferimento assoluto, definito durante la fase di "*Axes*", al sistema di riferimento della telecamera stessa localizzato nel centro di prospettiva rendendo possibile la stima della posizione 3D di un qualsiasi punto dello spazio inquadrato in contemporanea da almeno due camere.

3.2.4 Errori in stereofotogrammetria

L'accuratezza con la quale viene stimata la posizione di un punto nello spazio attraverso il sistema stereo fotogrammetrico è di fondamentale importanza per l'utilizzo in campo clinico e biomeccanico.
Le tipologie di errore che possono intervenire a compromettere tale accuratezza sono multiple e possono essere classificate secondo quanto segue:

- errori strumentali: si tratta di errori intrinseci al sistema di misura e possono essere di due tipi, sistematici o casuali;
- errori nella determinazione delle coordinate locali dei punti di repere anatomico: tali errori consistono nella dislocazione dei punti di repere anatomico e si propagano al calcolo della posa dei sistemi di riferimento anatomico e quindi agli angoli articolari;
- errori derivanti da Artefatti da Tessuto Molle (ATM): la cute, il tessuto adiposo e i muscoli, interposti tra i marcatori e i segmenti ossei sottostanti, fanno sì che durante l'esecuzione del compito motorio, si verifichi un movimento relativo tra queste due entità.

Tutte le tipologie di errori appena illustrate fanno si che, durante l'acquisizione, vengano registrati movimenti spuri dei marker che vanno ad influire sulla bontà dei dati ottenuti.
È importante sottolineare fin da subito che mentre gli errori strumentali possono essere facilmente quantificati ed eventualmente compensati, gli altri due tipi di errori sono casuali e quindi molto più difficili da quantificare e al limite compensare.

Nell'indagare le differenti tipologie di errori, in relazione allo specifico lavoro di tesi svolto, è importante sottolineare il contributo degli errori di tipo strumentale che insorge nel lavorare con dispositivi che forniscono dati la cui accuratezza e precisione non sono note all'operatore.

3.2.4.1 Errori strumentali

Si tratta di errori intrinseci nel sistema di misura stereo fotogrammetrico e fanno sì che anche in condizioni statiche, le coordinate ricostruite dei marcatori non risultano essere tempo invarianti. [44].
Questi errori necessariamente si propagano al calcolo della posa del sistema tecnico e quindi della cinematica articolare andando a compromettere la bontà dei risultati dell'analisi.
Gli errori strumentali possono essere di due tipi:

1. errori sistematici;
2. errori casuali.

I primi sono errori generalmente in bassa frequenza sempre associati ad un modello del sistema di misura di validità limitata che può derivare o da inaccuratezze nella calibrazione del sistema (cattiva stima dei parametri del modello) o da non linearità non considerate in fase di calibrazione (si parla allora di modello inadeguato ed è l'esempio del caso della distorsione dovuto all'ottica delle telecamere).
Ad esempio, la distorsione, generata dalla forma dell'ottica, è un fenomeno sempre presente, in forma più o meno marcata che deve quindi essere tenuto in considerazione da parte del modello di sistema.
L'entità degli errori sistematici dipende di conseguenza dall'ampiezza del campo di lavoro e dalla posizione che il *marker* assume al suo interno. Un marker posizionato di fronte all'obiettivo sarà meno soggetto alla presenza di distorsioni rispetto ad un marcatore in posizione defilata rispetto al campo visivo della camera.

È buona norma quindi, prima di ogni sessione sperimentale, procedere alla quantificazione degli errori strumentali mediante prove appositamente sviluppate, al fine di stimare quanto la loro presenza possa influenzare la misura stessa. In questo modo è possibile determinare le caratteristiche dell'errore stesso e decidere se procedere o meno all'adozione di tecniche di compensazione.

Risulta evidente che per quanto riguarda la componente sistematica di questa tipologia d'errore la calibrazione del sistema streofotogrammetrico deve essere tale da minimizzarne la presenza, pertanto deve essere eseguita in modo accurato dall'operatore ed eventualmente ripetuta più volte per mantenere inalterate le prestazioni del sistema.

Per quanto concerne invece gli errori casuali, si tratta di errori in frequenze maggiori prodotti in genere da rumore elettronico (*flickering*) legato all'imprecisione con cui l'immagine dei marcatori è convertita in punto immagine, e dalla quantizzazione intrinseca al processo di digitalizzazione, che trasforma le coordinate immagine del marcatore in valori numerici.

È interessante far notare come lavorando con dispositivi di cui non è nota l'elaborazione interna dei dati la presenza di errori strumentali sia da ritenersi una delle maggiori fonti di errore ed incertezza nella fase di calibrazione.

3.2.4.2 Tecniche di compensazione degli errori strumentali

Gli errori strumentali si presentano sottoforma di rumore additivo (e non correlato) a banda larga sovrapposto ai dati del movimento umano caratterizzati in genere da uno spettro di frequenza limitato.

In base alle caratteristiche statistiche di tali errori è possibile adottare diverse tecniche di filtraggio nel dominio del tempo (interpolazioni *spline*, tecniche di *smoothing*...) o delle frequenze (es. filtro di *Butterworth* ordine = 1- 4, cutoff = 3-6 Hz) utilizzando procedure di ottimizzazione per il *self-tuning* della frequenza di taglio.

Queste tecniche assumono però di essere in presenza di segnali stazionari, cosa non del tutto realistica in contesti come l'attività sportiva, dove sono presenti rapide variazioni del contenuto in frequenza dei segnali e fenomeni di impatto che comportano brusche variazioni nella traiettoria dei *marker*, costringendo il più delle volte ad adottare tecniche di filtraggio tempo-frequenza.

Si può quindi supporre che, seguendo le istruzioni fornite dal costruttore, la calibrazione del sistema operi bene nel correggere gli errori strumentali di carattere sistematico e che siano disponibili buone tecniche di filtraggio in grado di ridurne la componente casuale .

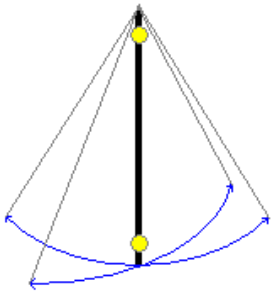
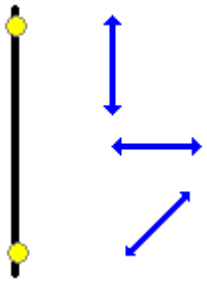
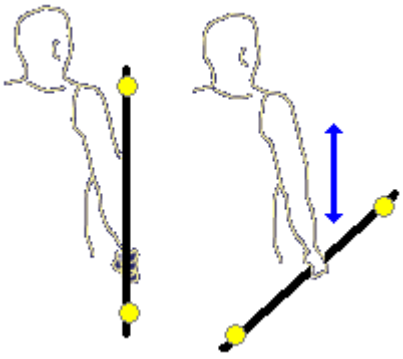
Nonostante ciò le prestazioni del sistema, in termini di accuratezza e precisione, possono dipendere da tutti quei parametri associati al *set up* del laboratorio specifico, come il numero e il posizionamento delle telecamere, le dimensioni del volume di lavoro e dell'oggetto di calibrazione utilizzato e ancora la cura dell'utilizzatore durante la procedura di calibrazione.

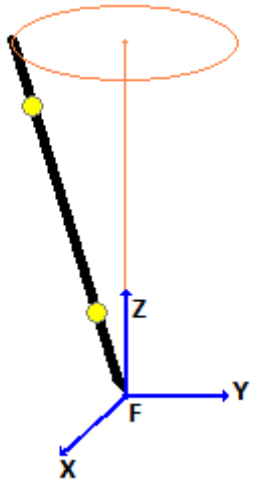
Comprendere e quantificare la bontà del sistema di analisi è essenziale per una corretta stima della cinematica articolare. È quindi di notevole importanza operare una stima ad hoc dell'accuratezza e della precisione effettive nella misura della posizione dei marker tramite l'esecuzione di *spot – checks*, ossia prove per la verifica della bontà del sistema e del mantenimento delle sue prestazioni.

Gli esempi classici di questo tipo di test sono:

- misura della distanza tra marcatori: metodo che mira a rilevare il mantenimento della distanza relativa tra marcatori montati su un supporto rigido guidato dalloperatore su tutto il volume di misura, come indice del buon mantenimento delle caratteristiche del sistema. In tabella sono riportate le principali tipologie di test appartenenti a questa classe;
- misura dello spostamento tra marcatori: si utilizza spesso un dispositivo motorizzato in grado di imporre un moto noto ai marcatori all'interno del campo di misura. Si tratta di

tecniche efficaci per la determinazione di problemi specifici ma generalmente meno adatti all'applicazione sistematica.

Nome	Immagine	Caratteristiche
Pendulum Test		Un pendolo rigido su cui sono attaccati due marcatori in posizione nota viene fatto oscillare su due piani ortogonali. Le coordinate tridimensionali dei marcatori vengono acquisite per almeno due oscillazioni complete. Il test può essere effettuato sia nel centro che in periferia del volume calibrato.
Full volume test		Un marcatore sferico è fissato ad ogni estremo di un'asta rigida. L'asse verticale dell'asta è allineato approssimativamente col l'asse verticale del laboratorio, quindi l'asta viene mossa parallelamente ad ogni asse, attraverso tutto il volume di misura mantenedo la velocità di movimento costante.
Walking test		Molto simile al precedente. L'operatore cammina in modo tale che i suoi piedi si posino sequenzialmente su quattro segni sul pavimento, tenendo in mano una barra con diverse orientazioni, poi si ferma e mantiene la posizione eretta.

MAL test		Su un'asta rigida sono montati due marcatori sferici (A,B). un punto Target T coincidente con la punta dell'asta, viene assunto in posizione nota rispetto alla posizione dei due marcatori. Questo viene posizionato in un punto fisso (F) sul pavimento all'interno del volume calibrato. Vengono fatte alcune acquisizioni con l'asta ferma (test statico) e mentre viene fatto ruotare intorno a T, muovendo l'altra estremità lungo una traiettoria pseudo circolare o due archi ortogonali. La manovra dinamica viene eseguita manualmente ad una velocità assimilabile a quella dell'esercizio fisico in esame.

Tab. 1 Principali test di *spot-checks*

Gli errori descritti finora sono considerati di tipo additivo e si sovrappongono alle coordinate globali dei *marker* in uscita dal sistema stereofotogrammetrico col risultato di ottenere delle traiettorie affette da errore. In particolare per il marker i-esimo:

$$\hat{p}_i(t)_{glo} = p_i(t)_{glo} + e(t)_{glo}$$

Per come viene definito l'errore esso si propaga anche al calcolo delle matrici di orientamento e del vettore di posizione dei *marker* nel sistema di riferimento tecnico e anatomico introducendo un certo errore nella stima di tali grandezze.

Esistono diversi metodi di analisi finalizzati alla minimizzazione della propagazione di tali errori:

- Stimatori non ottimi: queste procedure non tengono in considerazione la presenza di eventuali errori strumentali e ricostruiscono il sistema tecnico tramite semplici operazioni geometriche sulle posizioni dei *marker* considerate esatte.
- Stimatori ottimi ai minimi quadrati: questi metodi compensano la deformabilità del *cluster*, dovuta all'errore fotogrammetrico, sfruttando la ridondanza ossia la sovrabbondanza nel numero di marcatori adoperati.

Particolare attenzione viene posta su quest'ultimi stimatori in quanto mirano a ricostruire in maniera ottima la posa del segmento. Il problema consiste nello stimare i parametri di rototraslazione (tre rotazioni e una traslazione)del sistema tecnico rispetto al sistema globale, partendo dalle coordinate, considerate affette da rumore additivo, di almeno tre *marker* non allineati associati al segmento corporeo, considerato.

Si definiscono i seguenti parametri necessari poi nella risoluzione del problema di stima ai minimi quadrati:

- ✓ $p_i(t)_{glo}$ e $p_i(t)_{loc}$, coordinate del marcatore i-esimo rispettivamente nel sistema globale e locale(o tecnico);
- ✓ $\bar{p}_{glo}$ e $\bar{p}_{loc}$ centroidi del *cluster* rispettivamente nel sistema globale e tecnico;
- ✓ P_{glo} e P_{loc} *cluster* globale e locale relativi al proprio centroide (per esempio $P_{glo} = [p_{1,glo} - \bar{p}_{glo}, \ldots, p_{m,glo} - \bar{p}_{glo}]$);
- ✓ $K = P_{glo}P_{glo}^T/m$, matrice di dispersione del *cluster*;
- ✓ $G = P_{glo}P_{loc}^T$, matrice di cross-dispersione.

Si vogliono quindi trovare la matrice di rotazione ortonormale R ed il vettore posizione T che mappano in maniera ottima i punti $p_{i,loc}$ nei punti $p_{i,glo}$ con i=1….m.
A causa degli errori di misura la mappatura non è esatta, in particolare risulta:

$$p_{i,glo} = R(p_{i,loc}) + T + e_i$$

dove e_i rappresenta il rumore additivo.

A causa della mancanza di informazione sulle proprietà statistiche del rumore, uno stimatore ai minimi quadrati non pesati risulta essere il più adatto. In tal caso ci si riconduce alla soluzione del seguente problema di minimizzazione:

$$\min_{R,T} \sum_{i=1}^{2}(R,T) = \sum_{i=1}^{m}\|e_i\|^2 = \sum_{i=1}^{m}\|R(p_{i,loc}) + T - p_{i,glo}\|$$

Dove m è il numero di marker.

Per la soluzione del problema di minimo espresso in Y è possibile adottare il metodo di scomposizione ai valori singolari (*Singular Value Decomposition*, SVD). Tale scomposizione viene applicata alla matrice di cross – dispersione $G=p_{glo}p_{loc}^T/m$.

L'algoritmo si sviluppa in tre passi:

1) SVD della matrice di cross – dispersione:

$$G = UVD$$

dove U e V sono matrici ortogonali 3X3 tali per cui: det(U)=±1, det(V)=±1 e $D=diag(\sigma_1,\sigma_2,\sigma_3)$ con $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq 0$ valori singolari di G.

2) Stima della matrice di orientamento:

$$\hat{R} = U \cdot diag(1,1,\det(UV^T))V^T$$

3) Stima del vettore di posizione:

$$\hat{T} = \bar{p}_{glo} - \hat{R}\bar{p}_{loc}$$

I principali vantaggi di questo metodo sono la stabilità, essenziale nel caso di problemi mal condizionati, e la facilità di implementazione ed analisi. Esso, però, consente solo di pesare diversamente l'informazione associata ai diversi marcatori ma non le informazioni a priori sul rumore sovrapposto, non essendo basato sul calcolo delle funzioni di sensitività.

3.3 Sistemi ottici markerless

La possibilità di ricostruire il movimento umano a partire unicamente da sequenze di immagini è un argomento che, negli ultimi vent'anni, ha riscosso un notevole interesse nella comunità scientifica, soprattutto grazie alle innumerevoli applicazioni nei più svariati campi: dalla robotica alla *computer vision* dalla sorveglianza intelligente, all'identificazione e controllo, interfacce percettive, animazione di personaggi, realtà virtuale e analisi del movimento [7].
In particolare i campi della *computer vision* e *machine-learning*, dove l'analisi del movimento umano da parte del computer sta suscitando un sempre crescente interesse, hanno permesso lo sviluppo di una grande varietà di sistemi *vision-based* per la ricostruzione del movimento umano.

Per quanto riguarda la *motion analysis*, eliminare i marcatori (approccio *markerless)*rappresenta una via risolutiva nei confronti di molti, anche se non tutti,quei problemi che caratterizzano i sistemi ottici *marker-based.*

In primo luogo, con la rimozione di qualsiasi apparecchiatura applicata al corpo si annulla totalmente l'ingombro sul soggetto, eliminando eventuali stimoli indesiderati al sistema neuro-sensoriale e permettendo l'acquisizione di movimenti naturali, liberi da costrizioni e appartenenti quindi ad un *pattern* fisiologico. In secondo luogo, l'eliminazione dei *markers*, comporta una netta riduzione dei tempi di preparazione del paziente e potenzialmente mette in condizioni di disporre di valutazioni e diagnosi più corrette e accurate. Infine, la tecnica *markerless* presenta una potenziale applicabilità anche in contesti in cui l'utilizzo di una tecnica *marker-based* risulta più difficile o addirittura proibitiva: basti pensare al caso di movimento in ambiente acquatico.

I sistemi *MMC* sviluppati ad oggi vengono suddivisi in due categorie: attivi e passivi.

I sistemi attivi prevedono l'emissione di segnali luminosi nello spettro del visibile o dell'infrarosso sottoforma di laser, pattern luminosi o impulsi modulati mentre quelli passivi si basano sulla semplice acquisizione di immagini. In generale i sistemi *MMC* attivi come scanner laser, sistemi a luce strutturata e sensori *time-of-flight* [4] forniscono misure nello spazio 3D molto accurate ma necessitano di un ambiente di laboratorio controllato e "pulito" e spesso si limitano a misure di tipo statico.

A titolo d'esempio, uno scanner laser impiega tipicamente diversi secondi per acquisire la superficie totale di un corpo umano, ordine di grandezza incompatibile se si pensa che il movimento umano può raggiungere frequenze di 200Hz nell'esercizio del salto.

I sistemi passivi sono vantaggiosi in quanto si affidano solamente sull'acquisizione di immagini fornendo le condizioni ideali per quanto riguarda l'acquisizione del movimento umano.

Di questi metodi è possibile fornire un'ulteriore classificazione in base all'utilizzo o meno di un modello per il soggetto, distinguendo tra i metodi *model-free*, e *model-based* in cui vi può essere un uso indiretto del modello, o un uso diretto [6].
Nei metodi *model-free* la ricostruzione della posa 3D del corpo avviene senza l'utilizzo di informazioni a priori di tipo anatomico, funzionale e strutturale fornite della conoscenza di un modello del soggetto stesso.

Nei metodi a uso indiretto del modello, questo viene invece unicamente utilizzato come riferimento per guidare l'interpretazione dei dati misurati, mentre nei metodi a uso diretto il modello stesso viene assunto come rappresentazione del soggetto osservato, in grado di fornire in ogni istante qualunque informazione desiderata, inclusa la posa del soggetto in quel dato istante [6].

La maggior parte degli approcci è di tipo *model-based*: in questa tecnica un modello definito a priori, contenente le informazioni sia anatomiche che cinematiche necessarie, viene utilizzato per il matching con i dati, che possono avere o una rappresentazione bidimensionale sul piano delle telecamere (*sihlouette*) o tridimensionale sottoforma, ad esempio, di *visual hull* [6,7,11], come verrà descritto in seguito.

È facile intuire come l'uso di un modello introduca importanti vantaggi, come la possibilità di gestire meglio eventuali errori e artefatti sui dati, e la semplicità con cui diversi vincoli cinematici possono essere incorporati nel sistema, limitando in questo modo lo spazio di ricerca delle possibili pose assunte dal soggetto a quelle fisicamente ammissibili.

3.3.1 Calibrazione sistemi markerless

Prima di poter procedere alla fase di acquisizione dei dati, anche per questi sistemi occorre effettuare la calibrazione per ciascuna camera del sistema.

Nel caso di sistemi stereofotogrammetrici standard (composti da telecamere standard ossia videocamere o fotocamere digitali) esistono in letteratura diversi metodi per effettuare il processo di calibrazione.

Tipicamente questa operazione viene eseguita basandosi sull'utilizzo di particolari tool geometrici dei quali sono note con precisione le caratteristiche quali dimensione e posizione delle features presenti nel pattern. Mediante l'acquisizione di tali pattern posti in diverse posizioni all'interno dell'area di lavoro è possibile stimare i parametri intrinseci ed estrinseci che caratterizzano il sistema stereoscopico.

In particolare il metodo, fortemente ispirato all'algoritmo proposto da Z.Zhag in [22],prevede che a partire dall'immagine catturata dalla telecamera si vada a confrontare le proiezioni di particolari punti del pattern sul piano immagine con quanto si conosce a priori del pattern stesso .

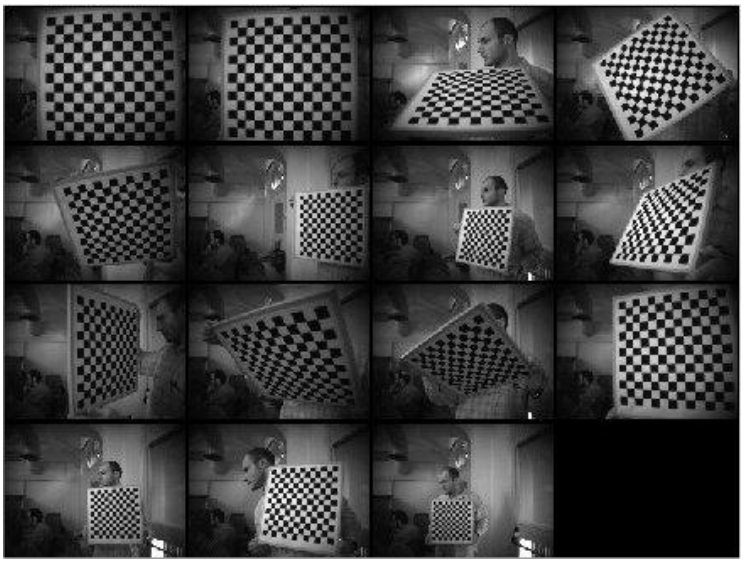


Fig. 11 Immagini utilizzate per la calibrazione di un sistema stereoscopico mediante l'utilizzo di un pattern planare (scacchiera). [33]

In [33] è possibile trovare un release del "*Camera Calibration Toolbox for Matlab*® " sviluppato in linguaggio *matlab* da Jean-Yves Bouguet.
Il toolbox in questione viene usato generalmente per la calibrazione di sistemi stereo fotogrammetrici standard.
Il metodo si articola in diversi punti di seguito illustrati e si basa sull'utilizzo di un particolare tool costituito da una scacchiera classica a caselle nere e bianche.
Innanzitutto si procede all'acquisizione di diverse immagini da tutte le telecamere che si intendono calibrare mentre nell'area di lavoro è presente il pattern di identificazione.
Tali immagini devono essere quindi salvate in un folder specifico e caricate in matlab per l'elaborazione. A questo punto se tutte le operazioni sono state effettuate correttamente occorre fornire in input il numero di tasselli per riga (*wintx*) o per colonna (*winty*) presenti nella scacchiera (di default il valore di *wintx=winty* è uguale a 5). Il toolmette a disposizione dell'utente un meccanismo automatico di conteggio e assegnazione dei valori di *wintx* e *winty.* Questo strumento è particolarmente utile quando si lavora con un numero significativo di immagini in quanto si risparmia all'utente il dover immettere manualmente il numero di quadrati in entrambe le direzioni x e y del modello. In alcune rare occasioni tuttavia, in particolare in presenza di obiettivi caratterizzati da distorsioni molto elevate, l'algoritmo può conteggiare un valore errato di tasselli determinando un errore nel processo di calibrazione.

A questo punto del procedimento all'utente si presenta una schermata simile a quella riportata in figura 12 dove viene richiesto di procedere alla selezione manuale dei quattro vertici estremità della scacchiera.

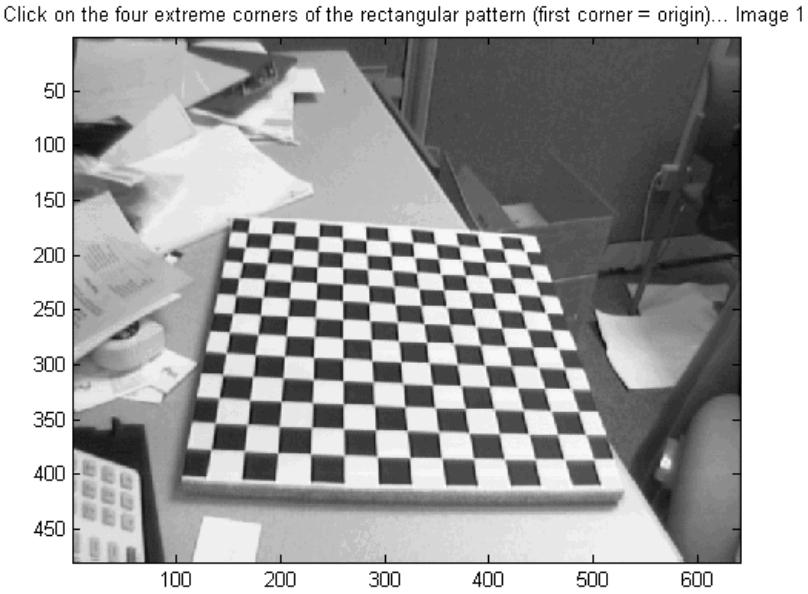


Fig.12 esempio di schermata presentata dal toolbox Matlab [33]

Occorre far notare che l'ordine di selezione dei vertici non è casuale. In particolareil primo punto cliccato viene associato al punto di origine del sistema di riferimento collegato alla singola immagine. Gli altri tre punti sulla griglia rettangolare possonoessere selezionati in qualsiasi ordine in quanto il software è in grado di costruire la terna locale unicamente basandosi sulla posizione dell'origine e sull'identificazione del piano della scacchiera (definito dai 4 punti totali). Occorre ribadire quindi che la prima selezione è estremamente importante in quanto è quella che permette di associare ad ogni pattern il proprio sistema di riferimento e di identificare quindi in un secondo momento punti corrispondenti in immagini successive.

Questo rappresenta un punto cruciale nel processo: una selezione poco accurata dei punti o un ordine errato nella scelta dei punti introduce un errore nella stima dei parametri che compromette l'intero processo di calibrazione.

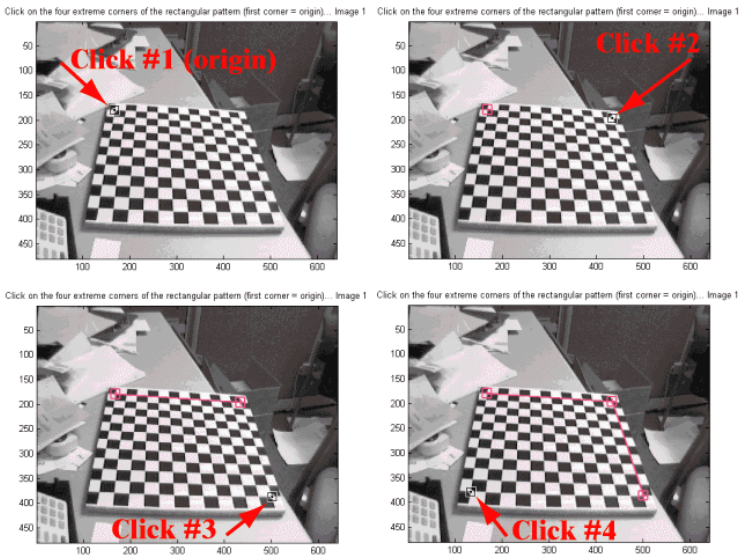


Fig.13 Esempio di schermata presentata all'operatore nella fase di selezione dei punti di controllo. [33]

All'utente viene quindi richiesto di riportare la dimensione reale dei tasselli nella scacchiera in modo che l'algoritmo possa stimare la posizione dei vertici di ogni casella nell'immagine fornendo in output una rappresentazione dei risultati.
Attraverso un'ispezione visiva si controlla la bontà dei risultati che se non sufficientemente accurati possono essere ricalcolati introducendo particolari fattori di correzione.
Questa procedura viene ripetuta per tutte le immagini acquisite.
Una volta aver effettuato tale procedura per ogni singola immagine è possibile procedere alla vera e propria fase di stima dei parametri intrinseci ed estrinseci delle camere.

La calibrazione viene effettuata in due fasi: una prima inizializzazione dei dati, e quindi un'ottimizzazione non lineare.
La fase di inizializzazione calcola una soluzione in forma chiusa per i parametri di calibrazione escludendo la presenza di qualsiasi distorsione della lente che viene invece considerata nel successivo step di ottimizzazione non lineare dove si opera per minimizzare l'errore totale di retroproiezione su tutti i parametri di calibrazione (nove parametri intrinseci: focali, punto principale, coefficienti di distorsione, e
6 * n estrinseci => 6*n + 9 incognite totali con n = numero telecamere). L'ottimizzazione viene effettuata in modo iterativo diminuendo via via il gradiente della matrice jacobiana della trasformazione.

Quest'ultima fase fornisce tutta una serie di comunicazioni all'utente il quale ha il compito di supervisionare alla bontà dei diversi step.

Quella appena illustrata non rappresenta però l'unica tecnica per procedere alla calibrazione di un sistema stereo fotogrammetrico anche se l'idea di fondo per effettuare tale processo rimane sempre lo stesso: ossia basarsi sull'acquisizione di oggetti e pattern di caratteristiche note per stimare i parametri necessari allo scopo.

3.3.2 Visual hull e modello

Una volta aver calibrato tutte le camere è possibile procedere all'acquisizione delle immagini che verranno poi processate al fine di ricavare quelle informazioni necessarie per svolgere un'analisi del movimento non mediata da marcatori.

Ad oggi, diversi metodi sono stati proposti per eseguire l'analisi del movimento con tecnica markerless.
Uno dei più diffusi è un metodo model-based che sfrutta il matching di un modello del soggetto sui dati sperimentali che assumono la forma tridimensionale di *visualhull* [6,7,11].
Per *visualhull* si intende una approssimazione del volume occupato dal soggetto in un dato istante, esso rappresenta una sorta di involucro (letteralmente hull= contenitore) che descrive la superficie corporea del paziente in movimento.
Il primo passo per le elaborazioni con questo tipo di tecnica consiste nel riconoscimento, da parte del calcolatore, del soggetto rispetto al background, in tutti i frame acquisiti dalle telecamere presenti nel laboratorio.
Questo processo di riconoscimento è detto segmentazione e rappresenta una fase particolarmente critica in molte applicazioni in computer vision.

La situazione più favorevole per quanto riguarda il processo di segmentazione è quella che prevede un'acquisizione effettuata in ambiente controllato, dove pareti e pavimento costituenti lo sfondo presentano un colore a tinta unita assente nel soggetto (solitamente bluo verde) [10]: in questo caso sarà sufficiente classificare come *background* tutti e solo i pixel che presentano il colore associato allo sfondo o un colore simile. La maggior parte delle applicazioni, tuttavia, coinvolge ambienti relativamente meno controllati che richiedono un approccio più complesso.
Una tecnica spesso impiegata in questi casi per agevolare l'identificazione del soggetto è la sottrazione del *background*, ossia la sottrazione pixel a pixel da ognuno dei frame registrati, di un'immagine che modella lo sfondo, acquisita cioè in assenza del soggetto. In generale, l'effetto della sottrazione di due immagini è l'accentuazione delle differenze tra le due, ossia nel nostro caso specifico porre in evidenza il soggetto rispetto al *background*, facilitandone l'identificazione o tramite un semplice processo di sogliatura o tramite algoritmi di classificazione più complessi.

Indipendentemente dall'algoritmo utilizzato, il risultato della segmentazione è una binarizzazione dell'immagine, allo scopo di ottenere le sagome che,retroproiettate nello spazio a partire dai piani delle telecamere,permettono la creazione del *visualhull.*

Spesso, al termine della classificazione vengono applicati anche altri algoritmi correttivi che vadano a migliorare il risultato riempiendo eventuali buchi rimasti nella sagoma ed eliminando pixel spuri erroneamente classificati come *foreground.* Le immagini segmentate del soggetto così ottenute vengono chiamate *silhouettes* (Figura 14).

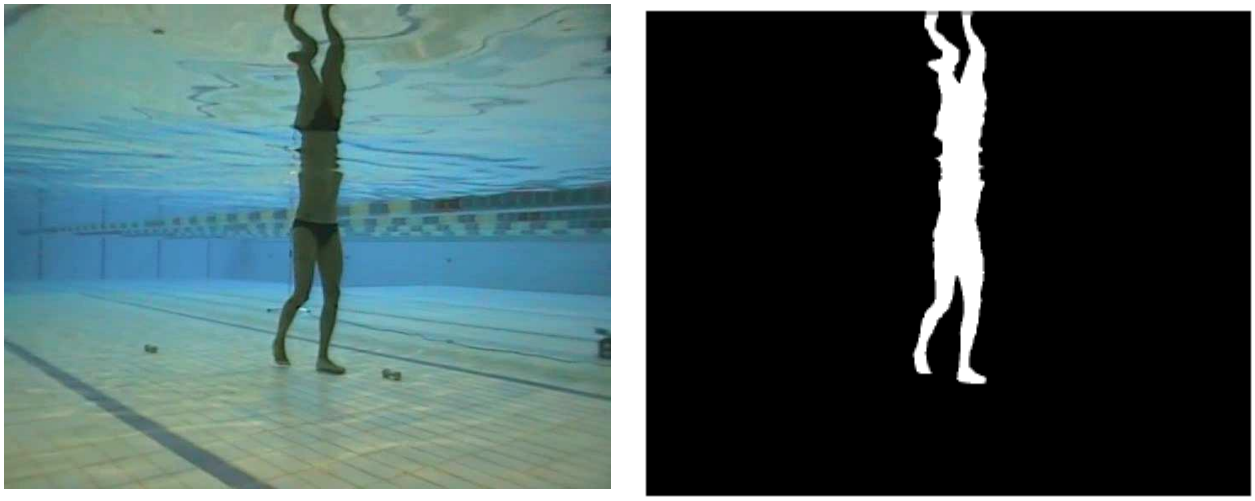


Fig.14 Esempio di frame originale del l ’acquisizione ed estrazione della sihlouette tramite segmentazione [45]

Le *silhouettes* ottenute dalla segmentazione di tutte le differenti viste sincronizzate vengono quindi impiegate per la generazione di un *visualhull*.
Dopo aver effettuato il processo di calibrazione, un generico cono costruito su ognuna delle silhouette definite in precedenza può essere retroproiettato nello spazio: il volume compreso nell’intersezione di tutti i coni costituisce il visual hull.

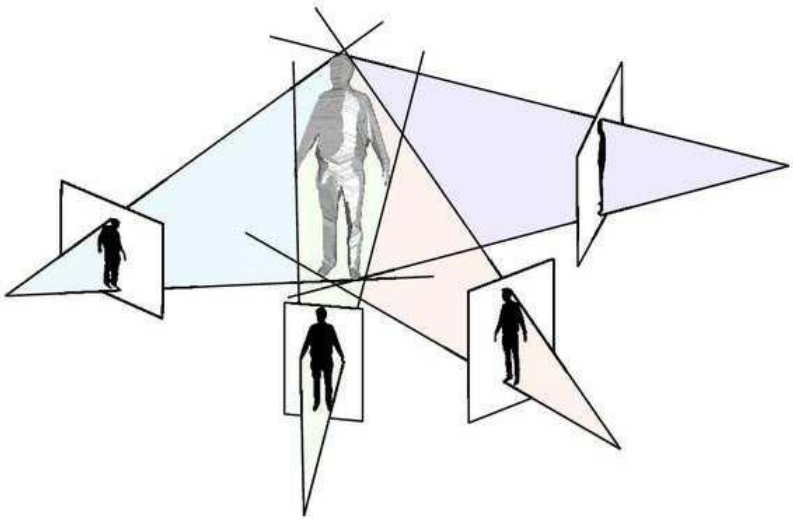


Fig. 15 Generazione di visualhull a partire da 4 silhouettes[45]

Ciò che si ottiene è costituito da un insieme di elementi discreti di volume, detti *voxels* (*volumetric picture elements*), ottenuti dalla partizione dello spazio calibrato in tanti piccoli volumi di forma cubica: solo i *voxels* le cuiproiezioni sui piani delle telecamere siano compresiin tutte lesilhouettes faranno parte del *visual hull*. La risoluzione del visual hull dipende naturalmente dalla dimensione dei *voxels*, parametro a sua volta correlato al numero di camere che inquadrano la scena.

Uno dei principali problemi legati all'uso dei *visual hull* è dovuto ad errori nella fase di ricostruzione e alla possibile insorgenza dei cosiddetti *phantom volumes.* I *phantom volumes, o volume fantasma* sono artefatti definiti come significativedeviazioni locali rispetto alla vera superficie corporea del soggetto [11], che compaiono quando una porzione del volume di interesse è occlusa alla vista delle telecamere. E' provato che un parametro critico nel controllo di questo problema è proprio il numero di telecamere impiegate: un numero maggiore di telecamere (almeno otto) permette una ricostruzione più accurata del visual hull riducendo la possibile insorgenza di tale artefatto.

Fig. 16 *Visual hull* affetto da phantom volume [45]

A questo punto, si rende necessaria la definizione di un modello per il soggetto che guidi l'interpretazione dei dati ottenuti sperimentalmente. Il modello rappresenta infatti la conoscenza a priori sia sulla morfologia
che sulla cinematica del corpo del soggetto e, con l'unica eccezione degli approcci *model-free,* è in definitiva necessario per l'interpretazione dei dati sperimentali permettendo di discriminare e riconoscere nel singolo *visual hull* i diversi segmenti corporei, ridurre l'influenza degli artefatti nella ricostruzione del visual hull e ricavare i parametri del movimento, come gli angoli articolari.

Occorre infatti ricordare che la nuvola di punti generata dalla ricostruzione del *visual hull* non contiene di per sé alcuna informazione riguardo ad esempio su quali punti appartengono a una determinata regione corporea, e nemmeno sulla correlazione tra i punti del *visual hull* generati in un frame e quelli generati nei frame precedenti o successivi.
La complessità nel costruire un buon modello consiste nel definire con sufficiente accuratezza il comportamento non lineare e non rigido delle sue componenti, in particolare a livello delle articolazioni, al fine di ottenere un'analisi del movimento il più possibile accurata e affidabile.

Nella realizzazione del modello, l'informazione anatomica è in genere ottenuta tramite una scansione del soggetto mediante un laser scanner: il risultato è un oggetto tridimensionale rappresentante la superficie esterna del soggetto, descritta al calcolatore come *mesh* poligonale. In genere, una *mesh* poligonale è espressa da un lista ordinata di punti 3D rappresentanti i vertici dei poligoni e dalla lista di poligoni che indicano come i vertici sono tra loro connessi. Ogni vertice è

univocamente identificato dalla sua posizione nella lista e definito dalle sue tre coordinate rispetto al sistema di riferimento globale, mentre ogni poligono è descritto dagli identificatori dei suoi vertici indicati in senso orario, guardando il poligono dall'esterno.

Le informazioni sulla cinematica possono essere invece rappresentate da unmodello articolato del corpo umano. Il modello articolato per l'analisi del movimento di tutto il corpo consiste generalmente in 15 segmenti rigidi, modellanti gli altrettanti segmenti corporei principali (testa, busto, bacino, braccia, avambracci, mani, cosce,gambe, piedi), connessi tra loro tramite articolazioni a 6 gradi di libertà (3traslazioni e 3 rotazioni).
Oltre a quello appena citato sono stati proposti negli anni modelli diversi con un diverso grado di complessità in relazione agli scopi e alle necessità prefissati.
In particolare i primi ad essere introdotti da Lee Hj. E Chen Z. nel 1984 furono dei modelli a sticko a cilindri. Successivamente sono stati sviluppati i modelli a superquadriche proposti da Gavrila D. e Davis L. per giungere infine a modelli più complessi, tutt'oggi utilizzati, realizzati al CAD.

Ognuno dei segmenti corporei definiti nel modello, ad eccezione della testa,ammette un segmento padre epuò avere più segmenti figli.
Un sistema di riferimento tecnico viene quindi definito per ognuno dei segmenti, secondo lo schema per cui il sistema di riferimento locale di ogni parte si trovi in una relazione nota con i sistemi di riferimento del segmento padre e dei segmenti figli. L'unica eccezione è costituita dal segmento testa, il cui sistema di riferimento è orientato come quello globale.

Giunti a questo punto il sistema articolato forma una catena cinematica, e laposa dell'intero modello è univocamente identificabile noti posizione eorientamento di ogni segmento rispetto al proprio segmento padre.

I vincoli imposti alle articolazioni sono inoltre parte di quella conoscenza a priori che può essere sfruttata e incorporata in questafase della modellazione. Questo tipo di informazione permette diammettere tra tutte le possibili configurazioni che andranno a fittare idati sperimentali solo quelle anatomicamente ammissibili e sensate.

A questo punto si può procederecol generare il modello complessivo,specifico per il soggetto, che include sia le informazioni anatomiche, precedentemente ottenute con scanner laser che le informazionicinematiche e che verrà quindi utilizzato per l'interpretazione dei dati.In sostanza, ciò che occorre fare è adattare il modello a15 segmenti rigidi,o quale altro utilizzato,alla *mesh* poligonale specificamente creata sulsoggetto attraverso la scannerizzazione laser, identificando sulla mesh i 15 segmenti corporei e le posizionidei centri articolari.
Si riuscirà in questo modo nell'intento di creare un modello specifico per il soggetto e utilizzabile per l'interpretazione dei dati, che contenga al suo interno le informazioni di tipo sia morfologico che cinematico.

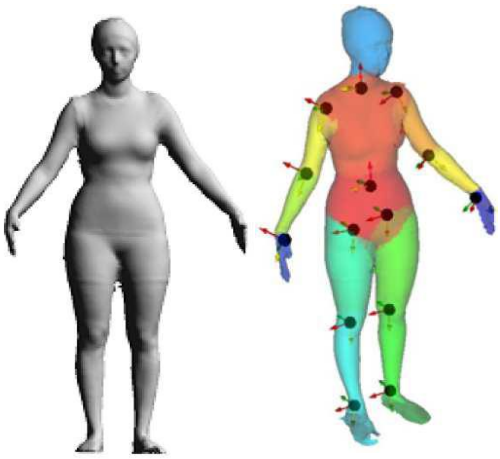


Fig. 17 Rappresentazione della superficie esterna del soggetto con mesh triangolare acquisita con laser scanner (a sinistra) e modello completo del soggetto dopo l 'applicazione del l 'algoritmo di generazione automatica del modello (a destra) [45]

Recentemente, è stato sviluppato un algoritmo per la generazione automatica di un modello specifico per il soggetto a partire dalla sua rappresentazione superficiale come mesh poligonale [7], che permette di includere direttamente le informazioni cinematiche nella *mesh* specifica del soggetto, ottenendo le posizioni dei centri articolari e la segmentazione della mesh nelle diverse parti del corpo.
E' stato provato che la procedura automatica [7] lavora bene anche quando si usa il *visual hull* come input anziché la *mesh* fornita dal laser scanner, che richiederebbe un costoso hardware dedicato allo scopo e non sempre disponibile. Nel caso in cui si voglia impiegare il *visual hull* come input, si richiede al soggetto di rimanere in piedi all'interno del volume di misura nella posizione di riferimento effettuando l'acquisizione per alcuni istanti. Il *visual hull* del soggetto in questa configurazione viene così utilizzato per generare il modello anatomico, che in questo caso risulta occupare un volume maggiore rispetto al soggetto vero, dato che il *visual hull* è un'approssimazione per eccesso del volume occupato dall'oggetto. Anche se il modello così creato è generalmente meno accurato rispetto all'utilizzo del laser scanner, può comunque rappresentare un input valido se il numero di telecamere utilizzate è sufficientemente elevato (in particolare in numero maggiore a 8 telecamere, possibilmente almeno 16) [6,8,11]

Una volta ottenuto il modello completo specifico per il soggetto, ciò che si intende fare è trovarne, istante per istante, la configurazione spaziale, tra tutte quelle permesse rispettando i vincoli imposti, che meglio descrive i dati. La giusta configurazione del modello può essere trovata facendo corrispondere ad ognuno dei punti *del visual hull* un punto del modello, usando ad esempio l'algoritmo *AICP* (*Articulated Iterative Closest Point*) introdotto da Corazza et al. e descritto in seguito.
Anche il matching nel senso opposto (modello sui dati) è tuttavia possibile: i due tipi di approccio differiscono solo nella loro robustezza rispetto ai dati corrotti. Se nel primo caso (dati sul modello) l'algoritmo è meno robusto rispetto ai *phantom volumes* eventualmente presenti nel *visual hull*, nel secondo tipo di approccio le parti mancanti del *visual hull* (specialmente arti mancanti), dovuti ad errori di ricostruzione, portano il modello ad assumere configurazioni completamente sbagliate.

