

Indice

Introduzione	3
Capitolo 1 – Stato dell’arte	5
1.1 Tipologie di sistemi di ricostruzione 3D	5
1.1.1 Scanner 3D ottici attivi.....	6
1.1.2 Scanner 3D ottici passivi	6
1.2 Sistemi di ricostruzione 3D basati su visione stereoscopica calibrata.....	6
1.3 Sistemi di ricostruzione 3D basati su visione stereoscopica non calibrata.....	8
Capitolo 2 – Teoria	11
2.1 La visione stereoscopica	11
2.1.1 Calibrazione e rettificazione	12
2.1.2 Calcolo delle corrispondenze.....	13
2.2 L’algoritmo SIFT.....	14
2.2.1 Ricerca degli estremanti nello scale space	15
2.2.2 Localizzazione dei keypoint	16
2.2.3 Assegnamento della direzione.....	17
2.2.4 Definizione del scrittore dei keypoint.....	17
2.2.5 Corrispondenze tra keypoint	18
2.2.6 Esempi pratici	19
2.3 Trasformazione tra viste	20
2.3.1 Trasformazione indotta da 3 punti.....	20
2.3.2 Quaternioni.....	22
2.3.3 Algoritmo iterativo	23
2.3.4 Algoritmo ai minimi quadrati.....	24
2.3.5 Confronto dei metodi	26

Capitolo 3 – Architettura del sistema	28
3.1 Panoramica del sistema	28
3.2 Acquisizione	29
3.3 Ricostruzione.....	30
3.3.1 Stima robusta del moto	31
3.3.2 Riproiezione dei punti.....	35
3.4 Quantizzazione e filtraggio.....	38
3.4.1 Octree	38
3.4.2 Quantizzazione.....	39
3.4.3 Filtraggio	39
Capitolo 4 – Risultati Sperimentali	41
4.1 Immagini reali.....	42
4.2 Immagini sintetiche.....	53
Capitolo 5 – Conclusioni	65
5.1 Considerazioni finali	65
5.2 Sviluppi futuri.....	66
Bibliografia	69

Introduzione

La ricostruzione tridimensionale (3D) di una scena è un argomento che nel campo della visione artificiale ha suscitato fin dagli inizi un notevole interesse. I sistemi di ricostruzione 3D, o scanner 3D, basati sulla visione artificiale di tipo passivo (che inferiscono la struttura 3D mediante la sola luce irradiata dagli oggetti analizzati) risultano particolarmente interessanti in quanto non invasivi e perché piuttosto economici. Le applicazioni in cui trovano impiego sono innumerevoli e spaziano in settori che vanno dall'ambito industriale all'architettura, dalla medicina alla conservazione di beni culturali, dal rilievo di territori alla cinematografia. Una delle tecniche per estrarre la struttura 3D usate da questi sistemi è la visione stereoscopica, che consente di determinare la profondità di un oggetto osservandolo da 2 o più punti di vista. D'ora in poi si farà riferimento alla visione stereoscopica a partire da 2 viste (detta anche visione binoculare). La ricostruzione ottenuta in questo modo è solo parziale e relativa alla porzione di oggetto inquadrata nelle 2 viste. Per ottenere una ricostruzione più completa occorre osservare l'oggetto da più punti di vista. Esistono particolari sistemi di ricostruzione che sfruttano supporti meccanici per imporre un movimento controllato al sistema stereo o all'oggetto, ad esempio la ripresa di un oggetto posto su un piatto rotante controllato meccanicamente. Tuttavia un approccio di questo tipo si rivela in alcuni casi eccessivamente vincolante e di conseguenza non applicabile. La possibilità di ottenere ricostruzioni da un sistema portatile diviene quindi una caratteristica particolarmente desiderabile, specialmente in applicazioni in cui non è concepibile alcuna valida alternativa.

Ottenere la ricostruzione di un oggetto semplicemente ruotandovi attorno il sistema stereo rende lo scanner ampiamente versatile ed integrabile in molteplici applicazioni. Nasce proprio da questa necessità il lavoro svolto, ovvero la progettazione e realizzazione di un sistema di ricostruzione 3D a partire da viste stereo multiple tra loro non calibrate. La problematica principale da affrontare è la determinazione di posizione e orientazione di ogni punto di vista, complicata ulteriormente dalla presenza di svariate incertezze nei dati in ingresso. L'idea alla base della ricostruzione è di individuare le corrispondenze fotometriche tra coppie di immagini in sequenza, risalire alle relative coordinate 3D e determinarne la relazione (in termini di rotazione e traslazione). Una volta determinate le relazioni che intercorrono tra i diversi punti di vista si procede con la fusione delle mappe al fine di ottenere un modello unico dell'intera scena.

Questo lavoro si articola su 5 capitoli a seguito brevemente descritti. Nel primo capitolo è analizzato lo stato dell'arte dei sistemi di ricostruzione 3D, con una breve rassegna sulle diverse tecnologie esistenti. L'attenzione è poi concentrata sui sistemi ottici che utilizzano la visione stereoscopica, sia in forma calibrata che non.

Nel secondo capitolo sono trattati tutti i concetti teorici necessari a capire le basi del lavoro svolto. Sono presentati i concetti fondamentali della proiezione prospettica, ovvero il modello geometrico che mette in relazione i punti 3D della scena con i punti 2D dell'immagine, e della visione stereoscopica, ovvero come la posizione di un punto 3D della scena sia determinabile a partire dalle sue proiezioni prospettiche in 2 immagini. In questo capitolo è inoltre illustrato l'algoritmo SIFT che consente di estrarre robuste corrispondenze fotometriche tra immagini. Sono poi presentati 2 metodi che determinano la relazione tra coppie di punti con coordinate misurate in differenti sistemi di riferimento.

Nel terzo capitolo è descritta l'intera architettura del sistema. Da una panoramica generale a blocchi si procede con un'analisi dettagliata delle fasi di acquisizione, ricostruzione e filtraggio. Nella fase di acquisizione sono illustrati i 2 algoritmi implementati che risolvono il problema delle corrispondenze e che permettono di inferire la struttura 3D della scena da una singola vista. Particolare attenzione è riservata alla fase di ricostruzione in cui sono descritti gli algoritmi utilizzati per la stima robusta del moto delle camere. Nella fase di filtraggio sono descritti i metodi che mirano a semplificare il modello generato e a ridurne il rumore.

Nel quarto capitolo sono presentati i risultati sperimentali ottenuti dal sistema. Sono divisi in due sezioni, in base alla modalità di acquisizione impiegata.

Nel quinto ed ultimo capitolo sono riportate le considerazioni finali. Sono analizzate le qualità e le principali limitazioni del sistema progettato. Infine, sono riportate alcune proposte di sviluppo, mirate al miglioramento del sistema.

1 - Stato dell'arte

In questo capitolo si analizza lo stato dell'arte dei sistemi di ricostruzione 3D (Scanner 3D). Nel paragrafo 1.1 viene presentata una breve rassegna delle tipologie di sistemi esistenti, dai sistemi a contatto a quelli ottici. Nei paragrafi 1.2 e 1.3 vengono presentati alcuni sistemi d'interesse che si basano sulla visione stereoscopica calibrata e non.

1.1 Tipologie di sistemi di ricostruzione 3D

Il punto di partenza per la ricostruzione 3D di un oggetto è il campionamento della sua superficie. Esistono diverse tecnologie che si prestano a questo scopo e possono essere classificate tramite l'albero mostrato in Figura 1.1.1. Una prima classificazione si può effettuare in base all'approccio con l'oggetto, in particolare se è necessario il contatto o meno. Gli scanner 3D a contatto necessitano di toccare fisicamente l'oggetto, rendendo l'approccio talvolta invasivo. Quelli non a contatto invece sono meno invasivi e dunque trovano maggior applicazione. Senza entrare nel dettaglio di ogni singola tipologia se ne descrivono le principali nella categoria degli scanner ottici.

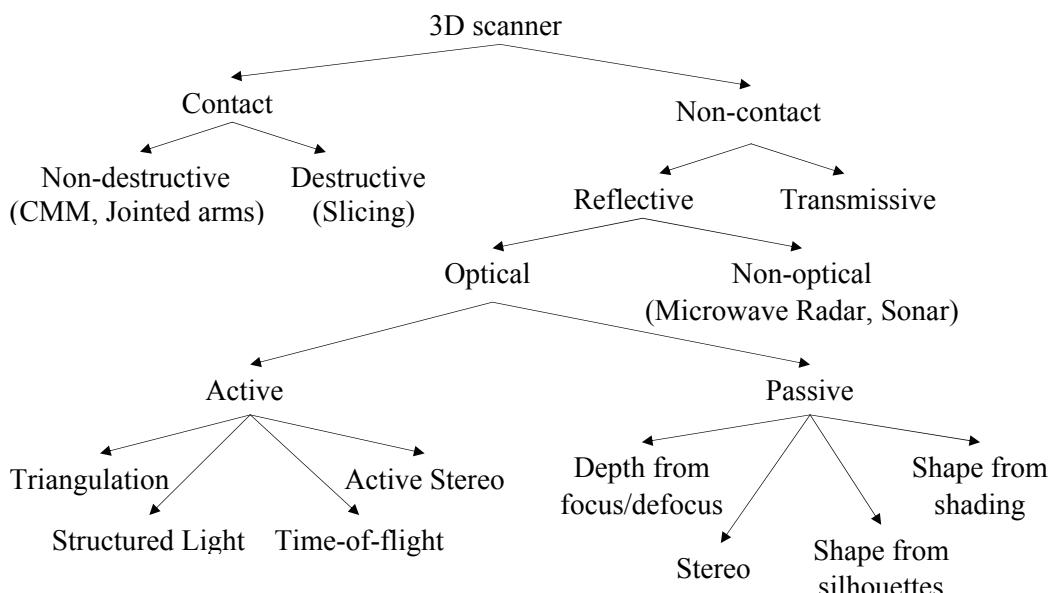


Figura 1.1.1 - Classificazione degli scanner 3D

1.1.1 Scanner 3D ottici attivi

Questi tipi di scanner 3D misurano la forma dell'oggetto tramite una certa emissione, come luce o ultrasuoni. Un sistema Time-of-flight misura il tempo di andata/ritorno di un impulso luminoso da cui ne ricava la distanza, mentre un sistema a triangolazione misura l'angolo di riflessione dell'impulso tramite un sensore CCD e da questo ne ricava la distanza (Figura 1.1.2).

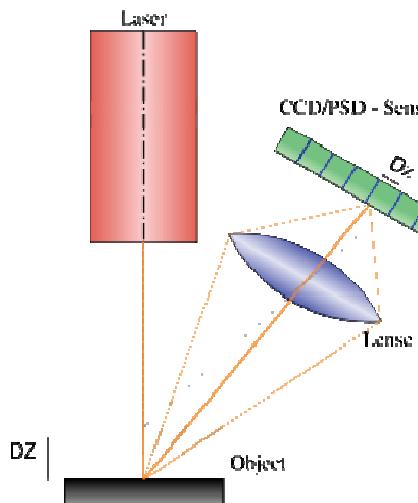


Figura 1.1.2 - Principio di funzionamento di uno scanner 3D a triangolazione

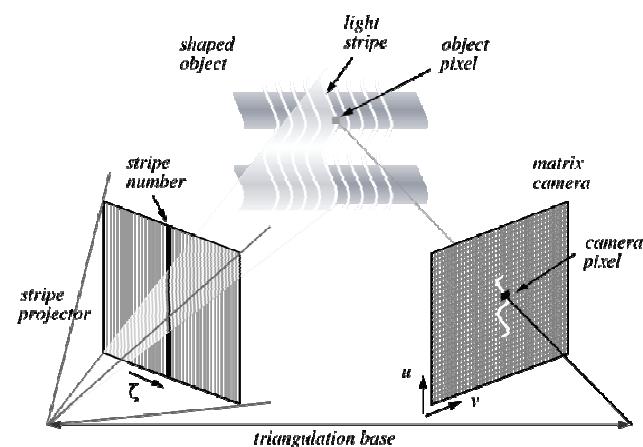


Figura 1.1.3 - Principio di funzionamento di uno scanner 3D a luce strutturata

Uno scanner 3D a luce strutturata proietta sull'oggetto un pattern noto, che può essere unidimensionale (es. linee) o bidimensionale (es. griglia), misurando poi la distorsione del pattern tramite una camera (Figura 1.1.3).

1.1.2 Scanner 3D ottici passivi

Gli scanner 3D ottici passivi mirano alla ricostruzione dell'oggetto mediante la sola luce irradiata. Una tipologia di questi ricostruisce l'oggetto a partire dalle silhouette, richiede quindi l'oggetto posto su uno sfondo a contrasto elevato. Noto l'angolo di rotazione di ogni silhouette si ricostruisce la superficie. Altri sistemi estraggono l'informazione 3D dalle ombre sull'oggetto (Shape from shading), mentre altri estraggono la profondità di un punto esaminando il livello di "blurring" di diverse immagini catturate a differenti lunghezze focali (Depth from focus/defocus). Infine esistono scanner 3D basati sulla visione stereoscopica che verranno esaminati più nel dettaglio nei paragrafi 1.2 e 1.3.