L'approccio *AICP* [6,11] sopra citato si basa sulla corrispondenza tra coppie di puntivicini su due *mesh* diverse: l'algoritmo agisce localmente, nonconsiderando tutte le possibili combinazioni di punti corrispondenti sulle due mesh. Questo riduce sensibilmente il costo computazionale,al prezzo di un aumento della sensibilità rispetto al posizionamento iniziale del modello.
Fondamentalmente, alla fine il sistema si riduce ad essere un tipico problema ai minimi quadrati non lineare.
Le due mesh in questione sono appunto rappresentate della superficie esterna del modello appena costruito e la mesh che descrive il *visualhull* al dato istante di tempo.
In particolare il modello articolato *M* viene rappresentato da un insieme discreto di punti $p_{1,\dots,}p_P$ sulla superficie del modello stesso, da un insieme $s_{1,\dots,}s_S$ di segmenti rigidi (solitamente S=15), e da un set $q_{1,\dots,}q_Q$ di articolazioni che congiungono gli S segmenti rigidi.

Il *visualhull*, elaborato per ogni istante di tempo, è rappresentato da un insieme di punti $V=v_{1,\dots,}v_N$ che descrive la superficie esterna del soggetto in quel particolare frame.

Per ogni frame della sequenza viene stimato un insieme di trasformazioni rigide T_j , una per ogni segmento rigido s_j, che allinea la superficie esterna del modello *M* con l'insieme dei punti *V* facendo attenzione a rispettare i vincoli introdotti dalle articolazioni *Q*.

In questo particolare caso l'algoritmo di *ICP* (*Iterative Closet Point*)lavora in due fasi; nel primo step ogni punto p_i del modello *M* è associato al suo nearest neighbor $v_{s(i)}$tra i punti del *visual hull*, dove *s(i)* definisce la mappatura dall'i-esimo punto p_i sulla superficie del segmento corporeo *s*.

Nel secondo step, dato l'insieme di coppie $(p_i, v_{s(i)})$ viene stimato un insieme di trasformazioni rigide *T* che permette di far corrispondere ad ogni p_i il corrispondente punto $v_{s(i)}$.

Questa seconda fase è costruita sulla definizione di una funzione costo che tiene conto delle variabili in gioco ed è definita come *F(T) = H(T) + G(T)* dove il termina *H(T)* assicura che le coppie di punti definite nella prima fase siano allineate. In particolare:

$$H(r,t) = w_h \sum_{i=1}^{P} \left\| R\left(r_{s(i)}\right) p_i + t_{s(i)} - v_i \right\|^2 \qquad [15]$$

La trasformazione T_j di ogni componente rigida s_j è parametrizzata con un vettore traslazione 3x1 t_j e un vettore rotazione 3x1 r_j, mentre$R(r_{s(i)})$ denota la matrice di rotazione costruita dai parametri $r_{s(i).}$

Il termine *G(T)*, dove ogni articolazione $q_{i,j}$ è quella che congiunge i segmenti *i* e *j*, assicura invece che i vincoli imposti dalle articolazioni siano rispettati.

Anche le trasformazioni T_i e T_j sono forzate a rispettare l'articolazione costantemente.

$$G(r,t) = w_G \sum_{(i,j)\in Q(M)} \left\| R(r_i)q_{i,j} + t_i - R(r_j)q_{i,j} - t_j \right\|^2 \quad [16]$$

La diminuzione del valore w_G permette un maggiore movimento in corrispondenza delle articolazioni che, potenzialmente, migliora il matching tra il segmento corporeo ed il *visual hull*.

È interessante notare che la stima del centro dell'articolazione permette una stima accurata del centro funzionale dell'articolazione stessa che risulta essere estremamente utile per il calcolo della componente dinamica negli studi di motion analysis.

Come risultato,il modello cinematico può essere ridefinito a posteriori al fine di ottenere una corrispondenza anatomica maggiore.

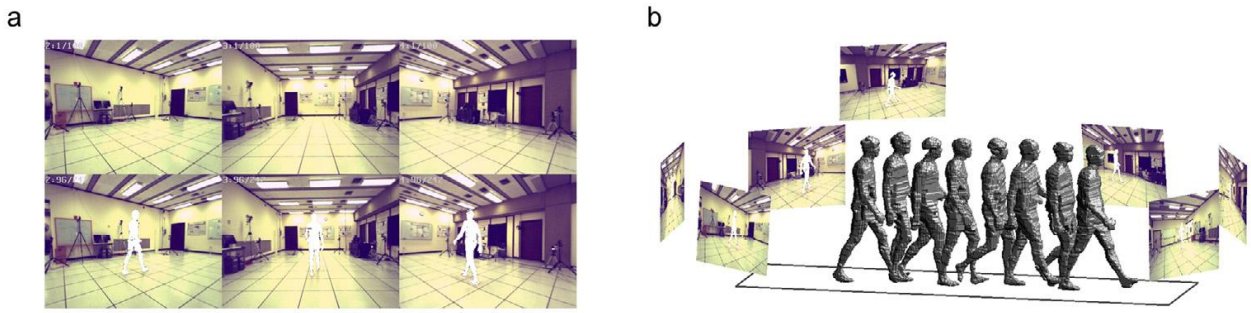


Fig. 18 A destra: Immagini del background (sopra) e successiva separazione del paziente tramite sottrazione sotto. A sinistra: configurazione delle camere e ricostruzione del visual hull [6]

Attraverso questo procedimento è possibile stimare, ad ogni istante, la posizione e la posa del modello e di tutte le sue componenti e da questo ricavare tutti i parametri di interesse per l'applicazione specifica.

Purtroppo il *markerless* non ha solo lati positivi ma presenta numerose problematiche di difficile risoluzione. In particolare il principale limite presentato ad oggi, per applicazioni cliniche e biomeccaniche è ancora purtroppo la scarsa accuratezza dei metodi correnti. L'accuratezza dipende tuttavia da molti fattori, tra cui la qualità della segmentazione iniziale, la risoluzione e soprattutto il numero di telecamere impiegate (parametro che influenza di molto la qualità della ricostruzione del *visualhull*): a seconda del data set impiegato, infatti, la differenza tra la ricostruzione della posizione dei centri articolari con tecnica markerless e marker-based può passare da 7-8 cm a 1-2 cm, con l'algoritmo proposto da Corazza et al [25].
Ad oggi quindi la ricerca di metodi e tecniche per migliorare le prestazioni del markerless è molto attiva anche se ad ora il sistema stereo *marker-based* rappresenta il *gold standard*.

Capitolo 4

Sensori analizzati

4.1 Sensori D-RGB

Come introdotto nei capitoli precedenti lo scopo principale della tesi è stato quello di utilizzare le mappe di profondità, ottenibili da dispositivi D-RGB (*Depth-RGB*) in commercio, al fine di studiarne l'applicabilità nel campo della *motion capture*.

Con l'acronimo D-RGB si sta ad indicare quei sistemi che oltre a fornire in output una rappresentazione RGB della scena permettono di ricostruirne una mappa di profondità della stessa.

La mappa di profondità o *depth map* è un'immagine *M* di dimensione *mxn*, in cui ciascun pixel *p(x, y)* codifica la distanza nella scena 3D del punto *(x,y)* dal sensore che l'ha generata.

L'utilizzo di immagini di profondità rispetto alle classiche immagini RGB o BW oltre a fornire informazioni riguardo la terza dimensione, semplifica molti problemi di computer vision e di interazione come ad esempio:

- rimozione del *background* e segmentazione della scena;
- *tracking* di oggetti e persone;
- ricostruzione 3D degli ambienti;
- riconoscimento della posa del corpo;
- implementazione di interfacce basate su gesti.

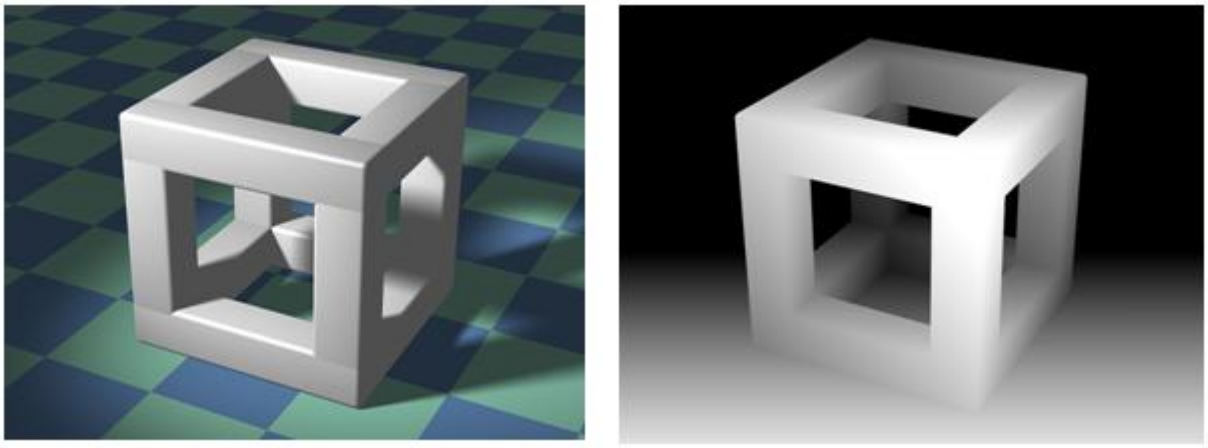


Fig. 1 Destra: mappa di profondità relativa alla scena presentata a sinistra. Si nota come le parti più vicine all'obiettivo sono caratterizzate da una tonalità di grigio meno intenso, mentre quelle più lontane da una tonalità più scura. Da notare come la *texture* presente nel pavimento non influisca minimamente la mappa di profondità [46]

Per determinare la mappa di profondità i dispositivi considerati utilizzano una tecnica a proiezione di pattern.
Questa si serve di un sistema di visione stereo costituito da una coppia proiettore-telecamera al fine di definire un processo di triangolazione attiva.

Nella scena viene proiettato un pattern luminoso (infrarosso) noto e la profondità degli oggetti è calcolata
studiando la sua distorsione sugli oggetti.
Nella pratica è possibile implementare questa tecnica con vari approcci:

- proiezione di linee e studio della loro curvatura sugli oggetti: non molto veloce e soggetta a disturbi quando gli oggetti sono in movimento;
- proiezione di pattern 2D periodici e studio del loro scostamento quando colpiscono gli oggetti: l'informazione 3D è ottenuta in real-time ma non è in grado di lavorare su lunghe distanze per
  via della distorsione del pattern;
- proiezione di pattern 2D pseudo-casuali: anche in questo caso i pattern sono 2D ma la loro casualità permette di ottenere accurate mappe 3D in real-time con un sistema molto semplice ed economico.

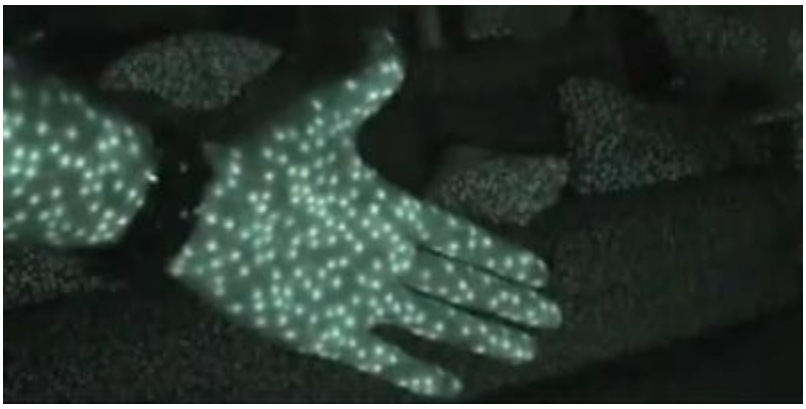


Fig. 2 Esempio di proiezione di pattern 2D generato da microsoft Kinect. [47]

In particolare, nei sensori analizzati per lo svolgimento del progetto di tesi, la mappa di profondità della scena viene costruita utilizzando la tecnica della proiezione di pattern pseudo-casuali. Questa tecnologia è stata brevettata nel 2005 da Zalevsky, Shpunt, Maizels e Garcia, sviluppata e integrata in un chip dalla compagnia israeliana PrimeSense e si basa su 3 elementi principali:

a) proiettore di pattern pseudo-casuali IR;
b) telecamera IR ;
c) unità di controllo;

Il proiettore è molto semplice ed economico ed è costituito da un emettitore di raggi IR e da un generatore di pattern che devia tali raggi nella scena imprimendo ad essi angolazioni pseudo-casuali note. Una volta proiettato il pattern, la telecamera acquisisce l'immagine IR della scena contenente il pattern distorto e la invia all'unità di controllo che costruisce così la mappa di profondità della scena.

Ma vediamo come nel dettaglio.
L'emettitore ad infrarossi proietta nell'ambiente esaminato un'immensità di spot luminosi la cui distribuzione pseudo-casuale è ottenuta grazie ad una mascherina posta davanti al led emettitore. Il pattern emesso è visibile spegnendo le luci e, puntando il sensore verso una superficie piana, inquadrando tale superficie con una camera digitale.

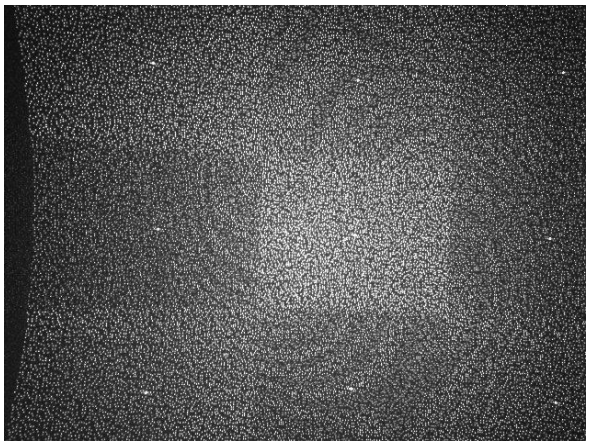


Fig. 3 Pattern infrarosso proiettato da distositivo D-RGB su di una parete parallela al piano della camera [48]

Passare da tale pattern all'informazione 3D è un'operazione matematica ben definita: la disposizione dei punti del pattern è nota a priori al software del sistema, che al suo interno ha memorizzato come il pattern dovrebbe essere visto dal sensore IR se fosse proiettato su una superficie posta ad una distanza ben definita e perfettamente parallela al piano della *depth* camera. Quest'ultimo quindi altro non è che un pattern di riferimento. Ma cosa succede quando gli spot luminosi proiettati impattano sugli oggetti della scena?
In tal caso viene a crearsi una deformazione della distanza tra uno spot e l'altro in base all'angolazione delle superfici (per intendere sarebbe come poggiare un lenzuolo a pois su un divano; è possibile farsi un'idea della forma del divano in base a come si dispongono i pallini), in quanto la fotocamera inquadra la proiezione bidimensionale della distribuzione tridimensionale degli *spot*.
La dimensione dei punti proiettati, la loro forma e orientazione non è quindi costante ma dipende dalla distanza dal sensore e da tali caratteristiche e in base alla densità degli spot luminosi si può avere indicazione sulla distanza dell'oggetto e riguardo l'angolazione di una superficie inquadrata. Per ognuno dei punti proiettati si può calcolare la distanza dal sensore triangolando la sua posizione attuale (espressa in pixel) con quella che avrebbe assunto nel pattern di riferimento. Si tratta di un processo di triangolazione attiva, una tecnica del tutto similare a quella implementata nella stereo visione per il calcolo della matrice di disparità.
In particolare dato lo schema in figura seguente, che riporta il caso di sistema passivo standard con due camere R 3 L, è possibile per le proprietà dei triangoli simili ricavare l'equazione:

$$z = \frac{b|f|}{d} \qquad [1]$$

dove $d = u_l - u_r$ rappresenta il valore di disparità del pixel, *b* sta a indicare il valore di baseline ossia la distanza tra le origini del sistema di riferimento della telecamera e del proiettore ed *f* è la distanza focale definita nel precedente capitolo.

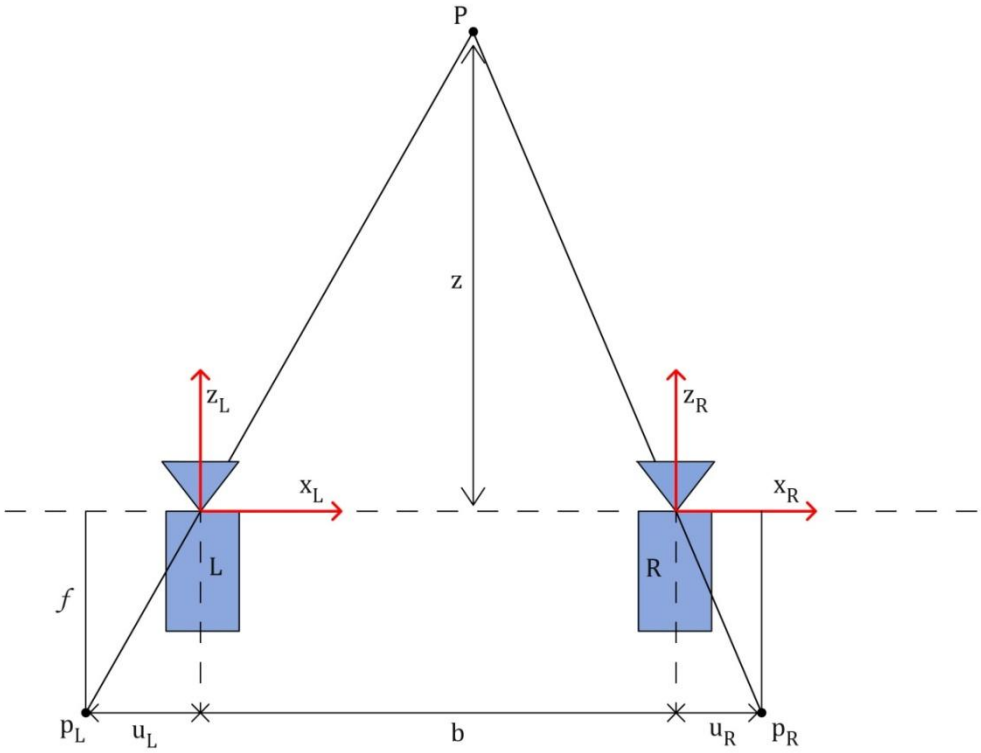


Fig. 4 Triangolazione tramite una coppia di camere rettificate e allineate

Nel caso in cui in un sistema stereo una telecamera venga sostituita da un proiettore lo schema di funzionamento è illustrato in figura 5:

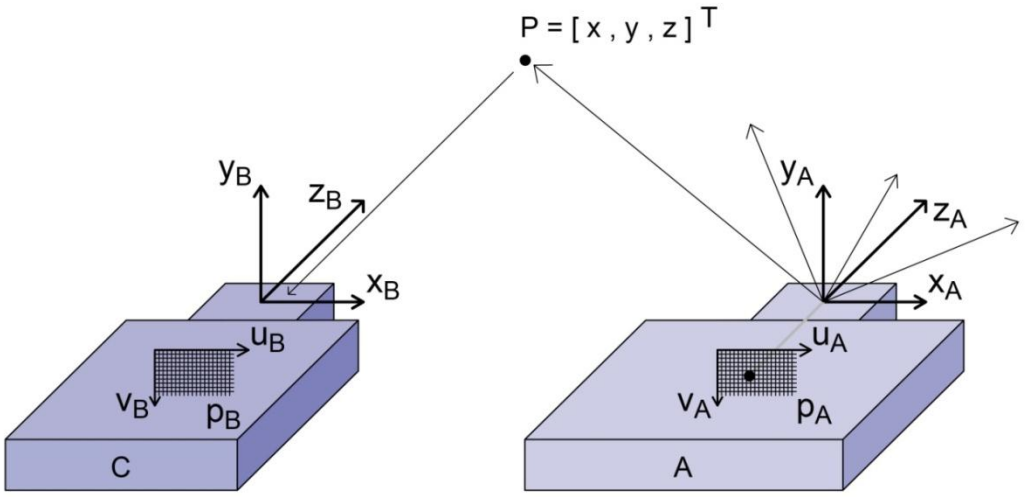


Fig. 5 Triangolazione attiva effettuata mediante un sistema composto da una telecamera ad infrarossi C e un proiettore A

Con la lettera *A* viene indicato il proiettore IR mentre con *C* viene indicata la telecamera sensibile alla stessa banda.
La camera *C* ha un proprio sistema di riferimento cartesiano (x_c,y_c,z_c) anche detto riferimento *C-3D* e un sistema di riferimento bidimensionale con coordinate (u_c,v_c) come mostrato in figura 5. Il proiettore A a sua volta, possiede un sistema di riferimento cartesiano (x_a,y_a,z_a) definito *A-3D* e un sistema 2D con coordinate (u_a,v_a).
Come nel caso di un sistema stereo passivo, un sistema attivo può essere calibrato e rettificato (i parametri intrinseci sono noti) al fine di semplificare la stima della profondità della scena.

La figura 5 mostra la proiezione del punto $p_A=[u_a,v_a]^T$ del pattern (secondo A-2D) nel punto reale 3D *P* di coordinate $P=[x,y,z,]^T$ nel sistema di riferimento *C-3D* .

Se il punto *P* è visibile alla camera esso proietta la luce emessa da A al pixel p_C di *C* stabilendo il triangolo p_CPp_A. Se il sistema è quindi calibrato (nota *f*) e rettificato (nota *b*), p_C ha coordinate $p_C=[u_C=u_A+d,\ v_C=v_A]$.
Come nel caso di un sistema stereo standard, finché p_A e p_C sono punti corrispondenti (o coniugati), una volta che le loro coordinate nel piano immagine sono note, la distanza *z* del punto *P* può essere facilmente calcolata attraverso la [1].

Risultato dell'elaborazione è un'immagine di profondità grezza (raw depth image), nel senso che i valori di profondità di un punto di quest'ultima non sono necessariamente espressi in una qualche unità di misura ma sicuramente sono legati attraverso una relazione nota alla distanza reale del punto dalla telecamera; è dunque necessaria una procedura di conversione (dipendente dai parametri interni del sensore quali: distanza focale, fattore di scala etc.) che metta in corrispondenza i valori grezzi con valori espressi in scala metrica.
Anche nel caso attivo infatti, le coordinate 3D di *P* possono essere ancora calcolate attraverso la relazione:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = K^{-1} \begin{pmatrix} u_C \\ v_C \\ 1 \end{pmatrix} z$$

Dove la matrice *K* è la matrice dei parametri intrinseci rettificata della camera *L*.

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$$

con $f_x = fk_u$ ossia la componente lungo $_x$ della distanza focale e f_y la corrispondente grandezza lungo $_y$ e c_x e c_y rappresentano le coordinate (u,v) dell'intersezione dell'asse ottico con il piano della telecamera.
Queste quantità sono espresse in [pixel] infatti, *f* è misurato in [mm] e k_u e k_v che rappresentano la dimensione del pixel nelle direzioni *u* e *v* assunti in [pixel]/[mm].

Sulla base di queste informazioni è quindi possibile assegnare una tonalità di grigio ad oggetti più o meno distanti ottenendo un' immagine in scala di grigi dove gli oggetti più vicini al sensore appaiono più chiari mentre gli oggetti più lontani risulteranno più scuri (per convenzione).

Fig. 6 Esempio di mappa di profondità

Il brevetto *PrimeSense* (uno dei primi ad aver sviluppato tale tecnologia) individua tre differenti tipologie di punti per tre differenti regioni dello spazio come mostrato in figura 7: una prima regione R1 (0.8 - 1.2 m) in cui si ha la massima risoluzione, una seconda regione R2 (1.2 - 2 m) con una buona accuratezza e una terza regione R3 (2 - 3.5 m) dove l'accuratezza è scarsa.

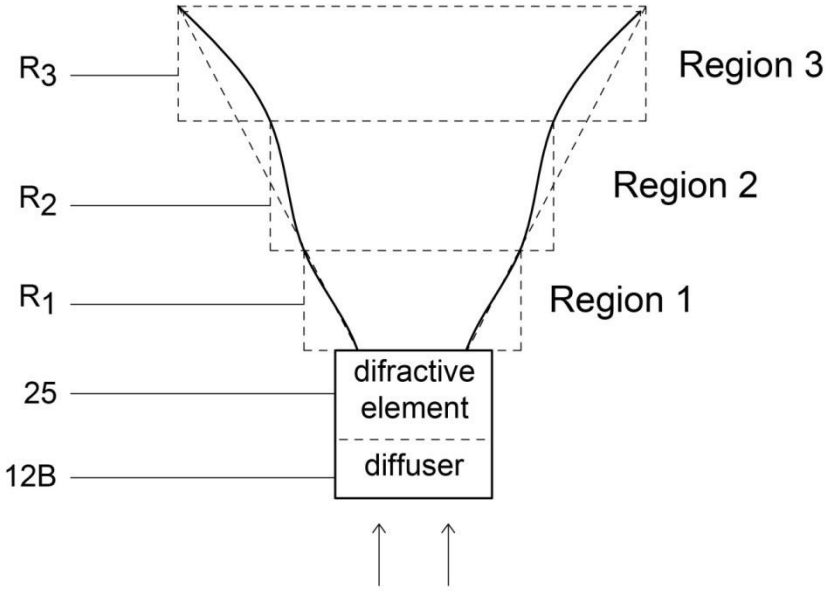


Fig. 7 Regioni dello spazio individuate dal brevetto PrimeSense.

PrimeSense non ha solo sviluppato un nuovo sistema di acquisizione della mappa 3D della scena, ma soprattutto una tecnologia capace di elaborare questi dati realizzando molti task di processing 3D.
Il chip presente nel sensore ha al suo interno molte funzionalità di processing per il tracciamento, la ricostruzione della scena e il riconoscimento di gesti.

4.2 Dispositivi per la generazione di depth maps

Grazie alla tecnologia sviluppata dalla compagnia israeliana *PrimeSense* sono stati commercializzati diversi tipi di sensori per la generazione di mappe di profondità.

Il primo ad essere immesso in commercio è stato *Microsoft Kinect*.
Inizialmente conosciuto con il nome *Project Natal*, *Kinect* è un accessorio, originariamente pensato per la consolle *Xbox 360*, sensibile al movimento del corpo umano ma che ben presto è diventato strumento cardine utilizzato da numerosi sviluppatori di software per le più disparate applicazioni.
A differenza del *Wiimote* della *Nintendo* e del *PlayStation Move* della *Sony* esso rende il giocatore controller della console senza l'uso di strumenti, come invece accade per i concorrenti.

Successivamente la società taiwanese *Asus* ha creato un sensore che sfrutta la stessa tecnologia col nome di *Xtion* disponibile in tre versioni *Xtion, XtionPro e XtionPro Live.*
Attualmente anche la stessa *PrimeSense* produce un proprio sensore con il nome *PrimeSense 3D Sensor*.

Fig. 8 Sensori di visione 3D. **(a)** PrimeSense 3D Sensor [49]. **(b)** Microsoft Kinect[50]. **(c)** Asus Xtion PRO[51]. **(d)** Asus Xtion PRO Live [52].

A differenza di quello sviluppato da *Microsoft* questi dispositivi sono nati con l'intento di fornire agli utenti uno strumento per lo sviluppo di soluzioni software PC motion sensing.

Grazie ad essi gli sviluppatori Windows e Linux hanno la possibilità di muoversi su un terreno davvero stimolante per realizzare una nuova generazione di applicazioni che possono rivoluzionare l'approccio tradizionale all'utilizzo del PC e non solo.

Come illustrato in precedenza tutti questi dispositivi montano un sensore di profondità a raggi infrarossi composto da un proiettore IR e da una telecamera sensibile alla stessa banda.
L'informazione ottenuta attraverso il sistema proiettore-ricevitore permette, previa elaborazione interna,di fornire in output una mappa di profondità della scena analizzata.

Questa tecnologia fa si che mediante una singola acquisizione di un singolo dispositivo sia possibile ottenere informazioni sullo spazio 3D osservato, informazioni utilizzabili in un'infinità di applicazioni: dal gioco alla sorveglianza, dalla *computer graphics* all'analisi del movimento.
E' possibile eliminare mouse, tastiere e telecomandi: persone disabili potrebbero utilizzare questi dispositivi per abbattere numerose barriere che impediscono loro l'utilizzo della tecnologia.
Nel campo della robotica è possibile, utilizzando la visione artificiale, far volare un elicottero, far muovere degli automi o un piccolo veicolo, evitando ostacoli mediante la creazione di una mappa 3D dell'ambiente.
In ambito medico è possibile risparmiare enormi budget per la realizzazione di un sistema di *MoCap* con applicazioni anche per l'intrattenimento e la *computer vision*.
La prima cosa che bisogna fare per poter lavorare con questi sensori è analizzarne l'hardware e il supporto software fornito agli sviluppatori e in secondo luogo indagare i driver e le funzionalità fornite dalle librerie annesse.

Nel scegliere il dispositivo da utilizzare per lo svolgimento della tesi si sono analizzati i dispositivi *Microsoft Kinect* [30] e quello sviluppato da *Asus: Xtion Pro Live* [32] .

4.2.1 Microsoft Kinect

La telecamera RGB montata sul sensore permette di acquisire immagini *UXGA c*on una risoluzione massima di 1600x1200 pixel e permette di lavorare fino a 60fps, mentre quella a infrarossi usa una matrice di 640x480 pixel con una frequenza di 30fps (è possibile lavorare anche a frequenze maggiori ma con risoluzioni inferiori pari a 320x240).

Il sensore presenta inoltre un array di microfoni utilizzato dal sistema per la calibrazione dell'ambiente in cui ci si trova, mediante l'analisi della riflessione del suono sulle pareti e sull'arredamento. In tal modo il rumore di fondo e i suoni del gioco vengono eliminati ed è possibile riconoscere correttamente i comandi vocali dell'operatore.
Il dispositivo, in aggiunta, è dotato di un *Tilt Motor:* si tratta di un motorino elettrico che consente i movimenti della videocamera, al fine di ottenere una migliore inquadratura dell'utente; *la A.M.U.* (*Acceleration Measurement Unit*) infine è un'unità di misura del vettore accelerazione: nello stato di quiete la *A.M.U.* fornisce il vettore dell'accelerazione gravitazionale.

Di fatto la periferica, inizialmente sviluppata per applicazioni ludiche in accoppiamento alla consolle *XBox 360*, permette all'utente di interagire con la console senza l'uso di alcun controller da impugnare, ma solo attraverso i movimenti del corpo, i comandi vocali o attraverso gli oggetti presenti nell'ambiente.
La tecnologia di *Kinect* riesce a codificare le informazioni che le servono nel momento stesso in cui la luce viaggia, analizzando le deformazioni incontrate nel suo percorso.
Quello che ne deriva è una vera e propria renderizzazione 3D dell'ambiente in tempo reale, molto più precisa che in passato. Il sistema è teoricamente in grado di misurare le distanze all'interno di un area di 2 metri con un margine di errore di 1 cm, anche se questi parametri di precisione forniti da *Microsoft* non sono stati ancora testati realmente.
Attualmente è in studio un nuovo sistema di gestione della periferica, che permette di portare il tempo di lavorazione delle immagini da 150 a 5 ms arrivando ad ottenere 200 elaborazioni circa al secondo, in modo da annullare la lag.

Fig. 9 Microsoft Kinect

Le *SDK* ufficiali per l'utilizzo su PC sono state rilasciate da *Microsoft* solo nel luglio del 2011, mentre driver non ufficiali, basati sul *framework OpenNI* (*Open Natural Interaction*) introdotto nel prossimo capitolo, che sono usciti molti mesi prima hanno permesso agli sviluppatori di software di testare il funzionamento di *Kinect* e di creare una moltitudine di applicazioni basate sul dispositivo lanciato da *Microsoft*.

Di seguito sono riportate le specifiche tecniche di *Kinect*, e le caratteristiche dei suoi componenti.

	Kinect
Campo visivo (gradi)	*58° H, 45°V, 70°D **
Sensori	*RGB & Depth*
Range di lavoro	*tra 0.8 e 3,5 m*
Margine di lettura dei movimenti	*± 27°*
Audio	*16 Bit a 16 KHz*
Mappa di profondità	*VGA (640x480)* *QVGA (320x240)*
Risoluzione x/y (a 2 m dal sensore)	*3mm*
Risoluzione z (a 2 m dal sensore)	*10mm*
Immagine a colori RGB	*UXGA (1600x1200)*
Frame-rate	*30 fps VGA* *60 fps QVGA* *60fps UXGA*
Piattaforme	*Intel X86 & AMD*
SO supportati	*Windows 32/64 : XP, Win 7* *Linux Ubuntu 10.10 32/64 bit*
Interfaccia	*USB 2.0*
Programmazione	*C++/C#(Windows)* *C++ (Linux)*
Utilizzo	*Interno*
Dimensioni	*25 x 5 x 5 cm*
Sistema riconoscimento utenti	*Fino a 6 persone e* *19 articolazioni per singolo giocatore*
Temperatura di lavoro	*0° - 40° C*

**H: Horizontal, V: Vertical, D: Diagonal*

Tab. 1 Specifiche tecniche di Microsoft Kinect

4.2.2 Asus Xtion Pro Live

Il sensore *Xtion PRO Live* [13] è il più completo dei tre sviluppati da *Asus*. Come quello fornito da *Microsoft* oltre al sistema proiettore-telecamera IR per l'informazione di profondità dispone di una telecamera RGB e di un set di microfoni per il comando vocale.

Fig. 10 Asus Xtion Pro Live

La telecamera *motion sensing Xtion Pro Live*, a differenza di *Kinect*, è stato inizialmente pensato per il pc e non per il *gaming*.
La videocamera è stata proposta con un kit di sviluppo software per chi intende sfruttare il nuovo controller del movimento corporeo e dare vita ad una nuova generazione di applicazioni.
Asus Xtion Pro è in grado di rilevare suoni e gesti fino a 3,5 metri di distanza con ampio raggio di azione. E' in grado di calcolare oltre alla posizione anche le rotazioni dei punti in movimento, caratteristica che ne permette un campo di utilizzo ancora maggiore.
Il dispositivo è del tutto compatibile con i driver *PrimeSense* e con le librerie *OpenNI* e *NITE*. In sostanza quindi le stesse applicazioni che si sviluppano per il *Kinect*, utilizzando tali librerie, possono essere utilizzate indifferentemente con l'*ASUS Xtion PRO Live*.

Una importante innovazione, rispetto a Kinect, è la possibila di sfruttare la trasmissione wireles mediante il sistema chiamato *Wavi Xtion*. Si tratta di un *set-top-box* composto da:

- le due basi *Wavi*, che trasmettono e ricevono segnali; ad esempio trasferire via *HDMI* flussi video in alta definizione a un display esterno o l'audio fino a 7.1 canali a un sistema audio, ma anche trasferire in modalità wireless gli input ricevuti da periferiche USB, come ad esempio tastiera o mouse.
- **l**a videocamera.

Per un corretto funzionamento una base *WAVI* va posizionata vicino al computer e una accanto al televisore o al monitor, mentre alla videocamera, posta davanti allo schermo, è affidato il compito di riprendere i movimenti. Questa disposizione dei componenti è differente rispetto a quella che si presenta con *Microsoft Kinect*, dove la webcam è direttamente connessa alla console o al pc.

L’unità *Wavi* non è però stata utilizzata nel progetto di tesi.

Fig. 11 ASUS Wavi Xtion Live [53]

Sempre in linea con la politica che vede *Xtion* come strumento nato per lo sviluppo di applicazioni non necessariamente rivolte al gioco, *Asus* mette a disposizione anche una piattaforma di store online dove gli sviluppatori possono caricare le proprie applicazioni per raggiungere tutti i potenziali utenti di questa tecnologia.
+

	Xtion PRO Live
Campo visivo (gradi)	*58° H, 45°V, 70°D **
Sensori	*RGB & Depth*
Range di lavoro	*tra 0.8 e 3,5 m*
Mappa di profondità	*VGA (640x480)* *QVGA (320x240)*
Risoluzione x/y (a 2 m dal sensore)	*3mm*
Risoluzione z (a 2 m dal sensore)	*10mm*
Immagine a colori RGB	*SXGA (1280x1024)*
Frame-rate	*30 fps VGA* *60 fps QVGA*
Piattaforme	*Intel X86 & AMD*
SO supportati	*Windows 32/64 : XP, Win 7*

	Linux Ubuntu 10.10 32/64 bit
Interfaccia	*USB 2.0*
Programmazione	*C++/C#(Windows)* *C++ (Linux)*
Utilizzo	*Interno*
Dimensioni	*18 x 3.5 x 5 cm*
Note	*In abbinamento al kit di sviluppo Software*

**H: Horizontal, V: Vertical, D: Diagonal*

Tab. 2 Specifiche tecniche di Asus Xtion Pro Live

4.2.3 Confronto

Il sensore sviluppato da casa *Microsoft* e l'*ASUS Xtion PRO Live*, condividendo la stessa tecnologia sviluppata da *PrimeSense* hanno caratteristiche molto simili se non per alcune particolarità che possono essere poste in secondo piano per il lavoro che si intende svolgere nel progetto di tesi.

Le sostanziali differenze di *Asus Xtion Pro Live* rispetto al sensore di casa *Microsoft* sono l'assenza del motore che regola l'inclinazione della barra orizzontale, il fatto che *Xtion Pro* presenta dimensioni più compatte e che non necessita di un cavo di alimentazione aggiuntivo, come nel caso del *Kinect*. Il sensore infatti si alimenta direttamente con il cavo USB in dotazione. L'ultima, e forse l'unica, sostanziale differenza è rappresentata dal fatto che il dispositivo marchiato *Asus* non supporta i driver *SDK* forniti da *Microsoft*, obbligando quindi gli sviluppatori ad adottare le librerie *OpenNI* e *NITE*.

Nonostante ciò le dimensioni ridotte e l'assenza di un alimentatore esterno, in aggiunta alla possibilità di utilizzo *wireless* mediante la stazione *Wavi* hanno fatto sì che la scelta aziendale ricadesse sul dispositivo *Asus*.

Device	**Pro**	**Contro**
Microsoft kinect	*Alta qualità dei driver* *Stabilità con hardware diversi* *Motorizzato*	*Dimensioni* *Peso* *Alimentazione*
Asus Xtion PRO Live	*Dimensioni più compatte* *Più leggero* *Alimentato via USB* *Frame-rate di 60 fps* *Possibilità di utilizzo Wireless*	*Meno diffuso* *Non compatibile col le SDK Microsoft* *Non motorizzato*

Tab. 3 Confronto tra i due dispositivi *Microsoft Kinect* e *Asus Xtion Pro Live*

4.3 Calibrazione di telecamere D-RGB

In letteratura [4] sono riportate diverse applicazioni dove l'informazione fornita dalla mappa di profondità viene integrata con le immagini provenienti da una o più telecamere standard che inquadrano la scena. Anche in questo caso si rende necessaria una fase preliminare di calibrazione in modo da rendere consistente l'informazione proveniente dai vari dispositivi.

In casi come questo di sistemi eterogenei la calibrazione viene effettuata in maniera del tutto similare a quanto già illustrato per telecamere standard nel caso *markerless* adottando particolari accorgimenti per quanto riguarda il dispositivo D-RGB.

L'idea è quella di procedere alla calibrazione mediante il metodo di *Bouget* utilizzando però non le mappe di profondità bensì le immagini acquisite dalla telecamera IR montata sul dispositivo.

Per far ciò occorre escludere temporaneamente il proiettore IR in modo da non ottenere nelle immagini acquisite il pattern proiettato che ne degrada fortemente la qualità riducendo l'accuratezza con cui è possibile localizzare nel piano immagine la posizione dei punti di controllo (nel caso della scacchiera gli spigoli delle diverse caselle).

Il grosso limite di utilizzo è pertanto la necessità di illuminare la scena con un illuminatore IR esterno (di cui spesso non si ha disponibilità) in modo che le immagini acquisite dalla telecamera, sensibile a questa stessa lunghezza d'onda, contengano informazione utile e utilizzabile. In questo modo è quindi possibile procedere alla calibrazione del sistema secondo quanto già illustrato nei capitoli precedenti senza la necessità di introdurre nuove particolari elaborazioni o procedimenti.

La stessa tecnica può essere adottata per la calibrazione di sistemi omogenei interamente composti da sensori D-RGB. Anche in questo caso, lavorando sulle immagini acquisite dalle telecamere a infrarossi montate sui dispositivi, è possibile risalire ad una stima dei parametri intrinseci ed estrinseci del sistema , purché vengano adottate quelle accortezze descritte in precedenza.

Una tecnica leggermente diversa per la calibrazione di telecamere D-RGB è illustrata da [1]. L'autore sostiene la possibilità di effettuare tale processo di calibrazione attraverso il metodo di Bouget senza passare per l'immagine IR, con le problematiche annesse legate all'illuminazione esterna della scena, ma lavorando direttamente sulle mappe di profondità per le quali questi dispositivi sono stati pensati.

Anche questa tecnica, come la precedente,non sfrutta, l'informazione sulla tridimensionalità ma richiede l'utilizzo di una specifica apparecchiatura per poter procedere.

Se, a scopo d'esempio, si acquisisce una mappa di profondità di una scacchiera tradizionale, il risultato che si ottiene è qualcosa di molto simile a quello riportato nella seconda riga di figura 12 dove non si evince alcuna informazione sulla posizione delle caselle di diverso colore.

Secondo *Kai Berger et al.*, la soluzione al problema risiede nella costruzione di una particolare scacchiera nella quale i tasselli neri (o bianchi indifferentemente) vengono sostituiti da quadri delle stesse dimensioni di alluminio rifrangente: la presenza dei fogli di alluminio fa si che il pattern IR proiettato venga riflesso "all'infinito" rendendo totalmente assente l'informazione acquisita dalla telecamera in corrispondenza di tali regioni.

Quello che si ottiene è rappresentato in figura:

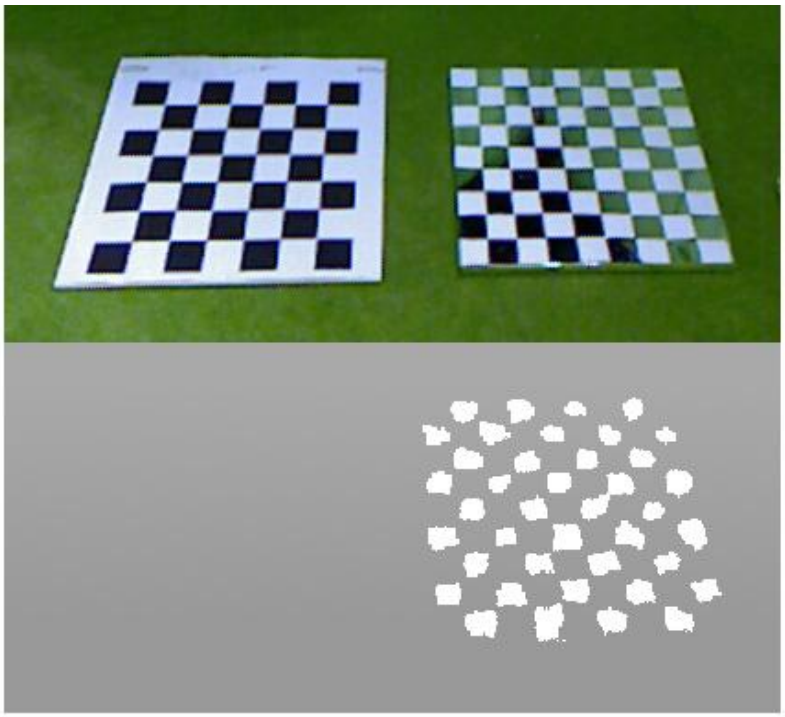


Fig. 12 In alto a sinistra scacchiere utilizzata per la calibrazione di telecamere standar, a destra tool per la calibrazione di sensori D-RGB. In basso mappa di profondità relativa alla scena riportata nella prima riga [1]

Nella prima riga è riportata, partendo da sinistra, l'immagine RGB della scacchiera normale e della scacchiera costruita secondo le indicazioni di Berger at al..

Nella seconda riga viene invece riportata la mappa di profondità corrispondente alla scena sopra. Ancora una volta si sottolinea come la mappa della scacchiera classica non rechi alcuna informazione mentre nell'immagine a destra è possibile distinguere i tasselli ricoperti d'alluminio in quanto caratterizzati da un colore bianco.

In questa seconda colonna infatti, ai tasselli neri (non alterati rispetto alla scacchiera precedente) viene assegnata una tonalità di grigio in tutto identica a quella con cui viene rappresentato lo sfondo in quanto stimati esser posti alla stessa distanza dall'obbiettivo (se non per la differenza legata allo spessore della scacchiera stessa).
I pixel appartenenti ai tasselli ricoperti d'alluminio, invece, rivelano una intensità nulla: il che non significa siano mappati ad una distanza nulla dall'obiettivo, bensì che da quelle regioni non viene acquisito dalla telecamera alcun segnale.

In questo modo è possibile, senza dover ricorrere all'utilizzo dell'immagine IR e di un illuminatore esterno, risalire alla posizione nel piano immagine dei vertici della scacchiera e dei tasselli in modo da semplificare il processo di calibrazione secondo il metodo classico di *Bouget*.

L'implementazione del metodo illustrato in [1], in realtà, non ha fornito i risultati sperati: dalla stessa figura 12 fornita dagli autori, si nota infatti che la forma dei tasselli viene ricostruita in maniera assai distorta impedendo una accurata ricostruzione dei vertici degli stessi, facendo così venir meno l'affidabilità della calibrazione.

Oltre a quello appena descritto entrambe le tecniche presentano un limite comune che risiede nel fatto che entrambe permettono la stima dei parametri estrinseci, vere incognite del problema dato che il dispositivo di per sé è in grado di determinare la posizione 3D di un punto inquadrato, basandosi sulle immagini IR e non sui dati di profondità per cui il sensore stesso è stato costruito. È quindi possibile che il modello utilizzato dai progettisti, e sconosciuto all'operatore, non sia lo stesso applicato in fase di stima dei suddetti parametri rendendo inconsistenti i risultati ottenuti.

A questo punto la domanda sorge spontanea: a quale scopo calibrare due o più sensori di profondità se già con un singolo dispositivo è possibile risalire alla mappa di profondità della scena?

Lo scopo di ciò nonché scopo stesso del lavoro di tesi è quello di definire se e quanto l'utilizzo contemporaneo di due o più dispositivi di questo tipo, oltre a permettere di inquadrare la scena da punti di vista multipli eliminando o riducendo il problema dell'occlusione, permette una migliore accuratezza nel risalire all'informazione 3D della scena inquadrata.

4.4 Librerie per l'utilizzo dei sensori

4.4.1 Librerie OpenNI

A differenza del dispositivo sviluppato da *Microsoft* (*Kinect*), nel caso del sensore *XtionPro Live* è presente sul mercato un solo tipo di driver, che ne consenta l'utilizzo con il computer:

- *PrimeSense OpenNI* ;

Questi driver rilasciati sul mercato hanno permesso agli sviluppatori di software di testare il funzionamento del sensore e di creare delle applicazioni basate su questa videocamera.
OpenNI ovvero *Open Natural Interaction* (*NI*) è un *framework* multi-lingua, *cross platform* (cioè indipendente dal sistema operativo utilizzato) rilasciato sotto licenza GNU GPL che definisce le *Application Programming Interface* (*API*) per scrivere applicazioni che sfruttano la *Natural Interaction*.
Le *API* di OpenNI presentano un insieme di interfacce per la scrittura di applicazioni *NI* che prevedono l'utilizzo di dispositivi come *Kinect* o l'intera linea *Asus Xtion*.
L'intento principale del *framework* è quello di rendere disponibili delle *API* standard che consentono la comunicazione con:

- sensori audio e video;
- *middleware* di percezione audio-visiva (componenti software che analizzano ed elaborano i dati audio e video che vengono registrati dalla scena).

Il *framework OpenNI* è un livello astratto che fornisce l'interfaccia sia per i dispositivi fisici che per i componenti *middleware* (come mostrato in figura 13).
Le *API* permettono la registrazione di più componenti nel *framework OpenNI* che sono indicati come moduli, e sono utilizzati per produrre ed elaborare i dati forniti dai sensori (audio e video) mentre il *middleware* permette di scrivere applicazioni che elaborano dati senza doversi preoccupare del sensore che li ha prodotti.

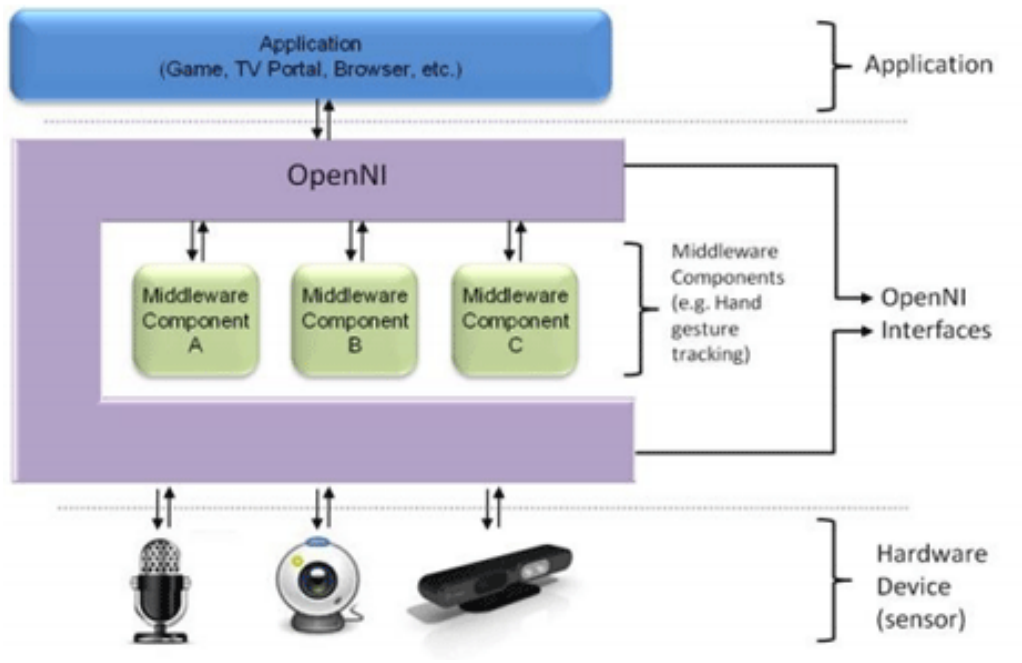


Fig. 13 Astrazione dei livelli di funzionamento di OpenNI [31].

Il livello più alto rappresenta il software che fa uso di *OpenNI* implementando le *Natural Interaction*. Il livello centrale (*middleware*) rappresenta invece l'interfaccia *OpenNI* con le sue capacità di comunicare con i vari sensori e con le funzionalità disponibili (*Skeleton Tracking*, *Hand Tracking*, ecc.).
Il livello più basso è il livello hardware composto da tutti i sensori che possono inviare dati audio o visivi. Questi componenti sono indicati come moduli ed attualmente quelli supportati dalle *API* sono:

Moduli per i sensori

- sensore 3D
- fotocamera RGB
- dispositivo audio (un microfono o un *array* di microfoni)

Moduli *middleware*

- Analisi totale del corpo: è un componente software che elabora i dati sensoriali e genera informazioni relative al corpo come una struttura dati che descrive le articolazioni, il loro orientamento, il centro di massa del corpo e molto altro.
- Analisi della mano: è un componente software che elabora i dati sensoriali, individua la sagoma di una mano assegnando un punto alla posizione del palmo.
- Rilevamento gesto: è un componente software che identifica gesti predefiniti (*Push, Wave, Circle*) associandoli ad eventi.
- Analisi della scena: è un componente software che analizza l'immagine della scena al fine di produrre informazioni come individuare più persone nella scena, trovare le coordinate del piano, separare oggetti in primo piano da quelli sullo sfondo.

OpenNI ha rilasciato i driver e i propri codici sorgente per far sì che le sue librerie vengano implementate su più architetture possibili e su qualsiasi sistema operativo, in modo da accelerare l'introduzione di applicazioni di *Natural Interaction* sul mercato.
Con l'uscita del *Kinect* la popolarità di *OpenNI* è nettamente aumentata, grazie anche alla creatività dei numerosi sviluppatori che lavorano con queste librerie. Va sottolineato che *OpenNI* non è *Kinect* o *Xtion Pro*, ma la facilità del framework di comunicare con qualsiasi sensore ha semplificato l'uso di tali dispositivi.

Le classi messe a disposizione dalla libreria *OpenNI* permettono di ricevere ed utilizzare i dati ottenuti da un generico sensore, senza preoccuparsi del tipo di hardware che li fornisce.
Per quanto riguarda l'uso di dispositivi D-RGB la creazione di mappe di profondità non è l'unica funzionalità che *OpenNI* fornisce. La libreria mette a disposizione tutta una serie di classi che permettono di individuare un nuovo utente quando questo entra nel campo visivo del sensore ed associa ad esso un colore per distinguerlo da altri. Fornisce metodi ed algoritmi di ricerca di una determinata posa, permettendo la calibrazione dell'utente e consentire la creazione di uno scheletro basato sul proprio corpo.
Tale scheletro tiene traccia dei movimenti dei vari arti del corpo in uno spazio in tre dimensioni.
Altre classi permettono di individuare il movimento delle mani, di riconoscerle e di tenerne traccia per mezzo dell'assegnazione di un "*hand point*". Tale punto identifica il palmo della mano dell'utente che verrà
usato per interfacciarsi con il sensore ed i programmi.
OpenNI mette a disposizione anche un nodo *AudioGenerator* che consente di raccogliere audio all'interno di un buffer con una qualità definita dall'utente in base alla qualità dei microfoni a disposizione.
Infine, come già detto, è possibile registrare i dati generati dal sensore e salvarli su file con il formato proprietario (*oni*).
Proprio come ROS11 anche *OpenNI* ha rilasciato i driver ed ha aperto i propri codici sorgente per far si che le sue librerie vengano implementate su più architetture possibili e su qualsiasi sistema operativo, in modo da accelerare l'introduzione di applicazione di *Natural Interaction* sul mercato.
In appendice C è riportato un esempio di utilizzo delle *OpenNI* per la generazione di mappe di profondità.

4.4.2 Librerie NITE

Il *PrimeSense NITE* ™ è il più avanzato e robusto *middleware* per la computer vision 3D oggi disponibile.
Mentre le librerie *OpenNI* restituiscono dati a basso livello come mappe di profondità, mappe RGB, audio e quant'altro *NITE* utilizza questi dati in entrata dal dispositivo hardware all'interno di algoritmi che permettono di ottenere dati ad un livello superiore.
Si tratta di un *toolbox* implementato sulle interfacce *OpenNI* che fornisce funzionalità aggiuntive e facilita la creazione di applicazioni di controllo basato sul movimento delle mani dell'utente e sullo scheletro.
Attualmente *NITE* è disponibile per i sistemi *Windows*, *Linux* e *Mac*.
NITE contiene implementazioni per tutti i moduli della libreria *OpenNI*. Il funzionamento di *NITE* è basato sui livelli illustrati in figura 14.

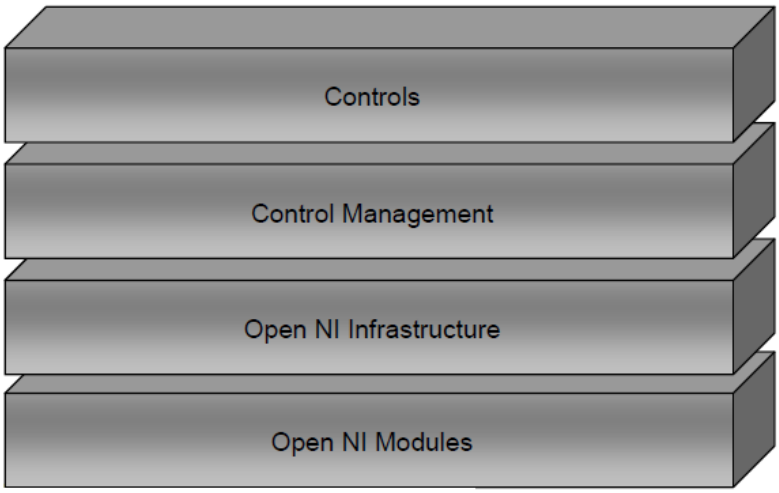


Fig. 14 Schema dei livelli NITE [29]

In particolare:

- ***openni modules*:** i moduli di *OpenNI* supportati da *NITE* sono *Gesture Generator*, *Hands Generator* e *User Generator*. Inoltre supporta lo *Skeleton Tracking*.
- ***openni infrastructure***.
- ***control management*:** riceve un insieme di punti e li indirizza verso il livello di controllo.
- ***controls*:** ogni controllo riceve un insieme di punti e li traduce in un'azione specifica di tale controllo. Successivamente viene richiamata una funzione *(callback*) dall'applicazione che può cambiare il modulo di controllo attivo nel livello *Control Management*. Questo definisce il flusso dell'applicazione.

Un esempio di controllo è il controllo del punto, che permette di registrare quando un punto viene creato, quando scompare e quando si muove. In realtà i controlli sono degli oggetti sempre in ascolto che attendono l'arrivo di nuovi dati ad ogni fotogramma. Le funzioni di *callback* saranno invocate solo quando un determinato evento si verifica.

Per capire con degli esempi di cosa si occupano i controllori è possibile fornire una lista di alcuni esempi con i relativi eventi associati:
- ***Push Detector*:** Il controllore cerca di riconoscere il moto della mano in avanti e indietro nella direzione del sensore come il movimento di una spinta. Riconoscere questo potrebbe permette ad un utente di aprire una cartella su desktop.
- ***Swipe Detector:*** Il controllore cerca di riconoscere il movimento di una mano che si muove in una direzione specifica (ad esempio su e giù o destra e sinistra), con una traiettoria rettilinea che termina in un punto. Questo evento potrebbe essere usato per far scorrere le pagine di un documento pdf.
- ***Steady Detector:*** Il controllore cerca di riconoscere quando una mano è ferma per qualche istante. Questo è utile per passare da un controllo ad un altro ed avere un punto di riferimento iniziale fisso.
- ***Wave Detector:*** Il controllore prende come evento l'onda, ovvero un movimento della mano che all'interno di un intervallo di tempo effettua un movimento contiguo con quattro cambi di direzione. Più avanti verrà spiegato nel dettaglio.
- ***SelectableSlider1D:*** Cerca di riconoscere il movimento della mano come punto di un cursore. Viene preso in considerazione uno spazio tridimensionale e gli assi sono così definiti: x (sinistra-destra), y (su–giù), z (avanti-indietro). Questo controllore può esser usato per implementare dei menù.
- ***SelectableSlider2D:*** Questo punto di controllo cerca di riconoscere il movimento della mano come punto di un dispositivo di scorrimento 2D sul piano XY.
- **Circle Detector:** L'utente disegna due circonferenze complete, in senso orario (considerato positivo) e antiorario (negativo), a questo punto il generatore di output seguirà i movimenti circolari successivi. Questo controllo molto probabilmente è stato utilizzato per far funzionare una versione simulata dell'interfaccia tastiera di *8Pen Android* per scrivere testo su *Windows 7*.

NITE permette inoltre di creare delle proprie classi che ereditano tutte le funzionalità delle classi base e mette a disposizione dei filtri che puliscono i dati ricevuti rimuovendo rumore o dati non utili. Da qui nasce il problema di capire cosa è rumore da filtrare e cosa no, per fare questo è possibile creare dei vincoli addizionali nei controllori. Ad esempio si può definire un intervallo di tempo all'interno del quale un gesto può essere riconosciuto, mentre tutto quello che viene immediatamente prima o dopo è rumore.
Introducendo le possibilità offerte agli sviluppatori da queste librerie è bene aprire una breve parentesi per illustrare un concetto di base nell'utilizzo di tali dispositivi.

Come facilmente ipotizzabile gli algoritmi sviluppati da *Microsoft* e da *PrimeSense* sono *closed-source* e non è perciò possibile modificare o adeguare l'elaborazione interna del sensore ai propri scopi; ecco quindi che le strade percorribili che si presentano agli sviluppatori sono due: da una parte è possibile porsi in quello che si può definire un "alto livello", ossia, fare affidamento a dati già processati internamente al sensore e forniti in output sottoforma di punti chiave in grado di descrivere la posa del soggetto, utilizzando tali informazioni all'interno dei propri algoritmi . Dall'altro lato è possibile invece utilizzare questi strumenti ponendosi ad un livello inferiore andando cioè a lavorare ed elaborare direttamente le mappe di profondità "grezze" sviluppando algoritmi in grado di ricavarne informazioni specifiche per i propri scopi.

In appendice C è riportata la procedura di *skeleton tracking* effettuata mediante le librerie *NITE*.

4.4.3 Librerie OpenCV

OpenCV è una libreria *open-source*, scritta in linguaggio *C#*, per lo *streaming video real-time*, ovvero la *computer vision*.

In quanto libreria multipiattaforma, *OpenCV* è compilabile sotto molti sistemi operativi (*Windows, Mac OS X, Linux, PSP, VCRT*).

La *computer vision* è una trasformazione dei dati provenienti da una fotocamera o videocamera verso una decisione o una loro nuova rappresentazione. Una tale trasformazione può essere effettuata per una moltitudine di possibili scopi.
Nel 2011 è stata rilasciata la versione 2.2.0, che supporta anche *Andriod*. Ora è disponibile la *release 2.3.1*.

La struttura di *OpenCV* consta delle seguenti 6 parti principali:

- **CV:** kernel delle funzioni di IP/CV
  - *Image processing*
  - *Motion analysis*
  - *Pattern recognition*
  - *3D reconstruction*
  - …
- **CxCore:** contiene la definizione di tutte le strutture dati e le funzioni per gestire immagini e video.
  - Strutture dati principali
  - Accesso alle strutture dati (creazione, inserimento/modifica valori, copia, ..)
  - Operazioni su matrici (aritmetiche, logiche, trasformazioni, permutazioni, ..)
  - Disegno (punti, linee, curve, figure, ..)
  - …
- **HighGui**: semplici operazioni di I/O, contiene le definizioni delle interfacce utenti (GUI):
  - Creazione/distruzione finestre per mostrare immagini
  - Apertura/salvataggio immagini
  - Gestione flusso video (da file e da cam)
  - …
- **CvCam:** contiene le interfacce per le webcam.
- **ML** (Machine Learning): contiene molte funzioni sul *Machine Learning*, quali il *clustering* e la classificazione.
- **CVAUX:** contiene algoritmi sperimentali per scopi diversi, ad esempio: segmentazione, sottrazione del *background*, modelli *HMM*, ecc.

La struttura di OpenCV può essere riassunta nello schema seguente.

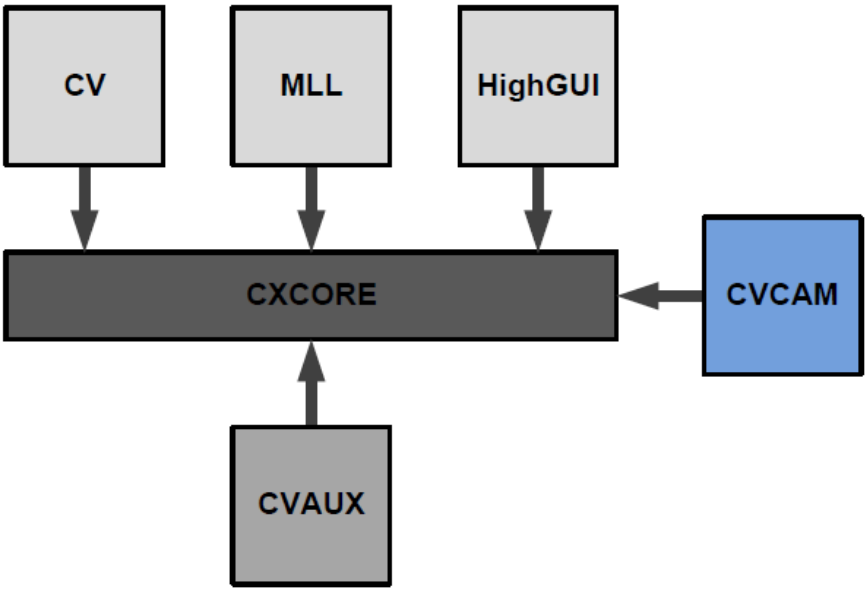


Fig. 15 schema delle diverse sezioni delle librerie OpenCV.

Di particolare interesse per i nostri scopi sono la sezione HighGUI utilizzata per caricare le immagini acquisite dal sensore e per la loro visualizzazione e la sezione CV utilizzata per effettuare il processamento e l'elaborazione delle immagini al fine di ricavarne le *features* di interesse.
La sezione CV prevede la definizione di un insieme di primitive utili per l'*image processing* e mette a disposizione dell'utente tutta una serie di filtraggi che sono stati impiegati nel progetto di tesi per l'elaborazione della mappa di profondità.

OpenCV utilizza le matrici come tipo di dato per memorizzare immagini e frame. All'interno di tali strutture troviamo i tipi elementari di punto, dimensione, rettangolo, che vengono definiti come riportato nella seguente tabella.

Struttura	**Contenuto**	**Descrizione**
CvPoint	*Int x,y*	*Punto in un'immagine*
CvPoint2D32f	*float x, y*	*Punto in R2*
CvPoint3D32f	*flaot x, y, z*	*Punto in R3*
CvSize	*int width, height*	*Dimensioni di un'immagine*
CvRect	*int x, y, width, height*	*Porzione di un'immagine*
CvScalar	*double val[4]*	*Valore RGBA*

Tab. 1 Principali tipi di dati implementati dalle *OpenCV*

Le matrici di dati di base presenti nelle *OpenCV* sono principalmente 2: *IplImage* e *cvMat*;
In generale l'uso delle *IplImage* è stato ormai rimpiazzato dalle più ampie e potenti *cvMat*.
In entrambi i casi si tratta comunque di immagini bidimensionali memorizzate sottoforma di *array* monodimensionali.

Sulle immagini è possibile utilizzare un insieme di oltre cento funzioni predefinite, che permettono di sommare, sottrarre, ruotare, determinare quantità statistiche, norme, ecc.
L'utilizzo di tali funzioni è abbastanza immediato. Per maggiori informazioni a riguardo, si rimanda alla guida di *OpenCV*, nonchè a [G. Bradski, A. Kaehler. Learnin OpenCV: Computer Vision with the OpenCV Library. O'Reilly, 2008.].

4.4.4 Librerie OpenGL

OpenGL (*Open Graphics Library*) è una specifica che definisce una API per più linguaggi e per più piattaforme per scrivere applicazioni che producono computer grafica 2D e 3D. Fra le caratteristiche possiamo trovare:

- Gestione delle viste ortogonali e prospettiche
- Illuminazione
- *Texturing*
- Trasparenze
- Effetti come nebbia, *motion blur* e messa a fuoco
- *Anti-aliasing*

L'interfaccia consiste in circa 250 diverse chiamate di funzione che si possono usare per disegnare complesse scene tridimensionali a partire da semplici primitive. Sviluppato nel 1992 dalla *Silicon Graphics Inc*., è ampiamente usato nell'industria dei videogiochi (nella quale compete con DirectX su Microsoft Windows), per applicazioni di CAD, realtà virtuale, e CAE. È lo standard di fatto per la computer grafica 3D in ambiente *Unix*.

A livello più basso *OpenGL* è una "specifica", ovvero si tratta semplicemente di un documento che descrive un insieme di funzioni ed il comportamento preciso che queste devono avere. Da questa specifica, i produttori di hardware creano implementazioni, ovvero librerie di funzioni create rispettando quanto riportato sulla specifica *OpenGL*, facendo uso dell'accelerazione hardware ove possibile. I produttori devono comunque superare dei test specifici per poter fregiare i loro prodotti della qualifica di implementazioni *OpenGL*.

OpenGL assolve due compiti fondamentali:

- nascondere la complessità di interfacciamento con acceleratori 3D differenti, offrendo al programmatore una API unica ed uniforme;
- nascondere le capacità offerte dai diversi acceleratori 3D, richiedendo che tutte le implementazioni supportino completamente l'insieme di funzioni *OpenGL*, ricorrendo ad un'emulazione software se necessario.

OpenGL è composto sostanzialmente da tre sotto-librerie:

- **GL** (*Graphics Library*) – Il nucleo della libreria grafica, qui vengono fornite le funzioni a basso livello come il tracciamento dei poligoni, l'illuminazione e la gestione dei vari buffer per il disegno.
- **GLU** (*OpenGL Utility Library*) – Estensione di GL con una serie di operazioni e primitive di utilità generale, implementate con le funzioni che si trovano in GL, qui si trovano quelle per il posizionamento di una telecamera o la gestione delle superfici *NURBS*.
- **GLUT** (*OpenGL Utility Toolkit*) – Libreria opzionale che fornisce al programmatore la possibilità di avere una gestione di un'interfaccia a finestre svincolata dal sistema operativo. In GLUT si trovano comandi in grado di aprire una finestra, controllarne lo stato, ricevere eventi dal mouse e dalla tastiera, gestire i menu ed in generale tutto quello che si può fare con un normale *window manager*.

Il compito di *OpenGL* è quello di ricevere primitive come punti, linee e poligoni, e di convertirli in pixel (*rasterizing* o rasterizzazione). Ciò è realizzato attraverso una pipeline grafica nota come (EN) *OpenGL state machine*. La maggior parte dei comandi *OpenGL* forniscono primitive alla pipeline grafica o istruiscono la pipeline su come elaborarle.

OpenGL è una *API* procedurale che opera a basso livello, che richiede perciò al programmatore i passi precisi per disegnare una scena. Questo approccio si pone in contrasto con le API descrittive ad alto livello le quali, operando su struttura dati ad albero (*scene graph*), richiedono al programmatore solo una descrizione generica della scena, occupandosi dei dettagli più complessi del *rendering.* La natura di *OpenGL* obbliga quindi i programmatori ad avere una buona conoscenza della pipeline grafica stessa, ma al contempo lascia una certa libertà per implementare complessi algoritmi di *rendering*.

Non include strutture dati o strumenti di alto livello per descrivere oggetti tridimensionali.
Scene complesse vengono disegnate in *OpenGL* tramite semplici primitive quali punti, linee e poligoni.
Per fare questo esegue a grandi linee i seguenti passi:

1) Gli oggetti vengono passati a *OpenGL* sottoforma di un insieme di primitive (punti, linee, poligoni, immagini e bitmap). Ne viene fornita una descrizione matematica.
2) Ogni elemento è posizionato nella scena e viene scelto un punto di vista
3) Viene calcolato il colore degli oggetti che può essere influenzato da un insieme di fattori (illuminazione, *texture*, proprietà del materiale)
4) La descrizione matematica degli oggetti e delle loro proprietà viene convertita in pixel sullo schermo (*rasterizzazione*).
In appendice A è riportato un esempio che utilizza le funzioni OpenGL per disegnare una piramide colorata.

4.4.5 QT

Qt è una libreria multipiattaforma per lo sviluppo di programmi con interfaccia grafica tramite l'uso di *widget* (congegni o elementi grafici). Qt, ampiamente utilizzato nell'ambiente *desktop KDE*, viene sviluppato dall'azienda Qt Software (meglio conosciuta come *Trolltech* o *Quasar Technologies*) di proprietà di *Digia*.

Qt usa il linguaggio *C++* standard con un estensivo uso del preprocessore *C#* per arricchire il linguaggio, ma esistono interfacce per *Java, Python, C#, Perl* e *PHP*. Gira sulle piattaforme principali ed integra funzioni per l'accesso ai database *SQL*, *parsing* di documenti *XML* ed *API* multipiattaforma per l'uso dei file.

Le librerie Qt sono disponibili per queste *piattaforme*:

- **Qt/X11** — Qt per *X Window System*;
- **Qt/Mac** — Qt per *Mac OS X di Apple*;
- **Qt/Windows** — Qt per *Microsoft Windows*;
- **Qt/Embedded** — Qt per piattaforme *embedded* (palmari e simili).

Attualmente vengono offerte in quattro edizioni, disponibili per ciascuna delle piattaforme:

- **Qt Console** — versione *embedded* per lo sviluppo senza interfaccia grafica;
- **Qt Desktop Light** — versione base per applicazioni con interfaccia grafica, senza supporto rete e *SQL*;
- **Qt Desktop** — la versione completa;
- **Qt Open Source Edition** — la versione completa, per sviluppo di software libero.

L'edizione commerciale per Windows supporta *Visual Studio*; tutte supportano comunque il compilatore *C++ GCC*.

La squadra di *KDE* ha anche rilasciato un'edizione di Qt, chiamata *Qt/Windows Free Edition* basata sul codice sorgente della versione Qt/X11 e licenziata sotto GPL, rilasciata per il progetto *KDE via Cygwin*. Questo è stato ispirato dal fatto che le versioni di Qt precedenti alla 4.0 non erano disponibili come software libero sotto Windows.

4.4.6 QtCreator e QtDesigner

Qt Creator è un ambiente di sviluppo integrato *cross-platform C++* , che fa parte del *SDK* di Qt. Esso comprende un *visual debugger* e un *designer* integrato per la creazione di *GUI*.

Qt Creator fornisce il supporto per la creazione e l'esecuzione di applicazioni Qt in ambienti *desktop* (*Windows, Linux, FreeBSD e Mac OS*) e mobili (dispositivi *Android ,Blackberry , Maemo e MeeGo*).

Qt Creator utilizza il compilatore *C++* della *GNU Compiler Collection* su *Linux* e *FreeBSD* . In Windows, con l'installazione di default, è possibile utilizzare *MinGW o MSVC* ma durante la compilazione dei file sorgenti è anche possibile usare cdb.

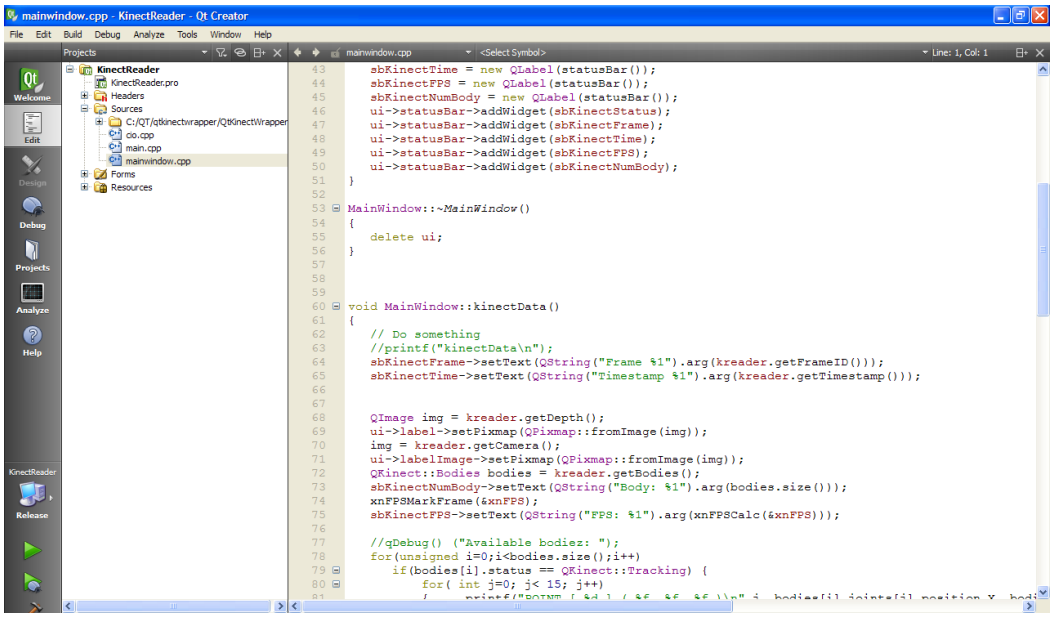


Fig. 16 Schermata di lavoro di *QtCreator*

Qt Creator include un *project menager* che utilizza un formato di file di progetto *cross-platform* del tipo .Pro. Un file di progetto può contenere informazioni sui file inclusi nel progetto stesso, sugli step di generazione personalizzata e le impostazioni per l'esecuzione delle applicazioni.

Qt Creator può utilizzare diversi sistemi di compilazione. Tuttavia, solo due di essi sono compresi da *Qt Creator: -qmake e CMake* . Altri possono essere utilizzati solo definendo il progetto come custom project.

Qt Creator include un editor di codice ed integra Qt Designer per la progettazione e la costruzione di interfacce utente grafiche (GUI) e di *widget* Qt.

L'editor di codice in *Qt Creator* oltre a supportare l'evidenziazione della sintassi per vari linguaggi è in grado di analizzare il codice in linguaggio C++ e QML e di fornire l'opzione di completamento del codice, di *context-sensitive help*,e *semantic navigation*.

Qt Designer è uno strumento per la progettazione e la costruzione di interfacce utente grafiche (GUI) e di *widget* Qt. E 'possibile comporre e personalizzare gli *widget* o le finestre di dialogo e verificarne le funzionalità utilizzando stili diversi direttamente nell'editor. Gli *Widget* e le forme create con *Qt Designer* sono quindi integrati con il codice utilizzando il meccanismo di trasmissione dei segnali "*signal & slot*" tipico del *framework Qt*.

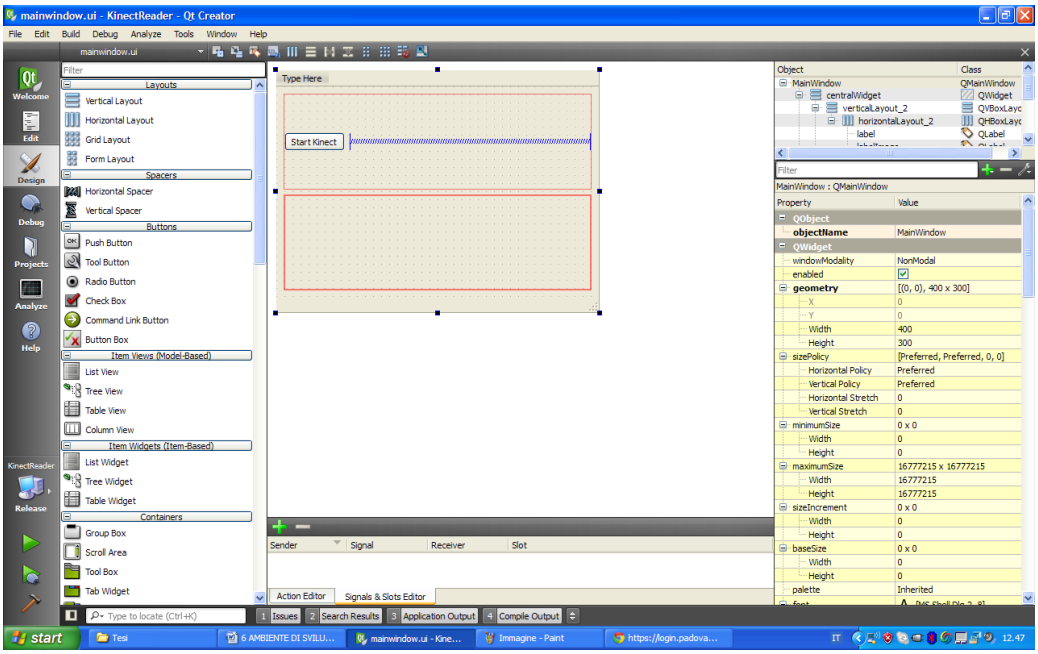


Fig. 17 Schermata di lavoro di QtDesigner

Qt Designer è uno strumento veloce per lo sviluppo di animazioni utilizzando un linguaggio di programmazione dichiarativa *QML* .

4.5 Applicazioni in letteratura

4.5.1 Introduzione

Costando poco più di 100$ non c'è da meravigliarsi che *Kinect* detenga il *Guinness World Record* per essere il dispositivo di elettronica di consumo più venduto.
Degli oltre 18 milioni di unità vendute fino ad oggi, la maggior parte viene impiegata dai consumatori desiderosi di esplorare nuovi modi di interagire con i loro giochi preferiti. Tuttavia, il dispositivo ha anche attirato l'interesse di sviluppatori software che desiderano scoprire se è possibile applicare il sistema 3-D di imaging a processi industriali, scientifici o riabilitativi.
A scopo d'esempio sono riportate di seguito alcune applicazioni nei vari campi della robotica e della medicina presenti in letteratura.

4.5.2 Robotica

Nel campo della robotica tale tecnologia è stata ampiamente analizzata al fine di creare una sorta di visione artificiale per comandare robot o manipolatori per l'industria o per definire la presenza di

difetti di fabbricazione durante il processo di costruzione di manufatti a livello industriale.

La possibilità di ottenere indicazioni sulla disposizione e conformazione dello spazio circostante attraverso l'elaborazione delle mappe di profondità ha permesso di compiere enormi passi in avanti nello sviluppo di robot in grado di orientarsi e farsi strada evitando gli ostacoli in modo automatico.

Per esplorare ambienti sconosciuti, i robot devono essere in grado di mappare e ricostruire l'ambiente circostante, stimando la distanza tra loro e le pareti vicine, l'altezza del soffitto o la presenza di ostacoli al fine di pianificare un percorso sicuro che gli permetta di raggiungere un qualche obiettivo.

Anche se una grande quantità di ricerca è stata dedicata allo sviluppo di mappe "una tantum" che i robot possono utilizzare per spostarsi all'interno di una ben determinata area, questi sistemi non sono in grado di adeguarsi ai cambiamenti che possono avvenire in continuazione nell'ambiente circostante nel corso del tempo; ad oggi se un robot riconosce degli oggetti non presenti in precedenza, è difficile incorporare tale informazione nella propria mappa, rendendo quasi impossibile la navigazione automatica in ambienti in evoluzione.
Il nuovo approccio in fase di sviluppo presso il Computer Science and Artificial Intelligence Laboratory (CSAIL) del MIT da parte del prof. John J. Leonard e lo studente laureato Hordur Johannsson, si basa su una nuova tecnica definita "*SLAM: Simultaneous Localization and Mapping*" (*localizzazione e mappatura simultanea*),e permetterà ai robot di aggiornare costantemente la mappa man mano che vengono apprese nuove informazioni nel corso del tempo. Il team ha già testato il metodo su robot dotati di costosi laser-scanner, ma ora grazie allo sviluppo di sensori quali *kinect e Asus Xtion PRO* ha inteso la possibilità di utilizzate questa nuova tecnologia a basso costo allo scopo.

Fig. 18 Robot sviluppato da John J. Leonard in collaborazione con Hordur Johannsson. Il sistema è in grado di muoversi all'interno di uno spazio in continua evoluzione grazie alla creazione di una mappa tridimensionale mediante l'utilizzo di Microsoft *Kinect* [54]

Grazie a questo nuovo tipo di approccio, come il robot attraversa una zona inesplorata, il sensore *Kinect* in base all'informazione unificata fornitagli dalla telecamera RGB e da quella IR è in grado di effettuare una scansione dei dintorni, costruendo un modello 3D delle pareti, della stanza e degli oggetti in essa contenuti. Quando poi il robot, spostandosi, raggiunge una zona non ancora mappata, il sistema confronta le caratteristiche della nuova immagine acquisita (come estremità delle pareti) con tutte le immagini precedenti finché non trova una corrispondenza.

Allo stesso tempo, il sistema di stima del movimento, utilizzando sensori di bordo che misurano la distanza e la rotazione delle ruote su cui è posto il robot è in grado, combinando l'informazione visiva con questi dati di movimento, di determinare dove il robot è posizionato all'interno dell'edificio in modo da eliminare gli errori che potrebbero generarsi se la localizzazione avvenisse unicamente in base ai sensori di bordo.

Una volta che il sistema è sicuro della sua posizione, le nuove informazioni che sono state immagazzinate dopo la scansione precedente possono essere incorporate all'interno della mappa combinando le immagini vecchie e nuove della scena aggiornando e ampliando la conoscenza del robot sull'ambiente che lo circonda.

Il team ha testato il sistema su una sedia a rotelle robotica, un robot *PR2* sviluppato da Willow Garage a Menlo Park, in California, e in un vestito- sensore portatile indossato da un volontario umano. È stato dimostrato essere in grado di mappare l'ambiente circostante muovendosi ad una velocità fino a 1,5 metri al secondo.

In definitiva, l'algoritmo potrebbe consentire ai robot di viaggiare in edifici, per uffici o in ospedale, pianificando le loro rotte senza alcun intervento umano. Secondo Seth Teller, capo della sezione *Vision Robotics* presso la *CSAIL* e ricercatore principale del progetto, potrebbe anche essere usato come dispositivo indossabile per i non vedenti, permettendo loro di muoversi , in maniera indipendente, in edifici anche di grandi dimensioni e affollati.

Questi studi prevedono inoltre numerose applicazioni militari, come la mappatura di una rete di bunker o di grotte per consentire una rapida uscita o un rientro in caso di necessità.

Un altro esempio di utilizzo di *Kinect* nel campo della robotica è stato sviluppato presso la Southwest Research Institute, dove i ricercatori hanno sviluppato un sistema basato su un manipolatore di casa Yaskawa Motoman Robotics dotato di un effettore a tre pinze in grado di individuare e raccogliere gli oggetti disposti casualmente all'interno dello spazio di lavoro (vedi fig. 19).