1.2 Sistemi di ricostruzione 3D basati su visione stereoscopica calibrata

Questi sistemi tracciano la posizione dei punti nello spazio tramite il principio di visione stereoscopica. I punti dell'oggetto vengono inquadrati da varie angolazioni estrapolandone l'informazione di profondità. Maggiori dettagli sulla visione stereoscopica sono rimandati al paragrafo 2.1. In questo paragrafo si vuole concentrare l'attenzione sui sistemi già esistenti che ne fanno uso. Il principio cardine è il seguente: per ogni vista si calcola la

relativa mappa di profondità, si procede con la fusione delle mappe ottenute (le varie viste sono calibrate, è quindi nota la trasformazione tra una mappa e l'altra) e si genera la mesh.

Bradley et al. in [3] propongono un sistema basato sulla fusione di mappe di profondità accurato e veloce. Il primo passo dell'algoritmo è calcolare le mappe di profondità, a questo scopo viene effettuata la rettificazione dell'immagine di una vista con la successiva mediante l'algoritmo proposto da Fusiello et al. in [5], calcolando poi la mappa di disparità mediante la funzione NCC¹ su finestre “rettificate” a diversi fattori di scala. Le mappe vengono poi pulite dal rumore ad alta frequenza tramite un opportuno filtro mediano e fuse insieme per ottenere la nuvola di punti del modello. La nuvola di punti viene poi semplificata rimuovendo l'informazione ridondante e filtrata con un algoritmo che preserva i punti in un intorno se appartenenti ad un piano. In Figura 1.2.1 viene riportato un esempio della precisione di questo metodo.

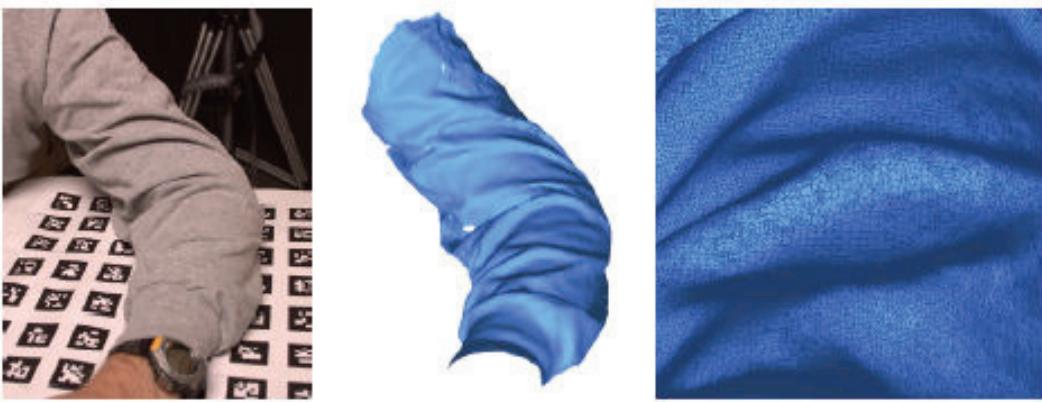


Figura 1.2.1 - A sinistra una delle immagini reali, al centro la ricostruzione finale, a destra uno zoom della ricostruzione che ne mostra il dettaglio

Furukawa e Ponce in [4] propongono un metodo di ricostruzione molto accurato che opera in 3 fasi denominate *matching*, *expansion* e *filtering*. Nella prima fase si dividono le immagini in *patch* e vengono selezionate le feature delle immagini con due diversi operatori: Harris² e DoG³. Per ogni feature individuata si cerca una corrispondenza nelle altre immagini (funzione NCC calcolata sulla proiezione) in prossimità delle linee epipolari, e si triangola il punto. In Figura 1.2.2 è mostrato un esempio di funzionamento della fase di *matching*. Nella seconda fase (*expansion*) si aggiungono altre patch a quelle già trovate cercando nelle celle adiacenti fino a ricoprire l'intera superficie visibile dell'immagine, in modo da ottenere una mappa densa di patch. Nella fase di *filtering* vengono eliminate tutte le patch classificate come *outlier*, in particolare patch che si trovano all'esterno dell'oggetto (se ne occlude altre) o all'interno (se la è occlusa da altre), un esempio in Figura 1.2.3.

¹ NCC (Normalised Cross-Correlation) è una funzione che permette di calcolare la similarità tra due finestre, robusta rispetto a variazioni fotometriche delle immagini.

² L' Harris Corner Detector è un operatore che consente di estrarre gli angoli in una immagine, con informazioni su direzione e intensità.

³ L'operatore DoG (Difference-of-Gaussians) viene impiegato per estrarre le feature “blob” da un'immagine.

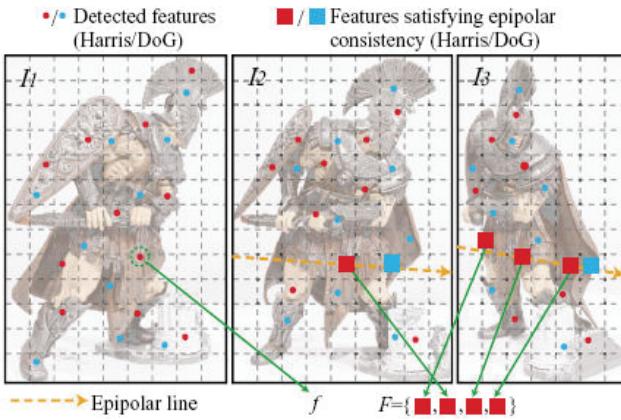


Figura 1.2.2 - Esempio di funzionamento della fase di Matching

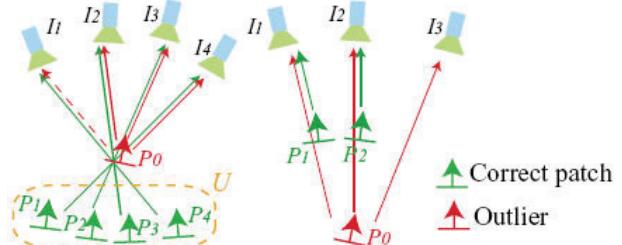


Figura 1.2.3 - Fase di filtering degli outlier esterni (sinistra) e interni (destra)

La mesh dell'oggetto viene ricostruita con un algoritmo a deformazione partendo dal guscio convesso delle patch sopravvissute. I risultati ottenuti da questo algoritmo sono notevoli (come mostrato in Figura 1.2.4), ed è al momento classificato il migliore come completezza e precisione nella valutazione definita in [6].



Figura 1.2.4 – Risultati dell'algoritmo

1.3 Sistemi di ricostruzione 3D basati su visione stereoscopica non calibrata

In questi sistemi di ricostruzione è necessario calcolare anche la posizione e l'angolazione delle viste in quanto non è un parametro noto a priori. A questo proposito si utilizzano tecniche che consentono di trovare elementi comuni in viste tra loro vicine e stabilirne una corrispondenza. Alcune applicazioni sfruttano vincoli specifici in base al problema da risolvere, in [15] si ricostruisce l'ambiente esplorato da un robot che si sposta liberamente in un piano (pavimento), il problema di fusione tra le varie viste ha quindi un vincolo in meno riportandolo ad un problema più semplice.

Park e Baek [1] propongono un sistema che utilizza una telecamera stereo spostabile liberamente (6 gradi di libertà). Il loro sistema, molto simile a quello presentato in questa tesi, effettua una doppia registrazione sulle immagini, salvando l'informazione geometrica e fotometrica. La selezione dei punti per la registrazione fotometrica viene effettuata mediante un KLT tracker¹, mentre per la registrazione geometrica usano un metodo di riproiezione iterativo che minimizza la distanza euclidea tra una superficie e la successiva. Oltre a questo aggiustamento effettuato per coppie si ricerca una ottimizzazione globale minimizzando l'errore tra un'immagine e le k precedenti. Le immagini sono poi registrate in un sistema di riferimento comune e viene effettuato il meshing dei punti con un algoritmo Marching Cubes. È riportato in Figura 1.3.1 e Figura 1.3.2 un esempio di ricostruzione di una statua (40 immagini).

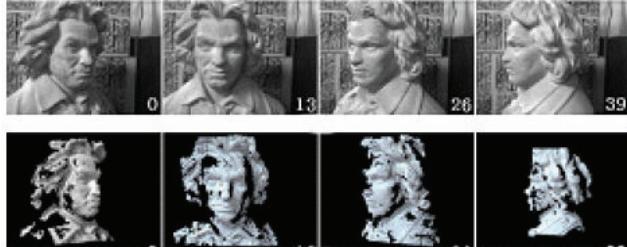


Figura 1.3.1 – Fase di acquisizione, immagini reali (sopra), e immagini con informazione geometrica (sotto)

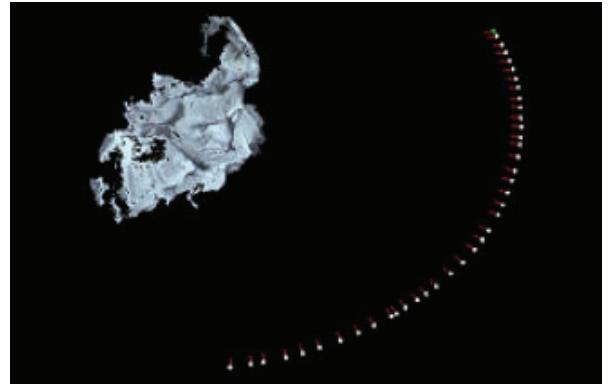


Figura 1.3.2 – Risultato finale

Un approccio diverso è quello di Se e Jasiobedzki in [8] che propongono un sistema per la ricostruzione 3D di una scena del crimine (ma non solo) denominato *instant scene modeler (iSM)*. Il calcolo delle mappe di disparità viene effettuato con un algoritmo proprietario (dalla libreria PGR's optimized Triclops library) e analogamente al sistema da noi progettato utilizza l'algoritmo SIFT per determinare il moto delle camere, ma con un differente approccio. Si estraggono le feature da entrambe le viste (sinistra e destra) al fine di creare delle feature SIFT 3D tenendo conto dei principali vincoli stereo (epipolare, unicità, gradiente di disparità, ordinamento) per scartare le false corrispondenze e vengono inserite in un database per poi effettuare dei confronti multiframe. Il movimento della camera viene quindi stimato con un approccio analogo al nostro, ovvero cercando il miglior allineamento tra le corrispondenze trovate implementando un metodo ai minimi quadrati e scartando quelle che apportano un errore eccessivo. Per la generazione della mesh utilizzano un metodo voxel-based [10] che riempie le lacune e lavora in modo efficiente anche con dati che presentano sovrapposizioni e ridondanze. Infine viene mappata la texture per ogni triangolo, selezionando la vista migliore su cui viene proiettata. In Figura 1.3.3 è mostrato un esempio di ricostruzione della facciata di una casa.

¹ Proposto da Kanade-Lucas-Tomasi [2], questo algoritmo consente di selezionare e tracciare alcuni punti da fotogramma a fotogramma in base a determinate caratteristiche.



Figura 1.3.3 – Acquisizione (sopra) e ricostruzione (sotto) con iSM della facciata di una casa

Analogamente al sistema da noi progettato, il sistema proposto da Yun et al. in [16] utilizza SIFT per estrarre le corrispondenze 3D, ma effettua la registrazione delle viste tramite l'algoritmo LMedS (Least Median of Square) che utilizza un approccio simile a RANSAC. In LMedS si estrae un campione di punti dalle corrispondenze SIFT, si stima la trasformazione del modello e si ordina l'errore residuo in modo crescente. Si effettuano una serie di iterazioni e si considera la trasformazione con errore residuo minore sul punto medio. In Figura 1.3.4 è mostrato un esempio del funzionamento dell'algoritmo LMedS con dimensione del campione pari ad m e numero di punti per stimare il modello pari a p .