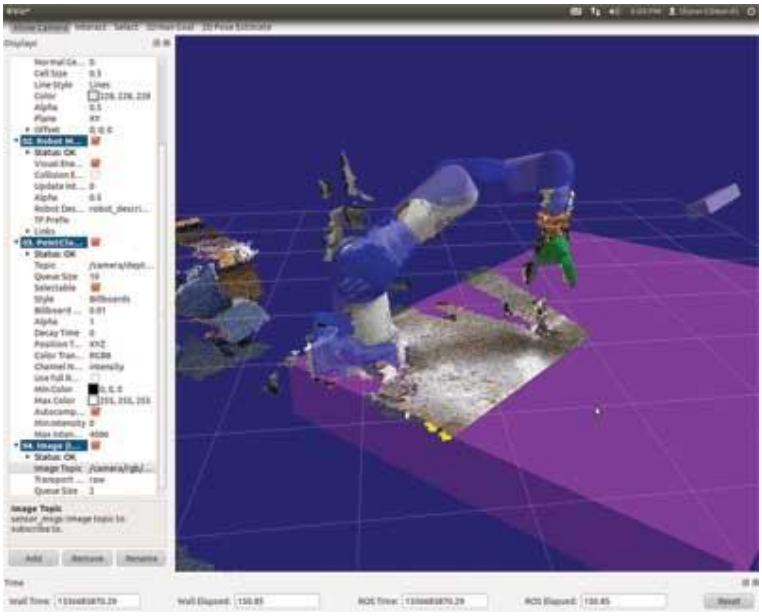


Fig. 19 Southwest Research Institute sviluppatori hanno costruito un sistema dotato di una pinza a tre dita robotico in grado di identificare e raccogliere oggetti posizionati in modo casuale. In questo caso, i dati processati da Kinect vengono visualizzati insieme ai dati CAD del robot e la pinza. [55]

Dopo aver acquisito i dati dal sensore *Kinect*, i punti catalogati come appartenenti allo sfondo vengono rimossi per isolare gli oggetti di interesse. I punti 3-D che rappresentano l'oggetto vengono quindi elaborati al fine di ricostruire un modello 3D dello stesso il quale viene analizzato in modo da identificarne il punto di presa più corretto. Grazie a queste informazioni attraverso un algoritmo di pianificazione il braccio robotico viene automaticamente posizionato nella posa più consona per afferrare l'oggetto.

4.5.3 Medicina e medical surgery

Oltre alla robotica l'utilizzo delle depth maps può risultare estremamente utile per un'innumerevole quantità di applicazioni in tutti i settori dell'ingegneria e non solo.

Per quanto riguarda il campo della medicina questo strumento è stato impiegato per implementare un setup di riabilitazione *low-cost*.
Misurare il progresso riabilitativo dei pazienti che soffrono di ictus o di altri disturbi neurologici è attualmente un processo estremamente complicato che necessita di valutazioni specifiche per il singolo soggetto.
È probabile che di fronte al medesimo paziente le diagnosi e le scelte siano discordanti se effettuate da operatori differenti in quanto non esiste tutt'oggi una modalità definitiva per misurare in modo oggettivo come il movimento di un paziente migliori nel corso del processo riabilitativo.

In genere, tale operazione viene svolta affidandosi a sistemi di motion capture molto costosi che richiedono lunghi tempi per la preparazione del paziente oltre che un laboratorio appositamente allestito.

Il dott. Levesley, docente presso il dipartimento di ingegneria meccanica dell'università di Leeds, insieme ad una equipe di studenti (Domenico Clark, Barnaby Cotter, e Chris Norman) ha sviluppato una soluzione portatile e poco costosa, progettata per eseguire questa operazione.

Sfruttando la tecnologia offerta da *Kinect*, il sistema è progettato per fornire stimoli sonori e grafici al paziente, mentre si registrata accuratamente la posizione del corpo e il suo movimento. Questo sistema è quindi in grado di fornire ai fisioterapisti valutazioni reali per estrarre informazioni dettagliate sul movimento del paziente permettendo così diagnosi oggettive e corrette . Soprannominato "*Kinestesia*", il sistema è composto dal sensore di movimento Kinect.
Il *SDK* di *Microsoft* è stato utilizzato per restituire le coordinate (x, y, z) di 20 punti corrispondenti a quelli che vengono classificati come punti chiave del corpo del soggetto; si tratta di alcune regioni del corpo che permettono di ricostruire, istante per istante la posa del soggetto.
Integrando i dati del punto precedente direttamente in un file. *Avi*, il video che ripropone il movimento del paziente può essere rivisto, modificato e salvato, fornendo, istante per istante, informazioni 3-D sulla posizione dei segmenti corporei , permettendo il rendering girevole dello scheletro del paziente insieme al materiale video originale. Utilizzando *Kinect* per tenere traccia dello skeleton dell'utente, il movimento può essere analizzato e i parametri metrici e cinematici relativi al paziente calcolati e forniti all'operatore che è quindi in possesso di tutte le informazioni necessarie per effettuare una valutazione oggettiva e reale.

Fig. 20 Screenshot del software kinestesia. [56]

Durante la fase di registrazione al paziente viene richiesto di effettuare alcuni task e movimenti definiti da protocolli riabilitativi specializzati basati su standard di settore come ad esempio *Action*

Research Arm Tests (*ARAT*) che include sub-test come *grip*, *grasp* e *pinch*; tali movimenti vengono quindi catturati e analizzati dal software che fornisce in output le informazioni elaborate. Attualmente, Levesley e colleghi stanno collaborando con un certo numero di centri medici tra cui il Dipartimento di Neurochirurgia dell'Università di Los Angeles (California) e gli ospedali locali di Leeds per sviluppare un software per l'analisi del cammino e *ARAT* per pazienti post-operatori.

Con lo sviluppo del sistema kinestesia, un paziente può eseguire gli esercizi riabilitativi direttamente da casa, per poi inviare i dati video al fisioterapista per analizzare le informazioni a distanza.
Ancora più avanzato è il progetto di un gruppo di studenti dell'Università di Washington, che stanno tuttora tentando di portare la tecnologia di *Kinect* in sala operatoria.
Nella chirurgia assistita da robot il chirurgo non ha attualmente sensazioni tattili. Il sistema di controllo remoto può opporre una "resistenza" all'azione del chirurgo, tuttavia è necessario che il robot abbia la percezione della sua posizione rispetto al campo operatorio, per sapere quando sta entrando in contatto con vene, arterie, organi o quant'altro.

Kinect, in questo senso, può essere utilizzato per ricreare delle mappe tridimensionali del corpo del paziente, in modo che il sistema computerizzato possa restituire un feedback diretto e affidabile al chirurgo su quanto sta effettivamente accadendo.
Si tratta di una tecnologia eccellente perché di basso costo ed estremamente accessibile anche se per ora utilizzato per fini puramente dimostrativi.

Il sistema richiede ancora una fase di perfezionamento, per l'ottimizzazione della risoluzione e del focus dei sensori, ma le prospettive sono incoraggianti.
Secondo gli sviluppatori senza *Kinect* una simile applicazione sfiorerebbe i 50.000 dollari di costo. Oggi, sembra invece che la tecnologia dei videogiochi abbia trovato un'altra "strada sociale", oltre al puro intrattenimento.

Capitolo 5

Materiali e metodi

5.1 Sviluppo di un modello 3D

L'intento da parte di BTS di adottare la tecnologia dei sensori D-RGB, indagando in particolare la possibilità di utilizzare dispositivi relativamente semplici e poco costosi, quali *Microsoft kinect* e *Asus Xtion Pro*, come strumento di localizzazione e controllo all'interno del progetto "Nirvana" ha determinato l'obiettivo del presente lavoro di tesi, ovvero l'indagine delle potenzialità offerte da questi strumenti, la loro calibrazione e utilizzo in numero di due sensori contemporanei.

Il prodotto Nirvana sviluppato da BTS opera in modo da immergere il paziente in una sorta di ambiente virtuale col quale può interagire mediante il proprio corpo al fine di compiere determinati task a scopi riabilitativi: si tratta di semplici esercizi a sfondo ludico che permettono al paziente il recupero della motilità persa in seguito a traumi o patologie. Per far ciò è di fondamentale importanza ricostruire, istante per istante, la posizione e la posa del paziente all'interno dell'area di lavoro, sia per monitorarne il movimento sia per capire come gli elementi grafici proiettati debbano comportarsi in risposta allo stimolo indotto dal paziente generando un *feedback* di tipo acustico o visivo.

Se, ad esempio, in un particolare esercizio al soggetto viene richiesto di raggiungere con l'arto un particolare punto proiettato a parete occorre sapere, con relativa precisione, se e quando egli riesce nell'impresa in modo da indicare il *task* come raggiunto ed emettere un segnale per notificare al paziente l'avvenuto contatto col punto.

Nell'attuale versione di "Nirvana" l'informazione sul raggiungimento o meno dell'obiettivo avviene attraverso l'elaborazione di una immagine 2D ottenuta da una telecamera posta posteriormente al

paziente. In particolare, ogni qualvolta il paziente si trova in una posizione tale per cui la proiezione del proprio corpo sulla parete (o pavimento) si sovrappone alla proiezione generata da Nirvana il software cataloga il *task* come raggiunto.

Per come viene oggi implementato, questo tipo di elaborazione non permette di ottenere alcun tipo di informazione riguardo alla profondità della scena: ricollegandosi all'esempio precedente, è possibile che il task venga catalogato come raggiunto anche in quei casi dove il paziente, pur interrompendo il fascio di luce dell'immagine, si trovi a notevole distanza dalla parete su cui è presente la proiezione.

Alla luce di ciò è intuibile il ruolo che potrebbe ricoprire uno strumento D-RGB all'interno di un simile progetto: la possibilità di risalire alla terza dimensione a partire dalla mappa di profondità può rappresentare un notevole passo avanti nel miglioramento delle prestazioni di Nirvana.

In questa prima parte si è quindi deciso di procedere secondo quello che in precedenza è stato definito "livello superiore", utilizzando i dati ad alto livello processati internamente al sensore e ottenibili in output mediante le primitive messe a disposizione dalle librerie *OpenNI* (vedi *skeleton tracking* in appendice C).

Grazie alle procedure di *skeleton tracking* è possibile risalire istante per istante alle coordinate 3D, espresse in mm rispetto al sistema di riferimento del sensore, dei 15 punti chiave che descrivono lo scheletro del paziente.

Il codice per ottenere le coordinate di tali punti in forma di punti *XnPoint3D* definiti dalle *OpenNI* è riportata in appendice C.

In questo modo è in via teorica possibile comprendere come il soggetto si stia spostando e stia interagendo con l'area di lavoro così da ottenere tutte quelle informazioni necessarie per le successive elaborazioni.

Occorre però determinare, almeno da un punto di vista qualitativo, l'accuratezza con cui, attraverso questi *Key points*, è possibile ricostruire la reale posa del paziente.

Allo scopo è stato costruito un modello 3D, una sorta di avatar in grado di replicare in ogni istante il movimento del paziente in modo tale da poter valutare, almeno da un punto di vista visivo, i risultati dello *skeleton tracking* forniti dal dispositivo.

Il modello che ricostruisce la figura del soggetto , elaborato sulla base dello scheletro basato su 15 *key points*, è stato costruito utilizzando le primitive *OpenGL* in modo da sfruttare le possibilità, offerte da questa libreria, di ricomporre una scena 3D che tenga conto degli effetti di prospettiva e illuminazione.

Innanzitutto è stato creato il piano su cui il modello potrà spostarsi: questo viene rappresentato come una sorta di griglia, la cui spaziatura può essere definita dall'utilizzatore, che permette di ricavare, tramite una semplice ispezione visiva, un'informazione di tipo qualitativo sulla posizione occupata dal soggetto rispetto alla posizione di centro stanza indicata da una croce rossa.

Il modello sviluppato consiste in un insieme di segmenti rigidi descritti da cilindri o coni a rappresentare i principali segmenti corporei, collegati tra loro da sfere a simulare il movimento di tipo puramente rotatorio supposto esserci a livello articolare.

Il modello riceve in ingresso, ad una frequenza di 30 Hz (quella a cui lavora *Asus Xtion Pro*), le coordinate 3D dei 15 punti chiave, ricavabili dall'elaborazione interna delle *depth map*; le dimensioni di ciascun segmento (distanze, lunghezze e larghezze) non sono note a priori ma sono modellate sulle distanze calcolate sugli stessi punti in input. In questo modo lo stesso modello può essere applicato a soggetti diversi caratterizzati da dimensioni e caratteristiche differenti.

In figura seguente è riportato lo schema utilizzato per la definizione del modello dove in rosso sono rappresentati i cosiddetti punti chiave.

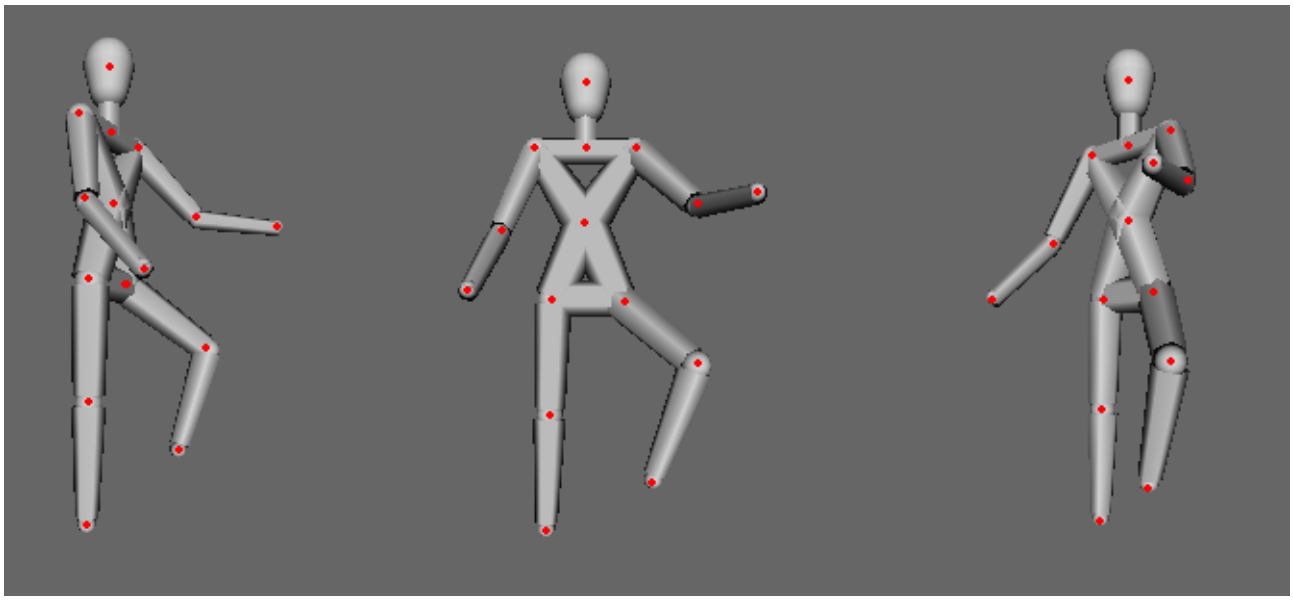


Fig. 1 Modello sviluppato. In rosso sono riportati i 15 punti chiave ottenibili dalle primitive *OpenNI* di *skeleton tracking*

La struttura ad "X" con cui viene rappresentato il busto permette al modello di simulare il movimento di rotazione nel piano sagittale, frontale e longitudinale attorno al punto centrale del busto stesso.

Alcune precisazioni sulla scelta del modello si rendono doverose.

È evidente che si tratta di un modello poco preciso e poco accurato che in modo estremamente approssimativo rappresenta la vera cinematica del corpo umano: innanzitutto è da considerare che in realtà i segmenti corporei non sono rigidi, e che quindi,solo in prima approssimazione, possono essere rappresentati da cilindri e coni rigidi. Inoltre il movimento che si realizza a livello delle articolazioni è tutt'altro che puramente rotatorio.

Tuttavia è bene contestualizzare l'ambito di applicazione in cui il suddetto modello sarà utilizzato: nel caso specifico occorre cioè ragionare su quello che è lo scopo del modello in questione.

In questa precisa situazione, il fine ultimo non è quello di eseguire un'analisi del movimento atta a stimare quei parametri cinematici quali angoli articolari o quant'altro, ma quello di capire come e dove, istante per istante, è posizionato il paziente all'interno dell'area di lavoro.

Ciò che serve al sistema, infatti, è capire se il paziente è in grado di compiere determinati movimenti e di raggiungere determinati oggetti proiettati da Nirvana. Ecco dunque che, sebbene l'accuratezza nel descrivere la cinematica da parte del modello non sia sufficiente per un'analisi quantitativa, essa è accettabile per un approccio di tipo qualitativo funzione allo scopo del progetto.

Il software sviluppato in questa tesi per l'analisi dello *skeleton tracking* mediante le primitive *OpenNI* è dotato di un'interfaccia grafica *user-friendly* che permette all'operatore di generare il modello e ottenere le informazioni sulla sua posa tramite la pressione di alcuni pulsanti.

Il software, oltre a mostrare a video in tempo reale l'immagine acquisita dalla telecamera RGB montata su *Xtion Pro Live* e il modello ricostruito al calcolatore, fornisce in una casella di testo le coordinate, assunte in ogni istante ed espresse in mm secondo il sistema di riferimento del sensore, dei 15 punti chiave che definiscono il modello. L'interfaccia permette inoltre, attraverso la pressione di un tasto, di procedere al calcolo e alla registrazione in un array dell'andamento nel tempo degli angoli articolari di maggiore interesse (braccio-avambraccio, femoro-tibiale). Infine, mediante l'utilizzo di alcuni widget quali tasti e slider (oggetti tipici dell'ambiente QT) posizionati nell'area di lavoro, è possibile ruotare la scena per indagare l'evolversi del movimento da differenti punti di vista diversi da quello del sensore.

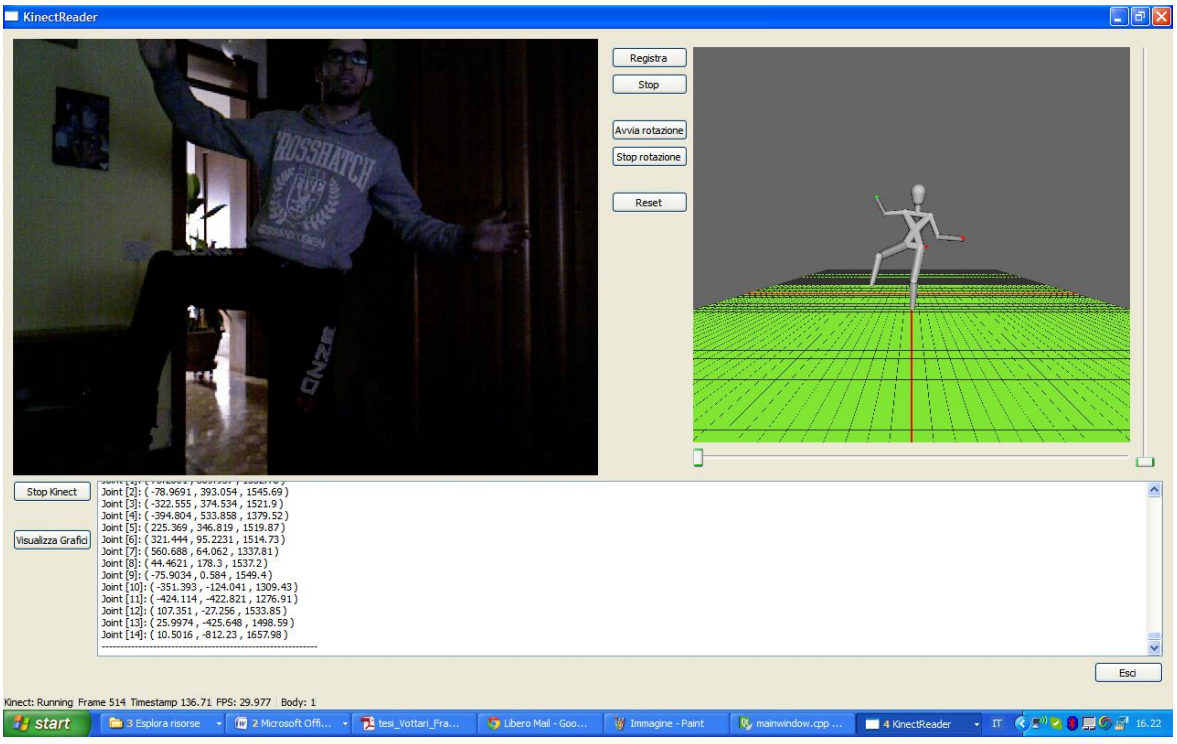


Fig. 2 Screenshot dell'interfaccia di lavoro del software sviluppato

Una volta realizzata una prima versione, sono stati effettuati dei test a livello qualitativo sul buon funzionamento del modello. Bisogna infatti confrontare quelle che sono le reali condizioni di utilizzo del sensore all'interno del progetto Nirvana con quella che è la logica per cui è stato concepito tale strumento: di fatto il dispositivo, originariamente predisposto per lavorare in posizione frontale rispetto all'utente, nel progetto viene posto alle spalle del paziente. Sulla base di ciò, si è reso necessario effettuare dei test per capire il comportamento del sensore in questa differente condizione d'uso.

Si sono svolte quindi diverse simulazioni in cui si inquadrava un soggetto in movimento da diverse angolature da cui, tramite ispezione visiva, è stato possibile determinare la bontà della risposta del modello.

Questi primi test hanno purtroppo manifestato alcune problematiche non considerate inizialmente che hanno portato a sospendere, almeno momentaneamente, questo tipo di soluzione per indagare una strada alternativa.

C'è da considerare il fatto che, affinché lo strumento riesca a fornire dati attendibili, occorre che siano soddisfatte alcune condizioni.

Innanzitutto occorre che il soggetto non sia a contatto con nessun altro oggetto del background, e ciò va quindi a limitare il suo utilizzo per quanto riguarda le applicazioni su tavolo, inoltre occorre che il soggetto non compia movimenti rapidi in quanto non rilevabili dal sensore che lavora a non più di 30fps introducendo quindi una componente di errore nel *tracking* dell'utente. Inoltre, uno dei fattori maggiormente limitanti lavorando con un'unica sorgente di segnale è quello dell'occlusione e in particolare dell' auto occlusione. Se, come era intenzione iniziale, si pone la telecamera dietro al soggetto, la sagoma dello stesso impedirà il corretto tracciamento dello scheletro durante il movimento.

Il progetto Nirvana, per la maggior parte delle applicazioni prevede che l'oggetto con cui il paziente deve interagire venga proiettato a parete. Ora, tenendo in considerazione che occorre posizionarsi ad una distanza minima di 50 cm (meglio se 150-200 cm in modo che il cono acquisito dalla telecamera copra tutto il paziente) dall'obiettivo affinché lo strumento possa elaborare correttamente i dati, si elimina qualsiasi possibilità di interporre Asus Xtion Pro tra il paziente e la parete stessa.

Si è deciso quindi di montare il sensore sulla parete, oggetto dell'interazione col paziente, in una posizione laterale e sopraelevata, come mostrato in figura seguente, in modo da avere, per quanto possibile, una vista latero-frontale del soggetto.

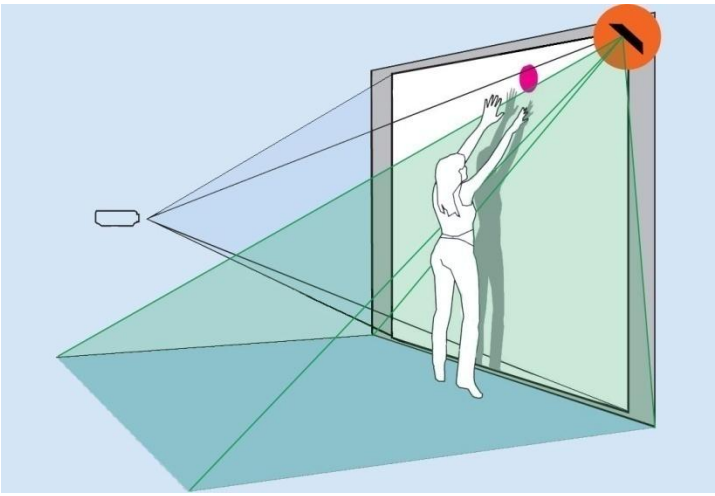


Fig. 3 Disposizione del sensore *Asus Xtion Pro Live* nell'area di lavoro di Nirvana. È l'unica disposizione che permette, per quanto possibile, una vista frontale del soggetto

Tuttavia questa nuova disposizione non rappresenta una soluzione robusta al problema: innanzitutto sorge l'impossibilità di utilizzare i dati forniti, sottoforma di coordinate 3D, dalle primitive ad alto livello delle OpenNI e NITE in quanto, data la posizione laterale, l'elaborazione della mappa di profondità grezza non genera risultati attendibili costringendo lo sviluppatore ad implementare un nuovo algoritmo di tracking a partire dalla depth map; inoltre, l'auto-occlusione diventa in questa nuova configurazione un fattore non più trascurabile che rende i risultati incompatibili con lo scopo preposto.

Un altro fattore finora non considerato, ma di fondamentale rilevanza, è quello per cui tutti i punti ottenuti dal dispositivo sono espressi secondo il sistema di riferimento del sensore stesso la cui posa è a priori sconosciuta da parte dell'utilizzatore. È quindi necessario definire una strategia che permetta di determinare la rototraslazione in grado di mappare punti descritti secondo il sistema del sensore in punti espressi secondo un sistema di riferimento globale definibile dall'operatore. Questa però è una questione che verrà affrontata in seguito, in particolare nella seconda parte della tesi.

Alla luce di queste riflessioni occorre dunque definire un nuovo setup di lavoro che permetta di risolvere i numerosi problemi riscontrati.

L'utilizzo, in collaborazione al primo, di un secondo sensore posizionato sempre sulla parete ma in posizione opposta al primo può rappresentare una condizione che permette di migliorare i risultati forniti dall'elaborazione congiunta della scena attraverso l'integrazione dei dati provenienti dai due dispositivi.

Ogni dispositivo genera però una mappa di profondità le cui coordinate sono definite rispetto al sistema di riferimento dello strumento che l'ha generata. Ecco dunque che, per ottenere dei dati confrontabili e assimilabili, occorre conoscere come mappare i punti acquisiti dall'una nel sistema di riferimento dell'altra.

Si rende necessaria quindi l'introduzione di una fase preliminare di calibrazione tra i due dispositivi che permetta di stimare i parametri di rototraslazione tra i due sistemi di riferimento in questione.

Tale tema rappresenta lo scopo di tutta la seconda parte del progetto di tesi che verrà illustrata nel prossimo paragrafo.

5.2 Calibrazione di due sensori D-RGB

Date le problematiche descritte nel paragrafo precedente, in questa seconda fase della tesi viene analizzato ed implementato un software con la relativa interfaccia grafica per la calibrazione automatica di due dispositivi D-RGB.

A differenza delle tecniche presenti in letteratura, l'algoritmo sfrutta l'informazione sulla tridimensionalità presente nelle mappe di profondità che questi dispositivi generano.

Da qui in poi si farà riferimento al caso di due soli sensori ma la stessa tecnica può essere adottata per un numero arbitrario di dispositivi.

Per capire come si intende procedere basta considerare il caso in cui, per un punto inquadrato allo stesso istante da due dispositivi D-RGB, sia possibile stimarne le coordinate espresse nei rispettivi sistemi di riferimento da entrambe le telecamere. Procedendo in questo modo è quindi possibile ottenere le coordinate nei due sistemi di un numero arbitrario di punti purché visibili contemporaneamente dagli obiettivi presenti sulla scena.

Sfruttando questi dati è possibile implementare una tecnica di calibrazione che prescindendo dal calcolo dei parametri intrinseci delle telecamere, noti allo strumento per costruire la depth map, consiste unicamente nella stima dei parametri di rototraslazione in grado di mappare i punti visti da un dispositivo nei corrispondenti punti dell'altro.

L'idea che sta alla base dell'algoritmo sviluppato è estremamente semplice e consiste nel far riconoscere dall'una e dall'altra telecamera la posizione nello spazio 3D, espressa secondo i relativi sistemi di riferimento, di alcuni punti noti, detti *punti di controllo*, e calcolare da questi le tre rotazioni e le tre traslazioni che permettono di sovrapporre il sistema di riferimento della prima telecamera a quello della seconda.

Come noto i parametri di calibrazione estrinseci sono sei: tre rotazioni e tre traslazioni; essendo un punto P nello spazio definito attraverso tre scalari [Xp,Yp,Zp], idealmente sono sufficienti tre soli punti non allineati per determinare la matrice di rototraslazione cercata.

Per procedere alla calibrazione secondo quanto appena esposto è necessario che un numero N>3 di punti sia contemporaneamente riconosciuto da entrambi i sensori. Affinché questo sia possibile, trascurando il problema dell'occlusione, la forma geometrica che più si presta per la definizione del singolo punto di controllo è quella della sfera.

Grazie alla simmetria di questa figura infatti, per ognuna delle telecamere è possibile identificare le coordinate del centro di un numero arbitrario di sfere inquadrate i cui centri formeranno il dataset su cui lavorare per procedere alla calibrazione.

In sostanza facendo riconoscere contemporaneamente ai due sensori almeno 3 sfere collocate nell'area di lavoro, dalla posizione stimata del centro di ciascuna di esse è possibile risalire alla trasformazione tra i due sistemi in esame.

Ovviamente il numero di punti influenza significativamente la bontà della stima: bisogna considerare infatti l'errore strumentale con cui il dispositivo ricostruisce la tridimensionalità della scena e che necessariamente incide sulla stima dei punti di controllo. Più punti vengono utilizzati per il calcolo e meno i risultati saranno influenzati da tale errore. Dall'altra parte occorre però che ogni punto di controllo sia identificato e marcato in modo da correlarlo al suo corrispondente nel sistema dell'altra telecamera, processo questo che va via via complicandosi all'aumentare della numerosità dei punti, senza considerare il problema di occlusione parziale o totale che incorre nel posizionare sulla scena un numero elevato di sfere.

Al fine di valutare l'affidabilità dell'algoritmo e del metodo di calibrazione sviluppato, è necessario disporre di un *gold standard*. Nel presente lavoro si è scelto di utilizzare quale *gold standard* la calibrazione eseguita mediante il sistema stereofotogrammetrico prodotto da BTS. A tal fine è necessario definire un algoritmo che preveda lo stesso procedimento operativo utilizzato per la calibrazione di sistemi stereofotogrammetrici standard secondo il metodo di calibrazione thor2 utilizzato dai sistemi BTS. Tale scelta consente di utilizzare lo stesso *tool* per effettuare la contemporanea calibrazione del sistema stereo formato dai sensori D-RGB e del sistema stereofotogrammetrico SMART-DX (sviluppato da BTS e utilizzato come termine di paragone nella fase di test) rendendo così possibile un confronto tra i risultati di una calibrazione effettuata mediante le stesse apparecchiature.

In particolare il metodo prevede un set up iniziale dove viene effettuata una prima stima mediante una terna a nove sfere e una seconda fase effettuata mediante una *Wand*, ovvero una bacchetta a tre sfere per il raffinamento dei parametri calcolati in precedenza.

Pertanto in un primo momento sono state adottate 9 sfere posizionate su di un *tool* a replicare quello utilizzato per la calibrazione di telecamere standard da BTS.

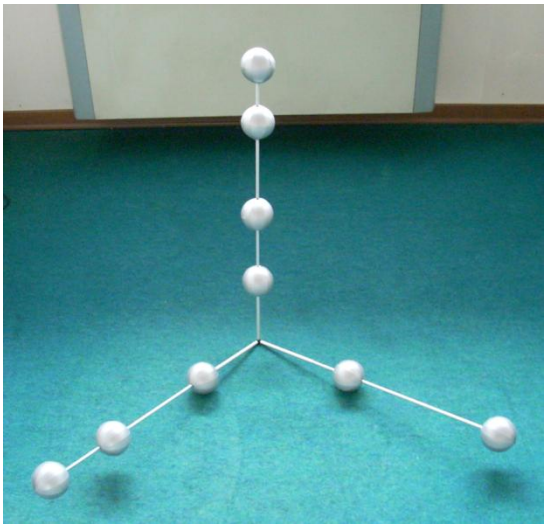


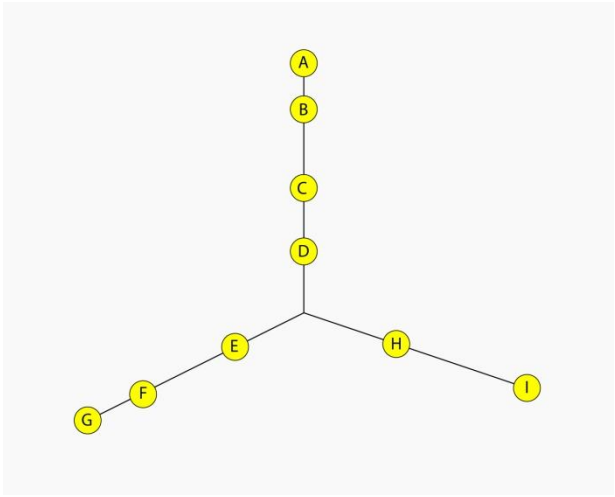
Fig. 4 Tool a nove sfere utilizzato nella fase di Axes per la calibrazione del sistema D-RGB

In questo caso la terna è stata costruita a mano cosicché necessariamente le sue dimensioni non risultano del tutto precise. Questo fatto rappresenta un problema relativo solamente al sistema SMART poiché, per il principio che si intende adottare, tale caratteristica non influisce sui risultati della calibrazione per il sistema D-RGB per il quale è sufficiente che le sfere viste dall'una telecamera siano le stesse viste dall'altra indipendentemente dalla posizione che occupano.

Purtroppo, per i motivi che verranno ora illustrati, il *tool* costruito non può essere in tutto e per tutto uguale a quello utilizzato da BTS. Le sfere utilizzate per la calibrazione di questi sistemi devono infatti avere dimensioni molto maggiori rispetto ai markers (1-2 cm) utilizzati da BTS affinché siano riconoscibili e identificabili dai sensori D-RGB.

Inizialmente sono state utilizzate sfere dal diametro di 4 cm; questa dimensione non permette però il loro riconoscimento non appena siano poste a più di 2 metri dall'obiettivo cioè ancora all'interno dell'area utile di lavoro. Quindi, dopo aver effettuato alcune prove, sono state adottate sfere del diametro di 7 cm. Si tratta di sfere in poliuretano espanso caratterizzate da una superficie liscia ma non lucida in modo da non incorrere in fenomeni di riflessione del pattern per la stima 3D.

In particolare di seguito sono riportate lo schema e le relative dimensioni del *tool* costruito.



Sfere	Distanza
A-B	120 mm
B-C	205 mm
C-D	165 mm
D-O	160 mm
O-E	200 mm
E-F	270 mm
F-G	160 mm
O-H	255 mm
H-I	360 mm

Fig. 5 Schema di costruzione del *tool* di calibrazione a nove sfere

È interessante notare come l'i utilizzo di un tool identico dal punto di vista strutturale ma non dimensionale non rappresenti un limite per il sistema SMART: per la calibrazione, esso non richiede che le dimensioni dei marker siano esattamente quelle standard e nemmeno che le interdistanze siano quelle definite da BTS. È sufficiente infatti fornire queste nuove dimensioni al software andando a settarne nuovamente i parametri e il sistema, in automatico, sarà in grado di stimare i parametri necessari.

Ciò che invece impedisce l'implementazione di tale tecnica è che, affinché le sfere possano essere riconosciute dal sistema optoelettronico, occorre che queste siano ricoperte da materiale rifrangente in microsfere di vetro. Questa condizione rappresenta però il limite principale per la stima della depth map da parte del sensore D-RGB dando luogo a fenomeni di riflessione del pattern che non permettono la stima sulla distanza in corrispondenza delle sfere.

Nonostante ciò è comunque possibile articolare il processo nelle due fasi di *Axes* e *Wand* procedendo però in due momenti distinti alla calibrazione dei due sistemi. Prima viene effettuata la stima per il sistema D-RGB e, solo dopo aver ricoperto le sfere presenti nei *tool*s con un particolare nastro rifrangente, proseguire alla stima dei parametri per lo SMART.

In particolare nella fase di *Axes* è possibile risalire ad una stima iniziale dei parametri in base all'informazione sui nove punti di controllo per poi procedere al raffinamento degli stessi dall'analisi di ogni frame contenente tre sfere, acquisito nella fase di *Wand*.

Una semplice analisi del metodo fa intuire come, ancora una volta, la cosa non risulti concretizzabile. Come già sottolineato infatti, per il sistema D-RGB occorre che le tre sfere siano non allineate costringendo quindi l'operatore ad adottare un *tool* differente.

Inoltre si nota come, a causa dell'errore introdotto nel calcolo delle coordinate di tre soli punti di controllo, le diverse stime ottenute sono caratterizzate da una variabilità non accettabile su frame successivi generando risultati non attendibili.

Per ovviare a questi inconvenienti e definire un setup che sia il più robusto possibile è stata abbandonata l'idea di portare sul sistema di sensori D-RGB il metodo che altresì lavora in modo ottimo per il sistema SMART e di implementare una nuova tecnica.

Questa nuova tecnica, per quanto possibile, dovrà servirsi di *tools* molto simili permettendo la calibrazione sulla base di un dataset composto da un numero N arbitrario di punti, in modo da ridurre l'influenza dell'errore di stima del centro, consentendo comunque la fase necessaria di identificazione degli stessi.

Si è deciso quindi di non avvalersi in seguito dell'utilizzo della terna a nove sfere che può essere comunque adottata, come verrà descritto in seguito, per determinare un sistema di riferimento globale definito dall'operatore al fine di esprimere le posizioni dei vari punti ricavati dalla telecamera ma di procedere secondo quanto segue.

Una bacchetta contenente tre sfere allineate e poste a distanze differenti l'una dall'altra (stesso schema che per lo SMART), viene mossa all'interno dell'area di lavoro. Per ogni frame acquisito dalla stessa camere si procede all'identificazione dei punti di controllo e si va a salvare i risultati in un singolo *array*. Per ogni frame quindi vengono costruiti tre nuovi punti, facilmente identificabili, che saranno aggiunti ad un'unica lista ordinata permettendo la stima dei parametri sulla base di una *point cloud* di N=3 x K (K=numero frame acquisiti) punti sparsi su tutta l'area di lavoro.

Di seguito viene illustrato nel dettaglio il metodo adottato.

5.2.1 Acquisizione dei dati di calibrazione

Si procede al posizionamento dei sensori facendo particolare attenzione che l'angolo tra le rette uscenti da ogni dispositivo e parallele al raggio ottico formino un angolo non inferiore a 90 gradi. In caso contrario è possibile che il pattern proiettato da un sensore A, risulti visibile da un secondo sensore B il quale tenterà di processare tale pattern introducendo una serie di errori nella costruzione della mappa di profondità.

Con l'utilizzo di soli due dispositivi tale condizione è facilmente evitabile; basterà infatti disporre gli stessi in modo da formare un angolo superiore ad un angolo retto tra i raggi ottici uscenti.

Qualora si decidesse di lavorare con un numero superiore di sensori occorre adottare particolari contromisure illustrate in Appendice A.

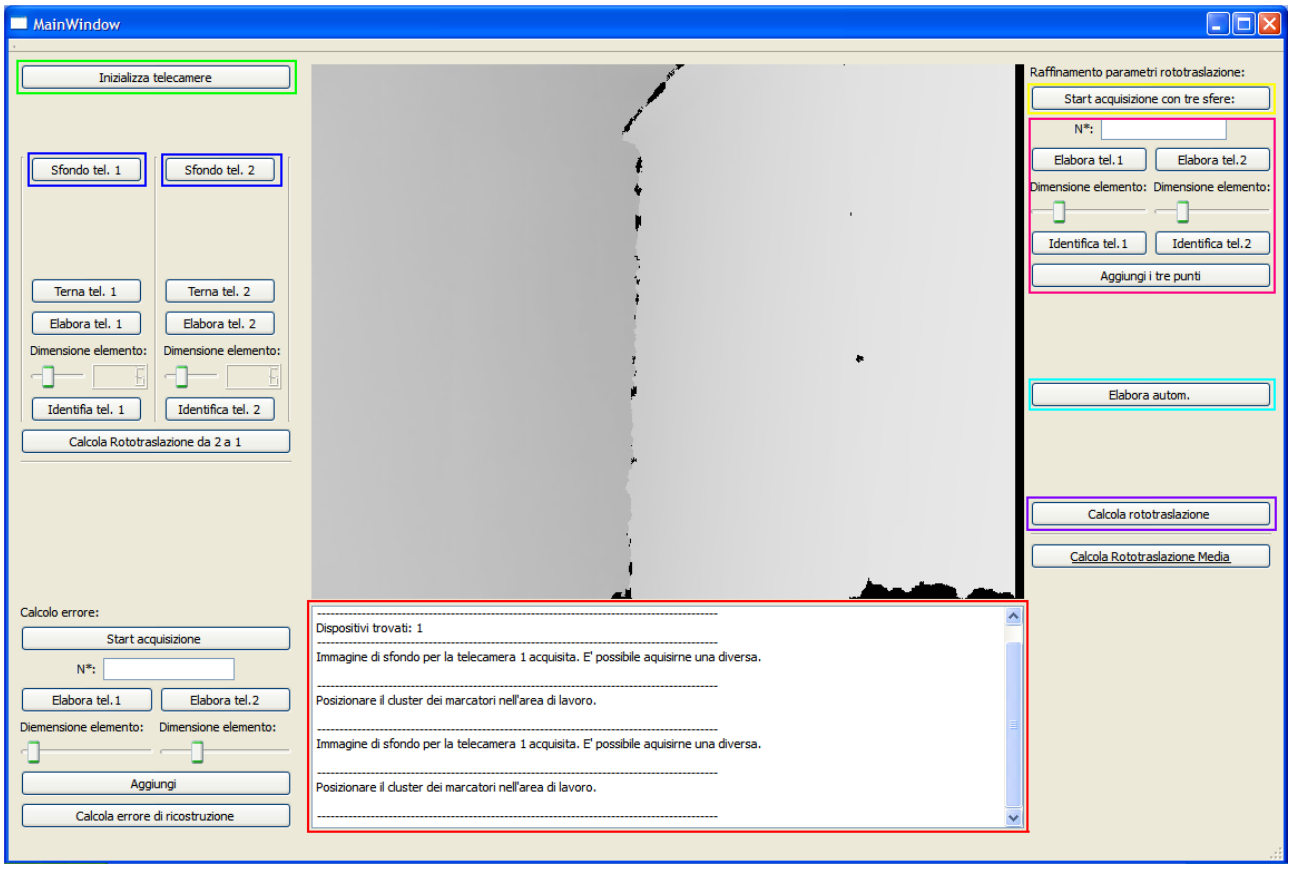


Fig. 6 Screen shot della schermata di lavoro del software sviluppato. Con i diversi colori sono indicati i vari strumenti che permettono di effettuare la calibrazione. Le sezioni non evidenziate sono state utilizzate nella fase di test o nella definizione del sistema globale.