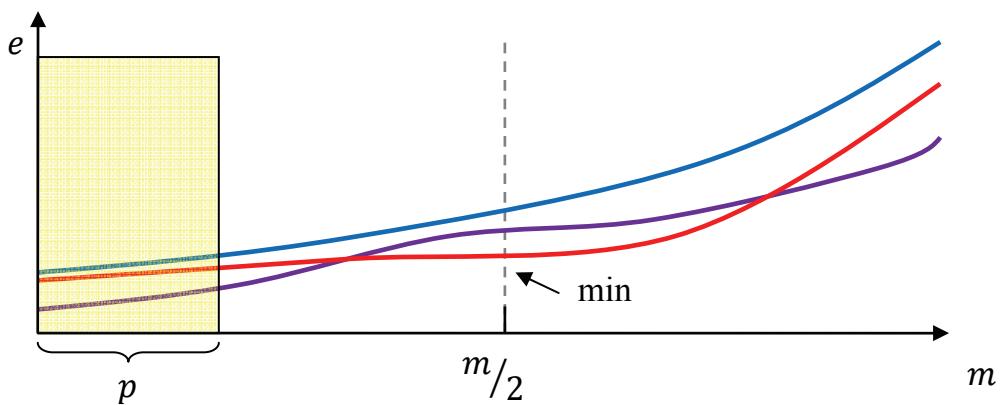


Figura 1.3.4 – Determinazione del campione nell'algoritmo LMedS

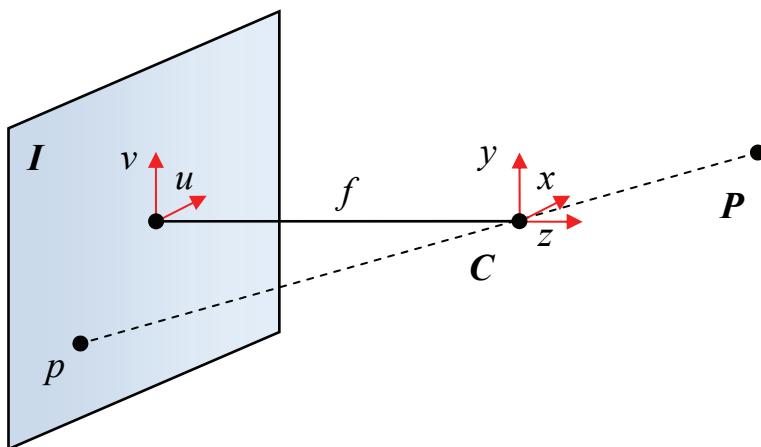
2 - Teoria

In questo capitolo vengono presentati alcuni concetti fondamentali per capire il lavoro svolto, le idee che ne stanno alla base e la terminologia usata. Nel paragrafo 2.1 si illustrano brevemente i principi cardine della visione stereoscopica, ovvero la proiezione prospettica e la geometria di un sistema stereo. Nel paragrafo 2.2 si illustra l'algoritmo SIFT che permette di stabilire corrispondenze robuste tra immagini. Nel paragrafo 2.3 si propongono e discutono alcuni metodi che stimano la trasformazione tra due viste a partire da corrispondenze.

2.1 La visione stereoscopica

Attraverso la visione stereoscopica (o visione stereo) è possibile determinare la posizione spaziale dei punti della scena. È opportuno capire prima di tutto il modello geometrico delle camere utilizzate, cioè la relazione che mappa un punto 3D della scena sul piano immagine: la proiezione prospettica (Figura 2.1.1). Considerando un sistema di riferimento centrato nel centro ottico con l'asse z ortogonale al piano immagine e gli assi x e y paralleli agli assi dell'immagine, le equazioni che legano un punto della scena (coordinate x, y, z) e un punto dell'immagine (coordinate u, v) sono

$$\frac{u}{x} = \frac{v}{y} = -\frac{f}{z} \Rightarrow u = -\frac{fx}{z}; v = -\frac{fy}{z}$$



- I : Piano immagine
- C : Centro ottico
- P : Punto 3D della scena
- p : Punto 2D proiettato
- f : Lunghezza focale

Figura 2.1.1 – Proiezione prospettica

È evidente che in questa proiezione si ha una perdita di informazione, in quanto allo stesso punto immagine corrispondono infiniti punti della scena, in particolare si perde l'informazione legata alla profondità. È però possibile risalire alla posizione del punto con un'ulteriore vista, si parla in questo caso di visione binoculare o stereoscopica (Figura 2.1.2).

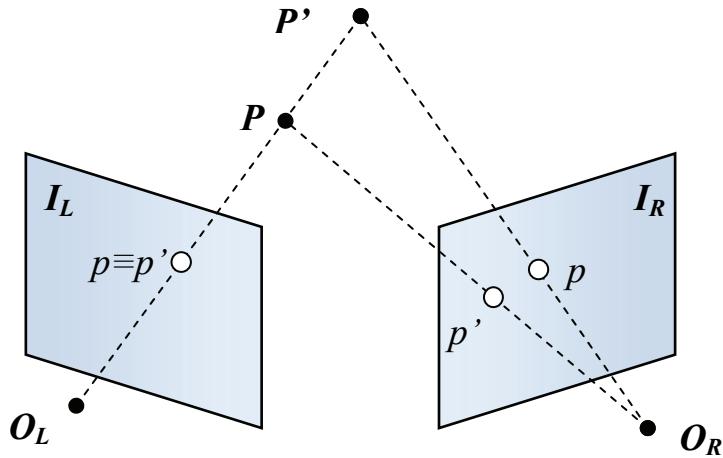


Figura 2.1.2 – Principio di visione stereoscopica

Nota la proiezione di uno stesso punto su entrambe le viste è possibile calcolarne la posizione 3D mediante triangolazione. A tal fine è necessario stabilire la corrispondenza tra le proiezioni di un punto sulle due immagini (punti omologhi). Sfruttando un particolare vincolo, denominato vincolo epipolare, è possibile limitare la ricerca di punti omologhi in uno spazio monodimensionale, ovvero lungo le rette epipolari. Tuttavia prima di procedere con la ricerca dei punti omologhi è necessario determinare alcuni parametri del sistema, in particolare come i punti vengono mappati nelle immagini e la posizione nello spazio delle due camere, questa procedura prende il nome di calibrazione.

2.1.1 Calibrazione e rettificazione

La procedura di calibrazione consente di definire in modo completo un sistema stereo determinando i parametri intrinseci ed estrinseci. I parametri intrinseci descrivono il modo in cui un punto della scena viene mappato in un'immagine e sono la distanza focale, le coordinate del punto principale (intersezione tra piano immagine e retta ortogonale passante per il centro ottico) ed altri parametri che descrivono la forma del pixel o la distorsione delle lenti. I parametri estrinseci descrivono la posizione delle camere rispetto ad un particolare sistema di riferimento (in genere quello di una delle due camere). I parametri estrinseci contengono quindi una traslazione ed una rotazione e permettono il calcolo della distanza tra i centri ottici delle immagini, la *baseline*, parametro fondamentale per la triangolazione. La procedura di calibrazione si effettua generalmente utilizzando un pattern contenente punti di calibrazione le cui coordinate sono note per costruzione, successivamente con tecniche di elaborazione dell'immagine se ne calcola la proiezione. La stima dei parametri è strettamente legata all'accuratezza della misura dei punti del pattern.

La rettificazione è un processo che permette, noti i parametri intrinseci ed estrinseci, di trasformare le immagini in modo da rendere le linee epipolari parallele ad un asse (generalmente l'asse orizzontale). Una coppia di immagini che soddisfano questo vincolo si dicono in forma standard, e risulta molto utile calcolarle per semplificare in modo significativo la ricerca delle corrispondenze (punti omologhi). In Figura 2.1.3 è mostrato un sistema stereo in forma standard.

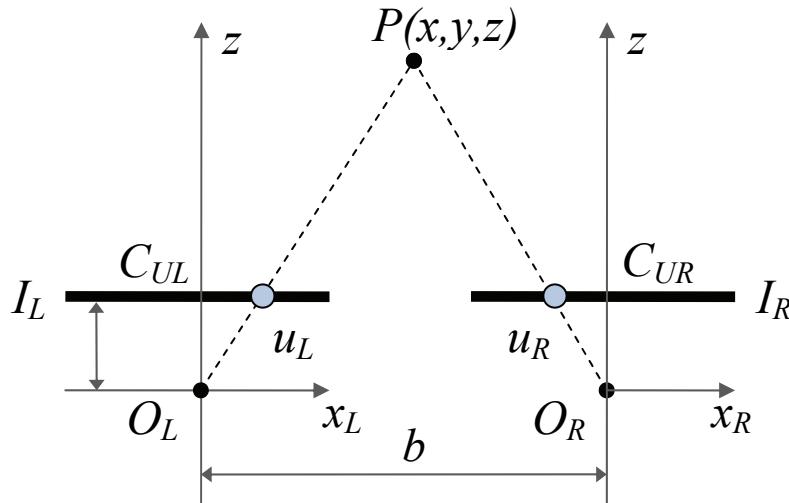


Figura 2.1.3 – Sistema stereoscopico in forma standard

Le equazioni che permettono di triangolare il punto P di coordinate (x,y,z) in un sistema in forma standard sono

$$z = \frac{bf}{d}; x = \frac{z}{f}(u - c_u); y = \frac{z}{f}(v - c_v)$$

In cui d è la disparità misurata in pixel del punto, u e v sono le coordinate del punto immagine mentre c_u e c_v sono le coordinate del punto principale.

2.1.2 Calcolo delle corrispondenze

Una volta ottenuta una coppia di immagini in forma standard, è necessario determinare quali siano i punti omologhi, ovvero quali punti delle immagini sono proiettati dallo stesso punto della scena. Questo problema è noto nel campo della visione artificiale come calcolo delle corrispondenze.

Il calcolo delle corrispondenze porta con sé alcuni problemi intrinseci derivanti dal differente punto di vista:

- **Occlusioni:** esistono punti che vengono proiettati in una sola delle due immagini, in quanto occlusi da un altro punto nell'altra vista. È chiaro che non è possibile associare alcuna corrispondenza ad un punto occluso, e quindi nessuna disparità.

- **Distorsioni fotometriche:** un differente angolo di vista può comportare differenti livelli di illuminazione e colorazione sulle superfici non lambertiane¹ (superficie con componenti riflettenti o trasparenti).
- **Distorsioni prospettiche:** a causa della proiezione prospettica un oggetto non parallelo al piano immagine può assumere dimensioni differenti nelle 2 immagini.

In Figura 2.1.4 a sinistra e al centro vengono evidenziati i problemi appena descritti in una coppia di immagini (già rettificate). In rosso è evidenziata un’occlusione, nell’immagine di sinistra si intravede un oggetto che non compare nell’immagine destra. In verde è mostrata una distorsione fotometrica, in questo caso un riflesso (o highlight). In azzurro è evidenziata una distorsione prospettica, si noti come la stessa parte nell’immagine di sinistra presenti un’area molto minore e un angolo molto più stretto. Questi problemi inoltre diventano più rilevanti al crescere della *baseline*. Esistono sostanzialmente due categorie di algoritmi che risolvono il problema delle corrispondenze: *area-based* o *feature-based*. Gli algoritmi *area-based* considerano un’area generalmente rettangolare (finestra) e ricercano nell’altra vista l’area ad essa più somigliante (lungo le linee epipolari) costruendo una mappa di disparità densa. La stima di similarità tra le due finestre viene calcolata da una funzione opportuna, in genere SAD, SSD o la più robusta NCC. In Figura 2.1.4 a destra viene mostrata la mappa di disparità calcolata con l’algoritmo di corrispondenza SMP [18].

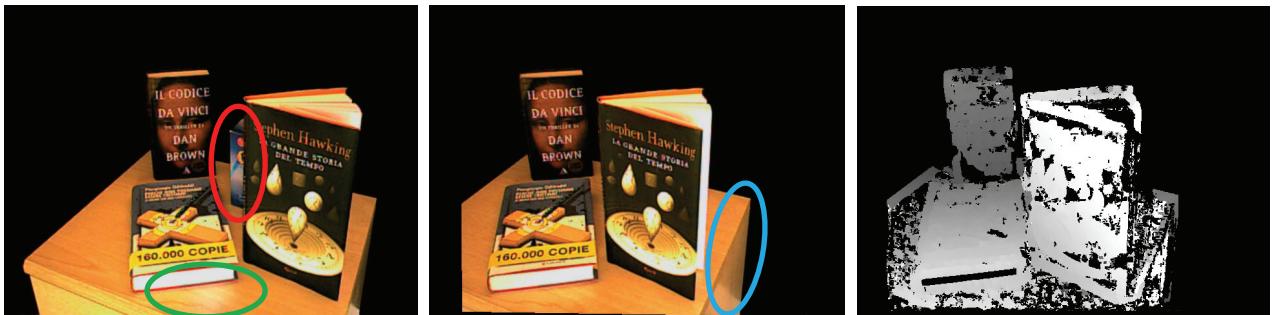


Figura 2.1.4 – Coppia di immagini rettificate e mappa di disparità SMP

Gli algoritmi *feature-based* invece considerano determinate caratteristiche geometriche dell’immagine come angoli, segmenti, curve o blob. Ogni *feature* ha associato un descrittore che ne definisce le caratteristiche salienti, ad esempio una *feature* segmento conterrà informazioni geometriche come lunghezza, angolazione, posizione ma potrà contenere anche informazioni fotometriche come contrasto o colorazione. Vengono poi confrontati i descrittori coerenti con il vincolo epipolare, associandone la disparità. Questi algoritmi generano solitamente mappe di disparità sparse.

2.2 L’algoritmo SIFT

Davide G. Lowe in [9] propone un algoritmo che permette di estrarre da immagini feature invarianti per scala, rotazione e traslazione, e parzialmente invarianti a distorsioni fotometriche e prospettiche (cambi di illuminazione, punto di vista 3D). Le feature rilevate

¹ Una superficie si definisce lambertiana quando irradia la luce in modo uniforme in tutte le direzioni (superficie opaca).

sono altamente distintive, cioè consentono di essere accoppiate allo stesso punto in un'altra immagine con buona probabilità. Questo algoritmo prende il nome di Scale Invariant Feature Transform (SIFT) e si divide in 4 passi computazionali:

- Ricerca degli estremanti nello *scale space*
- Localizzazione dei *keypoint*
- Assegnamento della direzione
- Definizione del descrittore dei *keypoint*

2.2.1 Ricerca degli estremanti nello scale space

La ricerca di punti candidati invarianti per scala viene effettuata in un piramide di immagini ottenute con una procedura di filtraggio in cascata utilizzando la funzione scala (*scale space*). È possibile dimostrare che l'unica funzione che costituisce un kernel per lo *scale space* è la funzione gaussiana. Per cui lo *scale space* di un'immagine è definito come la convoluzione dell'immagine con la funzione Gaussiana

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

$$\text{In cui } I(x, y) \text{ è l'immagine e } G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

L'immagine viene quindi ripetutamente convoluta con Gaussiane a differenti varianze σ^2 per ottenere una serie di livelli nell'ottava corrente, poi una di queste immagini¹ viene sottocampionata ad intervalli regolari per ottenere l'immagine di partenza per l'ottava successiva. I punti salienti vengono poi ricercati in una funzione definita come differenza di Gaussiane (DoG)

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma)$$

In Figura 2.2.2 è mostrato lo *scale space* e le relative mappe calcolate con DoG. È evidente che una volta costruito lo *scale space* è immediato calcolare la funzione DoG (semplici sottrazioni tra immagini). Questa funzione è una buona approssimazione della funzione LoG (Laplaciano di Gaussiana) che produce feature più stabili rispetto ad altre funzioni come gradiente, Hessiana o Harris Corner Detector. I punti di interesse sono quindi gli estremanti della funzione DoG, per rilevarli si procede col confronto del punto con i suoi 8 vicini nella stessa immagine, e i suoi 18 vicini nella scala immediatamente superiore ed inferiore, se il punto è maggiore/minore dei suoi 26 vicini verrà classificato come massimo/minimo locale (Figura 2.2.1).

¹ Si seleziona l'immagine con varianza doppia rispetto alla varianza dell'ottava precedente.

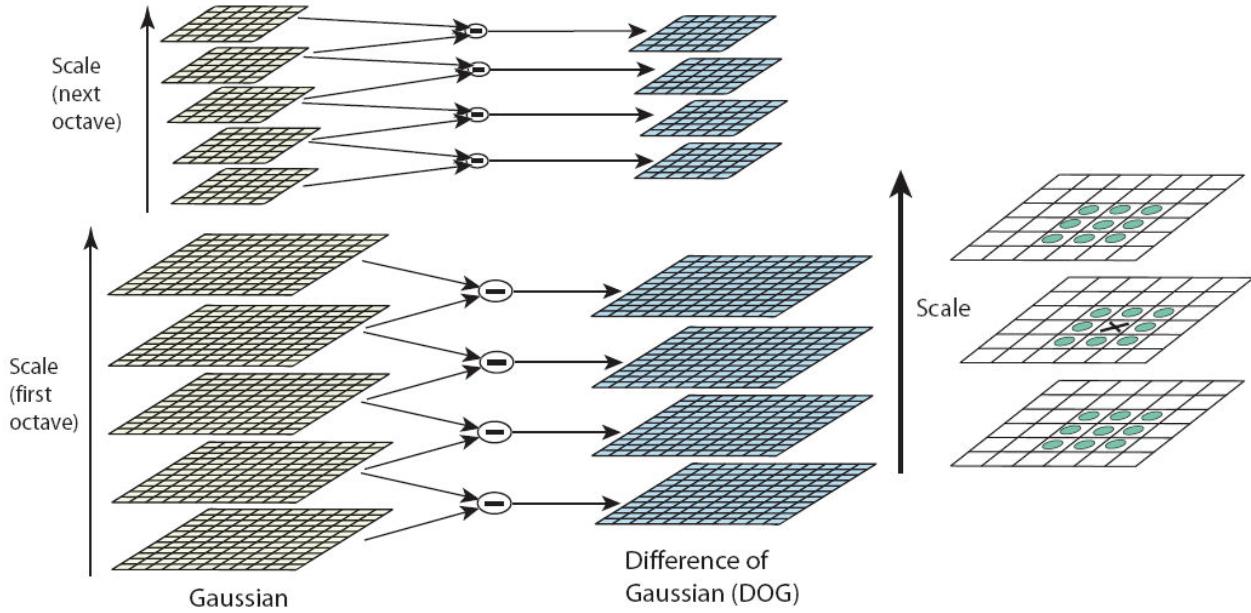


Figura 2.2.2 – Scale space e DoG

Figura 2.2.1 – Ricerca degli estremanti locali

2.2.2 Localizzazione dei keypoint

In questa fase viene testata la stabilità dei candidati keypoint analizzandone il contrasto, rilevando la presenza di bordi e interpolando la posizione per migliorare la precisione.

La posizione del massimo/minimo locale viene calcolata con precisione utilizzando una funzione 3D quadrica che interpola i punti locali vicini, utilizzando lo sviluppo in serie di Taylor della DoG. Si scartano poi tutti i punti a basso contrasto, in quanto maggiormente soggetti ad errori. L'eliminazione dei punti a basso contrasto non è comunque sufficiente, in quanto la DoG ha forti risposte anche in presenza di spigoli. Per questo motivo, mediante lo studio sulla matrice Hessiana della DoG nel punto, si scartando i keypoint con un rapporto delle curvature principali oltre una certa soglia.

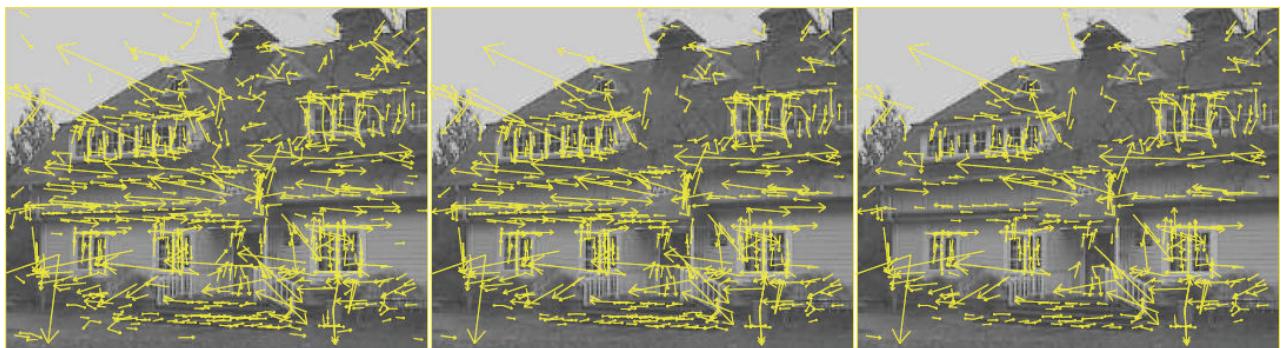


Figura 2.2.3 – Fase di selezione dei keypoint dai possibili candidati

In Figura 2.2.3 è mostrata la fase di selezione dei keypoint, a sinistra sono mostrati tutti gli estremi locali della DoG, al centro i keypoint sopravvissuti al vincolo di contrasto mentre a destra sono evidenziati i keypoint dopo la fase di sogliatura sul rapporto delle curvature principali.

2.2.3 Assegnamento della direzione

Ad ogni keypoint viene associato una direzione basata su alcune proprietà locali dell'immagine. Si utilizza la scala per determinare l'immagine dello scale space più vicina $L(x, y)$ in modo tale da poter operare in modo invariante per scala. A seguito vengono calcolati modulo e direzione del gradiente nel punto

$$m(x, y) = \sqrt{(L(x + 1, y) - L(x - 1, y))^2 + (L(x, y + 1) - L(x, y - 1))^2}$$

$$\theta(x, y) = \tan^{-1} \left(\frac{L(x, y + 1) - L(x, y - 1)}{L(x + 1, y) - L(x - 1, y)} \right)$$

Si costruisce poi un istogramma dalle direzioni dei gradienti dei punti campionati in un'area attorno al keypoint (istogramma a 36 intervalli che copre 360°). Ogni punto campionato aggiunge come contributo il modulo del gradiente pesato (peso Gaussiano sulla finestra), i picchi sull'istogramma determinano le direzioni dominanti dei gradienti. Si considera poi il picco massimo e gli eventuali altri picchi che superino l'80% del massimo (creando quindi un altro keypoint con la stessa posizione, stessa scala ma differente direzione). La direzione viene poi calcolata con un'interpolazione parabolica dei punti vicini per migliorarne la precisione. Un esempio in Figura 2.2.4.

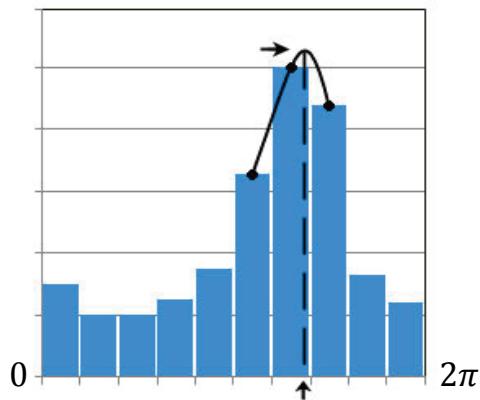


Figura 2.2.4 – Iistogramma delle direzioni

campionato aggiunge come contributo il modulo del gradiente pesato (peso Gaussiano sulla finestra), i picchi sull'istogramma determinano le direzioni dominanti dei gradienti. Si considera poi il picco massimo e gli eventuali altri picchi che superino l'80% del massimo (creando quindi un altro keypoint con la stessa posizione, stessa scala ma differente direzione). La direzione viene poi calcolata con un'interpolazione parabolica dei punti vicini per migliorarne la precisione. Un esempio in Figura 2.2.4.

2.2.4 Definizione del scrittore dei keypoint

A questo punto ad ogni keypoint è stata associata una scala, una posizione e una direzione. Occorre ora assegnare un descrittore opportuno che renda il keypoint il più possibile invariante a distorsioni fotometriche e prospettiche.

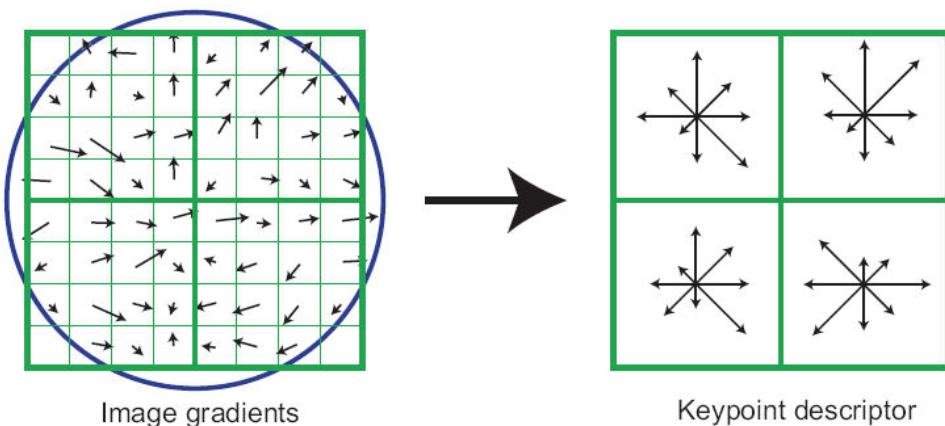


Figura 2.2.5 – Descrittore di un keypoint

Il descrittore di un keypoint viene calcolato selezionando dimensione e direzione dei gradienti dei punti campionati vicini (pesati su finestra Gaussiana), e si costruisce l’istogramma delle direzioni per sottoregioni. A sinistra della Figura 2.2.5 è rappresentata la fase di campionamento dei gradienti, a destra è mostrato il descrittore. In questo esempio viene calcolato un descrittore 2x2 a partire da una finestra 8x8 di campioni (vettore a 32 dimensioni, 2x2x8), nell’implementazione proposta da Lowe si utilizzano descrittori 4x4 da finestre 16x16 (vettore a 128 dimensioni, 4x4x8). Per evitare bruschi cambi di direzione durante lo scostamento tra un istogramma e l’altro si ricorre ad una interpolazione trilineare.

Infine, per ridurre la sensibilità del descrittore a cambi di illuminazione, il vettore viene normalizzato, in questo modo il descrittore risulta insensibile a cambi proporzionali di illuminazione (una costante aggiunta non comporta alcuna variazione in quanto i gradienti sono calcolati per differenze).