A questo punto mediante la pressione del tasto "Inizializza telecamere" , presente nell'interfaccia del software sviluppato, è possibile procedere al riconoscimento dei due sensori, indicati con sensors[0] e sensors[1].

Agendo quindi sui pulsanti evidenziati in blu in figura 6, da entrambi i sensori si acquisisce un'immagine senza che nell'area di lavoro sia posizionato il *tool* di calibrazione. Questa immagine indicata con B (*background*) andrà a modellare lo sfondo e sarà utilizzata per la fase di segmentazione delle immagini.

A questo punto, mediante la pressione del tasto evidenziato in giallo in figura 6, viene lanciato un algoritmo che per un tempo di M secondi, durante il quale l'utente muove nell'area di lavoro la bacchetta con le tre sfere, ad intervalli regolari di 500 millisecondi acquisisce da ogni camera un'immagine indicata con la lettera F (*foreground*).

Nel muovere la bacchetta occorre fare attenzione a non entrare col proprio corpo nell'inquadratura di uno dei due sensori in quanto si andrebbe a falsare le informazioni sullo sfondo contenute in B. Per questo è utile servirsi di una barra con cui prolungare il *tool* a tre sfere.

Punti	Distanza [mm]
A-B	160
A-C	430
B-C	270

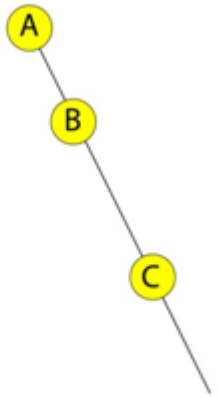


Fig. 7 *Tool* a tre sfere utilizzato per la calibrazione del sistema

Nell'acquisire il *tool* in movimento, occorre considerare con particolare attenzione il problema di sincronizzazione dei due dispositivi.

Se i due sensori non acquisiscono le immagini allo stesso istante le posizioni stimate nei due sistemi di riferimento non sono confrontabili. Per questo i dispositivi sono comandati da un timer, il quale ad intervalli regolari fornisce l'input ad entrambi i dispositivi sincronizzando il sistema. La presenza di eventuali ritardi minimi dovuti a differenze temporali nell'elaborazione interna è considerata trascurabile in quanto la bacchetta di calibrazione viene mossa lentamente.

5.2.2 Identificazione dei blob e stima del centro

A questo punto occorre procedere all'elaborazione di ogni frame acquisito al punto precedente al fine di stimare le posizioni dei centri di ognuna delle tre sfere nei rispettivi sistemi di riferimento.

Questa fase può essere effettuata in modo manuale, agendo sui widget evidenziati in fucsia in figura 6, andando quindi ad analizzare ogni singola immagine in modo da controllare di volta in volta i risultati dell'elaborazione e eventualmente scartare quei frame che visivamente non

conducono a risultati accurati. Oppure è possibile (tasto azzurro di fig. 6) lanciare un algoritmo automatico che fornirà i risultati permettendo, in questo caso, l'ispezione visiva dell'elaborazione solo a posteriori.
Nello specifico questa operazione, comunque si decida di procedere, si articola in diversi step successivi di seguito analizzati.
L'immagine B 640x480 dello sfondo acquisita in precedenza viene sottratta pixel per pixel dall'immagine F ottenendo l'immagine D (*difference*) delle stesse dimensioni di B e F.
Lavorando con fotocamere e telecamere digitali standard questo processo definito "*background subtraction*" permette di ottenere un'immagine i cui pixel sono diversi da zero solo in quelle zone che presentano differenze tra l'immagine B e l'immagine F permettendo, nel caso ideale, l'individuazione di tutte e sole quelle regioni occupate dai marcatori.
Lavorando con immagini di profondità la cosa risulta più complessa.
Se si acquisisce, tramite dispositivi D-RGB, la mappa di profondità di una scena statica si nota che l'immagine in alcune regioni più o meno ampie continua a variare di frame in frame creando una sorta di sfarfallio. Questo fenomeno si presenta in quanto l'informazione che la telecamera IR riceve da queste zone non è costante sebbene la scena sia immobile. In dettaglio entrano in gioco tutta una serie di fattori [4] tra cui l'orientazione delle superfici presenti, il materiale di cui sono composte, il loro colore o il loro potere riflettente che influenzano notevolmente il calcolo della depth map e sono alla base dello sfarfallio appena descritto.

Tornando all'algoritmo è evidente quindi che la sottrazione tra due mappe di profondità non genera il risultato sperato. L'immagine D ottenuta necessita di una ulteriore elaborazione mediante filtri morfologici (che permettono di effettuare una sorta di *blob analysis* andando ad eliminare eventuali proiezioni di dimensioni inferiori a quelle attese) e processi di sogliatura in modo da eliminare tutti i falsi positivi dovuti a elementi posti a distanze incompatibili con l'esperimento (oggetti stimati troppo lontani o troppo vicini) ed ottenere così un'immagine binaria contenente informazione solo in quelle regioni effettivamente occupate dalla proiezione sul piano immagine dei markers.
Queste regioni rappresentano i *blob* già introdotti nei capitoli precedenti.

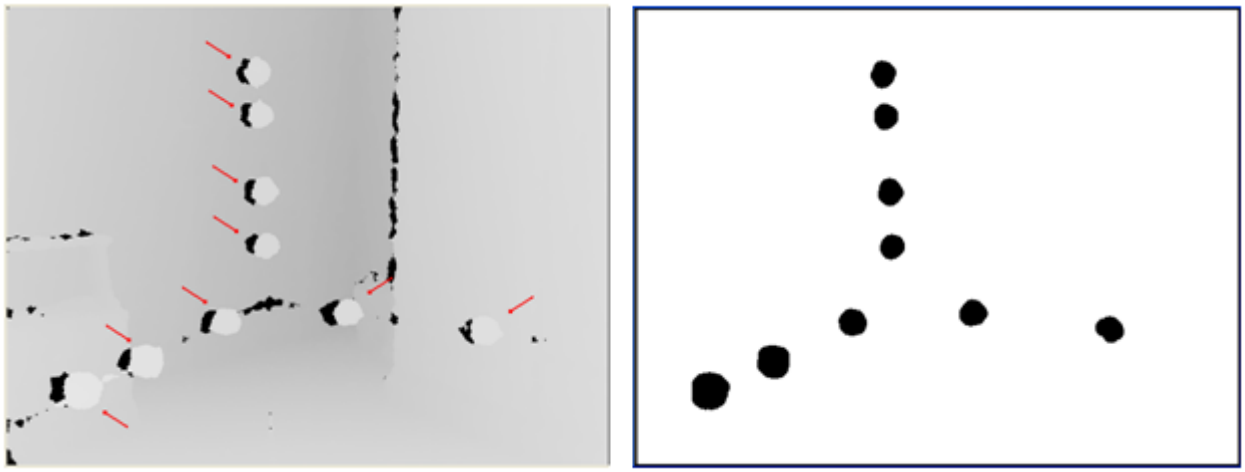


Fig. 8 destra depth map acquisita in presenza del pattern contenente le nove sfere; a destra risultato del processo di segmentazione per l'individuazione dei *blob*.

L’individuazione del *blob* è un’operazione fondamentale per ottenere risultati affidabili nella successiva fase di stima dei parametri di calibrazione ed è quindi necessario prestare molta attenzione alle tecniche di image processing adottate.

È facile notare come l’algoritmo di segmentazione permetta l’estrazione di un *blob* che sempre e comunque risulta essere pari o al limite inscritto nell’area effettivamente occupata dalla proiezione del marcatore. Questa considerazione tornerà utile in seguito quando si andrà a discutere la stima del centro di ogni singolo *blob*.

Una volta ottenuta un’immagine binaria contenente i soli *blob* si passa allo step successivo.

In questa fase si procede all’elaborazione delle immagini di cui al punto precedente con lo scopo di stimare il punto centrale di ciascun *blob* 2D nonché proiezione del centro del marker sulla superficie della sfera stessa in direzione del raggio ottico entrante nella telecamera.

Si sottolinea fin da subito che i diversi *blob* estratti nella fase di segmentazione sono caratterizzati dall’assumere una forma tutt’altro che regolare. Questo fatto non è dovuto alla fase di image processing bensì al principio di funzionamento del sensore stesso(vedi capitolo 4) e in particolare a come questo genera la mappa di profondità.

In particolare il pattern infrarosso proiettato sulla scena impattando con superfici quasi parallele al raggio ottico viene ampiamente distorto e non permette la corretta elaborazione dell’informazione proveniente da tali aree.

Lavorando con marcatori sferici il problema assume una forte rilevanza in concomitanza dei bordi delle sfere come schematizzato in figura seguente.

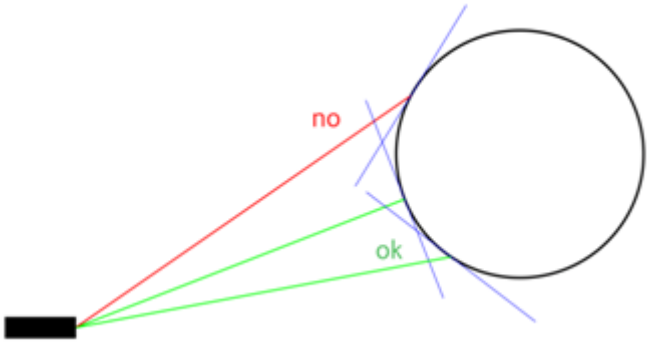


Fig. 9 Schematizzazione dell’errore di ricostruzione ai bordi di una sfera.

L’algoritmo di stima del centro deve quindi essere tale da elaborare anche quei *blob* caratterizzati da una forma che ben si discosta dal poter essere associata alla proiezione di una sfera e, nel caso, procedere a posteriori alla sua eliminazione.

In particolare l’algoritmo implementato permette, in una sola “passata” dell’immagine di identificare ogni singolo *blob* presente e di localizzarne il centro.

L'ipotesi su cui si basa l'algoritmo è quella per cui ogni *blob* estratto nella fase di segmentazione è caratterizzato da un bordo 8-connesso chiuso. Per assicurarsi che questa condizione sia verificata, prima di applicare un filtraggio passa-alto per l'estrazione del contorno, l'immagine viene sottoposta ad un'operazione di dilation con elemento strutturante di dimensione minima in modo da eliminare eventuali spigoli che darebbero luogo a "buchi" nel contorno.

L'immagine viene quindi trattata mediante filtraggio alla *Canny* in modo da ottenere un'immagine binaria con il solo bordo dei diversi *blob* che rappresenta l'input per la successiva elaborazione.

L'algoritmo inizia così a scorrere l'immagine per righe: una volta incontrato un primo valore a 1 significa che è stato incontrato un *blob*. A questo punto l'algoritmo inizia a scorrere il bordo cancellando di volta in volta il pixel su cui sta lavorando ma salvandone le coordinate in un buffer. Una volta percorso tutto il bordo sono note tutte le informazioni che permettono il calcolo del centro mentre nell'immagine originale quel *blob* sarà stato completamente eliminato. A questo punto l'algoritmo riprende a scorrere l'immagine dal pixel in cui si era fermato in precedenza fino ad incontrare un nuovo *blob*.

Grazie alle informazioni memorizzate nello scorrere il perimetro è possibile a posteriori effettuare una sorte di *blob-analysis* andando a scartare eventuali proiezioni caratterizzate da fattori di forma lontani da quello di una circonferenza o di dimensioni fuori da un range atteso definito a priori.

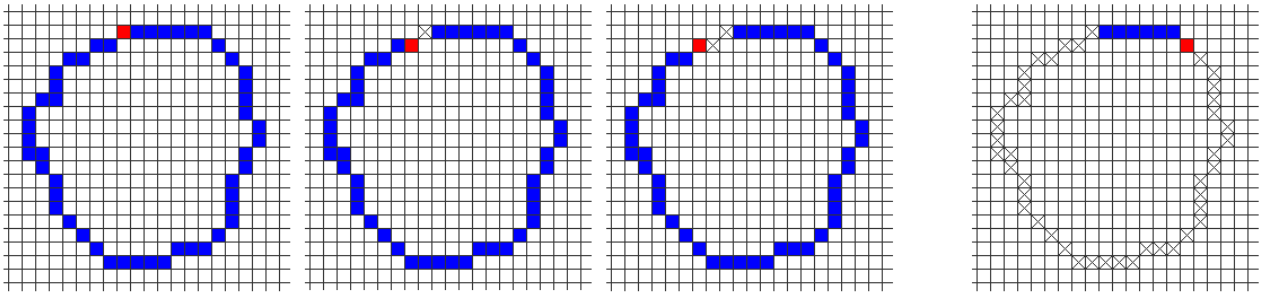


Fig. 10 Schema di lavoro dell'algoritmo sviluppato. In blu è presentato il perimetro del singolo marker. Da sinistra a destra sono presentati alcuni step successivi dell'elaborazione. Il pixel indicato in rosso rappresenta il pixel su cui sta lavorando temporaneamente l'algoritmo. I pixel barrati con una X sono i pixel elaborati e poi eliminati.

A partire dai dati ottenuti durante la "scansione" del perimetro è possibile infine risalire alle coordinate nel piano immagine dei vertici A,B,C,D del rettangolo di area minima che circoscrive il *blob* stesso rendendo poi immediato risalire alla circonferenza che alla stesso modo localizza la singola proiezione.

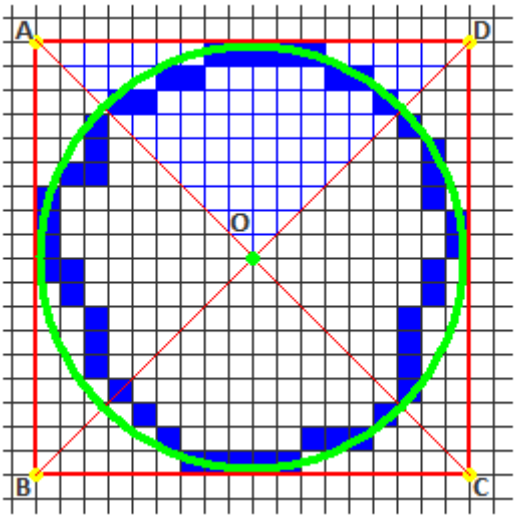


Fig. 11 Schematizzazione dei risultati ottenuti dall'algoritmo di stima del centro.

Il centro del *blob* è calcolato come centro del rettangolo ABCD che necessariamente corrisponde anche al centro della circonferenza cercata.

Come già evidenziato, il *blob* selezionato è generalmente inscritto nell'area occupata dall'immagine del marker, grazie a ciò il calcolo del centro attraverso la stima della circonferenza circoscritta fornisce risultati visivamente accurati.

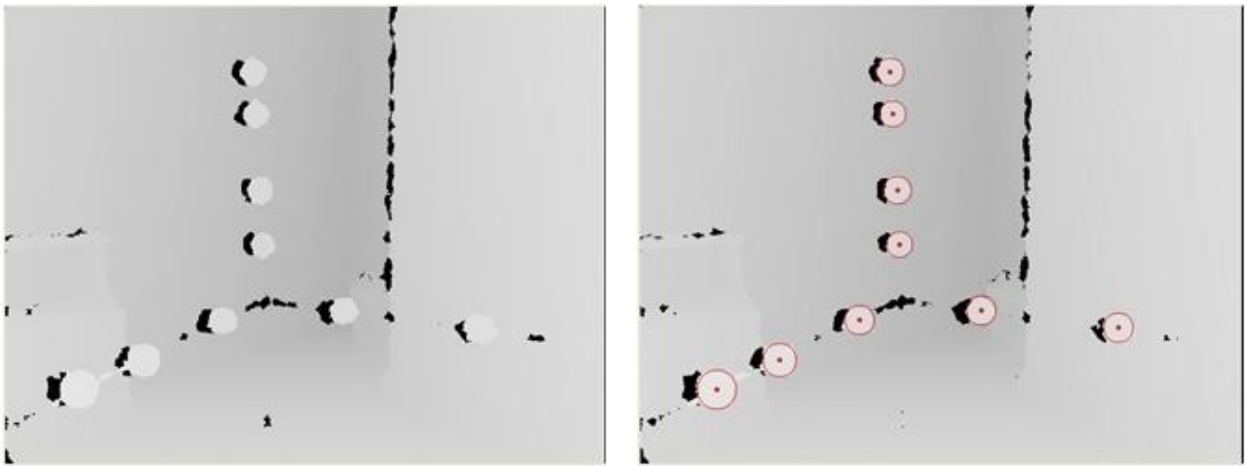


Fig. 12 A sinistra mappa di profondità della terna a nove sfere; a sinistra il risultato della stima della circonferenza che meglio modella il *blob*

È doveroso giunti a questo punto riportare alcune osservazioni per giustificare le scelte procedurali effettuate.

La stime delle coordinate X_q, Y_q, Z_q del centro di ciascun marker rappresenta la fase fondamentale per tutto il processo di calibrazione: un errore di qualche pixel nel piano immagine corrisponde ad un errore dell'ordine di millimetri e in alcuni casi di centimetri nella stima delle coordinate 3D del centro con una conseguente propagazione degli errori al calcolo dei parametri estrinseci.

La strada che potrebbe condurre alla stima ottima delle coordinate del centro è quella di sfruttare l'informazione fornita dalle depth maps sulla terza dimensione e implementare così una Hough trasformata [16] in 3 dimensioni (usando la sequenza (X_o,Y_o,Z_o,r) per descrivere la sfera) o di procedere al matching col modello di una sfera.

In realtà l'implementazione della versione 2D della Hough trasform [16] ha portato alla luce diverse complicazioni.
Come già specificato la forma irregolare del *blob* risulta essere la causa prima del fallimento della tecnica ma l'introduzione di dati aggiuntivi sulla terza dimensione dovrebbe comportare un netto miglioramento delle prestazioni dell'algoritmo. In realtà un'analisi sui valori assunti dalla depth map in corrispondenza dei pixel associati al *blob* evidenzia come questi si discostino fortemente dal caso ideale rappresentato in figura 13:

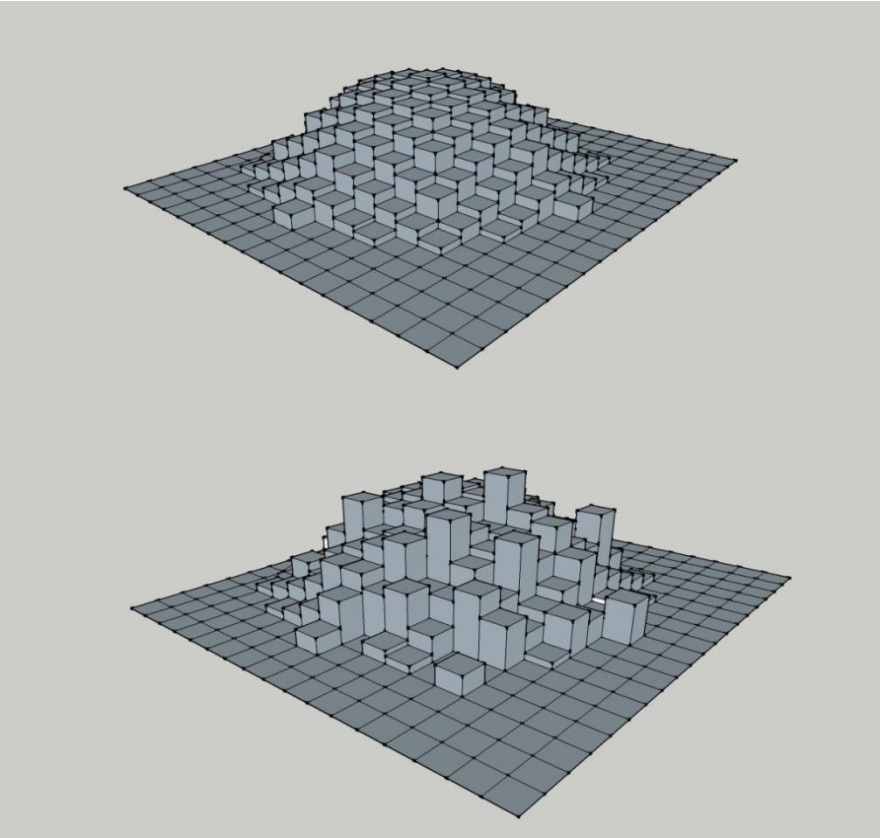


Fig. 13 In alto: ricostruzione ideale di una semisfera mediante mappa di profondità. In basso: caso reale ottenuto mediante sensore *Asus Xtion Pro Live*

Ecco dunque che l'accuratezza con cui vengono stimate le distanze dei pixel corrispondenti al *blob* non è tale da permettere l'uso di tecniche che sfruttano la terza dimensione.

Per lo stesso motivo si è scelto di non determinare il centro del marker basandosi sulla ricerca del pixel più vicino ovvero cha assume, all'interno del *blob*, il valore inferiore (più scuro) che sempre in via teorica rappresenta esattamente il punto sulla superficie del marker più vicino all'obiettivo, nonché la proiezione del centro della sfera sulla superficie in direzione del raggio ottico.

Alla luce di ciò è giustificata la scelta di procedere all'elaborazione 2D dell'immagine nonostante la possibilità di risalire all'informazione 3D.

Per evitare di lavorare con immagini a bassa risoluzione, nel definire i bordi del *blob* e ottenerne una stima più robusta delle coordinate del centro è stata considerata la possibilità di lavorare non sulla mappa di profondità ma a livello di immagine IR, sicuramente meno rumorosa.

Tale procedimento sembra fornire buoni risultati ma si tratta di una tecnica molto laboriosa.
Lo svantaggio di tale metodo si rende esplicito nell'esigenza di articolarsi necessariamente in due step: innanzitutto si deve prevedere una prima acquisizione della scena con l'esclusione del proiettore per le ragioni esposte al capitolo precedente, mentre, in un secondo momento, occorrerà procedere con una nuova acquisizione, questa volta mantenendo attivo il proiettore, in modo da poter ottenere l'informazione relativa alla tridimensionalità.

A questo punto è nota una stima delle coordinate X_p e Y_p secondo O_i (Sistema di riferimento dell'immagine) del centro P del cerchio 2D proiezione della sfera sul piano immagine.

Ora, con riferimento all'immagine 19, dalle grandezze X_P, Y_P e noto il valore assunto dal corrispondente pixel nella mappa di profondità , è possibile, sfruttando le primitive fornite dalle librerie OpenNI[31], ricavare le coordinate 3D del punto Q nello spazio reale proiettato in quel determinato pixel.

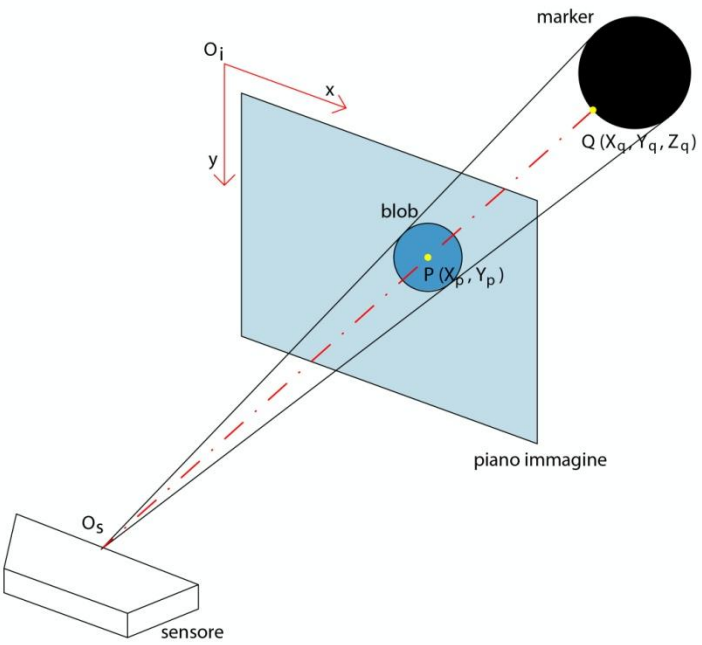


Fig. 14 Proiezione della sfera nel piano immagine del sensore.

La funzione in questione è la seguente:

depthGenerator.ConvertProjectiveToRealWorld(int num_point,XnPoint3D &proj,XnPoint3D &real);

dove:

- depthGenerator rappresenta il nodo generatore delle depth maps
- num_point è il numero di punti che devono essere convertiti dalle coordinate proiettive a quelle reali
- proj è il puntatore al punto/i in coordinate proiettive : si tratta di un dato in forma XnPoint3D i cui campi sono: XnPoint3D.X che assume il valore della coordinata x espressa nel sistema di coordinate del piano immagine, XnPoint3D.Y valore della coordinata Y e XnPoint3D.Z ossia il valore assunto dal pixel.
- real è il puntatore restituito ai punti in coordinate reali

La funzione restituisce un puntatore al vettore o al singolo punto (in base al numero di punti che si stanno convertendo) XnPoint3D dove i diversi campi XnPoint3D.X , XnPoint3D.Y, XnPoint3D.Z contengono le coordinate espresse in mm secondo il sistema di coordinate del sensore raffigurato in figura seguente che d'ora in poi indicheremo con O_s:

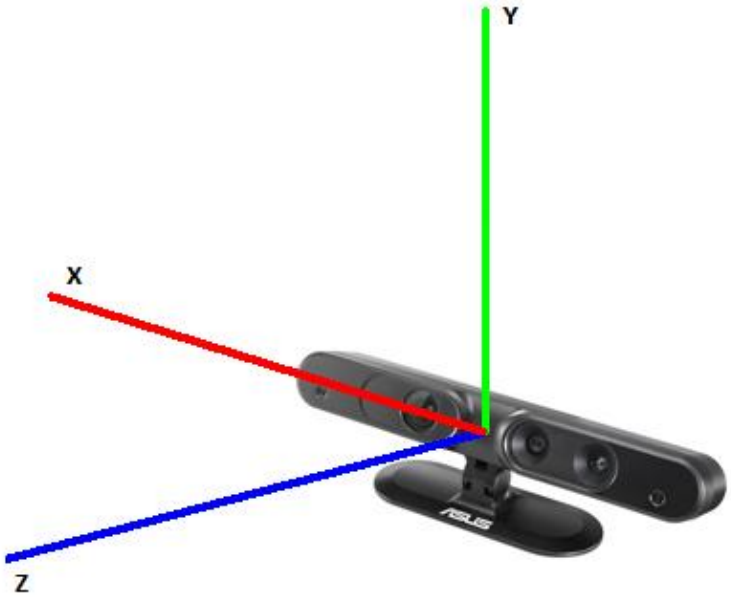


Fig. 15 Sistema di riferimento del sensore.

Con riferimento all'immagine 19 a questo punto sono note per ogni marker le coordinate 3D, espresse in mm secondo Os, del punto Q sulla superficie della sfera più vicino al sensore.

Per stimare le coordinate del punto di controllo P_i i=1,….,9, nonché centro delle sfera i-esima è possibile procedere in questo modo: per ogni marker ,ad ogni coordinata di Q_i (secondo O_s), si somma la componente, rispettivamente scomposta secondo gli assi X_s,Y_s,Z_s di O_s , del raggio del marker pari a 35mm.

Riportandosi al problema nel piano XZ con riferimento alla figura 16:

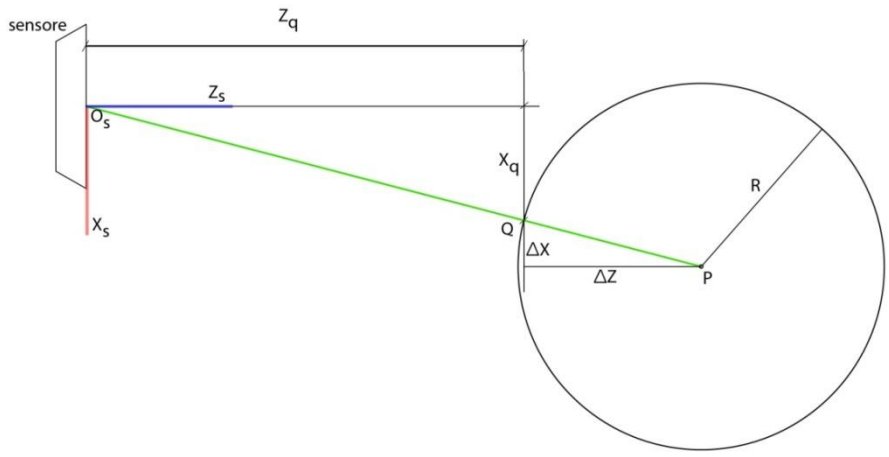


Fig. 16 Calcolo del centro P della sfera a partire dalle coordinate di Q nel caso 2D

Note le coordinate di $Q=[X_Q,Y_Q,Z_Q]$ e noto il raggio R del marker, allora:

$$X_P = X_Q + \Delta x$$

$$Z_P = Z_Q + \Delta z$$

dove per la similitudine dei triangoli:

$$\Delta x = \frac{R \cdot X_Q}{\overline{QO_S}}$$

$$\Delta z = \frac{R \cdot Z_Q}{\overline{QO_s}}$$

Per ognuno dei marcatori è quindi possibile stimare la posizione del centro P_i per i=1,…,9; che rappresentano i punti di controllo che andranno a formare la point cloud per il calcolo dei parametri estrinseci di calibrazione.

5.2.3 Identificazione dei markers

Una volta note le coordinate dei centri di ciascuna sfera, secondo i sistemi di riferimento dei due sensori, occorre procedere all'identificazione dei marker in modo tale da poter mappare punti della prima telecamera in punti corrispondenti della seconda.

In particolare il sistema di marcatori utilizzato prevede l'utilizzo di 3 sfere allineate lungo una bacchetta in alluminio con interdistanze note a priori e tali da permetterne un'identificazione univoca. Così facendo è possibile associare ad ogni sfera un label di identificazione e aggiungere le coordinate di ognuna in due macrovettori ordinati (*realPoint1[N]*, *realPoint2[N]* con N=3 x K, K = numero acquisizioni) che conterranno tutti i punti della point cloud rispettivamente per il sensore sensors[0] e sensors[1].

L'algoritmo di identificazione in forma di pseudocodice è riportato in appendice C.

I risultati di questa fase di elaborazione vengono in seguito visualizzati a video assegnando ad ogni sfera una particolare tonalità di colore. Nel caso delle tre sfere ad A è associato il colore blu, a B il colore verde e a C il rosso.

Nel caso invece si utilizzi il *tool* a nove sfere (indicate con le lettere A, B, C,D,E,F,H,I secondo lo schema riportato in precedenza) per la definizione di un sistema di coordinate globale noto all'operatore, viene assegnato un colore per ogni asse (Z=blu; X=rosso; Y=verde) e ad ogni sfera di quell'asse una sua particolare sfumatura:

- Sfere A,B,C,D colore blu di intensità crescente a partire dal marker D al C.

- Sfere E,F,G colore rosso di intensità crescente a partire dal marker E al G.
- Sfere H,I colore verde di intensità crescente a partire dal marker H al I.

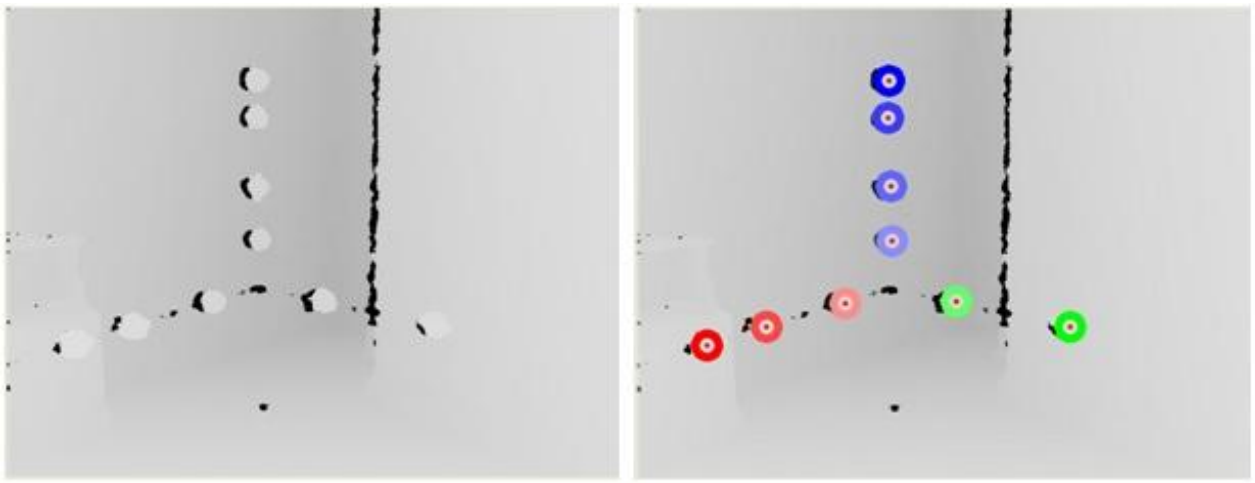


Fig. 17 Risultato della fase di identificazione nel caso di pattern a nove sfere.

Una volta stimata la posizione dei punti di controllo in un numero K (circa 300) di frame e memorizzate, in modo ordinato, le loro posizioni in *realPoint1[N]* e *realPoint2[N*] è possibile avanzare allo step successivo di calcolo dei parametri di rototraslazione attraverso il tasto che in figura 6 è evidenziato in viola.

5.2.4 Stima dei parametri di rototraslazione: l'algoritmo di Horn

A questo punto non resta che procedere alla stima della trasformazione di coordinate che permette di sovrapporre il sistema di riferimento del primo dispositivo con quello del secondo a partire dalle coordinate dei punti di controllo contenuti nei vettori *realPoint1[N]* e *realPoint2[N]*.

Questo problema, in letteratura, viene indicato come: "absolut orientation problem" storicamente risolto mediante metodi grafici, empirici o iterativi. Solo nel 1987, per primo Berthold K.P.Horn in [17] riporta la soluzione al problema in "forma chiusa".

L'algoritmo presentato da Horn permette di lavorare su un numero di punti arbitrario e di stimare i parametri di rototraslazione tramite una formulazione in forma chiusa rispettando quelli che sono i vincoli imposti al problema (quali ortonormalità della matrice di rotazione etc.).

L' "*absolut orientation problem*" presenta sette gradi di libertà: un vettore traslazione (tre scalari), tre gradi di libertà legati alla rotazione (direzione dell'asse secondo cui avviene la rotazione e angolo di rotazione attorno ad esso) e infine uno scalare a rappresentare il fattore di scala. La posizione nota di tre punti non allineati nello spazio fornisce all'operatore un totale di 9 vincoli (3

coordinate per ogni punto) rendendo il problema determinato. Nel caso specifico non è stata considerata la presenza di un possibile fattore di scala dato che le coordinate in output dai due sensori sono espresse nelle stesse unità metriche.

Un netto vantaggio nell'utilizzo di una formula risolutiva in forma chiusa è il fatto di fornire la stima dei migliori parametri possibili in un solo passaggio senza quindi la necessità di iterazioni multiple e senza dover conoscere a priori una buona stima iniziale dei parametri stessi. La soluzione viene fornita in forma di quaternione, una particolare formulazione che permette di definire una rotazione senza dover tener conto dei fenomeni di singolarità presenti nelle più note forme Euleriana o Cardanica.

Horn, in particolare, dimostra come il quaternione che descrive la rotazione R in grado di sovrapporre i due sistemi di riferimento in esame sia l'autovettore corrispondente all'autovalore massimo della matrice simmetrica 4x4 i cui elementi sono semplicemente combinazione di somme e prodotti delle coordinate dei punti noti.

L'affermazione è dettagliatamente argomentata in [17], qui mi limito a riportare come procedere operativamente.

Ciò che l'algoritmo permette di stimate sono i parametri di una trasformazione del tipo:

$$r_r = R(r_l) + r_0 \qquad [1]$$

che permette di descrivere i punti visti dal sistema di coordinate di sinistra, indicato col la lettera l (left), secondo il sistema di destra r (right).

Dati N punti, con $\{r_{l,i}\}$ $\{r_{r,i}\}$ ($i=1...N$) vengono indicate le coordinate del punto i-esimo secondo il sistema di riferimento rispettivamente di l e di r..

Per prima cosa, per entrambi i sistemi, viene calcolato il centroide $\bar{r}_r$ e $\bar{r}_l$ dei punti misurati. Ovvero:

$$\bar{r}_r = \frac{1}{n}\sum_{i=1}^{N} r_{r,i} \qquad \bar{r}_l = \frac{1}{n}\sum_{i=1}^{N} r_{l,i} \qquad [2]$$

A questo punto tutte le coordinate dei punti vengono riscritte in rapporto al centroide di riferimento, ossia si definiscono le grandezze:

$$r'_{l,i} = r_{l,i} - \bar{r}_l \quad \text{e} \quad r'_{r,i} = r_{r,i} - \bar{r}_r \quad \text{con} \quad i = 1,...., N$$

Per ogni coppia di punti (secondo r e secondo l) vengono calcolati i prodotti di tutte le nove combinazione possibili che si ottengono prendendo una componente del punto i-esimo espresso secondo r e una componente del punto corrispondente secondo l, ossia:

$$x'_{l,i}\, x'_{r,i}\, , x'_{l,i}\, y'_{r,i}\, , \ldots\ldots, z'_{l,i}\, z'_{r,i} \quad \text{con} \quad i = 1, \ldots, N$$

che vengono poi utilizzati per determinare gli elementi della matrice *M* 3x3 così definita:

$$M = \begin{pmatrix} S_{xx} & xy & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{pmatrix} \quad \text{dove } S_{xy} = \sum_{i=1}^{N} x'_{l,i}\, y'^{T}_{r,i} \quad \text{e} \quad S_{xx} = \sum_{i=1}^{N} x'_{l,i}\, x'^{T}_{r,i} \qquad [3]$$

La matrice *M* contiene tutte le informazioni necessarie per risolvere il problema ai minimi quadrati della stima dei parametri di rotazione tra i due sistemi *r* e *l*.

Gli elementi della matrice *M* vengono combinati a loro volta per determinare i dieci elementi indipendenti che compongono la matrice fondamentale, reale e simmetrica , *N*.

La matrice *N* 4x4 è così composta:

$$N = \begin{pmatrix} (S_{xx} + S_{yy} + S_{zz}) & S_{yz} - S_{zy} & S_{zx} - S_{xz} & S_{xy} - S_{yx} \\ S_{yz} - S_{zy} & (S_{xx} - S_{yy} - S_{zz}) & S_{xy} + S_{yx} & S_{zx} + S_{xz} \\ S_{zx} - S_{xz} & S_{xy} + S_{yx} & (-S_{xx} + S_{yy} - S_{zz}) & S_{yz} + S_{zy} \\ S_{xy} - S_{yx} & S_{zx} + S_{xz} & S_{yz} + S_{zy} & (-S_{xx} - S_{yy} + S_{zz}) \end{pmatrix}$$

Da questi elementi è possibile calcolare i coefficienti del polinomio di quarto ordine che permettono di ottenere gli auto valori λ_j $j=1,\ldots,4$ della matrice *N* stessa.

Dato che *N* è simmetrica e reale i λ_j sono reali. Selezionato il maggiore è possibile calcolarne l'autovettore reale associato risolvendo l'equazione omogenea associata:

$$\det(N - \lambda_j I) = 0 \quad \text{dove } I = matrice\,identità = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Ora il quaternione che rappresenta la rotazione cercata è il vettore unitario *Q* nella stessa direzione dell'autovettore appena calcolato.

$$Q = \begin{pmatrix} q_0 \\ q_x \\ q_y \\ q_z \end{pmatrix}$$

Per descrivere la rotazione nella forma più nota di matrice *R* 3x3 ortonormale, si può far riferimento alla rappresentazione in forma matriciale di un quaternione per cui *Q* viene riscritto come:

$$Q = \begin{pmatrix} q_0 & -q_x & -q_y & -q_z \\ q_x & q_0 & -q_z & q_y \\ q_y & q_z & q_0 & -q_x \\ q_z & -q_y & q_x & q_0 \end{pmatrix} \quad \text{e il suo coniugato } \overline{Q} \text{ come} \quad \overline{Q} = \begin{pmatrix} q_0 & -q_x & -q_y & -q_z \\ q_x & q_0 & q_z & -q_y \\ q_y & -q_z & q_0 & q_x \\ q_z & q_y & -q_x & q_0 \end{pmatrix}$$

In tal caso R rappresenta la sottomatrice 3x3 in basso a destra della matrice 4x4 ottenuta dal prodotto tra Q e $\overline{Q}^T$:

$$\overline{Q}^T Q = \begin{pmatrix} \dot{q} \cdot \dot{q} & 0 & 0 & 0 \\ 0 & (q_0^2 + q_x^2 - q_y^2 - q_z^2) & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 0 & 2(q_y q_x + q_0 q_z) & (q_0^2 - q_x^2 + q_y^2 - q_z^2) & 2(q_y q_z - q_0 q_x) \\ 0 & 2(q_z q_x - q_0 q_y) & 2(q_z q_y + q_0 q_x) & (q_0^2 - q_x^2 - q_y^2 + q_z^2) \end{pmatrix}$$

cioè

$$R = \begin{pmatrix} (q_0^2 + q_x^2 - q_y^2 - q_z^2) & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 2(q_y q_x + q_0 q_z) & (q_0^2 - q_x^2 + q_y^2 - q_z^2) & 2(q_y q_z - q_0 q_x) \\ 2(q_z q_x - q_0 q_y) & 2(q_z q_y + q_0 q_x) & (q_0^2 - q_x^2 - q_y^2 + q_z^2) \end{pmatrix}$$

Ora non resta che calcolare i valori dei tre parametri relativi alla traslazione r_0 di eq. [1].

Le componenti di r_0 sono calcolate come differenza tra il centroide delle misure secondo *r* e il centroide secondo *l* ruotato e scalato secondo la matrice *R* e *s*, ossia:

$$r_0 = \bar{r}_r - R(\bar{r}_l)$$

In questo modo tutti i parametri di [1] sono stati stimati e risulta quindi possibile effettuare il cambio di coordinate tra sistema di riferimento sinistro (l) e destro (r).

L'algoritmo, com'è ovvio che sia, dipende fortemente dall'accuratezza con cui vengono stimate le coordinate iniziali dei punti di controllo. Ecco dunque che le fasi di elaborazione dell'immagine, di localizzazione dei *blob* e la stima dei relativi centri, che dipendono a loro volta dalla bontà con cui sono forniti dal sensore i dati sulla profondità, condizionano fortemente i risultati della calibrazione.

5.3 Tool a 9 sfere per definire un sistema di riferimento globale

Solitamente in un laboratorio di analisi del movimento occorre che tutte le misure effettuate siano esprimibili secondo un sistema di riferimento globale definito dall'operatore in modo da poter rendere tali grandezze consistenti e confrontabili.

Lo stesso procedimento utilizzato per determinare una stima dei parametri estrinseci nel caso di due sensori può essere portato al caso in cui, note le coordinate di almeno tre punti non allineati rispetto ad un sistema di riferimento qualsiasi O_{Glo}, e note le posizioni di questi stessi punti rispetto un dispositivo D-RGB si voglia stimare la trasformazione per mappare ogni punto visto dal sensore in questione in coordinate espresse secondo il sistema O_{Glo}.