2.2.5 Corrispondenze tra keypoint

In diverse applicazioni è necessario dover ritrovare il punto in un’altra immagine, a questo scopo è importante garantire l’unicità nella corrispondenza tra descrittori. Il miglior candidato per similarità sarà il punto con descrittore più vicino (distanza euclidea). È comunque possibile che un punto non abbia alcuna corrispondenza (e.g., occlusione), risulta quindi necessario scartare le corrispondenze classificate come “deboli”. A questo scopo si considera il rapporto tra la distanza del descrittore più vicino d_1 e quella del secondo più vicino d_2

$$r = \frac{d_1}{d_2}$$

Si considera il descrittore più vicino come corrispondenza se il rapporto r è inferiore ad una certa soglia, cioè se sufficientemente discriminato dal secondo, e di conseguenza da tutti gli altri. La ricerca dei punti vicini richiede confronti su vettori a molte dimensioni (nell’implementazione di Lowe la ricerca è su spazi a 128 dimensioni), dimensione tale da rendere ricerche efficienti come sui kd-tree senza significativi miglioramenti rispetto ad una ricerca esaustiva. A tal proposito Lowe propone un algoritmo di ricerca approssimato chiamato Best-Bin-First (BBF), che punta a trovare il descrittore più vicino con buona probabilità. L’algoritmo BBF utilizza un sistema di ricerca modificato per kd-tree cercando nello spazio delle feature in ordine di distanza dal punto di query e bloccando la ricerca dopo un certo numero di descrittori trovati. Dai risultati sperimentali, su un database di 100.000 keypoint, si è ottenuto un miglioramento sul tempo computazionale di circa 2 ordini di grandezza superiore rispetto alla ricerca esatta, con meno del 5% di perdita della corrispondenza esatta. Una ragione che porta a lavorare bene questo algoritmo (efficiente seppur non esaustivo) è che il discriminante della corrispondenza tra i due descrittori è il rapporto tra le loro distanze, per cui non è necessario risolvere esattamente la ricerca in cui si troverebbero altri descrittori vicini ma con distanze simili.

2.2.6 Esempi pratici

In Figura 2.2.6 è mostrato un esempio di utilizzo di SIFT per un'applicazione di riconoscimento di oggetti. Gli oggetti nella scena rispetto agli oggetti da ricercare sono ruotati e scalati, nell'immagine di destra è evidenziato il riconoscimento. I quadrati più piccoli indicano la corrispondenza tra keypoint, mentre il rettangolo che li racchiude è la trasformazione affine dei contorni dell'immagine campione risolta in base ai keypoint rilevati. Si nota che l'applicazione riconosce gli oggetti correttamente anche in presenza di forti occlusioni.



Figura 2.2.6 – Esempio di riconoscimento di oggetti

In Figura 2.2.7 si mostra un esempio di risoluzione di corrispondenze tra viste diverse di una scena. Si nota la corrispondenza corretta tra punti nonostante la presenza di molteplici distorsioni causate dal differente angolo di inquadratura. Si noti la presenza di occlusioni e distorsioni prospettiche notevoli nella scatola di cd, e distorsioni fotometriche nel libro a destra.

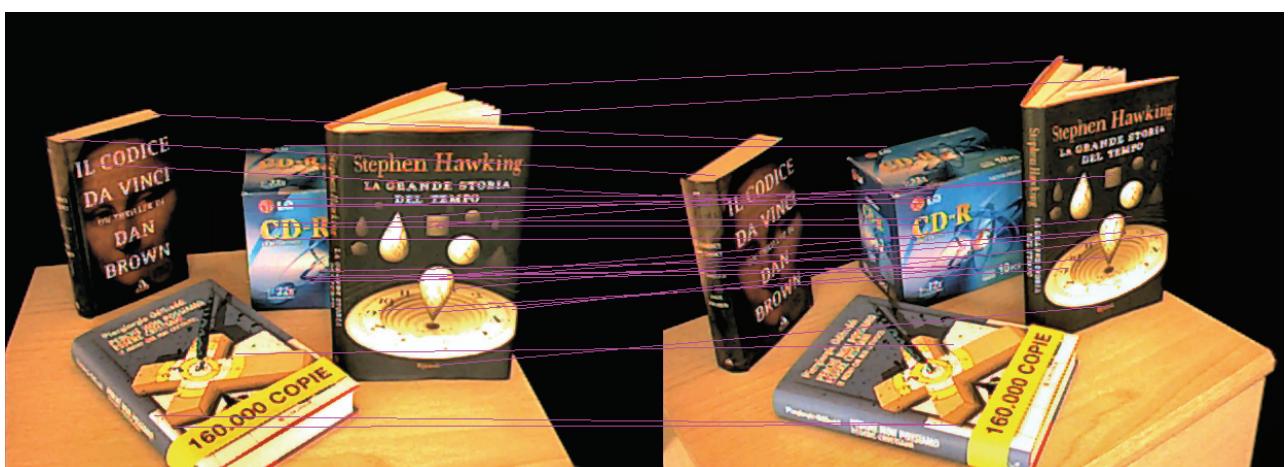


Figura 2.2.7 – Esempio di risoluzione delle corrispondenze tra viste diverse

2.3 Trasformazione tra viste

In questo paragrafo si affronta il problema della determinazione della trasformazione tra due viste, ovvero si cerca la relazione tra due sistemi di riferimento differenti a partire dalle misure di alcuni punti.

2.3.1 Trasformazione indotta da 3 punti

La trasformazione tra due sistemi di riferimento è definita da una traslazione, una rotazione e una scala. Ci sono 3 gradi di libertà per una traslazione e altri 3 per una rotazione, la scala comporta un ulteriore grado di libertà portando il totale a 7. Sono quindi necessari 3 punti non allineati nello spazio 3D che impongono 9 vincoli al sistema (Figura 2.3.1).

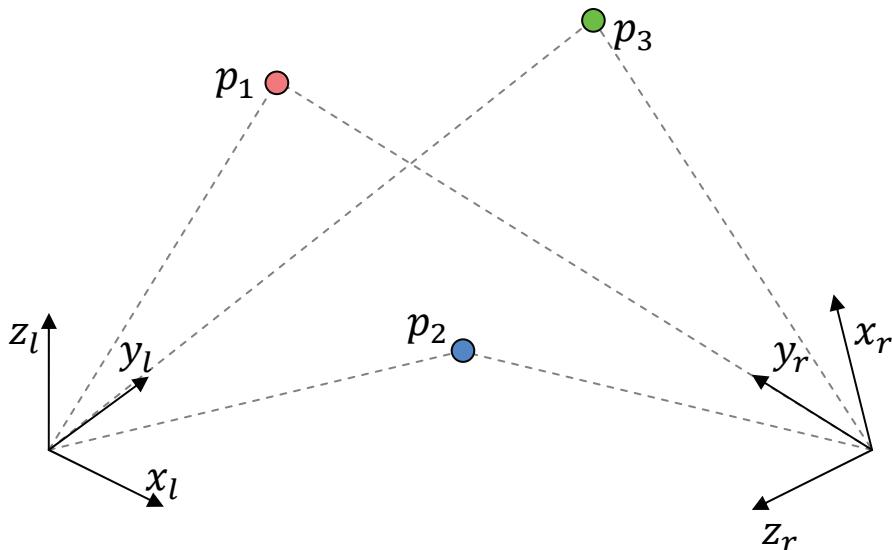


Figura 2.3.1 – Misura di 3 punti in due sistemi di riferimento

Definiamo i due sistemi di riferimento come sinistro e destro (l e r), dati 3 punti e le rispettive coordinate (p_{l1}, p_{l2}, p_{l3} e p_{r1}, p_{r2}, p_{r3}) si costruiscono due nuovi sistemi di riferimento su questi punti. La trasformazione che porta un sistema nell'altro è la soluzione al problema. Si costruisce il primo sistema di riferimento (e.g., sinistro) centrato nel primo punto nelle relative coordinate. L'asse X si ottiene normalizzando il vettore orientato dal primo punto verso il secondo (Figura 2.3.2).

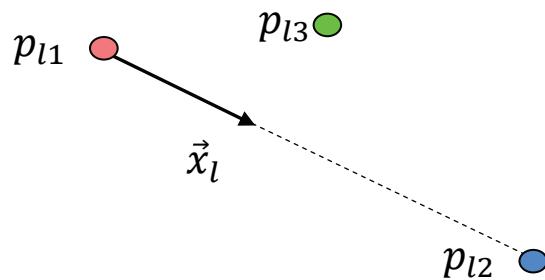


Figura 2.3.2 – Costruzione asse X

Si costruisce quindi il vettore

$$\vec{v}_{xl} = p_{l2} - p_{l1}$$

E lo si normalizza

$$\vec{x}_l = \frac{\vec{v}_{xl}}{\|\vec{v}_{xl}\|}$$

L'asse Y è costruito in modo che il piano XY comprenda il terzo punto. A tal fine si calcola la componente perpendicolare all'asse X del vettore che porta il primo punto nel terzo (Figura 2.3.3). Quindi

$$\vec{v}_{yl} = (p_{l3} - p_{l1}) - [(p_{l3} - p_{l1}) \cdot x_l] x_l$$

Poi si normalizza

$$\vec{y}_l = \frac{\vec{v}_{yl}}{\|\vec{v}_{yl}\|}$$

L'asse Z deve essere ortogonale al piano XY , si costruisce quindi mediante un semplice prodotto vettoriale (il verso è determinato dalla regola della mano destra).

$$\vec{z}_l = \vec{x}_l \times \vec{y}_l$$

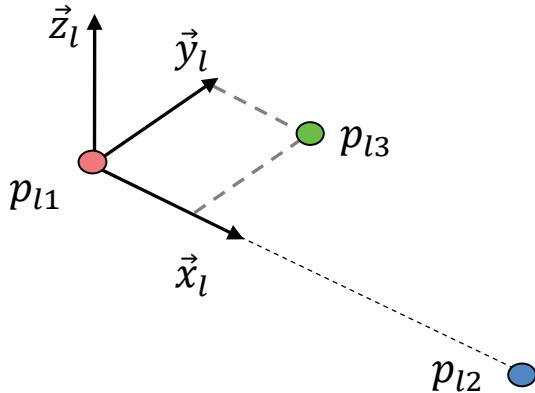


Figura 2.3.3 – Costruzione assi Y e Z

Si procede allo stesso modo nel sistema di riferimento r per la creazione degli assi $(\vec{x}_r, \vec{y}_r, \vec{z}_r)$. Tramite l'unione di questi vettori si ottengono le matrici

$$M_l = |\vec{x}_l \ \vec{y}_l \ \vec{z}_l| \quad e \quad M_r = |\vec{x}_r \ \vec{y}_r \ \vec{z}_r|$$

Dato un punto generico p_{l1} nel sistema di riferimento l , il prodotto di p_{l1} con M_l^T è un punto che ha per componenti le sue proiezioni lungo gli assi. Moltiplicando per M_r si ottiene il vettore mappato nel sistema di coordinate r , quindi

$$v_r = M_r M_l^T v_l$$

La matrice di rotazione tra i due sistemi di riferimento sarà dunque

$$R = M_r M_l^T$$

La matrice R è ortonormale in quanto sia M_l che M_r lo sono per costruzione.

Si è definito un metodo per risolvere in forma chiusa la trasformazione tra due sistemi di riferimento a partire da 3 punti. Tuttavia in molte applicazioni pratiche si hanno a disposizione molti più punti, e certamente non esenti da errore. È opportuno utilizzare l'intero contenuto informativo al fine di minimizzare tale errore. Nei paragrafi 2.3.3 e 2.3.4 vengono presentati due diversi approcci alla risoluzione del problema, il primo iterativo ed il secondo ai minimi quadrati. Nel paragrafo 2.3.5 si procede al confronto dei metodi, ma prima di tutto è necessario introdurre l'entità matematica utilizzata per rappresentare le rotazioni: i quaternioni (paragrafo 2.3.2).

2.3.2 Quaternioni

I quaternioni (Hamilton, 1843) sono entità introdotte come estensione dei numeri complessi e descrivono spazi vettoriali a 4 dimensioni (una parte reale e tre parti immaginarie). Vengono utilizzati per descrivere rotazioni su spazi tridimensionali in quanto il calcolo risulta molto più stabile rispetto all'impiego di matrici di rotazione o angoli di Eulero¹. Si considera il seguente quaternione come rotazione di un angolo θ attorno all'asse v . I quaternioni si indicano con un puntino sopra la lettera.

$$\dot{q} = [w, x, y, z] = \left[\cos \frac{\theta}{2}, v \left(\sin \frac{\theta}{2} \right) \right]$$

Con v vettore unitario di componenti $[ax, ay, az]$, quindi

$$w = \cos \frac{\theta}{2}; x = ax \sin \frac{\theta}{2}; y = ay \sin \frac{\theta}{2}; z = az \sin \frac{\theta}{2}$$

In Figura 2.3.4 è mostrato il quaternione \dot{q} nella forma asse/angolo.

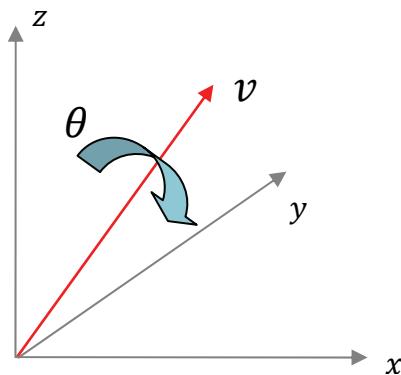


Figura 2.3.4 – Quaternione

¹ L'impiego degli angoli di Eulero (e.g., roll, pitch e yaw) può portare al fenomeno del Gimbal Lock, ovvero la perdita di un grado di libertà nella rotazione a causa dell'allineamento tra 2 assi.

Dato un quaternione è possibile ricavare la relativa matrice di rotazione.

$$M = \begin{bmatrix} 1 - 2y^2 - 2z^2 & 2xy - 2zw & 2xz + 2yw & 0 \\ 2xy + 2zw & 1 - 2x^2 - 2z^2 & 2yz - 2xw & 0 \\ 2xz - 2yw & 2yz + 2xw & 1 - 2x^2 - 2y^2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Analogamente da una matrice di rotazione si può calcolare il relativo quaternione.

$$w = \frac{1}{2} \sqrt{M_{00} + M_{11} + M_{22} + M_{33}}$$

$$x = \frac{M_{21} - M_{12}}{4w} \quad y = \frac{M_{02} - M_{20}}{4w} \quad z = \frac{M_{10} - M_{01}}{4w}$$

La norma di un quaternione si calcola come la radice del prodotto scalare del quaternione per sé stesso. La composizione di rotazioni si ottiene mediante il prodotto vettoriale tra quaternioni (non commutativo).

2.3.3 Algoritmo iterativo

L'idea alla base di questo approccio è estrarre iterativamente un certo numero di terne di punti, calcolarne le trasformazioni e determinare la migliore. Si estraе una terna di punti mediante un generatore di numeri casuale in modo che i punti siano non eccessivamente allineati. Il fattore di allineamento è determinato mediante il prodotto scalare dei vettori normalizzati costruiti sui punti, in Figura 2.3.5 è mostrato un esempio di controllo di allineamento con soglia s e origine in p_1 .

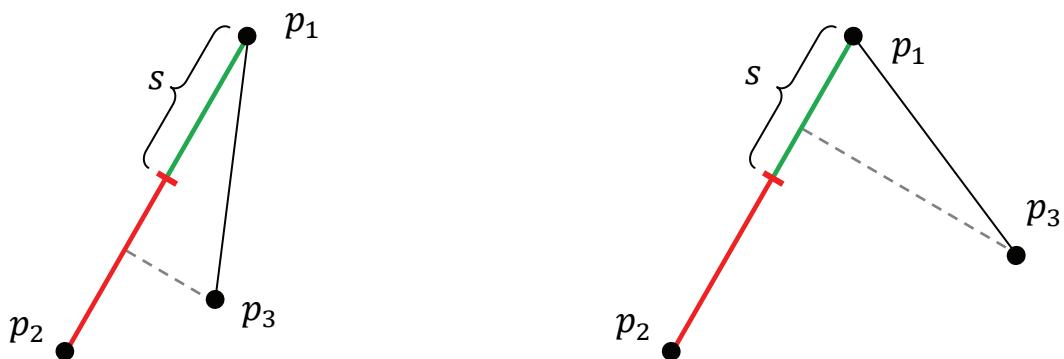


Figura 2.3.5 – Controllo di allineamento, terna scartata (sinistra) e terna valida (destra)

La terna migliore viene selezionata in base ad uno dei seguenti approcci:

- Rotazione media
- Criterio di similarità (SAD o SSD) sulla matrice media

Per trovare la rotazione media si calcola la media dei quaternioni, mentre nella misura di similarità si mediano le matrici di rotazione e si determina quale di queste é a distanza minima (calcolata con la funzione SAD o SSD). In Figura 2.3.6 sono riportati i risultati dell'applicazione di questo algoritmo su un campione di 5000 punti a cui è stato applicato del rumore Gaussiano. I diversi criteri vengono valutati in base al numero di iterazioni, in cui per ogni criterio vengono estratte le stesse terne. L'errore viene calcolato come distanza euclidea tra il quaternione trovato e il quaternione di riferimento (cioè privo di errore).

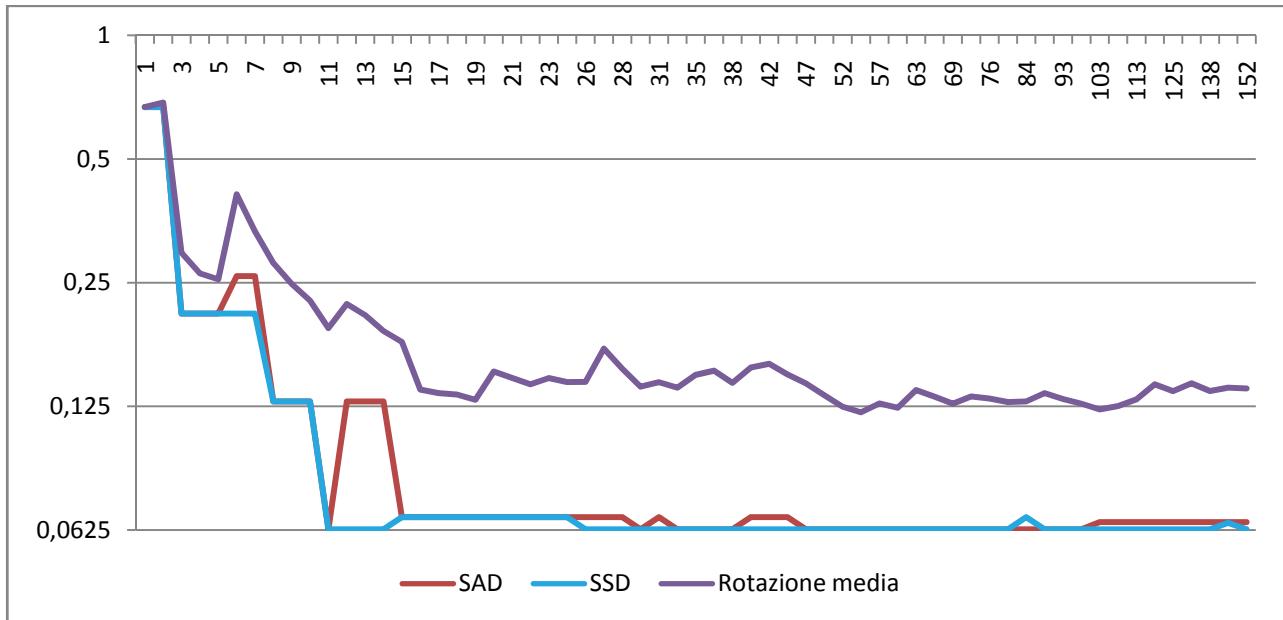


Figura 2.3.6 – Errore calcolato in base al numero di iterazioni

Dai risultati ottenuti è evidente che SSD é il criterio che garantisce maggior precisione e stabilità.

2.3.4 Algoritmo ai minimi quadrati

Berthold K. P. Horn in [11] propone una soluzione in forma chiusa con un metodo ai minimi quadrati. Questo metodo utilizza i quaternioni unitari (cioè a norma unitaria) per risolvere la rotazione.

L'algoritmo mira a minimizzare l'errore residuo a seguito della trasformazione, quindi dati n punti nei sistemi di riferimento sinistro e destro (p_{li} e p_{ri}), si calcolano traslazione t , rotazione R e scala s al fine di minimizzare l'errore

$$\sum_{i=1}^n \|e_i\|^2 \quad \text{con } e_i = p_{ri} - sR(p_{li}) - t$$

Risulta conveniente riferire tutte le misure con il relativo baricentro nell'origine

$$c_l = \frac{1}{n} \sum_{i=1}^n p_{li}; \quad c_r = \frac{1}{n} \sum_{i=1}^n p_{ri}; \quad p'_{li} = p_{li} - c_l; \quad p'_{ri} = p_{ri} - c_r$$

L'errore residuo diventa quindi

$$e_i = p'_{ri} - sR(p'_{li}) - t' \quad \text{in cui} \quad t' = t - c_r + sR(c_l)$$

E la somma dei quadrati degli errori

$$\sum_{i=1}^n \|p'_{ri} - sR(p'_{li}) - t'\|^2 = \sum_{i=1}^n \|p'_{ri} - sR(p'_{li})\|^2 - 2t' \underbrace{\sum_{i=1}^n [p'_{ri} - sR(p'_{li})]}_{=0} + n\|t'\|^2$$

La sommatoria al centro è nulla in quanto le misure sono riferite al baricentro. La prima sommatoria non dipende dalla traslazione t' , per cui rimane da minimizzare solo la quantità non negativa $n\|t'\|^2$, minima quando t' è nullo, ossia

$$t = c_r - sR(c_l)$$

Si conclude che la traslazione ottimale è la differenza tra il baricentro di sinistra e il baricentro del sistema di destra opportunamente scalato e ruotato. Per determinare la scala si considera che il modulo di un vettore rimane invariato a seguito di una rotazione, cioè

$$\|R(p'_{li})\|^2 = \|p'_{li}\|^2$$

L'errore da minimizzare diventa

$$\sum_{i=1}^n \|p'_{ri}\|^2 - 2s \sum_{i=1}^n p'_{ri} R(p'_{li}) + s^2 \sum_{i=1}^n \|p'_{li}\|^2 = S_r - 2sD + s^2 S_l$$

In cui S_l e S_r sono la somma dei quadrati dei moduli dei punti relativi al baricentro, mentre D è la somma dei prodotti scalari dei punti del sistema di destra con i punti ruotati del sistema di sinistra. Completando il quadrato, si ottiene

$$\left(s\sqrt{S_l} - \frac{D}{\sqrt{S_l}} \right)^2 + \frac{S_l S_r - D^2}{S_l}$$

Il secondo termine non dipende dalla scala s , per cui il minimo si ottiene con $s = D/S_l$, quindi

$$s = \frac{\sum_{i=1}^n \vec{p}'_{ri} R(\vec{p}'_{li})}{\sum_{i=1}^n \|\vec{p}'_{li}\|^2}$$

La rotazione viene espressa come il seguente prodotto¹

$$\dot{p}' = \dot{q} \dot{p} \dot{q}^*$$

In cui \dot{p} è il quaternione che rappresenta il punto (parte reale nulla), e \dot{q}^* è il coniugato di \dot{q} .

¹ Ottenuto mediante la formula di rotazione di Rodrigues

La rotazione migliore si ottiene massimizzando la seguente sommatoria

$$\sum_{i=1}^n (\dot{q} \dot{p}'_{li} \dot{q}^*) \cdot \dot{p}'_{ri} = \sum_{i=1}^n (\dot{q} \dot{p}'_{li}) \cdot (\dot{p}'_{ri} \dot{q})$$

I prodotti della sommatoria possono essere riscritti nella forma matriciale

$$\dot{q} \dot{p}'_{li} = \begin{bmatrix} 0 & -x'_{li} & -y'_{li} & -z'_{li} \\ x'_{li} & 0 & z'_{li} & -y'_{li} \\ y'_{li} & -z'_{li} & 0 & x'_{li} \\ z'_{li} & y'_{li} & -x'_{li} & 0 \end{bmatrix} \dot{q} = \bar{\mathbb{R}}_{li} \dot{q} \text{ e } \dot{p}'_{ri} \dot{q} = \begin{bmatrix} 0 & -x'_{ri} & -y'_{ri} & -z'_{ri} \\ x'_{ri} & 0 & -z'_{ri} & y'_{ri} \\ y'_{ri} & z'_{ri} & 0 & -x'_{ri} \\ z'_{ri} & -y'_{ri} & x'_{ri} & 0 \end{bmatrix} \dot{q} = \mathbb{R}_{ri} \dot{q}$$

Di conseguenza la somma diventa

$$\sum_{i=1}^n (\bar{\mathbb{R}}_{li} \dot{q}) \cdot (\mathbb{R}_{ri} \dot{q}) = \dot{q}^T \left(\sum_{i=1}^n \bar{\mathbb{R}}_{li}^T \cdot \mathbb{R}_{ri} \right) \dot{q} = \dot{q}^T \left(\sum_{i=1}^n N_i \right) \dot{q} = \dot{q}^T N \dot{q}$$

In cui $N_i = \bar{\mathbb{R}}_{li}^T \mathbb{R}_{ri}$ e $N = \sum_{i=1}^n N_i$. Ogni matrice N_i è simmetrica, quindi lo è anche N . Si introduce ora una matrice i cui elementi sono le somme dei prodotti delle coordinate nei due sistemi di riferimento. Questa matrice contiene tutte le informazioni per risolvere il problema ai minimi quadrati.

$$M = \sum_{i=1}^n p'_{li} {p'}_{ri}^T = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix}$$

La matrice N diventa quindi

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

Che è una matrice simmetrica a traccia nulla, il quaternione unitario che massimizza $\dot{q} \dot{p} \dot{q}^*$ è l'autovettore corrispondente all'autovalore più grande della matrice N . Il calcolo degli autovalori di N richiede il calcolo degli zeri dell'equazione di quarto grado $\det(N - \lambda I)$, a tal fine esistono metodi specifici di risoluzione (e.g., metodo di Ferrari).