Per definire la posa di O_{Glo}, che rappresenta nel caso specifico il sistema di riferimento globale del laboratorio, è possibile sfruttare il *tool* costruito in precedenza.

Ecco dunque che, una volta assegnato a ciascuna delle tre barre della terna con le 9 sfere un asse cartesiano (X,Y,Z) è possibile ottenere le informazioni necessarie allo scopo.

Ipotizzando la perfetta ortogonalità delle barre che formano il sistema e quindi della sfere su di esse posizionate è possibile considerare il *tool* stesso come sistema di riferimento globale, che può quindi essere posizionato e orientato a piacere all'interno dell'area di lavoro. In questo caso la posizione delle nove sfere, facilmente descrivibili secondo la terna stessa, forniscono i punti di controllo per le successive elaborazioni.

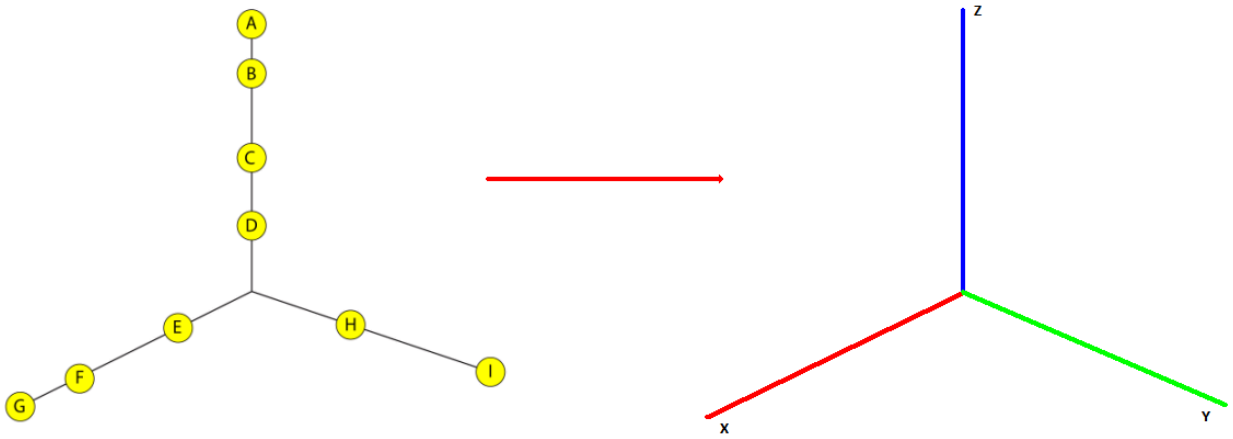


Fig. 18 Definizione di un sistema di riferimento globale a partire dalla terna a nove sfere

In questo senso la sfera A, data la distanza OA pari a 650mm, secondo il sistema globale avrà coordinate (0,0,650), mentre la sfera F, fissata OF=470, sarà rappresentata dal punto (470,0,0) e così via per tutte le altre sette sfere.

Non resta quindi che procedere all'identificazione, secondo quanto prima descritto, delle nove sfere da parte del sensore D-RGB in modo da avere a disposizione tutti i dati per applicare l'algoritmo di Horn e stimare la trasformazione che permette di esprimere qualsiasi punto fornito dal sensore in coordinate rispetto alla terna che definisce ora il sistema di riferimento assoluto.

Nel caso si decida di lavorare con due o più sensori questa trasformazione è comunque possibile e anzi all'operatore si presentato due alternative.

Da una parte è possibile stimare la trasformazione in grado di mappare, per ogni sensore, i propri punti in punti del sistema globale in modo tale da avere alla fine tutti le coordinate espresse secondo il sistema assoluto O_{Glo}. In formule è possibile cioè risalire alle ${}^{G}R_i$ e ${}^{G}O_i$ per $i=1....n$ con n=numero sensori tali per cui:

$$ {}^{G}P = s_i \, {}^{G}R_i ({}^{i}P) + {}^{G}O_i $$

dove ${}^{i}P$ rappresenta un generico punto espresso secondo il sistema di riferimento del generico sensore i-esimo mentre ${}^{G}P$ indica lo stesso punto visto dal sistema globale.

In questo modo tutti i punti di ciascun sensore sono riportati al sistema globale O_{glo}.

Dall'altra parte è possibile stimare dapprima la trasformazione tra il sistema di un sensore definito principale e il sistema globale per poi stimare la trasformazione tra tutti i singoli sensori e il principale.

Procedendo in questo secondo modo è possibile procedere alla stima di ${}^{G}R_{Pr}$ e ${}^{G}O_P$ tali per cui:

$$ {}^{G}P = {}^{G}R_{\mathrm{Pr}} ({}^{\mathrm{Pr}}P) + {}^{G}O_{\mathrm{Pr}} \qquad [4] $$

Dove Pr sta ad indicare il sensore considerato come principale. Si definisce così un cambio di coordinate dal Pr al sistema globale, per poi, per i = 1….n-1 con n = numero sensori, procedere alla stima delle ${}^{\mathrm{Pr}}R_{\mathrm{i\,e}}$ ${}^{\mathrm{Pr}}O_{\mathrm{i}}$ per cui:

$$ {}^{\mathrm{Pr}}P = {}^{\mathrm{Pr}}R_i ({}^{i}P) + {}^{\mathrm{Pr}}O_i \qquad [5] $$

In questo modo dalla composizione della [4] e della [5] si ottiene la trasformazione per esprimere il punto visto dalla camera i-esima secondo il sistema O_{glo}:

$$ {}^{G}P = {}^{G}R_{\mathrm{Pr}} ({}^{\mathrm{Pr}}R_i ({}^{i}P) + {}^{\mathrm{Pr}}O_i) + {}^{G}O_{\mathrm{Pr}} $$

Occorre fare alcune osservazioni sulla tecnica appena illustrata:

essendo la stima dei parametri di rototraslazione tra il sistema di riferimento di una camera qualsiasi e il sistema globale effettuato su un numero massimo di nove punti di controllo la strada che per via teorica permette di ottenere risultati più accurati è sicuramente la seconda. I parametri tra due sensori di profondità vengono infatti calcolati su un dataset con un minimo di 600 punti offrendo quindi le condizioni per una stima più corretta e robusta.

Bisogna considerare infatti che la terna utilizzata nel lavoro di tesi è stata realizzata a mano ed è quindi caratterizzata da una certa incertezza che può introdurre una componente d'errore sulla stima delle coordinate delle nove sfere secondo O_{Glo} che può compromettere la bontà del calcolo.

È da sottolineare infine come il sistema di riferimento utilizzato dal sensore sia un sistema a convenzione destrogira. Tale scelta costringe l'operatore ad effettuare un preventivo cambio di coordinate per permettere il matching delle sfere con il riferimento globale, definito nella più usuale forma levogira.

Ogni punto $p_{or}(x_{or}, y_{or}, z_{or})$ espresso secondo il sistema del sensore deve cioè essere mappato nel punto $p_{el}(x_{el}, y_{el}, z_{el}) = (y_{or}, x_{or}, z_{or})$ da fornire in ingresso al metodo di Horn [17].

Tale procedimento è stato testato in via puramente qualitativa in quanto non sono quantificabili gli errori introdotti nel calcolo dalla costruzione manuale del *tool*.

Se il *tool* fosse prodotto mediante tecniche a controllo numerico sarebbe possibile ridurre le fonti d'errore alla sola stima del centro di ciascuna sfera da parte del sensore rendendo l'algoritmo più robusto.

5.4 Verifica dell'accuratezza della calibrazione

Al fine di quantificare da un punto di vista numerico la bontà della calibrazione ottenibile secondo la tecnica sviluppata si sono svolti dei test in modo da replicare la prova di spot-checks per la misura di distanza tra marcatori.

Lo scopo è quello di ricostruire la posizione di due punti nello spazio a partire dalle depth maps congiunte dei due sensori e valutare la variabilità della loro stima dell'interdistanza in acquisizioni successive.

Come termine di confronto è utilizzata l'interdistanza tra gli stessi punti valutata attraverso il sistema stereofotogrammetrico SMART -DX 100, che rappresenta il gold standard per quanto riguarda la *motion capture* mediante sistemi *marker-based*.

5.4.1 Setup strumentale

Strumenti:

- 2 tripodi
- Sistema di motion capture SMART-DX 100:
  - due telecamere *Basler* e rispettivi illuminatori
  - workstation
- 2 sensori *ASUS Xtion Pro Live*
- *Tool*s di calibrazione:
  - terna con le 9 sfere ricoperte in materiale rifrangente
  - bacchetta a tre sfere per calibrazione
  - bacchetta a due sfere per test

Il setup per lo svolgimento dei test prevede la definizione di due sistemi di acquisizione: il primo, sotto test formato da due dispositivi D-RGB, e il secondo, considerato come *gold standard* formato da altrettante telecamere infrarossi e relativo sistema di illuminazione.

L'obiettivo primo che si è cercato di perseguire è quello di ricreare nei due sistemi, le stesse condizioni di lavoro in modo da rendere i dati acquisiti confrontabili e robusti.

Il sistema sotto test è composto da due dispositivi *ASUS Xtion Pro Live* d'ora in poi indicati con *Depth1* e *Depth2* le cui caratteristiche tecniche sono state già ampiamente discusse nei capitoli precedenti. In particolare la calibrazione del sistema sarà tale da stimare la trasformazione per mappare punti visti dal dispositivo *Depth2* in punti visti da *Depth1* considerato come principale.

Il sistema di *MoCap* SMART-DX/100 in questa specifica configurazione è costituito da due telecamere a infrarossi aventi risoluzione simile (659 x 494) ai sensori D-RGB e da un'unica workstation per l'acquisizione, l'elaborazione e l'analisi dei dati.

Le telecamere BTS usano una tecnologia *CCD* (*Charge Coupled Device*) poiché caratterizzata da un rapporto segnale/rumore superiore ai sensori *CMOS* e da una frequenza di lavoro superiore fino a 200 Hz.

In particolare il sistema è composto da:

- **Illuminatori**: sono montati posteriormente al piano focale di ogni telecamera in modo che la radiazione emessa non disturbi l'acquisizione delle videocamera su cui è montata. Tali illuminatori emettono impulsi di luce infrarossa di lunghezza d'onda 880 nm ad alta potenza controllati digitalmente;
- **Telecamere:** nel caso specifico sono state adottate telecamere *Basler* modello *scA640-120gm*. Ogni telecamera monta un'ottica *Goyo* modello *GM23514MCN* così da avere un campo visivo molto simile a quello ottenuto da Depth1 e Depth2. Le specifiche tecniche delle telecamere e delle lenti sono riportate in appendice B.
- **Workstation**: permette l'acquisizione video delle 2 telecamere e si interfaccia all'utente come un personal computer provvisto di scheda di acquisizione e conversione analogico – digitale dei segnali acquisiti e del software per la gestione del sistema.

L'elaborazione e l'integrazione dei dati ottenuti con le acquisizioni, così come la procedura di acquisizione stessa, necessitano di software specifici per poter essere analizzati. Questi programmi sono contenuti nel pacchetto *SMART-SUITE* che può essere suddiviso nelle componenti: *SMARTcapture, SMARTtracker, SMARTanalyzer*.
In appendice B sono riportate le caratteristiche tecniche del sistema SMART DX100 utilizzato insieme ad una breve descrizione delle tre sezioni.

Sui ogni tripode è montata una telecamera sopra la quale è fissato un sensore D-RGB tra *Depth1* e *Depth2*. La posizione e l'orientamento con cui applicare ogni dispositivo sulle telecamere sono definiti in modo da ottenere, quanto più possibile, la stessa inquadratura. Per far ciò è possibile centrare il campo visivo di ogni camera sui 6 *spot* IR principali proiettati dal sensore nell'area di lavoro.

A questo punto i due tripodi sono posizionati in modo tale da rispettare quei vincoli per l'utilizzo contemporaneo di due sensori D-RGB.

La distanza tra i due tripodi è stata misurata circa 350 cm e sono disposti in modo da tale che l'angolo tra i raggi ottici delle camere formino un angolo di circa 120°-130°.

I tripodi sono quindi rialzati per cui entrambi gli obiettivi delle camere infrarossi siano posti a circa un metro da terra, mentre i sensori D-RGB montati sopra si trovino circa a 110 cm.

Nello specifico in figura seguente è riportata la particolare disposizione adottata nella fase di test.

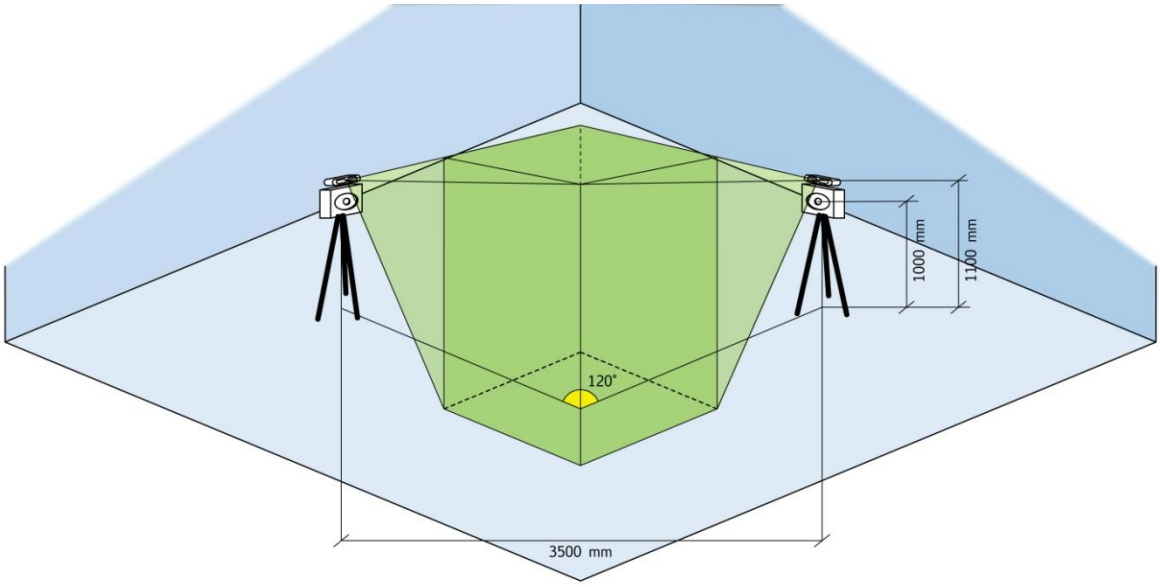


Fig. 19 Disposizione della strumentazione nel laboratorio

5.4.2 Acquisizioni sistema Depth1-Depth2

Una volta definito il setup di lavoro è possibile procedere alla calibrazione del sistema *Depth1-Depth2* secondo quanto descritto nei paragrafi precedenti.

Allo scopo viene mossa all'interno dello spazio di lavoro la bacchetta a tre sfere, appositamente costruita, in modo da generare la point cloud per la stima dei parametri estrinseci del sistema. La bacchetta viene mossa in modo da coprire tutto il volume di spazio visibile ad entrambi i sensori servendosi di una prolunga in modo da non entrare col corpo nell'area di lavoro.

Questa fase ha una durata di 175 secondi durante la quale viene acquisita un'immagine ad intervalli regolari di 500 msec per un totale di 350 frame.

Dei 1050 (350 x 3) punti di controllo ottenibili, l'algoritmo automatico di elaborazione e stima ha fornito in output la posizione di 858 punti considerati effettivamente corretti.

Da questi dati è possibile passare alla fase vera e propria di stima dei parametri estrinseci di calibrazione che viene effettuata sempre in automatico dal software.

A questo punto è possibile procedere alla fase di test.

Secondo un processo del tutto similare a quello appena descritto viene spostata all'interno dell'area di lavoro una bacchetta contenente questa volta due sole sfere.

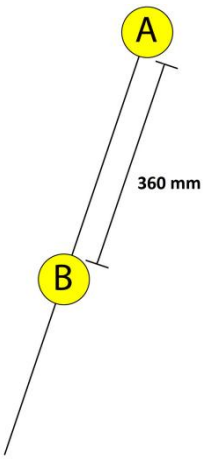


Fig. 20 *Tool* a due sfere per la fase di test

Durante il movimento della bacchetta occorre spazzolare tutto il volume di lavoro del sistema in modo da poter valutare la bontà della calibrazione nelle diverse zone. Durante questa fase occorre spostare la bacchetta orientandola in tutte le possibili direzioni al fine di valutare l'accuratezza del sistema nel ricostruire le posizioni delle sfere, e quindi l'interdistanza, lungo le diverse direzioni dello spazio.

Alla fine della fase di acquisizione è possibile salvare tre file contenenti i dati: il primo riporta la posizione 3D di tutte le coppie ordinate dei punti appena calcolati espressi nel sistema di riferimento di Depth1 (file ordinato così costituito: Riga 1: Sfera_A_Acq.1, Sfera_B_Acq.1, Riga 2: Sfera_A_Acq.2, Sfera_B_Acq.2,…., Riga N: Sfera_A_Acq.N, Sfera_B_Acq.N), il secondo contiene le coordinate delle stesse coppie di punti espresse secondo il sistema di Depth2 e trasformati mediante i parametri di calibrazione stimati e infine un terzo file contenente i punti costruiti mediando i valori contenuti nei file precedenti, ossia i punti ricostruiti dall'informazione congiunta di *Depth1* e *Depth2*.

Per ogni serie di misure, per ogni coppia ordinata di punti (SferaA-SferaB) è stata calcolata l'interdistanza che verrà in seguito utilizzata per la quantificazione della bontà della calibrazione.

Tutto il processo di calibrazione e di test è stato effettuato per quattro volte senza modificare la disposizione dei sensori, permettendo così di acquisire quattro serie di misure per l'interdistanza misurata da *Depth1*, tre serie da *Depth 2* e tre serie di misure dell'interdistanza tra i punti ricostruibili da *Depth1* e *Depth2* congiunti.

In tutti e quattro i test sono stati acquisiti un totale di 110 frame ma, solo una parte di questi sono stati correttamente processati fornendo la misura di 90 interdistanze. Il processo di calibrazione invece è stato effettuato su un numero di punti differente di volta in volta in base ai risultati elaborati dall'algoritmo.

Nel primo tentativo il processo automatico di elaborazione e stima ha reso disponibile un totale di 858 punti degli 1050 ottenibili su cui effettuare la calibrazione.

Nel secondo esperimento la point cloud è costituita da 870 punti .

Nella terza prova sono stati considerati effettivamente corretti un totale di 855 punti mentre, nella quarta ed ultima prova si è lavorato su un data set di 708 punti per la stima della rototraslazione.

5.4.3 Acquisizioni sistema SMART-DX/100

Si procede quindi all'acquisizione dei dati dal sistema SMART con cui confrontare i valori ottenuti in precedenza.

Innanzitutto occorre effettuare la calibrazione del sistema formato dalle due camere mediante il protocollo thor2 di BTS.

Come già illustrato, nella sequenza *axes* viene definita la posizione della terna di riferimento del laboratorio; mentre nella successiva fase di *Wand* si definisce lo spazio fisico dove effettuare le misurazioni di interesse, andando a raffinare i dati inizializzati in precedenza.

Per la fase di *Axes* la terna adottata è quella discussa in precedenza contenente le nove sfere che sono state per l'occasione ricoperte di materiale rifrangente. Si è deciso di utilizzare questa terna in modo da far sì che i due sistemi risultino calibrati con strumenti aventi precisioni comparabili.

Per la seconda fase è stato utilizzato lo stesso *tool* di tre sfere utilizzato in precedenza per la calibrazione del sistema *Depth1-Depth2* , solo, questa volta, ricoperte di materiale rifrangente.

Una volta calibrato il sistema è possibile procedere alla successiva fase di test.

Come per il sistema *Depth1-Depth2*, la medesima bacchetta con applicate le due sfere ricoperte, viene spostata all'interno dell'area di lavoro cercando di coprire lo stesso volume che in precedenza e orientando la bacchetta in tutte le varie direzioni. Questa fase di acquisizione è stata ripetuta tre volte per ognuna delle quali sono stati acquisite 6000 frame ad una frequenza di 100 Hz. Nei tre casi all'elaborazione con lo *smart tracker* e *smart analyzer* (vedi appendice B) si sono ottenute tre serie rispettivamente di 5904, 5106 e 5715 misure di interdistanza tra markers.

Il processo di copertura delle sfere mediante il particolare nastro a microsfere di vetro, indispensabile per effettuare il test col sistema SMART, può introdurre un errore o meglio una variazione delle caratteristiche del *tool* utilizzato nel test.

Le sfere prima di procedere alle diverse acquisizioni sono quindi state fermamente fissate alla bacchetta (in modo da non spostarle maneggiando la bacchetta) e dopo aver effettuato il test per il

sistema D-RGB sono state ricoperte cercando di ottenere il più possibile una superficie regolare senza sovrapposizioni di nastro.

Fig. 21 Sfera ricoperta in materiale rifrangente a microsfere di vetro

Il risultato è visivamente accurato e l'ordine di grandezza dello spessore del nastro risulta trascurabile per il sistema sotto test.

Capitolo 6

Risultati e discussioni

6.1 Confronto D-RGB e stereofotogrammetria

I dati acquisiti in fase di test possono quindi essere riassunti secondo quanto segue:

- Da test su sistema D-RGB:
  - 4 set di parametri di calibrazione
  - 4 serie di 90 coppie di coordinate 3D ricavate da *Depth1*
    → 4 serie di 90 interdistanze da *Depth1*
  - 4 serie di 90 coppie di coordinate 3D ricavate da *Depth2*
    → 4 serie di 90 interdistanze da *Depth2*
  - 4 serie di 90 coppie di coordinate 3D ricavate dall'informazione congiunta di *Depth1* e *Depth2*
    → 4 serie di 90 interdistanze da *Depth1 e Depth2*

- Da test su sistema SMART-DX/100
  - 3 serie di coppie di punti ricavate dall'analisi con SMART
    → 1° serie: 5904 interdistanze da sistema SMART
    → 2° serie: 5106 interdistanze da sistema SMART
    → 3° serie: 5715 interdistanze da sistema SMART

In tabella seguente riporto i valori stimati per i parametri di calibrazione per le 4 diverse acquisizioni col sistema D-RGB.

Traslazione				
	Serie 1	Serie 2	Serie 3	Serie 4
X [mm]	-1762.33	-1761.14	-1759.08	-1761.67
Y [mm]	682.824	683.093	692.629	687.814
Z [mm]	2699.14	2706.38	2694.29	2696.47

Rotazione				
	Serie 1	Serie 2	Serie 3	Serie 4
R[1][1]	-0.474874	-0.477464	-0.475028	-0.4746
R[1][2]	0.182146	0.181994	0.189182	0.183359
R[1][3]	0.860998	0.859596	0.859395	0.860891
R[2][1]	-0.24076	-0.24244	-0.241685	-0.243939
R[2][2]	0.914136	0.913047	0.911011	0.91235
R[2][3]	-0.326176	-0.327974	-0.334136	-0.328801
R[3][1]	-0.846481	-0.844542	-0.846131	-0.845723
R[3][2]	-0.362186	-0.364996	-0.366427	-0.366054
R[3][3]	-0.390246	-0.391825	-0.387033	-0.388273

Tab. 1 Parametri di calibrazione stimati nelle 4 serie di misure

Nonostante l'errore introdotto dall'algoritmo di segmentazione e l'errore di disretizzazione dello spazio 3D dovuto alla bassa risoluzione della mappa di profondità nella stima delle coordinate del centro, l'utilizzo di una nuvola costituita da un numero significativo di punti (>500) consente una stima robusta dei parametri di calibrazione.

Nello stimare i parametri di traslazione la differenza massima tra due valori corrispondenti è di circa 9 mm ossia inferiore all'errore stimato dalla casa produttrice nella generazione della *depth map*.

Sono di seguito riportati il valor medio, la varianza e la deviazione standard delle quattro serie di misure dell'interdistanza ottenute rispettivamente per la telecamera 1, la telecamera 2 e dalla ricostruzione da entrambe le camere 1 e 2.

	Serie 1			Serie 2		
	Tel 1	Tel 2	Tel 1+2	Tel 1	Tel 2	Tel 1+2
Media [mm]	362,218	364,875	361,848	361,107	364,780	362,824
Var [mm]	21,772	42,185	18,878	25,058	28,670	12,579
S.D. [mm]	4,666	6,495	4,344	5,006	5,354	3,546

	Serie 3			Serie 4		
	Tel 1	Tel 2	Tel 1+2	Tel 1	Tel 2	Tel 1+2
Media [mm]	360,858	364,592	362,617	361,411	365,100	363,165
Var [mm]	37,332	32,313	22,429	33,412	37,011	20,772
S.D. [mm]	6,110	5,684	4,746	5,780	6,084	4,558

* Il valor medio sulle quattro serie, dell'interdistanza tra i marker del tool di test, è pari a: 362,863 mm.

Tab. 2 Media, varianza e deviazione standard delle misure effettuate con sistema D-RGB

Dalla tabella si nota come l'utilizzo di due sensori riduca la varianza nel calcolo dell'interdistanza rendendo i valori ottenuti più attendibili.

In tabella seguente sono riportati il valor medio, la varianza e la *standard deviation* per ognuna delle tre serie di misure da sistema SMART.

	Serie 1	Serie 2	Serie 3
Media[mm]	360,531	360,357	360,711
Var [mm]	0,653	1,638	0,966
S.D. [mm]	0,808	1,279	0,983

*il valor medio sulle tre serie della distanza tra i marker vale: 360,533.

Tab. 3 Media, varianza e deviazione standard delle misure effettuate con sistema SMART

Di seguito vengono riportati il valor massimo, il valor minimo e il valor medio di *RMSD* intra-serie ed inter-serie tra tutte le possibili combinazioni tra le diverse serie di misure.

Il *root-mean-square deviation(RMSD) o root-mean-square error (RMSE),* è una misura di uso frequente delle differenze tra i valori ottenuti da un modello o da uno stimatore ed i valori effettivamente osservati. Il RMSD è una buona misura di precisione e serve per aggregare i valori degli errori nelle previsioni di una serie di acquisizioni in una singola misura di potere predittivo.

In alcune discipline, viene utilizzato per confrontare le differenze tra due grandezze che possono variare, nessuno dei quali è accettata come "standard".

Nel caso specifico, quando si misura la differenza media tra due serie di misure $x_1(i)$ e $x_2(i)$ per $i = 1....n$ con n numero campioni, i valori di RMSD intra-serie sono calcolati come:

$$RMSD_i = \sqrt{\frac{\sum_{k=1}^{n} (x_1(i) - x_1(k))^2}{n}}$$

E quindi il valor medio come:

$$\overline{RMSD} = \frac{\sum_{i=1}^{n} RMSD_i}{n}$$

I valori di RMSD inter-serie sono invece definiti come:

$$RMSD_i = \sqrt{\frac{\sum_{k=1}^{n} (x_1(i) - x_2(k))^2}{n}}$$

E quindi il valor medio come:

$$\overline{RMSD} = \frac{\sum_{i=1}^{n} RMSD_i}{n}$$

RMSD intraserie D-RGB			
	RMSD max	**RMSD min**	**RMSD medio**
Serie 1	8,716	4,786	7,361
Serie 2	7,884	4,809	6,543
Serie 3	8,637	6,416	7,681
Serie 4	8,725	6,190	7,637

*misure espresse in mm

Tab. 4 RMSD intra-serie per le misure effettuate con sistema D-RGB

Si sottolinea come il valor minimo di *RMSD* si presenti, per ogni serie, nel caso di confronto tra i dati congiunti delle camere 1 e 2, ad indicare appunto la miglior ripetibilità delle misure nel caso di utilizzo di dispositivi multipli.

RMSD D-RGB Vs SMART			
	RMSD Max	**RMSD Min**	**RMSD medio**
Serie 1 VS SMART 1	5,891	3,629	4,565
Serie 1 VS SMART 2	6,138	3,907	4,841
Serie 1 VS SMART 3	5,870	3,659	4,566
Serie 2 VS SMART 1	5,616	3,581	4,451
Serie 2 VS SMART 2	5,860	3,867	4,692
Serie 2 VS SMART 3	5,552	3,569	4,446
Serie 3 VS SMART 1	5,711	4,416	5,135
Serie 3 VS SMART 2	5,962	4,650	5,350
Serie 3 VS SMART 3	5,652	4,416	5,129
Serie 4 VS SMART 1	6,195	4,579	5,233
Serie 4 VS SMART 2	6,428	4,807	5,447
Serie 4 VS SMART 3	6,142	4,553	5,216

*misure espresse in mm

Tab. 5 RMSD Tra i dati ottenuti con sistema D-RGB e quelli ottenuti con sistema SMART

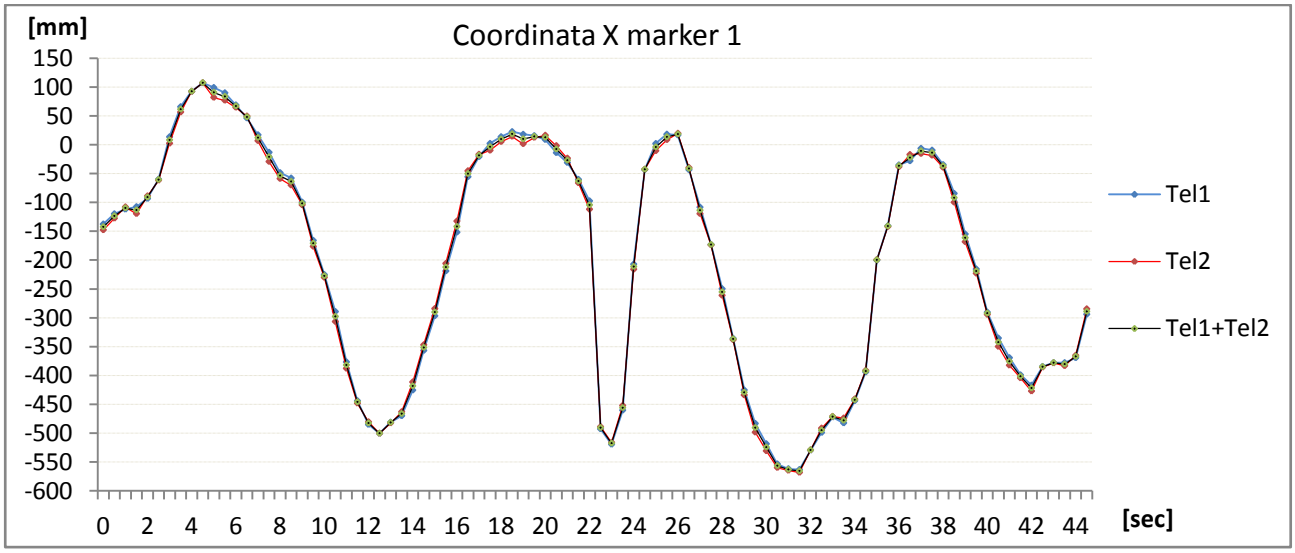
Anche in questo caso i valori minimi si presentano in presenza dei dati provenienti dai dati incrociati di entrambe le camere.

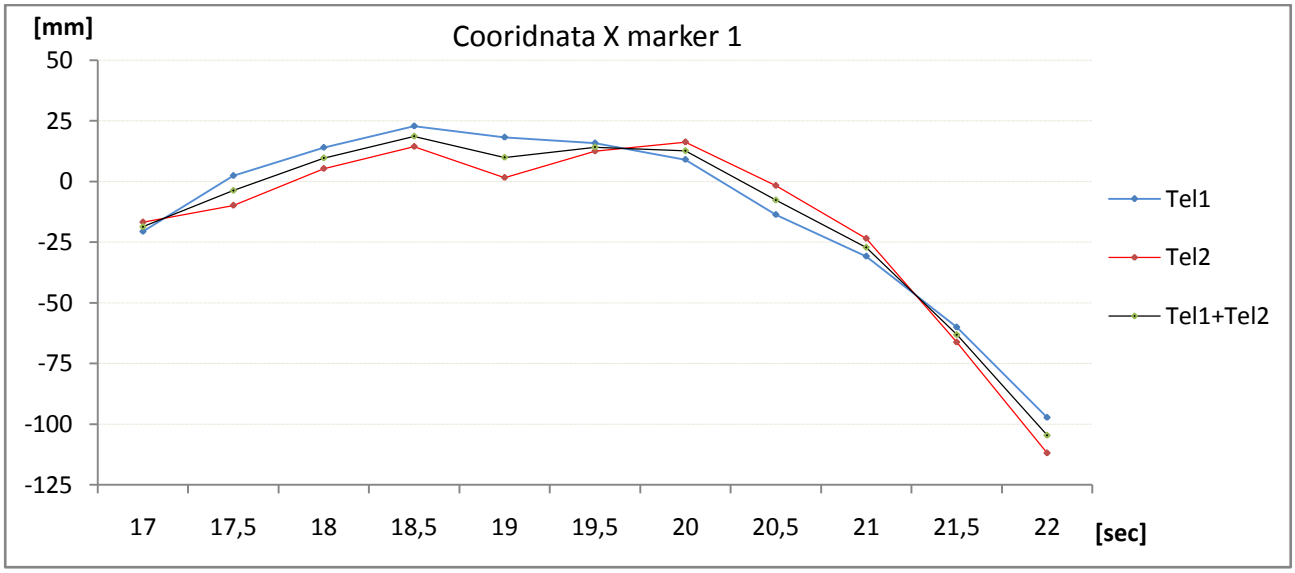
SMART VS SMART	
	RMSD
SMART 1 VS SMART 1	1,082
SMART 1 VS SMART 2	1,498
SMART 1 VS SMART 3	1,229
SMART 2 VS SMART 1	1,275
SMART 2 VS SMART 2	1,659
SMART 2 VS SMART 3	1,415
SMART 3 VS SMART 1	1,188
SMART 3 VS SMART 2	1,605
SMART 3 VS SMART 3	1,297

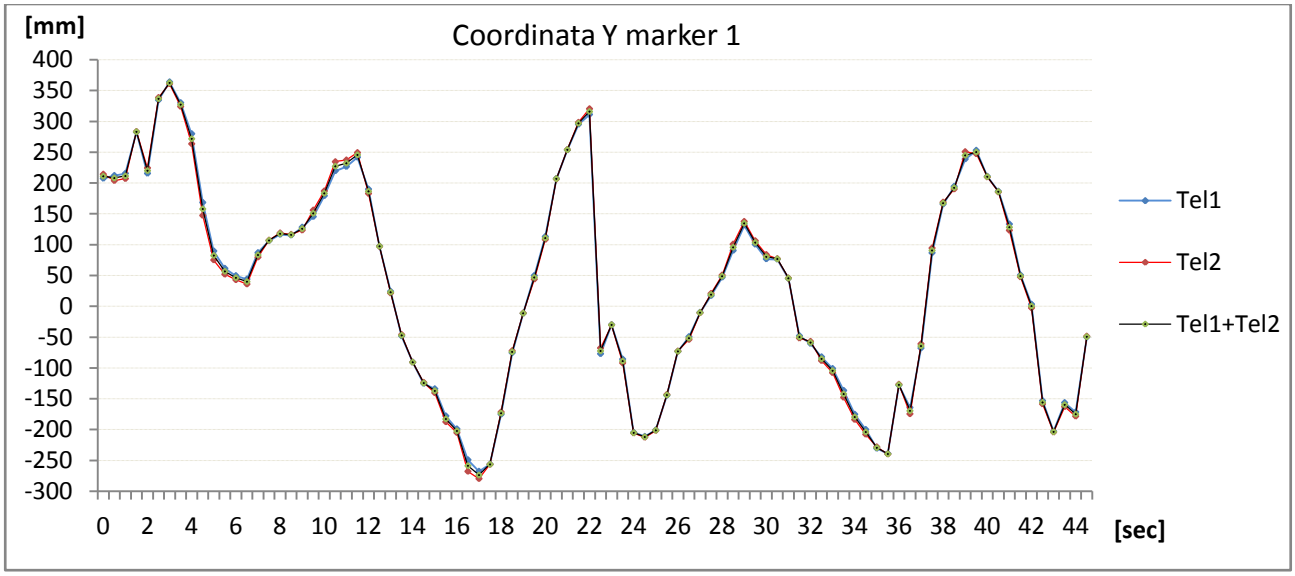
*misure espresse in mm

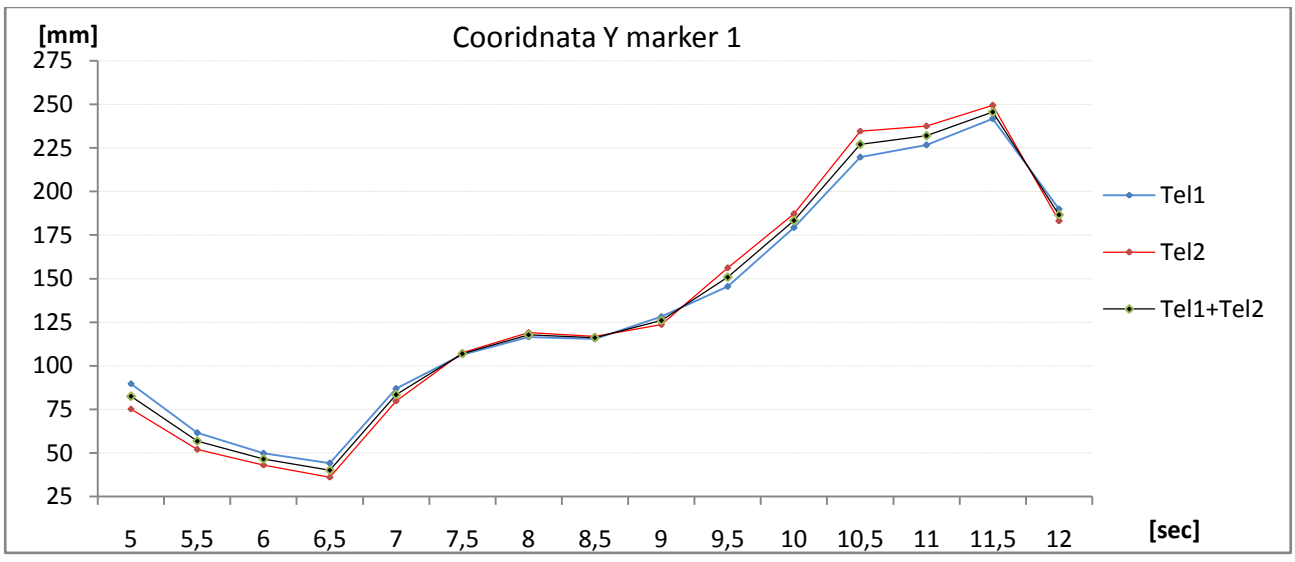
Tab. 6 RMSD intra-serie per le misure effettuate mediante sistema SMART

In seguito vengono riportati gli andamenti lungo i tre assi X,Y,Z delle traiettorie di uno dei due marcatori ricostruite da *Depth 1*, da *Depth2* e dall'informazione congiunta delle due nella prima serie di acquisizione mediante sistema D-RGB e sotto un dettaglio di ognuno. Tutti i dati sono espressi nel sistema di riferimento di *Depth1* considerato principale.




[mm]
Cooridnata X marker 1
50
25
0
-25
-50
-75
-100
-125
17
17,5
18
18,5
19
19,5
20
20,5
21
21,5
22
[sec]
Tel1
Tel2
Tel1+Tel2


[mm]
Coordinata Y marker 1
400
350
300
250
200
150
100
50
0
-50
-100
-150
-200
-250
-300
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
38
40
42
44
[sec]
Tel1
Tel2
Tel1+Tel2


[mm]
Cooridnata Y marker 1
275
250
225
200
175
150
125
100
75
50
25
5
5,5
6
6,5
7
7,5
8
8,5
9
9,5
10
10,5
11
11,5
12
[sec]
Tel1
Tel2
Tel1+Tel2

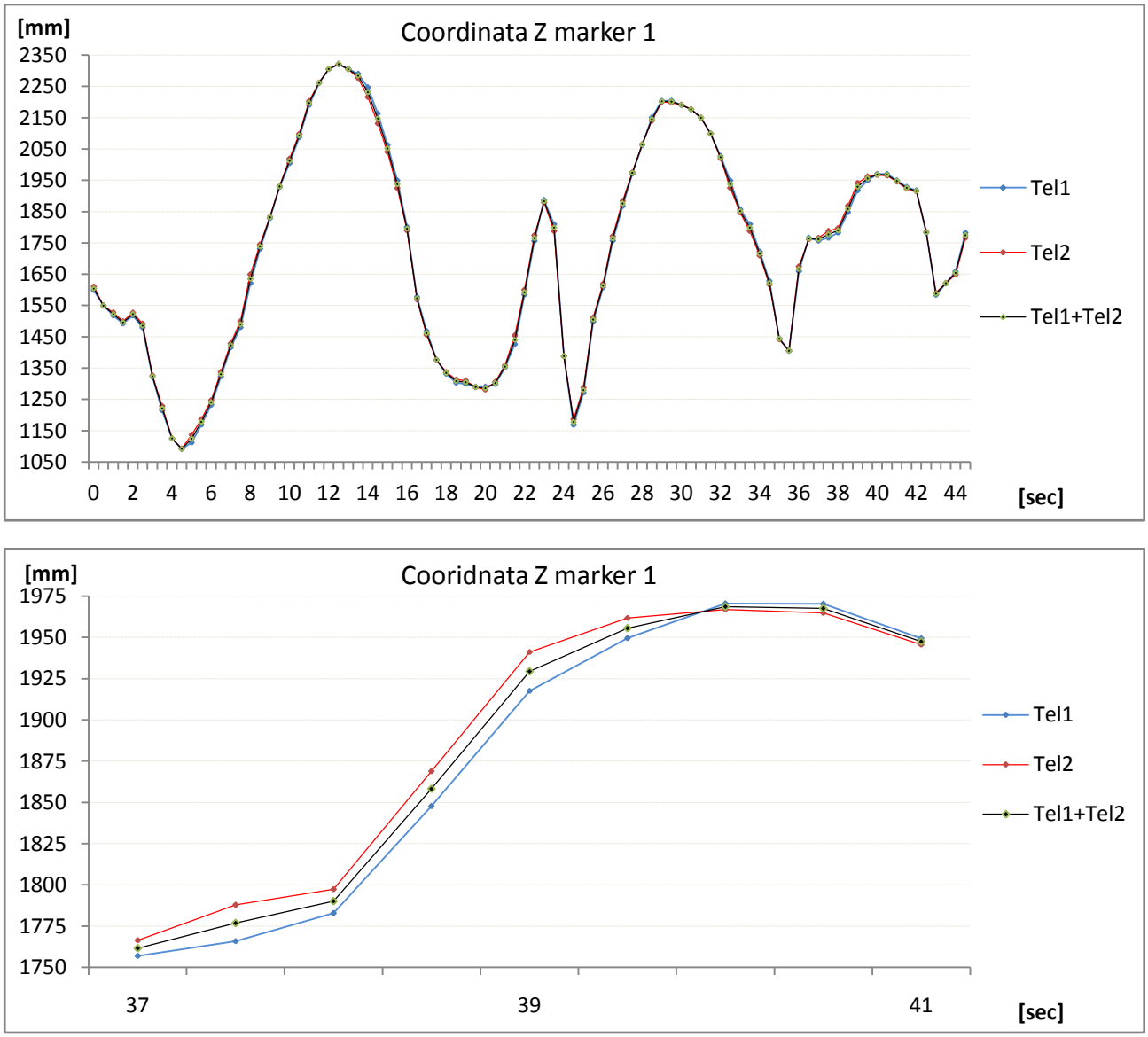


Fig. 1 Andamento nel tempo rispettivamente per le coordinate X, Y,Z del marker 1 misurate da Depth1, Depth2, e dalle due insieme.

Nell'elaborare i dati non sono stati considerati eventuali outliers presenti nelle misure. Con outliers si intendono quelle coordinate dei *marker 1* e *marker 2* la cui interdistanza si discosti di almeno 2 volte la standard deviantion dalla media delle misure stesse. Questa analisi ha condotto all'eliminazione di un solo campione acquisito durante la prima serie di misure in cui l'algoritmo di segmentazione ha evidentemente elaborato un falso positivo proveniente da un riflesso.

In seguito viene riportata la rappresentazione 3D del test effettuato per il sistema D-RGB.

In particolare le sfere di diametro maggiore collegate dalla barra rossa rappresentano la posizione all'istante zero del tool di test. Le altre sfere minori, rappresentano tutte le differenti posizioni occupate dal tool durante la fase di test (in particolare si tratta della prima serie di acquisizioni). Tutte le misure sono espresse nel sistema di riferimento di *Depth1* riportato in figura.

Il parallelepipedo in nero rappresenta il volume di lavoro calibrato all'interno del quale è stato effettuato il test.

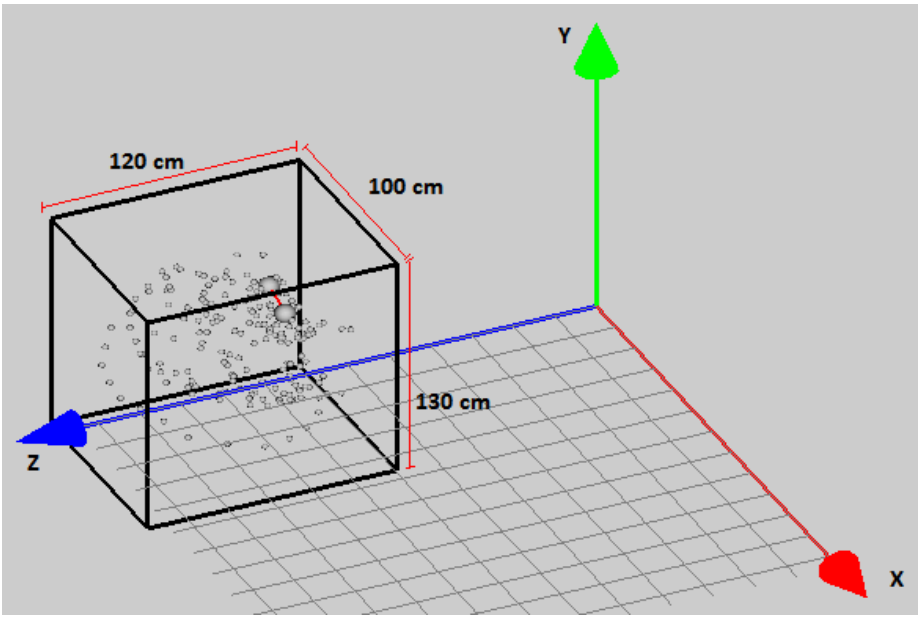


Fig. 2 Ricostruzione 3D della fase di test per il sistema D-RGB

In seguito sono riportati i risultati ottenuti nel test con lo SMART.

Nello specifico, di seguito è riportato uno screenshot della ricostruzione 3D dei punti effettuabile mediante il software SMART tracker di BTS. Tutti i punti sono espressi secondo il sistema di riferimento globale definito mediante il tool a nove sfere nella fase di *Axes*.

Anche in questo caso il parallelepipedo in verde rappresenta il volume di lavoro effettivamente calibrato mentre l'insieme di linee curve in bianco rappresentano le traiettorie seguite dai due marcatori durante l'acquisizione.

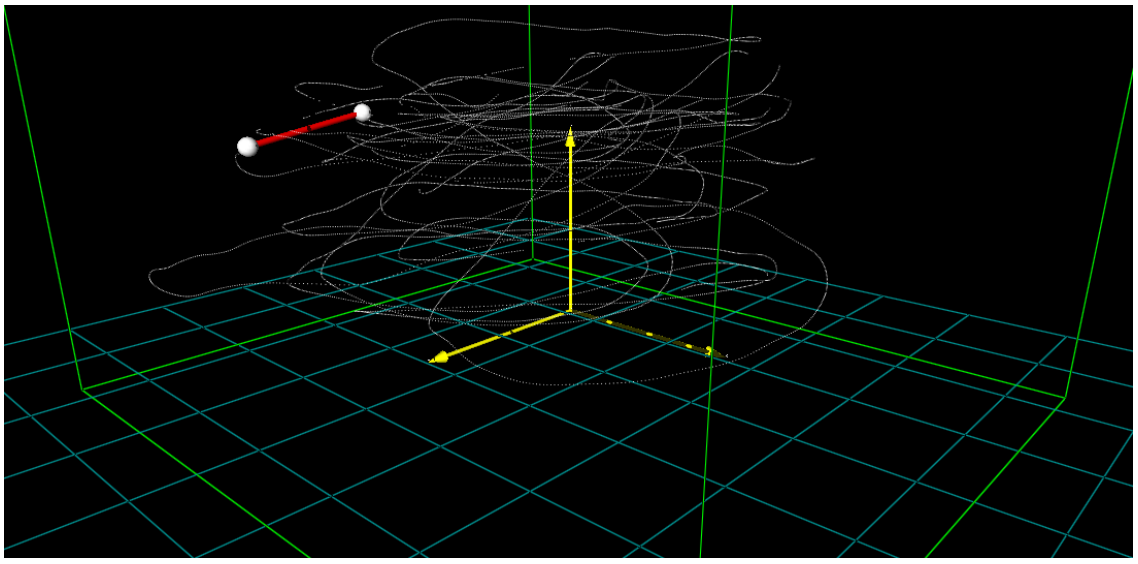


Fig. 3 Ricostruzione 3D delle traiettorie compiute dai marcatori durante la fase di test con sistema SMART

6.2 Ricostruzione di un punto 3D da depth map

La stima del centroide del blob nel piano immagine rappresenta una fase cruciale per la buona riuscita del processo di calibrazione.

È utile perciò quantificare quanto un eventuale errore sulla determinazione di tale punto influisca sul calcolo della posizione del punto di controllo nello spazio reale durante la fase di conversione da coordinate nel piano immagine a coordinate 3D.

Supponendo che l'informazione sulla profondità sia priva d'errore (ipotesi stringente), in quanto non quantificabile dall'operatore, è stata eseguita una prova per stimare l'errore commesso nel determinare la posizione del punto sulla superficie della sfera, che viene poi proiettato per la stima del centro della sfera, nel caso in cui si vada a selezionare un pixel diverso da quello centrale nel blob.

Per fare ciò sono state costruite 4 immagini di profondità 640x480 costituite da un unico piano posto rispettivamente a 1000, 1500, 2000 e 2500 mm dal sensore. Ogni immagine è stata campionata in 9 punti rappresentativi dell'intera immagine selezionati come in figura seguente:

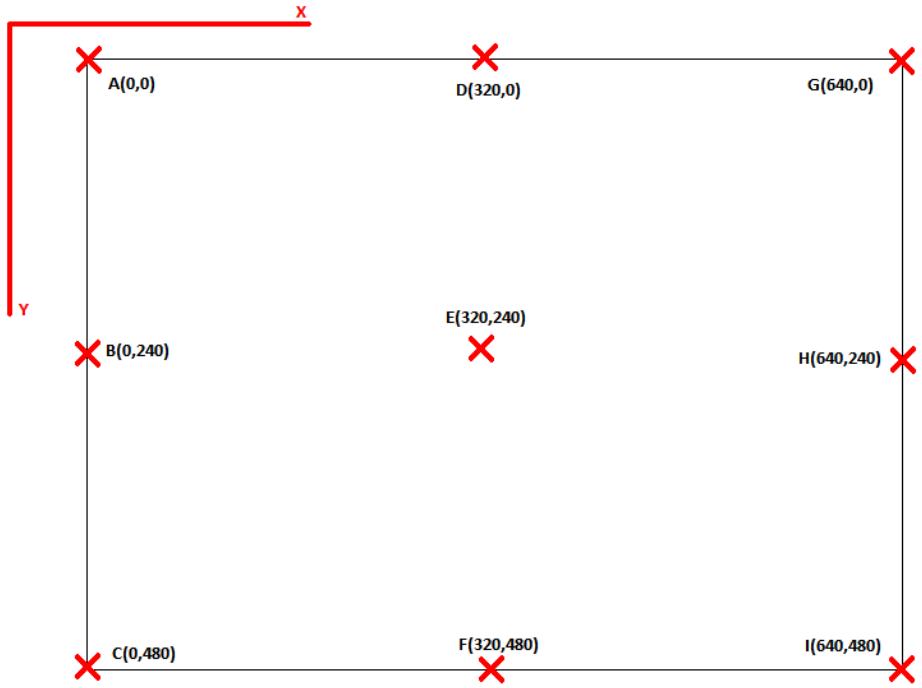


Fig.4 Schema di campionamento delle mappe di profondità per il calcolo dell'errore di ricostruzione

Per ogni punto (A,B,C,D,E,F,G,H,I) e per il suo contiguo prima lungo X e poi lungo Y (per A(0,0)→ (0,1) e (1,0), per I(640,480)→(639,480) e (640,479)) sono stati stimati, attraverso le primitive OpenNI, i rispettivi punti nello spazio 3D. E' stato quindi calcolato l'errore commesso nel selezionare il pixel vicino come distanza tra i punti nello spazio 3D appena definiti.

Con ΔX si intende l'errore in mm commesso nel selezionare un pixel vicino a quello corretto lungo l'asse x mentre ΔY assume lo stesso significato lungo l'asse Y del sensore.

1000 mm									
	A(0,0)	B(0,240)	C(0,480)	D(320,0)	E(320,240)	F(320,480)	G(640,0)	H(640,240)	I(640,480)
ΔX	1.754	1.754	1.754	1.754	1.754	1.754	1.754	1.754	1.754
ΔY	1.753	1.753	1.753	1.753	1.753	1.753	1.753	1.753	1.753

*le misure in tabella sono espresse in mm

1500 mm									
	A(0,0)	B(0,240)	C(0,480)	D(320,0)	E(320,240)	F(320,480)	G(640,0)	H(640,240)	I(640,480)
ΔX	2.630	2.630	2.630	2.630	2.630	2.630	2.630	2.630	2.630
ΔY	2.630	2.630	2.630	2.630	2.630	2.630	2.630	2.630	2.630

*le misure in tabella sono espresse in mm

2000 mm									
	A(0,0)	B(0,240)	C(0,480)	D(320,0)	E(320,240)	F(320,480)	G(640,0)	H(640,240)	I(640,480)
ΔX	3.506	3.506	3.506	3.506	3.506	3.506	3.506	3.506	3.506
ΔY	3.507	3.507	3.507	3.507	3.507	3.507	3.507	3.507	3.507

*le misure in tabella sono espresse in mm

2500 mm									
	A(0,0)	B(0,240)	C(0,480)	D(320,0)	E(320,240)	F(320,480)	G(640,0)	H(640,240)	I(640,480)
ΔX	4.383	4.383	4.383	4.383	4.383	4.383	4.383	4.383	4.383
ΔY	4.343	4.343	4.343	4.343	4.343	4.343	4.343	4.343	4.343

*le misure in tabella sono espresse in mm

Diversamente da quanto si potrebbe ipotizzare, per ogni profondità l'errore commesso è lo stesso lungo la direzione X e la direzione Y ma in particolar modo risulta essere lo stesso a prescindere dalla posizione del punto nel piano immagine.

Ecco dunque che nel caso il pixel selezionato dall'algoritmo automatico sia uno tra i 4-neighboor del pixel effettivamente rappresentante il centro del blob l'errore di stima è inferiore a:

- ✓ nel caso di marker posto a 1000mm: 1,75 mm o lungo X o lungo Y;
- ✓ nel caso di marker posto a 1500mm: 2,63 mm o lungo X o lungo Y;
- ✓ nel caso di marker posto a 2000mm: 3,51 mm o lungo X o lungo Y;
- ✓ nel caso di marker posto a 2500mm: 4,38 mm o lungo X o lungo Y;

mentre nel caso di selezione di un D-neighboor l'errore commesso è inferiore a:

- ✓ nel caso di marker posto a 1000mm: 2,47 mm (= caso peggiore sulla diagonale = (sqrt(2*1.75^2))) nel piano XY del sistema di riferimento del sensore;
- ✓ nel caso di marker posto a 1500mm: 3,71 mm nel piano XY del sistema di riferimento del sensore;
- ✓ nel caso di marker posto a 2000mm: 4,96 mm nel piano XY del sistema di riferimento del sensore;
- ✓ nel caso di marker posto a 2500mm: 6,19 mm nel piano XY del sistema di riferimento del sensore;

L'errore appena considerato rappresenta però l'incertezza sulla stima del punto appartenente alla superficie della sfera e non sulla stima del punto di controllo.

Inoltre è da considerare che con molta probabilità il pixel selezionato, seppur vicino, assumerà un diverso valore rispetto al pixel centrale aumentando l'errore nel cambio da coordinate proiettive a reali rispetto al caso ideale appena considerato.

6.3 Definizione di un visual hull

Alla fine del lavoro di tesi è stato sviluppato un software che, a partire dalle mappe di profondità generate da due sensori *Asus Xtion Pro Live* calibrati in modo da mappare punti del secondo in punti del primo, permette di ricostruire in real time (nel caso specifico lavora a 30fps) la point cloud che rappresenta il paziente inquadrato dalle telecamere.

Si tratta di un semplice programma che permette di ottenere dei risultati grafici qualitativi che però fanno intendere la concreta possibilità di adottare tale strumentazione per la generazione di un visual hull o per la definizione di un modello anatomico da poter utilizzare in una successiva analisi markerless.

In particolare, per ogni mappa di profondità generata dai due sensori, mediante un semplice algoritmo di *background subtraction*, vengono estratti tutti e soli quei pixel che riportano la proiezione della silhouette del soggetto inquadrato. Per ognuno di tali punti, mediante le librerie *OpenNI* viene ricostruito il corrispondente punto 3D dello spazio, il quale viene disegnato in uno spazio virtuale definito in *OpenGL*.

Nel caso dei punti provenienti dal secondo sensore, prima di procedere alla loro rappresentazione, si effettua una trasformazione di coordinate mediante i parametri calcolati in fase di calibrazione, in modo da esprimere le due *point clouds* secondo lo stesso sistema di riferimento.

A scopo d'esempio sono riportati in seguito alcuni screenshot dei risultati che si ottengono dalla fusione delle informazioni ricavate separatamente dai due sensori. In particolare in verde è presentata la *point cloud* costruita sulla base delle informazioni fornite dal primo sensore e in rosso quella costruita a partire dal secondo.

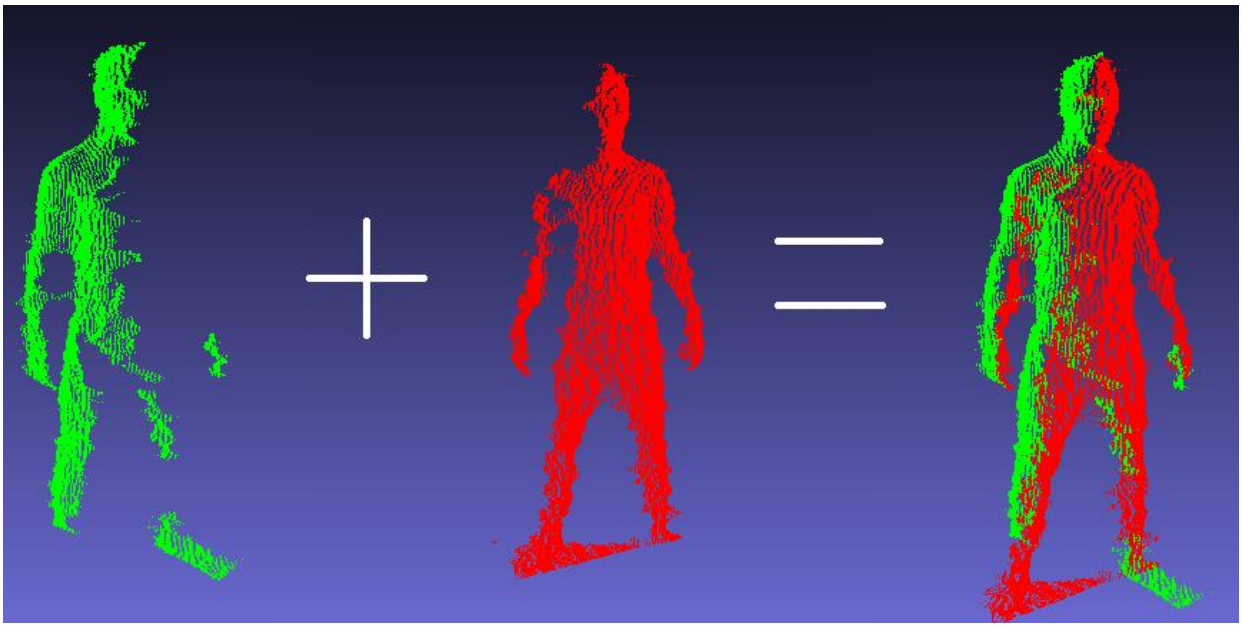


Fig. 5 In verde è rappresentata la point cloud generata dal sensore 1 mentre in rosso quella ricostruibile dal sensore 2 previa rototraslazione delle coordinate dei punti che la compongono. A destra fusione delle due nuvole.

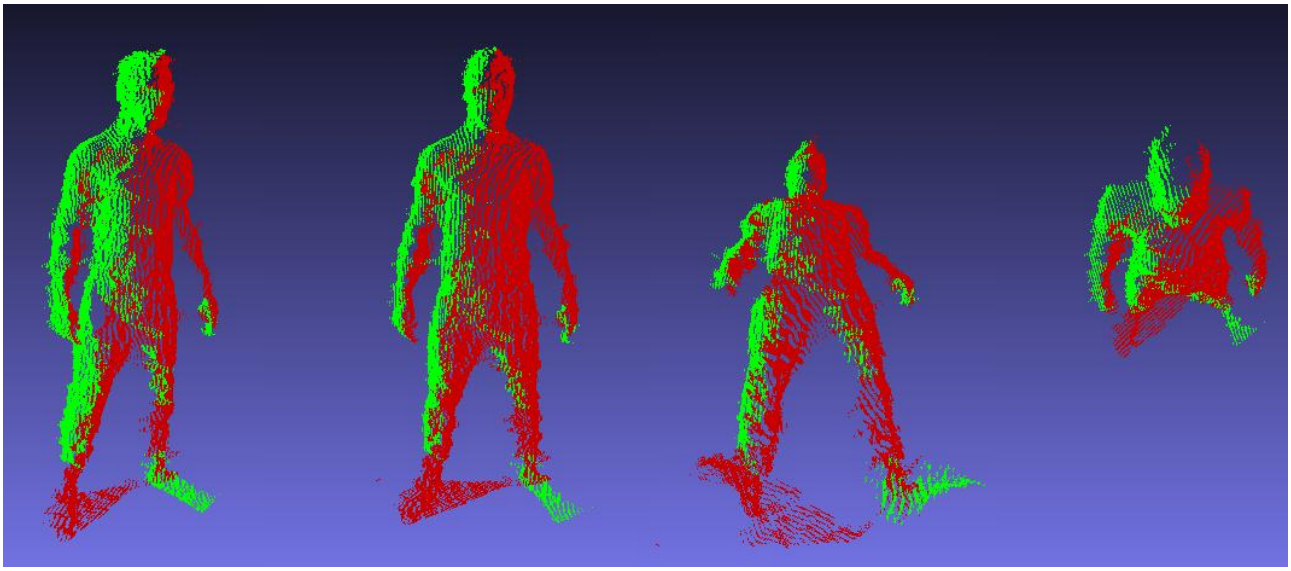


Fig. 6 Screenshot da diverse angolature dei risultati ottenuti dalla fusione delle point clouds generate dai due sensori di un sistema stereo D-RGB

Dall'analisi visiva dei risultati si nota come le due nuvole si fondano correttamente ad indicare una buona calibrazione del sistema.
Dalla nuvola totale è possibile quindi risalire alla mesh (in questo caso il processo è stato effettuato appoggiandosi al software *meshlab* [43], ma può essere effettuato mediante l'utilizzo di particolari librerie per l'elaborazione delle *point cloud* come ad esempio e P*CL: Point Cloud Library* [42]) che descrive il volume così ricostruito e che può quindi essere fornita in input ai vari algoritmi di analisi *markerless* per il matching col modello cinematico.
Un esempio delle mesh ottenibili è rappresentata in figura seguente.

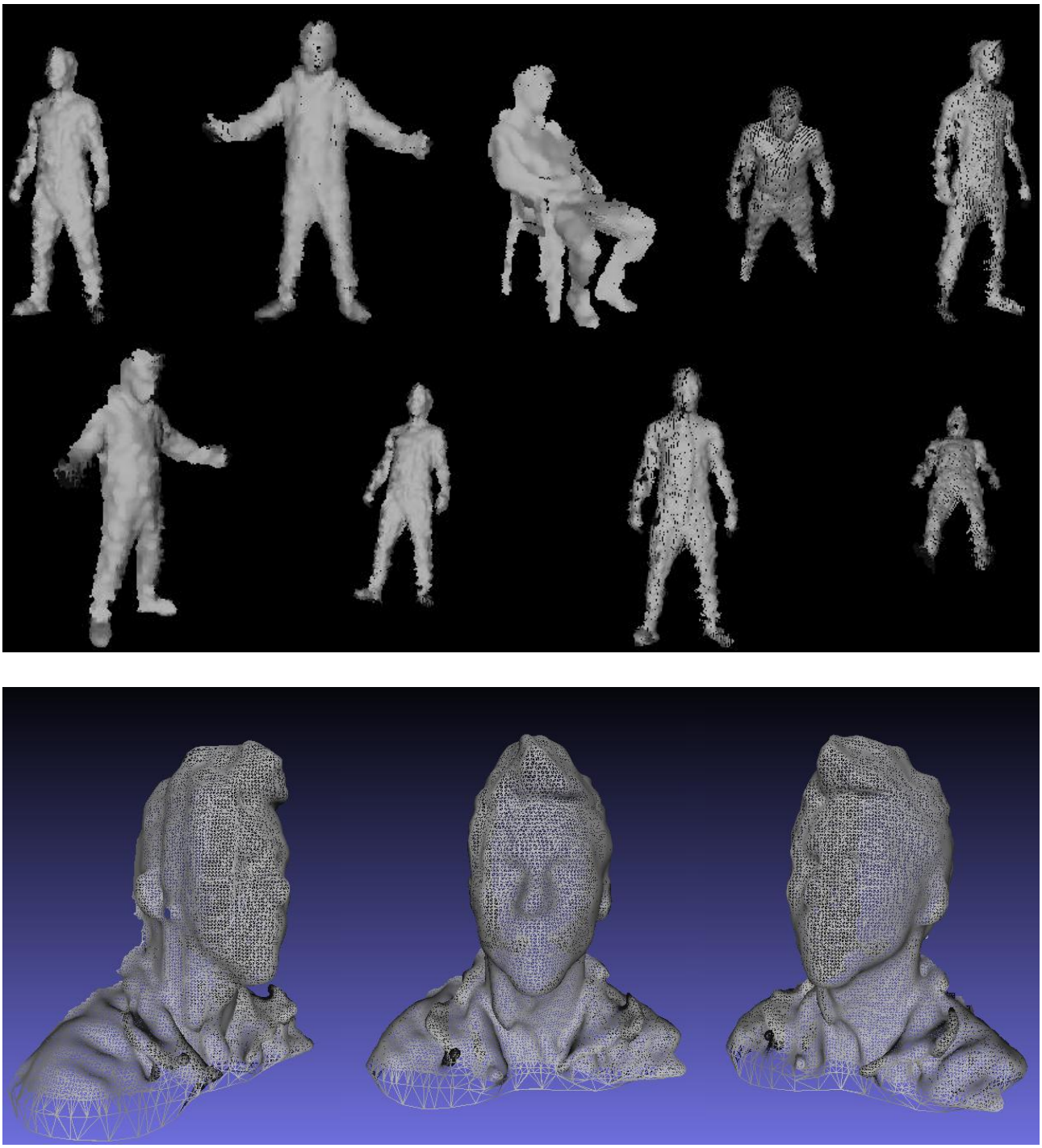


Fig. 7 Alcuni esempi di mesh ottenibili a a partire dalle point clouds ricostruite dai diversi sensori.

Visivamente, i risultati ottenuti appaiono molto buoni ed è sicuramente interessante approfondire l'utilizzo di tali dispositivi in campo di analisi *markerless* in modo particolare per la semplicità con cui permettono di ottenere i risultati grafici riportati in precedenza.
È da sottolineare infatti il basso costo computazionale richiesto per la generazione della point cloud del paziente, fattore questo che rappresenta un grosso vantaggio sulla possibile adozione di tale tecnica nel campo della *motion capture*.

6.4 Discussione

Dai test effettuati e dall'analisi dei dati raccolti è possibile notare come l'errore di calibrazione medio stimato sull'interdistanza tra due punti noti risulta essere inferiore all'incertezza indicata dalla casa produttrice *Asus* sulla ricostruzione della *depth map*. Questo fatto fa intuire la bontà del metodo sviluppato e la precisione dei *tools* utilizzati nella calibrazione e in fase di test.

Allo stesso modo, le differenze minime tra i valori dei parametri di calibrazione calcolati nelle quattro serie di misure indicano come il processo di calibrazione sia robusto nonostante l'errore introdotto e dall'algoritmo automatico di segmentazione e dall'errore strumentale sicuramente presente nella misura.

I risultati mostrano quindi come, l'utilizzo contemporaneo di più dispositivi permetta una diminuzione del la varianza sulla stima dell'interdistanza (che resta comunque ben maggiore di quella del sistema stereo fotogrammetrico) e in particolare riduca il valore di *RMSD* intra-serie e inter-serie ad indicare una maggiore ripetibilità della misura.

In sostanza si può affermare che, l'utilizzo di dispositivi multipli non permette solo di risolvere i problemi legati all'occlusione ma consente anche di migliorare l'accuratezza con cui si riesce a risalire alla posizione 3D degli oggetti nella scena.

Occorre però portare all'attenzione del lettore alcuni fattori che possono altresì limitare l'utilizzo di questi sistemi.

Innanzitutto occorre considerare il volume di spazio che si riesce ad elaborare correttamente e che rappresenta quindi l'area effettiva di lavoro del sistema.

Considerando la limitata apertura dell'obiettivo (HxVxD: *58°x45°x70°*) sommata alla distanza massima di corretto funzionamento pari a 3500 mm, lo spazio di lavoro rappresentato dall'intersezione di tutti i coni uscenti da ogni dispositivo, si riduce ad un volume pressoché cilindrico con base di diametro non superiore ai 120-130 cm ed un'altezza inferiore ai 2 metri.

Questo spazio ridotto si presta all'analisi solo di un numero limitato di movimenti: basti pensare al caso della posturografia o dell'analisi di salto sul posto.

Occorre inoltre considerare l'insorgere di fenomeni di interferenza tra i pattern proiettati dai diversi dispositivi che, sebbene possano essere limitati mediante l'adozione di alcuni accorgimenti riportati in appendice A, limitano il numero di sensori stessi utilizzabili in contemporanea.

Un ulteriore aspetto da considerare è il problema della sincronizzazione tra i vari dispositivi. Nel software sviluppato questo ostacolo è stato superato mediante l'adozione di un timer che, a cadenze regolari fornisce un input alle camere le quali acquisiscono le immagini.

Questa tecnica necessariamente riduce il frame rate massimo di acquisizione perciò nel caso si voglia adottare un sistema di più sensori da far lavorare in free run occorre quantificare gli eventuali ritardi introdotti dall'elaborazione interna nei differenti dispositivi.

Infine è utile riportare un ultimo elemento che rappresenta sicuramente un ostacolo nell'utilizzo di sensori D-RGB multipli ossia il problema di far riconoscere allo stesso PC più sensori. Questo è dovuto al fatto che ogni dispositivo deve essere collegato ad una porta USB che faccia capo a *HUBs* (ovvero un dispositivo di rete che funge da nodo di smistamento dati di una rete di comunicazione) indipendenti. Nel caso specifico il problema è stato risolto mediante l'utilizzo di una scheda *PCMCIA* in grado di aggiungere alle porte USB già presenti nel PC altre due ulteriori porte processate da un chip esterno indipendente.

Nonostante questa serie di ostacoli, la definizione di un sistema composto da sensori multipli D-RGB può porre le basi per lo sviluppo di tutta una serie di applicazioni nei più svariati campi: dalla robotica alla riabilitazione.

Ricollegandosi ad esempio al sistema "*Kinestesia*" introdotto in capitolo 3, l'utilizzo di una mappa maggiorata costruita sulla base delle informazioni provenienti da più sensori *Microsoft Kinect*, può comportare un netto miglioramento delle prestazioni fornendo al medico risultati più precisi e accurati.

Allo stesso modo, tutte le applicazioni che prevedono oggi l'utilizzo di un singolo dispositivo possono beneficiare e essere potenziate dall'azione simultanea di più sensori.

In questo senso anche l'utilizzo nel progetto Nirvana di un sensore posto posteriormente al paziente e due posizionati a parete secondo quanto già proposto, può consentire di ottenere un ottimo *rendering* del paziente permettendo la sua localizzazione e analisi.

Per quanto riguarda la possibilità di costruire un *visual hull* a partire da un sistema stereo di sensori D-RGB, i risultati mostrano, seppur qualitativamente, come, in seguito alla calibrazione, le nuvole di punti provenienti dai diversi sensori si fondano accuratamente ad indicare la concreta possibilità di adottare tale strategia in un sistema *markerless di MoCap*.

Capitolo 7

Conclusioni e sviluppi futuri

7.1 Conclusioni

Il lavoro di tesi, svolto in collaborazione con la sezione ricerca e sviluppo dell'azienda BTS di Padova [41] è stato interamente incentrato sullo studio di dispositivi D-RGB in grado di generare mappe di profondità.

In particolare si è analizzata la possibilità di migliorare le prestazioni di tali dispositivi, in particolare del prodotto *Xtion Pro Live* [32] sviluppato da *Asus*, e di migliorare l'accuratezza dei dati ottenibili mediante l'utilizzo simultaneo di dispositivi multipli presenti nell'area di lavoro.

Di qui la necessità di sviluppare un metodo di calibrazione che permettesse di integrare e confrontare i dati provenienti dai diversi sensori.

Il metodo sviluppato consiste nella stima dei parametri estrinseci di rototraslazione in grado di trasformare punti visti da un dispositivo in punti espressi secondo il sistema di riferimento di un differente sensore considerato primario.

Il metodo si basa sull'algoritmo di Horn [17] e permette il calcolo, mediante una formulazione in forma chiusa, dei migliori parametri sulla base di un data set composto da N>600 punti di controllo riconoscibili da entrambi i sensori in esame.

Il software sviluppato permette una calibrazione rapida (il tempo impiegato dal processo dipende unicamente dalla durata delle acquisizioni per costruire la nuvola di punti sulla base di cui stimare i parametri) e completamente automatica del sistema interfacciandosi all'utente attraverso un'interfaccia *user-frinedly* estremamente intuitiva. Il metodo, testato nel caso di un sistema stereo

composto da due dispositivi, è trasportabile al caso di un numero *N* generico di sensori permettendo quindi la calibrazione di un completo sistema stereo in grado di ricostruire da tutte le angolazioni la scena inquadrata.

I test svolti a validazione del metodo, in modo da replicare la prova di *spot-checks* per la misura di distanza tra marcatori, hanno permesso il confronto delle prestazioni di tale sistema con il sistema stereo fotogrammetrico SMART-DX/100 sviluppato da BTS, indicando la robustezza del metodo stesso e la possibilità di aumentare l'accuratezza sulla ricostruzione 3D della scena sulla base di informazioni multiple provenienti da diversi sensori, oltre ovviamente alla risoluzione di quei problemi di occlusione che incorrono nel lavorare con un'unica sorgente di segnale.

Il lavoro svolto intende quindi inserirsi in un *background* estremamente ampio ponendo delle basi per una moltitudine di applicazioni nei più svariati campi.

Tutte quelle applicazioni che prevedono oggi l'utilizzo di un singolo sensore di profondità possono beneficiare di questa tecnica e migliorare le proprie prestazioni nell'adottare un numero multiplo di sensori.

Già dalle specifiche fornite da *Asus* era noto come le precisioni ottenibili lavorando con questo tipo di dispositivi non potessero essere comparabili a quelle di un sistema stereofotogrammetrico oggi utilizzato nei laboratori di analisi del movimento.

Nonostante ciò, tale tecnologia, grazie anche alla limitata onerosità computazionale del sistema, può essere sicuramente adottata in tutte quelle applicazioni ove non sia richiesto un grado di accuratezza dell'ordine di frazioni di millimetri (vedi il suo impiego nel progetto Nirvana) rappresentando una soluzione economica, robusta ed elegante.

7.2 Sviluppi futuri

Relativamente al software sviluppato, si tratta di una prima versione che può essere oggetto di numerosi perfezionamenti e migliorie. Si rende necessario infatti un maggiore approfondimento riguardo la possibilità di stimare la posizione dei punti di controllo attraverso una fase di *matching* 3D con una sfera, metodo questo che in tutta probabilità garantirebbe risultati più accurati e affidabili. Purtroppo bisogna di continuo confrontarsi con l'errore strumentale con cui il sensore ricostruisce la profondità della scena che influisce significativamente sull'effettiva buona riuscita del metodo.

Allo stato attuale il limite principale che presenta il metodo è dovuto al fatto che il processo di segmentazione delle immagini, basandosi su un una prima fase di *background subtraction*, costringe l'operatore a muovere il *tool* di calibrazione a tre sfere senza entrare col proprio corpo nell'area di lavoro.

Questo rappresenta una "scomodità" dal punto di vista operativo che potrebbe essere risolta nel caso si riuscisse a definire uno stratagemma che, come il nastro a microsfere di vetro per la

stereofotogrammetria, permetta di identificare con facilità (mediante semplice sogliatura) particolari oggetti inquadrati dal sensore senza però impedire il calcolo della mappa di profondità.

In questo modo il processo di calibrazione risulterebbe operativamente semplificato ed ulteriormente velocizzato.

In riferimento alle possibili applicazioni del metodo sviluppato, l'utilizzo di un numero multiplo di sensori e la possibilità quindi di ricostruire una *mesh poligonale* del paziente inquadrato può porre le basi per una gran varietà di applicativi.

Per quanto riguarda l'analisi del movimento a causa dello spazio di lavoro limitato che si riesce ad elaborare, tale soluzione può essere usata in particolar modo in posturografia.

Nell'analisi posturografica la variabile che viene misurata è la posizione nel tempo del centro di massa corporeo (CoM) e il suo movimento nelle direzioni antero- posteriore(AP) e medio-laterale (ML). La misura diretta della traiettoria del CoM è assai laboriosa in quanto richiede la misura del CoM di ogni segmento corporeo per poi sommare il loro contributo pesato nel calcolo del CoM globale. Oggi, l'alternativa più diffusa consiste nel misurare il centro di pressione (CoP) per mezzo di una pedana di forza e quindi eseguire una doppia integrazione per ottenere la posizione del CoM.

La possibilità di ottenere mediante sistemi D-RGB una *mesh* completa del paziente introduce una nuova possibilità di fare posturografia. Come introdotto da Corazza e Andriacchi in [13] ipotizzando una densità costante del corpo del paziente, si rende possibile una misura diretta del movimento del CoM come calcolo del centro di massa del volume descritto dalla *mesh*.

Conoscendo infatti l'esatta posizione e dimensione del volume occupato in ogni istante dal corpo del paziente è possibile risalire alle coordinate del baricentro del volume stesso ottenendo una stima della posizione del CoM.

D'altro canto come già introdotto nel capitolo precedente, l'utilizzo di un sistema stereo di sensori D-RGB può consentire la definizione di un modello anatomico specifico del soggetto che può essere quindi *matchato* con il modello cinematico definito a priori e rappresentare quelle informazioni per l'interpretazione dei dati nel caso di analisi *markerless*.

I dispositivi D-RGB ad oggi sviluppati presentano tuttavia numerosi limiti al loro utilizzo in biomeccanica ma *Microsoft,* in particolare, ha già annunciato l'uscita alla fine di quest'anno di *Kinect* II.
Le caratteristiche annunciate dagli sviluppatori, possono migliorare i dati ottenibili sulla profondità della scena e potenzialmente rappresentare una soluzione a quei problemi che oggi limitano l'utilizzo di tale tecnologia.

Sono in seguito riportate le caratteristiche annunciate da *Microsoft.*

- Camera RGB: *stream* di qualità migliorata.
- Maggiore profondità di *stream* in modo da permettere la registrazione di oggetti più piccoli

- Maggiore precisione nello *stream* in modo da aumentare la precisione nella distinzione degli oggetti, soprattutto calcolando anche bordi e curvature
- *Stream* attivo ad infrarossi che consente un calcolo indipendente delle luci
- Maggiore frequenza di lavoro fino a 100 fps
- Riconoscimento differenziato di mani aperte o chiuse
- Riconoscimento di ulteriori articolazioni e rotazioni
- *Tracking* di articolazioni occluse
- Riconoscimento delle posizioni delle articolazioni

Sono certo che l'enorme impulso provocato dall'uscita in commercio di questi dispositivi sullo sviluppo di nuove applicazioni porterà le maggiori compagnie del settore a intensificare lo studio di questa tecnologia portando allo sviluppo in un tempo non così remoto di strumenti più precisi e potenti che rappresenteranno la soluzione a numerosi problemi oggi incontrati.

Appendice A

Generare mappe di profondità

La funzione base delle librerie *OpenNI* è quella di generare una mappa di profondità come quella illustrata in figura . Per farlo basta inizializzare un oggetto *Context* che useremo per creare e leggere i dati di profondità.

Fig. 1 Esempio di mappa di profondità.

La prima cosa da sapere è che *OpenNI* ha un meccanismo di controllo delle funzioni che ritorna un codice di errore di tipo *XnStatus* se si è verificato un problema. Possiamo creare una variabile di questo tipo ed assegnargli il valore **XN_STATUS_OK**: questo è l'unico valore per indicare che non ci sono stati problemi. Possiamo usare questa variabile per controllare i valori restituiti dalle varie funzioni e verificare che l'operazione sia andata a buon fine. Volendo una descrizione dell'errore si può usare la funzione **xnGetStatusString()**.
Si può così creare un controllo degli errori che useremo ogni qual volta ci sarà la creazione o l'inizializzazione di un oggetto. Se l'errore si verifica verrà stampato un messaggio a video e verrà chiusa l'applicazione.

A questo punto è possibile procedere alla generazione della mappa di profondità: il procedimento non è molto complesso e l'uso delle librerie *OpenNI* permette di ottenere il risultato in poche righe di codice all'interno dello stesso *main*.
La prima cosa da fare è creare un oggetto *Context* ed un oggetto generatore di profondità, inizializzarli e verificare che non ci siano stati errori. Un esempio di codice che permette di generare l'immagine in questione è riportato in appendice C.

Sarà così possibile iniziare a generare i dati acquisiti dal sensore che si sta utilizzando. La funzione che si utilizza è **StartGeneratingAll**(); anche questa come tutte le funzioni genera un messaggio di errore che dovrà esser controllato.
OpenNI ha messo a disposizione un oggetto **DepthMetaData** di supporto ai nodi generatori che creano mappe di profondità.
Quando generiamo la mappa di profondità useremo il metodo **Data()** di questo oggetto, che restituisce un puntatore ai dati di profondità veri e propri.

La mappa verrà generata sui dati forniti dal sensore ad ogni instante, per questo si è soliti adottare un ciclo *while* che userà al suo interno l'oggetto *Context* per richiedere nuovi dati facendo uso del metodo **WaitOneUpdateAll().** Nel momento in cui questo metodo verrà invocato, sarà inviata la richiesta di aggiornamento a tutti i nodi generatori del *Context* in questione.

Esempio di applicazione: Tracking del corpo umano

Per lo sviluppo della tesi ci si è focalizzati in particolari sulla possibilità, offerte dalle librerie *OpenNI* e *NITE* di effettuare una sorta di *tracking* del soggetto in modo da individuarne in ogni istante la posizione e la posa.

La combinazione *OpenNI-NITE* si basa sull'utilizzo delle librerie *OpenGL* per visualizzare lo scheletro dell'utente. Con la prima versione dei driver forniti, il tracciamento dello scheletro costringeva l'utente ad assumere una posa "a cactus" di calibrazione (vedi Figura 2), ovvero l'utente doveva rimanere in posizione eretta, allargare le braccia e posizionare gli avambracci con un angolo di novanta gradi circa rispetto alle braccia. Con l'ultima *release* delle *OpenNI* non è più necessaria alcuna posa base per far sì che l'algoritmo di calibrazione riconosca il soggetto in fronte alla telecamera.

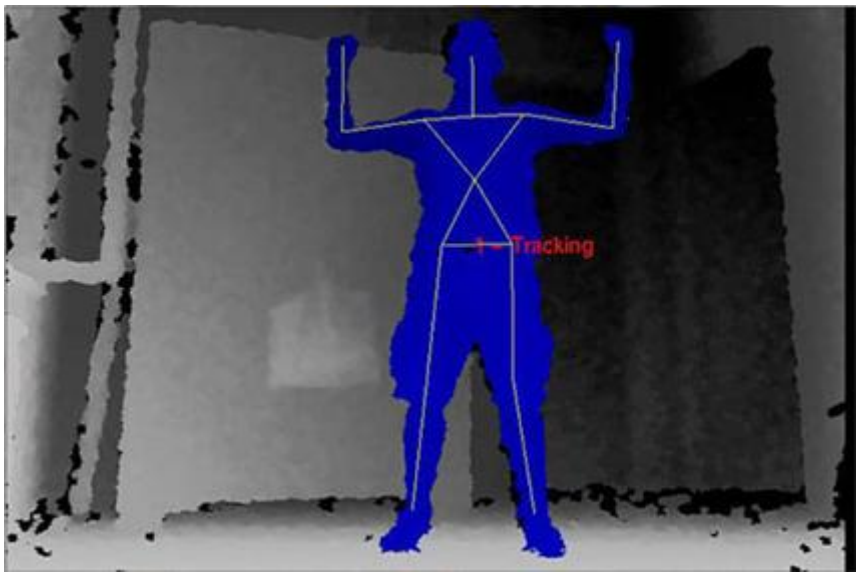


Fig. 2 Esempio di skeleton tracking

Per far ciò l'intero sistema segue delle linee guida riguardanti la conformazione generale del corpo. Questo permette in fase di calibrazione di non confondere gli oggetti appartenenti al background con le persone.