2.3.5 Confronto dei metodi

La comparazione dei metodi si è effettuata su un campione di 5000 punti a cui è stato applicato del rumore Gaussiano. La stima di precisione sulla rotazione viene calcolata in base alla distanza euclidea tra il quaternione trovato e il quaternione di riferimento ottenuto dal modello privo di errori. In Figura 2.3.6 è mostrato l'errore sul modello al variare dell'intensità del rumore, come algoritmo iterativo di riferimento è stato scelto SSD che ha mostrato la maggior precisione.

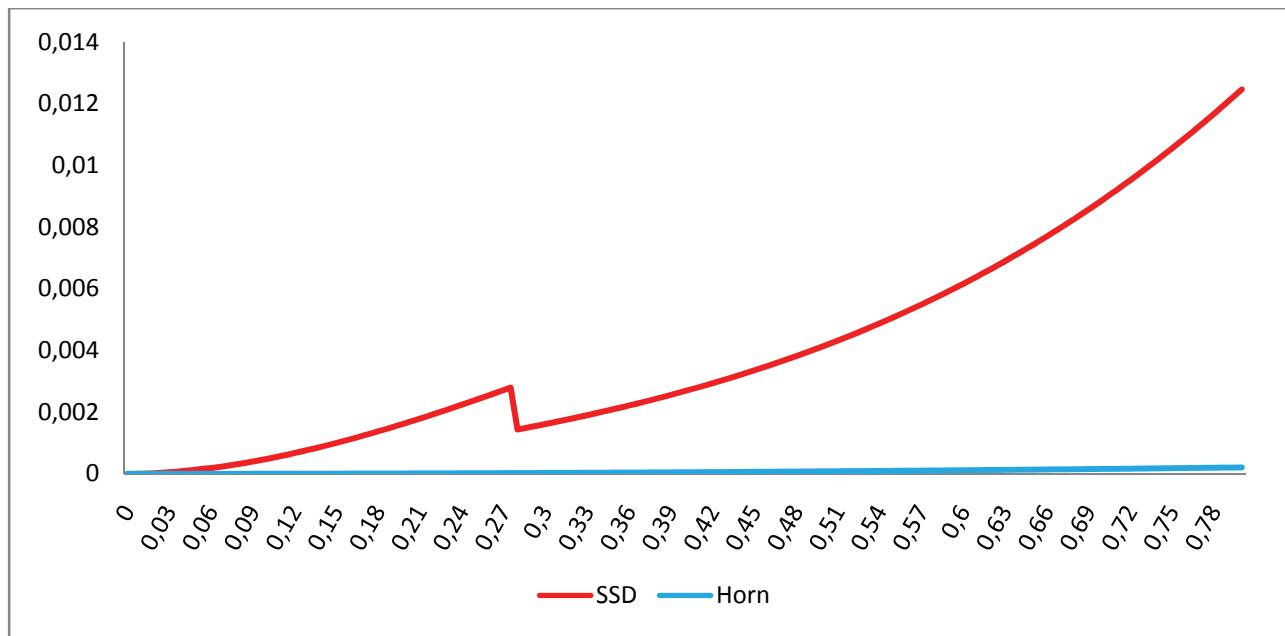


Figura 2.3.7 – Errore calcolato sui 2 metodi in base al rumore

È evidente che l'algoritmo ai minimi quadrati proposto da Horn ottiene una precisione maggiore rispetto a quelli iterativi, con tempi di calcolo inferiori in quanto non necessita di iterazioni.

3 - Architettura del sistema

In questo capitolo si analizza l'intera architettura del sistema da noi progettato. Una panoramica generale è illustrata nel paragrafo 3.1, mentre nei paragrafi successivi sono trattate nel dettaglio le principali fasi di elaborazione. Nel paragrafo 3.2 si descrive la fase di acquisizione assieme ad alcuni algoritmi implementati che risolvono il problema delle corrispondenze. Nel paragrafo 3.3 si esamina la fase di ricostruzione in cui si stima in modo robusto il movimento delle camere per ottenere la nuvola di punti finale. Nel paragrafo 3.4 si analizza l'ultima fase del sistema in cui la nuvola di punti viene opportunamente semplificata e filtrata per ottenere il modello finale.

3.1 Panoramica del sistema

Sono previste 3 fasi fondamentali nel sistema:

- Acquisizione
- Ricostruzione
- Filtraggio

Nella prima fase vengono acquisite le coppie di immagini da un sistema stereo calibrato, si rettificano e si calcolano le mappe di disparità mediante un algoritmo stereo. Attraverso i parametrici intrinseci ed estrinseci ottenuti dalla calibrazione si ricavano le nuvole 3D per ogni mappa. La stima del moto delle camere viene calcolata dalle corrispondenze SIFT tra coppie di immagini, associando i rispettivi punti 3D e scartando le corrispondenze che inducono maggior errore. Successivamente la stima viene raffinata con un meccanismo di riproiezione iterativo in cui i punti 3D di una vista vengono mappati nella successiva al fine di stabilire nuove corrispondenze. Calcolate le trasformazioni tra le viste si procede con la loro fusione, registrandole in un unico sistema di riferimento. La nuvola di punti viene poi quantizzata e filtrata per rimuovere gli *outlier*. In Figura 3.1.1 è mostrata l'architettura del sistema.

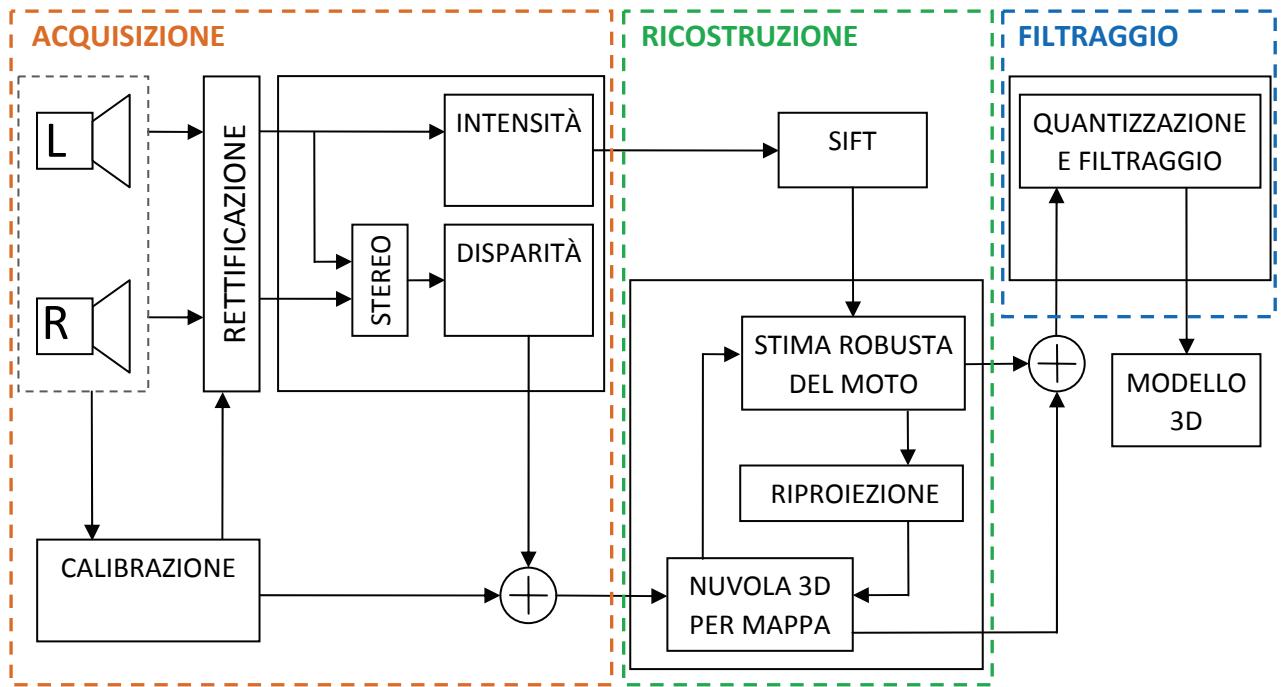


Figura 3.1.1 – Architettura del sistema

3.2 Acquisizione

Nella prima fase del sistema si acquisiscono una serie di immagini da una coppia di camere calibrate. Il movimento delle camere non è soggetto ad alcun vincolo, si possono muovere nello spazio lungo i 6 gradi di libertà (3 traslazioni e 3 rotazioni). Dai parametri intrinseci ed estrinseci ottenuti dalla calibrazione si effettua la rettificazione delle coppie di immagini (si veda paragrafo 2.1.1) e si procede con il calcolo delle mappe di disparità. Nel sistema sono integrati 2 algoritmi che risolvono il problema delle corrispondenze: SMP e Fast Aggregation.

SMP (Single Matching Phase, Di Stefano et al. [18]) è un algoritmo area-based di tipo correlativo molto veloce mentre Fast Aggregation (Tombari et al. in [19]) è un algoritmo area-based basato sulla segmentazione dell’immagine. In Figura 3.2.1 è mostrato un esempio di elaborazione di una coppia di immagini sintetiche già rettificate.

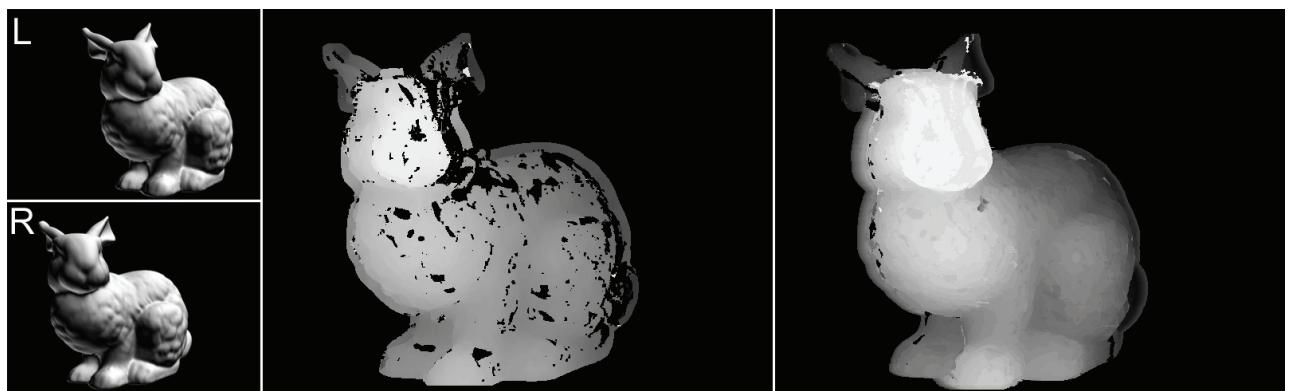


Figura 3.2.1 – Immagini di partenza (sinistra) e mappe di disparità calcolate con SMP (centro) e Fast Aggregation (destra). Per migliorare la visualizzazione è stato aumentato il contrasto.

Le rispettive nuvole di punti 3D sono riportate in Figura 3.2.2.



Figura 3.2.2 – 3 viste delle nuvole 3D generate da Fast Aggregation (sopra) e SMP (sotto)

3.3 Ricostruzione

Nella fase di ricostruzione si fondono le singole nuvole 3D in modo da ottenere la nuvola della scena completa. Questa è la fase più importante del sistema in cui è necessario calcolare il moto delle camere (definito come trasformazioni T_k , oppure in termini di rotazioni R_k e traslazioni t_k) nelle varie viste in modo da poter allineare correttamente le singole nuvole 3D. Il moto viene calcolato tra coppie di immagini, ovvero tra un'immagine e la successiva (con K immagini si ottengono $K - 1$ trasformazioni), infine le nuvole vengono riportate allo stesso sistema di riferimento. Per registrare ogni nuvola in un unico sistema di riferimento è necessario applicare in cascata tutte le trasformazioni intermedie. Ad esempio, per riportare la nuvola k al sistema di riferimento della prima vista è necessario applicare in sequenza le trasformazioni inverse $T_{k-1}^{-1}, T_{k-2}^{-1} \dots T_1^{-1}$, Figura 3.3.1.