La prima cosa che la periferica fa è quindi creare una mappa di profondità della scena separando l'utente dagli oggetti inanimati e distinguere dal resto le forme umane con colori diversi (blu, verde, rosso, e così via) rispetto al background (in scala di grigi). A questo punto si deve assegnare un ID all'utente; questo ID è persistente e permetterà di riconoscere diversi utenti presenti nella scena. Il processo di assegnazione dell'ID viene chiamato segmentazione degli utenti e restituisce una mappa delle etichette assegnate.

L'algoritmo andrà ad utilizzare questa mappa per generare uno scheletro per ogni diversi ID. Ogni possibile errore nella segmentazione dell'utente potrà influire sul *tracking* del soggetto.
I fattori che possono dare problemi alla fase di segmentazione ed alla fase di calibrazione sono:

- Utenti troppo vicini che si toccano tra di loro.
- Oggetti (come sedie o tavoli) troppo vicini che non permettono il corretto riconoscimento del corpo.
- Utente in movimento che non mantiene la posa di calibrazione.
- Spostamento del sensore mentre la fase di segmentazione è attiva.

La figura 3 rappresenta uno dei possibili problemi appena descritti, l'utente nella scena tenendo la mano sullo schienale della sedia ha portato il processo di segmentazione ad assegnare il proprio ID anche ai pixel che rappresentano la sedia. Il problema in questo caso non è permanente: non bisognerà riavviare l'applicazione ma semplicemente togliere la sedia dal campo visivo e verrà

ricreata una nuova mappa. Il colore blu è un *feedback* visivo per l'utente per capire che quei pixel della mappa di profondità appartengono a lui.

Fig. 3 Risultato della fase di segmentazione

Nel momento in cui sono presenti più utenti in scena può succedere che uno impedisca al sensore di monitorare gli altri. Ad esempio se due si mettono in fila indiana, l'ID del primo viene perso e successivamente dovrà essere riassegnato, oppure i due ID potrebbero anche essere scambiati. Inoltre se un utente esce dalla scena per più di 10 secondi il suo ID viene cancellato e potrà essere assegnato ad un altro utente in un secondo momento, in quanto gli ID sono riciclabili.
Dopo che un utente è stato individuato segue la fase di calibrazione, necessaria per poter monitorare in un secondo momento lo scheletro.
L'algoritmo interno al dispositivo che permette di determinare la posizione delle articolazioni principali a partire dalla *depth map* è stato brevettato da *PrimeSense* per quanto riguarda le *OpenNI* e da casa *Microsoft* per le *SDK* di *Kinect*. Ovviamente il codice non è disponibile agli utenti ma in [21] è possibile ottenere informazioni sulle linee guida utilizzate per il *tracking*.
Una volta che la calibrazione è terminata si avrà un completo monitoraggio delle principali articolazioni dello scheletro; in particolare saranno monitorati: testa, collo, spalla destra e sinistra, gomito destro e sinistro, torso, anca destra e sinistra, ginocchio destro e sinistro ed infine piede destro e sinistro.

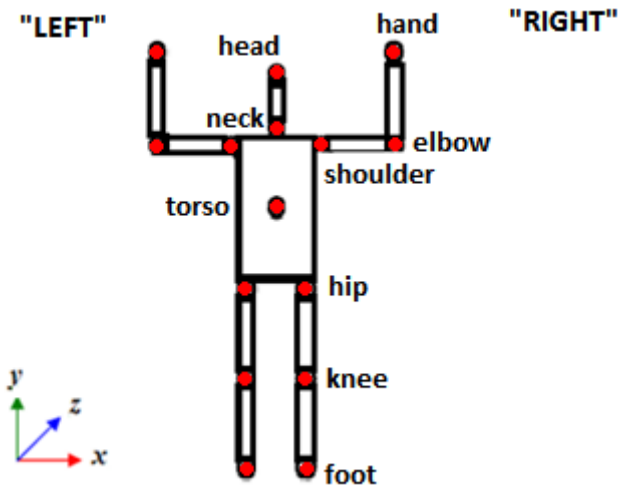


Fig. 4 testa; 2. collo; 3. busto; 4. spalla sinistra; 5. gomito sinistro; 6. mano sinistra; 7. spalla destra; 8. Gomito destra; 9. mano destra; 10. anca sinistra; 11. ginocchio sinistro; 12. piede sinistro; 13. anca destra; 14. ginocchio destro; 15. piede destro.

Le coordinate fornite dal sensore sono espresse secondo una terna cartesiana avente origine nel sensore stesso e disposta secondo quanto riportato in figura seguente.

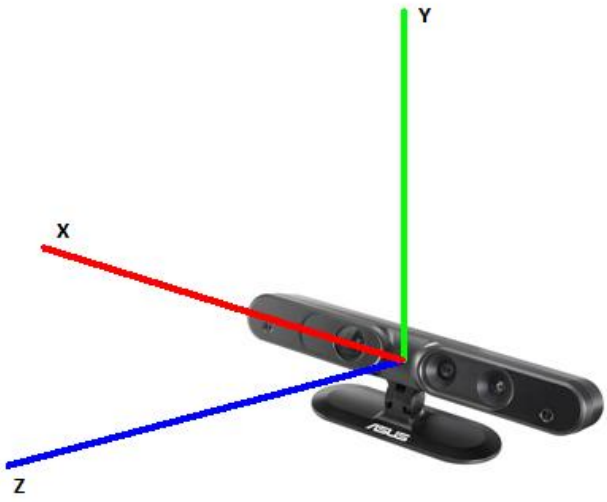


Fig. 5 Sistema di riferimento dei dati forniti dal dispositivo

E' interessante notare come i dati in uscita siano espressi secondo un sistema cartesiano definito in convenzione destrogira.
Gli orientamenti sono memorizzati in una matrice di rotazione 3X3 (ortonormale). Le tre colonne rappresentano rispettivamente la direzione dell'asse X, Y e Z. Durante la posa T (utente in piedi di fronte alla telecamera con braccia allargate) il nostro orientamento verrà rappresentato come una matrice identità, in quanto l'orientamento di ogni articolazione è allineato con il sistema di coordinate.

La successiva fase di *tracking* è ancora lenta, e può quindi capitare che se l'utente effettua movimenti relativamente rapidi, la risposta dello scheletro a monitor sarà soggetta ad un ritardo.

L'intero sistema lavora infatti a 30 fps (con una risoluzione di 640x480, è possibile lavorare fino ad una velocità di 60 fps ma la risoluzione della *depth image* pari a 320x240 risulta essere troppo scadente per qualsiasi tentativo di elaborazione) e contiene un database di 200 pose comuni per lo scheletro precaricate. Nel caso l'utente faccia un movimento che impedisca alla telecamera di riconoscerne la posa, l'algoritmo userà una delle pose tra quelle presenti in memoria che più si adatta al caso per non perdere il tracciamento dell'utente.
Va sottolineato infine che i dati forniti dal tracciamento dello scheletro oltre ad essere utilizzati in tempo reale in diverse applicazioni e giochi, possono essere registrati in un formato proprietario .ONI o con appropriate modifiche sul codice sorgente della demo *NITE* esportate in formato .bvh16.

Utilizzo di più sensori D-RGB: tecnica snake'n' sense

Il numero di sensori D-RGB che inquadrano la stessa scena è limitato dal fatto di dover adottare una conformazione tale per cui l'angolo tra i raggi ottici uscenti dal proiettore montato su ognuno di essi sia maggiore di un angolo retto. Questo fa intuire come il numero massimo utilizzabile sia pari a 4 e quindi disposti ai vertici di un quadrato che circoscrive il volume di acquisizione.

Tutto ciò è dovuto alle interferenze che sorgono quando pattern diversi proiettati da diversi dispositivi impattano sulla stessa scena introducendo una serie di distorsioni ed errori nel calcolo delle singole *depth maps.*

In [27,28] viene presentata una nuova tecnica in grado di mitigare l'interferenza causata dall'uso contemporaneo di più dispositivi D-RGB. La tecnica è particolarmente utile per quei strumenti, quali *Micrsoft Kinect* o *Asus Xtion Pro Live,* in cui la fonte di luce strutturata non è modulata. Si tratta di una soluzione puramente meccanica che non richiede la modifica dell'elettronica interna, del *firmware* o del software associato al dispositivo. Questo metodo non altera altresì il frame rate della fotocamera che rappresenta un problema in approcci che richiedono la modulazione della sorgente di luce strutturata e non influisce sui valori di profondità o sulla geometria della scena.

Il concetto chiave che sta alla base di questo metodo definito *Shake 'n'Sense* è quello di far vibrare minimamente il dispositivo utilizzando un carico eccentrico applicato ad un motore posto sulla "cassa" del sensore stesso in modo da introdurre artificialmente un effetto *blurring* sul pattern proiettato nella scena.

In questo modo sia l'illuminatore che la telecamera IR del dispositivo si muovono allo stesso modo,non creando alcun disturbo per il calcolo della profondità, ma semplicemente introducendo una lieve sfocatura nella mappa generata. Tuttavia, il movimento quasi impercettibile del sensore provoca la sfocatura dei pattern strutturati proiettati dalle altre unità che permette quindi di eliminare la maggior parte della diafonia.

A validazione del metodo sono stati effettuati svariati test in cui due dispositivi, posti parallelamente di fronte ad una parete (condizione di *worst case* per l'utilizzo contemporaneo di più sensori), una volta messo in vibrazione uno dei due ad ampiezze e frequenze variabili, si sono valutate la bontà delle mappe di profondità ottenute.

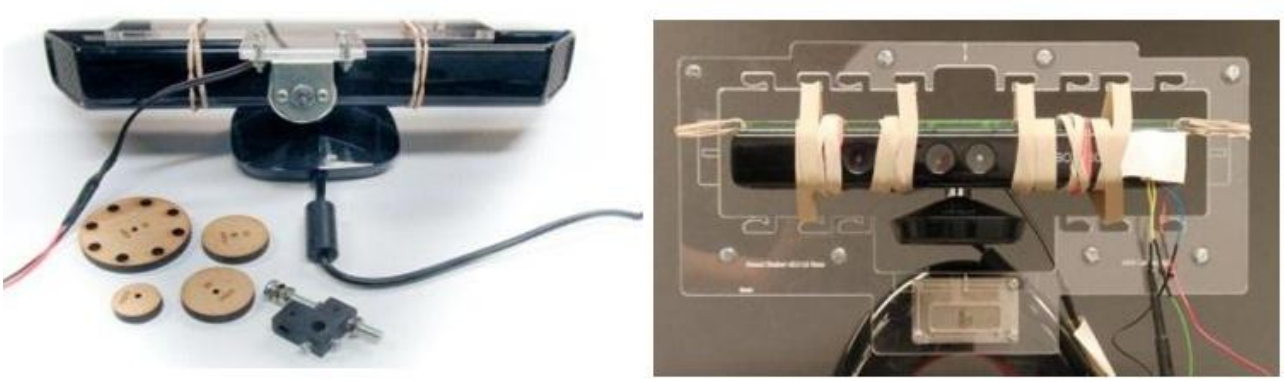


Fig. 6 A sinistra: il nostro setup Shake'n'Sense semplice per Ki-nect permette una consuetudine montato posteriormente offset peso vibrazioni del motore per indurre vibrazioni in tutto il gruppo. Destra: il gruppo telecamera è montata in un telaio acrilico usando elastici consentano di vibrare liberamente. Un ac-celerometer viene incollata alla unità (a destra) per consentire la frequenza di vibrazione da misurare. [28]

La frequenza della vibrazione è stata fatta variare da 15Hz a 120Hz con incrementi di 10Hz. Sono stati acquisiti 150 fotogrammi per ogni frequenza. La figura mostra i risultati di una vibrazione appena superiore ai 40Hz, anche se i test hanno indicato che la frequenza ottimale in quella particolare configurazione è compresa tra 60 Hz e 80Hz.

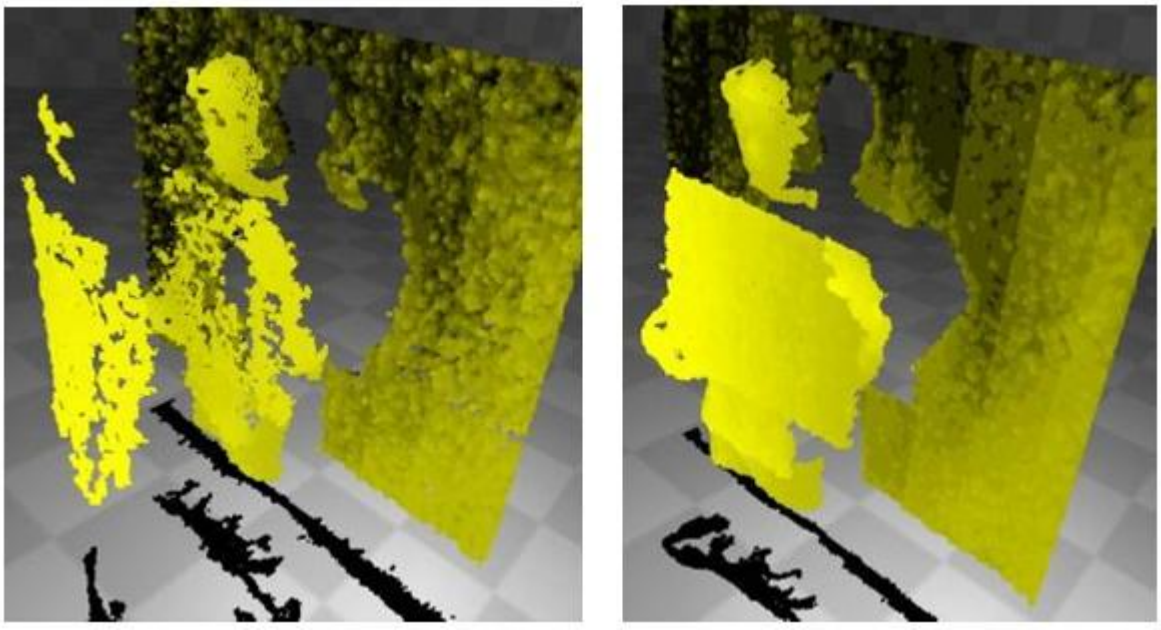


Fig. 7 Point-cloud renderings of a person standing in front of a wall holding a sheet of card. Left: significant error in depth due to cross-talk including depth values being hal-lucinated. Right: our method applied.

Gli stessi ricercatori hanno poi semplificato l'apparecchiatura utilizzata in modo da rendere il sistema facilmente implementabile da chiunque voglia confrontarsi con l'utilizzo simultaneo di più sensori.

In particolare, l'utilizzo di un semplice motorino presente nei più comuni giochi elettronici al quale viene applicato un carico eccentrico, permette di ottenere ottimi risultati per quanto riguarda la riduzione dell'interferenza tra i dispositivi.

Appendice B

Sistema stereo fotogrammetrico SMART-DX/100

Le diverse versioni della serie SMART DX si basano su telecamere digitali di nuova concezione che utilizzano sensori a elevata sensibilità e illuminatori dal design innovativo e funzionale, la cui alta potenza di irraggiamento, combinata all'alta risoluzione della telecamera (fino a 4 Megapixel), aumenta il volume di lavoro e consente la cattura di movimenti rapidi e impercettibili. Il sistema integra, sincronizza e gestisce in tempo reale tutte le informazioni cinematiche, cinetiche, elettromiografiche e video provenienti da dispositivi collegati quali piattaforme di forza, elettromiografi, *treadmill* sensorizzati, ecc.

Specifiche tecniche

Telecamere digitali a raggi infrarossi	Fino a 4 per singola workstation
Possibilità di collegare più workstation	Si
Risoluzione sensore	640x480
Frequenza di acquisizione alla massima risoluzione	100 Hz
Frequenza di acquisizione massima	200 Hz / 1000 Hz
Accuratezza	<0,2 mm su volume 4x3x3 m
Prepocessing	-
Preview	Full frame
Lunghezza d'onda LED emettitore	850nm
Numero marker gestiti	Illimitato
Trasmissione dati	Gigabit Ethernet

Alimentazione	Diretta da data station
Ottiche	Intercambiabili C-mount
Ottica a focale fissa	4,5-8 mm
Ottica zoom	6-12/25 mm

Compomenti del sistema SMART-SUITE

Smart Capture

Questo software permette la gestione completa e *real time* della calibrazione e dell'acquisizione dei dati. All'avvio del programma vengono eseguite una serie di diagnostiche di sistema affinché tutte le telecamere siano correttamente impostate e collegate alla *workstation*.
Una volta avviato si possono visualizzare in *real time* sia i segnali analogici acquisiti che le immagini video delle singole telecamere. Tale visione permette di gestire la sensibilità delle telecamere che acquisiscono immagini IR a 256 livelli di grigio. Prima di ogni serie di acquisizioni, o comunque obbligatoriamente ogni volta che vengono cambiati i parametri di acquisizione (cambio di posizione delle videocamere, cambio di ottiche,eliminazione o aggiunta di videocamere), è indispensabile eseguire la procedura di calibrazione.

Smart Tracker
È un ambiente grafico che permette la ricostruzione tridimensionale dei dati utilizzando i dati bidimensionali acquisiti dalle videocamere e quelli provenienti dalla calibrazione. Ad ogni *marker* viene assegnata una traiettoria (*tracking*) e successivamente è possibile assegnare alla traiettoria un nome (*labelling*).
Il *tracking* ricostruisce la traiettoria del *marker* collegando la posizione dello stesso in due frame successivi sulla base di algoritmi che usano stimatori ricorsivi dello stato dei sistemi dinamici (come il filtro di *Kalman*).
Questo software è stato utilizzato nella fase di test col sistema SMART-DX/100 permettendo di ricavare istante per istante le coordinate occupate dai due *markers* e di applicarvi il modello della bacchetta a due punti.
Questo strumento permette inoltre la costruzione dei grafici che riportano l'andamento nel tempo delle coordinate di ogni *marker* calcolate in precedenza.

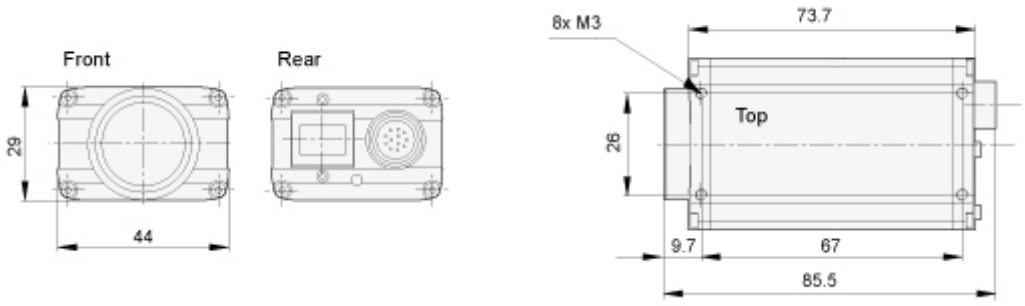
Smart Analyzer

Questo software consente di eseguire un'analisi biomeccanica e l'elaborazione dei dati di cinematica e dinamici. Alcune operazioni usate di frequente nello SMART *analyzer* sono:

- creazione di protocolli di analisi;
- importazione o esportazione dati acquisiti;
- visualizzazione 3D dell'acquisizione con il sistema di riferimento della calibrazione;
- creazione di report clinici.

Nel lavoro di tesi tale strumento è stato utilizzato, una volta ricavate le posizioni 3D dei diversi *marker* mediante lo *smart tracker*, per risalire al valore delle interdistanze tra *marker* 1 e *marker* 2 utilizzate poi per quantificare la bontà della calibrazione nel confronto con i dati da sistema D-RGB.

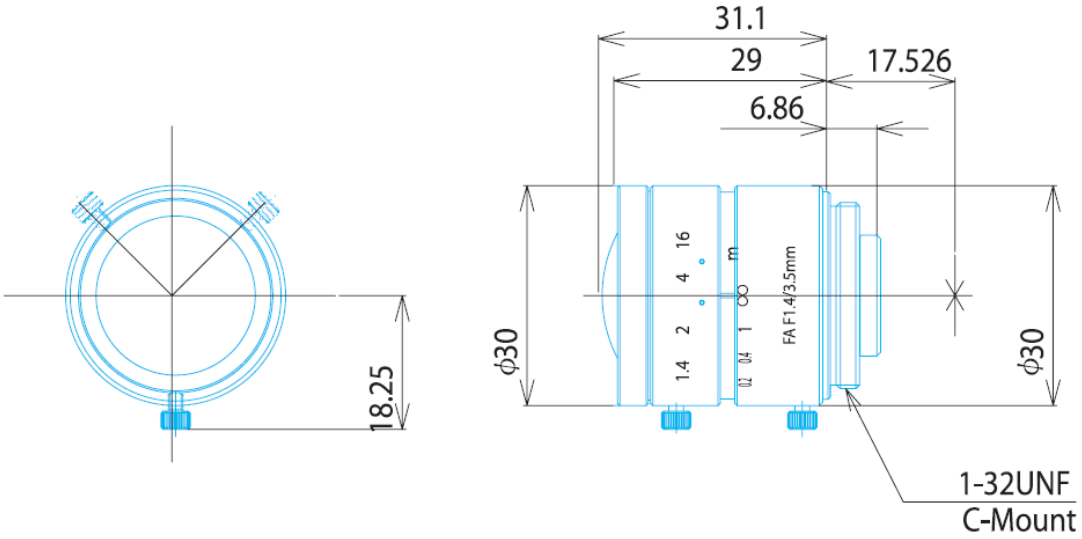
Telecamere a infrarossi BASLER scA640-120gm



Resolution horizontal/vertical	659 pixels x 494 pixels
Pixel Size horizontal/vertical	5.6 µm x 5.6 µm
Frame Rate	122 fps
Mono/Color	Mono
Interface	Gigabit Ethernet
Video Output Format	Mono 8, Mono 16, Mono 12 Packed, YUV 4:2:2 Packed, YUV 4:2:2 (YUYV) Packed
Pixel Bit Depth	12 bits
Synchronization	external trigger software

Exposure Control	programmable via the camera API
Housing Size (L x W x H) in mm	73.7 x 44 x 29
Housing Temperature	0 °C - 50 °C
Lens Mount	C-mount
Digital Input	2
Digital Output	4
Power Requirements	12-24 VDC
Power Consumption (typical)	3.5 W
Weight (typical)	160 g
Conformity	CE RoHS GenICam GigE Vision IP30 FCC
Sensor Vendor	Sony
Sensor Name	ICX618
Sensor Technology	Progressive Scan CCD, global shutter
Sensor Size (optical)	1/4 inch
Sensor Type	CCD
Sensor Size (mm)	3.69 mm x 2.77 mm
Item Number	104464

Lenti Goyo modello GM23514MCN



Focal length	3.5 mm
Iris Range	F 1.4 – Close
Angle of View (HxVxD) 1/2"	103.6° x 76.7° x 131.4°
MOD	0.1 m
Filter Thread	-
Dimention (DxL)	Ø30 x 31.5 mm
Weight	75 g

Appendice C

Codice

- **Il codice permette di verificare il corretto funzionamento del sistema e a fornire in caso contrario una diagnosi:**

```
#define CHECK_RC(rc, what)
if (rc != XN_STATUS_OK)
{
	printf("%s failed: %s\n", what, xnGetStatusString(rc));
	return rc;
}
```

- **Il codice riportato in seguito permette di inizializzare il sensore e di ottenere in output una mappa di profondità della scena in esame:**

```
using namespace xn;
int main()
{
	// Associo ad una variabile dei valori di status del sensore
	XnStatus nRetVal = XN_STATUS_OK;
	// Creazione ed inizializzazione dell'oggetto Context
	Context context;
	nRetVal = context.Init();
	//controllo eventuali errori
	CHECK_RC(nRetVal, "Initialize context");
	// Creo il generatore di profondità
	DepthGenerator depth;
	nRetVal = depth.Create(context);
	// Controllo eventuali errori
	CHECK_RC(nRetVal, "Create depth generator");
	nRetVal = context.StartGeneratingAll();
	CHECK_RC(nRetVal, "StartGeneratingAll");
```

```
        DepthMetaData depthMD;
        while (!xnOSWasKeyboardHit()) // mando in loop
        {
                // Attendo che i nuovi dati siano disponibili
                nRetVal = context.WaitOneUpdateAll(depth);
                controllo eventuali errori
                if (nRetVal != XN_STATUS_OK)
                {
                        printf("UpdateData failed: %s\n", xnGetStatusString(nRetVal));
                        continue;
                }
                // Ottengo gli attuali meta-dati di profondità
                depth.GetMetaData(depthMD);
                // Prendo l'attuale mappa di profondità
                const XnDepthPixel* pDepthMap = depthMD.Data();
                printf("Frame %d Middle point is: %u.\n",
                depthMD.FrameID(), depthMD(depthMD.XRes()/2,
                depthMD.YRes()/2));
        }
        context.Shutdown(); // Chiudo il Context
        return 0;
}
```

- **Il codice seguente sfrutta le primitive OpenNI per ottenere l'immagine IR nel caso si decida di procedere alla calibrazione attraverso l'analisi dell'immagine IR:**

```
XnStatus nRetVal;
xn::Context context;
xn::IRGenerator nIRGen;

nRetVal = XN_STATUS_OK;

// Initialize context object
nRetVal = context.Init();
// TODO: check error code

// Create a IRGenerator node
nRetVal = nIRGen.Create(context);

// Make it start generating data
nRetVal = context.StartGeneratingAll();

// Wait for new data to be available
nRetVal = context.WaitAnyUpdateAll();

if (nRetVal != XN_STATUS_OK)
   {
      printf("Failed updating data: %s\n", xnGetStatusString(nRetVal));
   }

const XnIRPixel* IRMap;= nIRGen.GetIRMap();
```

- **Nel codice riportato di seguito viene illustrato come identificare un nuovo utente nella scena e come traccare la posizione dei 15 punti chiave che ne descrivono lo scheletro. Nello specifico, esso stampa la posizione della testa dell'utilizzatore.**

```
XnStatus nRetVal = XN_STATUS_OK;
    xn::Context context;
    nRetVal = context.Init();
    // TODO: check error code
    // Create the user generator
    nRetVal = g_UserGenerator.Create(context);
    // TODO: check error code
    XnCallbackHandle h1, h2, h3;
    g_UserGenerator.RegisterUserCallbacks(User_NewUser, User_LostUser,
                                                        NULL, h1);
    g_UserGenerator.GetPoseDetectionCap().RegisterToPoseCallbacks(
                                Pose_Detected, NULL, NULL, h2);
    g_UserGenerator.GetSkeletonCap().RegisterCalibrationCallbacks(
                                Calibration_Start, Calibration_End, NULL, h3);
    // Set the profile
    g_UserGenerator.GetSkeletonCap().SetSkeletonProfile(
                            XN_SKEL_PROFILE_ALL);
    // Start generating
    nRetVal = context.StartGeneratingAll();
    // TODO: check error code
    while (TRUE)
    {
            // Update to next frame
            nRetVal = context.WaitAndUpdateAll();
            // TODO: check error code
            // Extract head position of each tracked user
            XnUserID aUsers[15];
            XnUInt16 nUsers = 15;
            g_UserGenerator.GetUsers(aUsers, nUsers);
            for (int i = 0; i < nUsers; ++i)
            {
                    if (g_UserGenerator.GetSkeletonCap().IsTracking(aUsers[i]))
                    {
                            XnSkeletonJointPosition Head;
                            g_UserGenerator.GetSkeletonCap().GetSkeletonJointPositio n( aUsers[i],
                            XN_SKEL_HEAD, Head);
                            printf("%d: (%f,%f,%f) [%f]\n", aUsers[i],
                            Head.position.X, Head.position.Y, Head.position.Z,
                            Head.fConfidence);
                    }
            }
    }
    // Clean up
    context.Shutdown();
```

- **Di seguito un semplice programma che disegna utilizzando le primitive OpenGL un triangolo. Si noti che questo programma fa uso di funzionalità deprecate a partire da OpenGL 3.0.**

```
#include <GL/gl.h>   // File header per la libreria OpenGL
```

```
#include <GL/glut.h> // File header per la libreria GLUT
#include <stdlib.h>  // File header per usare 'exit()'

/* Funzione invocata quando la finestra viene ridimensionata (anche quando viene creata) */
void resize (int width, int height)
{
        glViewport (0, 0, width, height); // Usiamo tutta finestra
        glMatrixMode (GL_PROJECTION);     // Seleziona di usare la matrice
                        // 'PROJECTION'
    glLoadIdentity ();              // Resetta la matrice 'PROJECTION'
    glMatrixMode (GL_MODELVIEW);      // Seleziona di usare la matrice
                // 'MODELVIEW'
    glLoadIdentity ();              // Resetta la matrice 'MODELVIEW'
}

/* Funzione di disegno */
void draw ()
{
        glClearColor (0.0f, 0.0f, 0.0f, 0.0f); // Sfondo nero
    glClear (GL_COLOR_BUFFER_BIT);        // Cancella la scena

glBegin (GL_TRIANGLES);               // Disegna un triangolo
        glColor3f (1.0f, 0.0f, 0.0f);         // Impostiamo il colore rosso
glVertex3f ( 0.0f, 0.5f, 0.0f);       // Angolo in alto
glColor3f (0.0f, 1.0f, 0.0f);         // Imposta il colore verde
glVertex3f (-0.5f, -0.5f, 0.0f);      // Angolo basso sinistro
glColor3f (0.0f, 0.0f, 1.0f);         // Imposta il colore blu
glVertex3f (0.5f, -0.5f, 0.0f);       // Angolo basso destro
    glEnd ();                         // Fine triangolo

         glutSwapBuffers ();              // Disegna!
}

/* Funzione invocata ogni volta che viene premuto un tasto */
void keyPressed (unsigned char key, int x, int y)
{
         if (key == 'q')   // Se il tasto premuto e` q,
exit (0);     // esce
}

/* Main */
int main (int argc, char **argv)
{
        glutInit (&argc, argv);               // Inizializza la libreria
                                              // GLUT
        glutInitDisplayMode (GLUT_RGB|GLUT_DOUBLE); // Seleziona il modo di
                //visualizzazione: usa RGB e double buffer
glutInitWindowSize (640, 480);        // Imposta la dimensione della
                        // finestra a 640x480
glutInitWindowPosition (0, 0);            // Imposta la posizione
                //dell'angolo alto sinistro della finestra
glutCreateWindow ("Esempio uso OpenGL");  // Crea la finestra
glutDisplayFunc (draw);           // Imposta la funzione di disegno
glutReshapeFunc (resize);             // Imposta la funzione di
                        // ridimensionamento
glutKeyboardFunc (keyPressed);        // Imposta la funzione per gli
                        // eventi della tastiera
glutMainLoop ();                      // Inizio
return (1);
}
```

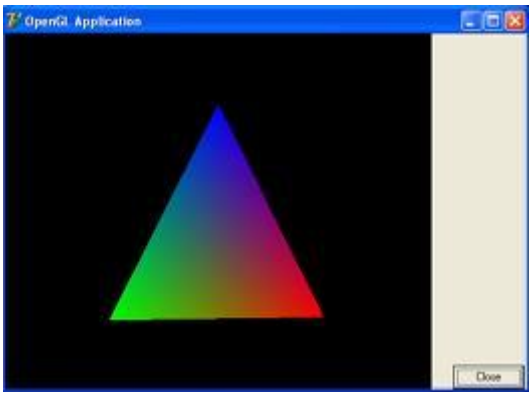


Fig. 1 Risultato del codice sopra

- **Algoritmo di identificazione del tool a nove sfere (in pseudocodice).**

```
//Contiene le distanze tra tutte le possibili coppie di sfere trovate //nell'immagine
double distanze[NUM_MARKERS][NUM_MARKERS];
   for (int c=0; c<NUM_MARKERS; c++)
   {
      for (int r=0; r<NUM_MARKERS; r++)
      {
         distanze[r][c]=calcolaDistanza(realPoint1[c],realPoint1[r]);
                  //realPoint1 contiene le coordinate di tutti i 9 punti trovati
         if (distanze[r][c] è pari alla distanza tra A e B )
         {
                              Risalgo alle posizioni nel vettore realPoint1 occupate  dalle sfere A e B
         }
      }
   }

   XnPoint3D a,b,c,d,e,f,g,h,i;
   bool trovatiABC = false;
   int colB;

   for (int r=0; r<NUM_MARKERS; r++)
   {
      if (distanze[r][colAoB[0]] è uguale alla distanza BC)
      //Se entra nell'if allora colAoB[0] contiene la colonna di B
      {
         Assegno A;
         Assegno B
      }
   }

   if (trovatiABC==false)//colAoB[0] contiene la sfera A
   {
      Assegno A;
         Assegno B

   }

   //Ho sicuramente assegnato A e B
   int colD;

   for (int r=0; r<NUM_MARKERS;r++)
```

```
{
   if (distanze[r][colB] è uguale a distanza BC)
   {
      Assegno C
   }
   else if (distanze[r][colB] è uguale a distanza BD )
   {
      Assegno D
   }

}
//Assegnate C e D

//Trovo la sfera F attraverso la distanza DF

int colF;

for (int r=0; r<NUM_MARKERS;r++)
{
   if (distanze[r][colD] è uguale a distanza DF)
   {
      Assegno F
   }
}

for (int r=0; r<NUM_MARKERS;r++)
{
   if (distanze[r][colF] è uguale a distanza FG )
   {
      Assegno G

   }
   else if(distanze[r][colF] > è uguale a distanza FE)
   {
      Assegno E
   }
}
//Trovo H in base a DH
for (int r=0; r<NUM_MARKERS;r++)
{
   if (distanze[r][colD] > è uguale a distanza DH )
   {
      Assegno H
   }
}

for (int s=0; s<NUM_MARKERS;s++)
{
   if (trovo il punto non ancora assegnato)
   {
      Assegno I
   }
 }
```

Bibliografia

[**1**] Yannic Schroder, Alexander Scholz, Kai Berger, Kai Ruhl, Dr. Stefan Guthe, Prof. Dr. Ing. Marcus (2011), ***Multiple Kinect Studies,*** Technical Report 2011-09-15, Computer Graphics Lab, TU Braunschweig.

[**2**] Kai Berger, Kai Ruhl, Yannic Schroeder, Christian Bruemmer , Alexander Scholzk, Marcus Magnor (2011), ***Markerless Motion Capture using multiple Color-Depth Sensors,*** Vision, Modeling, and Visualization (2011), The Eurographics Association 2011

[**3**] Angelo Cappello, Aurelio Cappozzo, Pietro Enrico di Prampero (2003), ***Bioingegneria della postura e del movimento,*** PATRON editore Bologna 2003.

[**4**] Carlo Dal Mutto, Pietro Zanuttigh, Guido M. Cortelazzo (2012), ***Time-of-flight Cameras and Microsoft Kinect,*** Springer briefs in electrical and computer engineering

[**5**] Brian W.Kernighan, Dennis M. Ritchie (1989), ***Linguaggio C,*** Gruppo editorial Jackson 1989

[**6**] Lars Mündermann, Stefano Corazza1 and Thomas P Andriacchi (2006), ***The evolution of methods for the capture of human movement leading to markerless motion capture for biomechanical applications,*** *Journal of NeuroEngineering and Rehabilitation* 2006, **3**:6 doi:10.1186/1743-0003-3-6

[**7**] Stefano Corazza, Emiliano Gambaretto, Lars M¨undermann, and Thomas P. Andriacchi (2010), ***Automatic Generation of a Subject-Specific Model for Accurate Markerless Motion***

Capture and Biomechanical Applications.

[**8**] Elena Ceseracciu a, Zimi Sawacha, Silvia Fantozzi, Matteo Cortesi, Giorgio Gatta, Stefano Corazza, Claudio Cobelli (2011), ***Markerless analysis of front crawl swimming,*** Journal of Biomechanics 44 (2011) 2236–2242.

[**9**] Stefano Corazzaa, Lars Mu¨ ndermanna, Tom Andriacchia (2007), ***A framework for the functional identification of joint centers using markerless motion capture, validation for the hip joint,*** Journal of Biomechanics 40 (2007) 3510–3515

[**10**] J. Jackman (2007), ***Bluescreen Compositing***, Burlington, USA: Focal Press, 2007.

[**11**] S. Corazza, L. Mundermann, A. M. Chaudhari, T. Demattio, C. Cobelli,T. P. Andriacchi (2006*), **A Markerless Motion Capture System to Study Musculoskeletal Biomechanics: Visual Hull and Simulated Annealing Approach,*** *Annals of Biomedical Engineering*, Vol. 34, No. 6, June 2006 pp. 1019–1029.

[**12**] G. Guerra-Filho (2005), ***Optical motion capture: theory and implementation***, Journal of Theoretical and Applied Informatics, vol 12, n° 2, pagg 61–89, 2005.

[**13**] StefanoCorazza, ThomasP.Andriacchi (2008), ***Posturographic analysis through markerless motion capture without ground reaction forces measurement,*** Journal of Biomechanics 42 (2009) 370–374.

[**14**] Ariel V. Dowling, Stefano Corazza, Ajit M. W. Chaudhari and Thomas P. Andriacchi (2010), ***Shoe-Surface Friction Influences Movement Strategies During a Sidestep Cutting Task : Implications for Anterior Cruciate Ligament Injury Risk,*** Am J Sports Med 2010 38: 478.

[**15**] Michal Tölgyessy, Peter Hubinský (2011), ***The Kinect Sensor in Robotics Education***.

[**16**] Dimitrios Ioannoua, Walter Hudab, Andrew F. Lainec (1998), ***Circle recognition through a 2D Hough Transform and radius histogramming,*** Image and Vision Computing 17 (1999) 15–26.

[**17**] Berthold K. P. Horn (1986), ***Closed-form solution of absolute orientation using unit quaternions,*** Journal of the Optical Society of America A, Vol. *4*, page 629, April 1987.

[**18**] Jamie Shotton Andrew Fitzgibbon Mat Cook Toby Sharp Mark Finocchio, Richard Moore Alex Kipman Andrew Blake (),***Real-Time Human Pose Recognition in Parts from Single Depth Images,*** Microsoft Research Cambridge & Xbox Incubation.

[**19**] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Andrew Fitzgibbon (2012),***KinectFusion: Real-Time Dense Surface Mapping and Tracking***, Microsoft Research (2012).

[**20**] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, Andrew Fitzgibbon (),***KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera,*** Microsoft Research Cambridge, UK UIST'11, October 16-19, 2011, Santa Barbara, CA, USA.

[**21**] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman Andrew Blake (),***Real-Time Human Pose Recognition in Parts from Single Depth Images: Supplementary Material,*** Microsoft Research Cambridge & Xbox Incubation.

[**22**] Zhengyou Zhang (1998), ***A Flexible New Technique for Camera Calibratio,*** *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[**23**] Andrea Giovanni Cutti, Gabriele Paolini, Marco Troncossi, Angelo Cappello, Angelo Davalli (2004***), Soft tissue artefact assessment in humeral axial rotation,*** Gait and Posture (2004).

[**24**] Sprigle S, Wootten M, Sawacha Z, Thielman G. (), ***Relationships among cushion type, backrest height, seated posture, and reach of wheelchair users with spinal cord injury.***

[**25**] Stefano Corazza, Lars Mündermann, Emiliano Gambaretto, Giancarlo Ferrigno, Thomas P. Andriacchi (2010), ***Markerless Motion Capture through Visual Hull, Articulated ICP and Subject Specific Model Generation.***

[**26**] A. Kontaxis, A.G. Cutti, G.R. Johnson, H.E.J. Veeger (2009), ***A framework for the definition of standardized protocols for measuring upper-extremity kinematics***, Clinical Biomechanics 24 (2009) 246–253.

[**27**] Andrew Maimone, Henry Fuchs (2012***), Reducing Interference Between Multiple Structured Light Depth Sensors Using Motion,*** I.4.3 [Computing Methodologies]: Image Processing and Computer Vision—Image Enhancement.

[**28**] Alex Butler, Shahram Izadi, Otmar Hilliges, David Molyneaux, Steve Hodges, David Kim (2012), ***Shake'n'Sense: Reducing Interference for Overlapping Structured Light Depth Cameras***

[**29**] Programmer's Guide, ***Prime Sensor™ NITE 1.3 Controls*** Version 1.0

[**30**] Manuale Kinect: ***http://en.wikipedia.org/wiki/Kinect***

[**31**] Manuale OpenNI: ***http://www.openni.org/***

[**32**] Manuale Asus Xtion Pro: ***http://www.asus.com/Multimedia/Xtion_PRO/***

[**33**] Matlab Calibration Toolbox: ***http://www.vision.caltech.edu/bouguetj/calib_doc/***

[**34**] Brochure prodotto "***Nirvana***"

[**35**] Forum **http: *//openkinect.org/wiki/Main_Page***

[**36**] Guida QT: ***http://www.qt-italia.org***

[**37**] Guida OpenGL:
http://nehe.gamedev.net/tutorial/creating_an_opengl_window_(win32)/13001/

[**38**] Reference page OpenGL: ***http://www.opengl.org/sdk/docs/man/xhtml/***

[**39**] ***http://www.goyooptical.com/products/industrial/GM23514MCN.pdf***

[**40**] ***http://www.baslerweb.com/products/scout.html?model=108&language=en***

[**41**] Sito azienda BTS Bioengineering: ***http://www.btsbioengineering.com/***

[**42**] Point Cloud Library: ***http://pointclouds.org/***

[**43**] ***http://meshlab.sourceforge.net/***

[**44**] Materiale corso ***"Bioingegneria del movimento e riabilitazione***", corso di laurea in Bioingegneria anno accademico 2011-2012

[**45**] Alice Mantoan, (2011), **"*Underwater gait analysis: a markerless approach"***

[**46**] ***http://www.flickriver.com/photos/dominicspics/sets/72157625795608025/***

[**47**] ***http://graphics.stanford.edu/~mdfisher/Kinect.html***

[**48**] ***http://www.ros.org/news/packages/***

[**49**] ***http://www.forbes.com/fdc/welcome_mjx.shtml***

[**50**] **http://*www.forbes.com/sites/kellyclay/2012/04/02/finalists-for-microsoft-kinect-accelerator-include-two- seattle-startups/***

[**51**] ***http://www.assodigitale.it/2011/04/05/con-asus-xtion-pro-il-pc-si-controlla-con-il-movimento/***

[**52**] ***http://www.asus.it/Multimedia/Motion_Sensor/Xtion_PRO_LIVE/***

[**53**] ***http://press.asus.com/computex2011/asus-innovation-beyond-expectations-at-computex-2011/***

[**54**] ***http://www.vision-systems.com/articles/print/volume-17/issue-9/features/kinect-api-makes-low- cost- 3-d-imaging-systems-attainable.html***

[**55**] ***http://www.vision-systems.com/articles/print/volume-17/issue-9/features/kinect-api-makes-low-cost-3-d-imaging-systems-attainable.html***

[**56**] ***http://www.vision-systems.com/articles/print/volume-17/issue-9/departments/technology-trends/motion-analysis-gesture-recognition-therapy-system-gets-physical-kinect-based-motion-analysis.html***