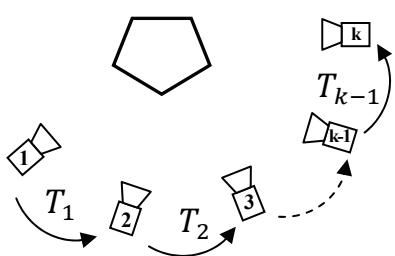


Figura 3.3.1 – Trasformazioni tra viste

Questo implica che un eventuale fallimento nella stima del moto delle camere produce il fallimento di tutte le viste successive. Una stima iniziale del moto viene calcolata mediante le corrispondenze SIFT tra coppie di immagini. Dalle corrispondenze 2D-2D, con l'informazione di disparità, si associano i relativi punti 3D. La trasformazione tra i due sistemi di riferimento viene calcolata con l'algoritmo ai minimi quadrati di Horn (si veda paragrafo 2.3.4). Nelle corrispondenze 3D sono tuttavia presenti una certa quantità di *outlier* che se inclusi nella stima possono

portare al fallimento della registrazione. Per rilevare e rimuovere gli *outlier* sono implementati 2 algoritmi di stima robusta: RANSAC e LWO (paragrafo 3.3.1). Si è poi osservato che la stima del moto può essere resa più accurata disponendo di un numero maggiore di corrispondenze rispetto a quello ottenute dai keypoint SIFT. A questo scopo è stato definito ed implementato un meccanismo di riproiezione originale che mira ad utilizzare il maggior numero di punti del modello al fine di ottenere un progressivo miglioramento della stima del moto (paragrafo 3.3.2).

3.3.1 Stima robusta del moto

Dalle corrispondenze 3D di viste diverse è necessario ricostruire il moto delle camere in modo robusto. Occorre un buon grado di robustezza in quanto nelle corrispondenze sono presenti un certo numero di *outlier* derivanti sia da corrispondenze errate tra keypoint SIFT sia da disparità errate (false corrispondenze individuate nella fase di matching stereo). Nel sistema sono implementati 2 algoritmi che provvedono al filtraggio degli *outlier* e quindi alla stima robusta del moto:

- RANSAC
- LWO

Con l'algoritmo RANSAC (Fischler et al. in [20]) si estraggono un certo numero di volte 3 corrispondenze (il numero minimo per stimare la trasformazione), si determina per ogni trasformazione il numero di corrispondenze che concordano con il modello e si sceglie la migliore. Con l'approccio LWO si stima il modello completo e si procede con l'eliminazione progressiva delle corrispondenze che contribuiscono col maggior errore. Si procede ora con la descrizione dell'algoritmo RANSAC e successivamente LWO per poi definire un criterio di confronto.

L'algoritmo RANSAC (RANDOM SAmples Consensus) si riassume nelle seguenti fasi:

- Selezione del campione e stima del modello
- Determinazione dei punti in accordo al modello
- Scelta del campione migliore

Nella prima fase attraverso un generatore di numeri casuali si sceglie una terna di corrispondenze e se ne calcola la trasformazione. Questa trasformazione è poi applicata all'intero insieme di corrispondenze per determinare i punti che concordano col modello (*Consensus Set*), ovvero i punti che stanno entro una certa distanza. Se la dimensione del *Consensus Set* non è sufficiente a garantire robustezza il processo viene reiterato.Terminate le iterazioni si considera la terna che ha prodotto il *Consensus Set* a dimensione maggiore e si ricalcola la trasformazione dalle sue corrispondenze.

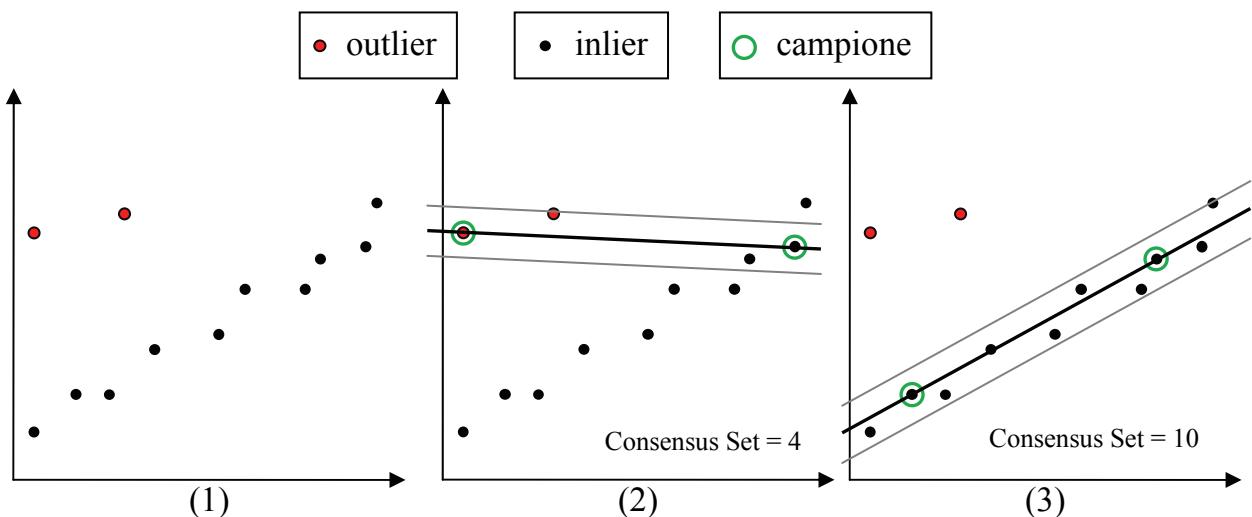


Figura 3.3.2 – Esempio di funzionamento di RANSAC

In Figura 3.3.2, per semplicità, è riportato un esempio di funzionamento di RANSAC per la stima robusta di una retta. L'estensione al caso di trasformazione tra corrispondenze 3D è di facile intuizione. In (1) sono evidenziati i punti in ingresso all'algoritmo, sono marcati in rosso gli *outlier* e in nero gli *inlier*. In (2) viene mostrata un'iterazione in cui nel campione di punti sono presenti degli *outlier*, mentre in (3) viene mostrata l'iterazione che comprende solo *inlier*, la retta in nero è la stima in base ai punti campione.

In un approccio di questo genere è evidente che risulta necessario definire il numero di iterazioni necessarie affinché almeno una di queste coinvolga solamente *inlier*. Viene quindi definito il numero di iterazioni in termini probabilistici

$$N = \frac{\log(1-p)}{\log(1-(1-\epsilon)^s)}$$

In cui p è la probabilità di successo desiderata, ϵ è la percentuale di *outlier* e s è la dimensione del campione. Il numero di iterazioni è quindi strettamente legato alla percentuale di *outlier* e indipendente dal numero di punti. Nel caso (tutt'altro che raro) in cui non sia nota la percentuale di *outlier* è possibile utilizzare un approccio adattivo per la determinazione del numero di iterazioni. In questo caso si tiene traccia del rapporto tra il numero di *inlier* trovati e il numero totale di punti, si considera il rapporto massimo e si aggiorna N ponendo ϵ pari a tale rapporto.

Con l'algoritmo LWO (Leave Worst Out) si considera inizialmente la stima calcolata su tutti i punti (compresi gli *outlier*). Nella fase successiva si scartano le corrispondenze che introducono il maggior errore. L'idea è quindi di scartare il punto che aggiunge l'errore massimo alla stima e ricalcolarla fino ad ottenere il grado di precisione desiderato. Un approccio di questo tipo, tuttavia, comporta un costo computazionale non indifferente, soprattutto in presenza di molti punti (complessità $O(n)$). Si procede quindi scartando la metà dei punti col maggior errore e ricalcolando la stima del modello finché è presente almeno un *outlier*. Se sono presenti solo *inlier* viene reintegrata la metà successiva, al fine di trovare il massimo numero di *inlier* (complessità $O(\log n)$). È comunque prevista una soglia per stabilire se l'insieme attuale di *inlier* è sufficiente a stimare correttamente il modello per ridurre i tempi di calcolo (analogamente a RANSAC).

In Figura 3.3.3 è mostrato un esempio di funzionamento dell'algoritmo nella stima di una retta. In nero sono indicati gli *inlier* e in rosso gli *outlier* che l'algoritmo dovrà riconoscere e scartare. Col cerchietto verde sono indicati i punti utilizzati per stimare il modello mentre la retta in nero è la stima attuale. È riportato anche il grafico delle distanze dei punti dal modello ordinate in modo crescente, la linea rossa rappresenta la soglia che distingue gli *inlier* dagli *outlier*.

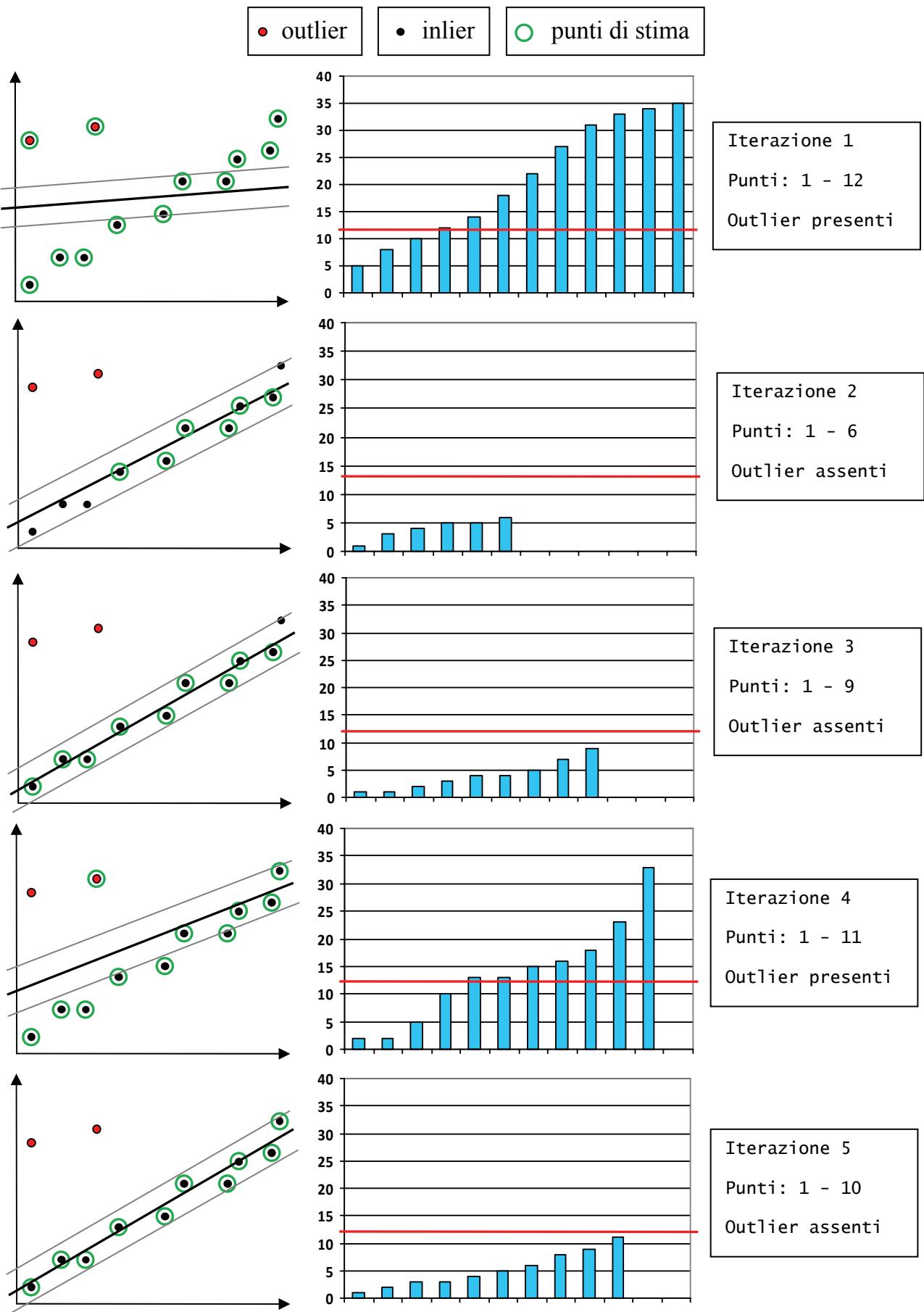


Figura 3.3.3 – Esempio di funzionamento dell'algoritmo LWO

Sono stati effettuati diversi test per determinare quale algoritmo risolve meglio la stima robusta del moto. A tal fine si è generata una nuvola di punti 3D di riferimento (in modo casuale), e una nuvola da stimare applicando a quella di riferimento del rumore. Il rumore aggiunto è generato da una distribuzione Gaussiana a varianza bassa per gli *inlier* e varianza alta per gli *outlier*. La trasformazione di riferimento è quindi la trasformazione identità ($R = I$ e $T = [0 \ 0 \ 0]$). Sono ora riportati 3 grafici a 3 diverse percentuali di *inlier* (10%, 50% e 90%), il criterio di accuratezza si riferisce alla rotazione ed è calcolato come distanza euclidea della matrice trovata rispetto alla matrice identità (grafico in basso), per cui una rotazione sarà tanto più accurata quanto più questa distanza tende a zero. I test sono effettuati in base al numero di punti (espresso in migliaia) e si tiene conto anche del tempo di calcolo (espresso in ms, grafico in alto).

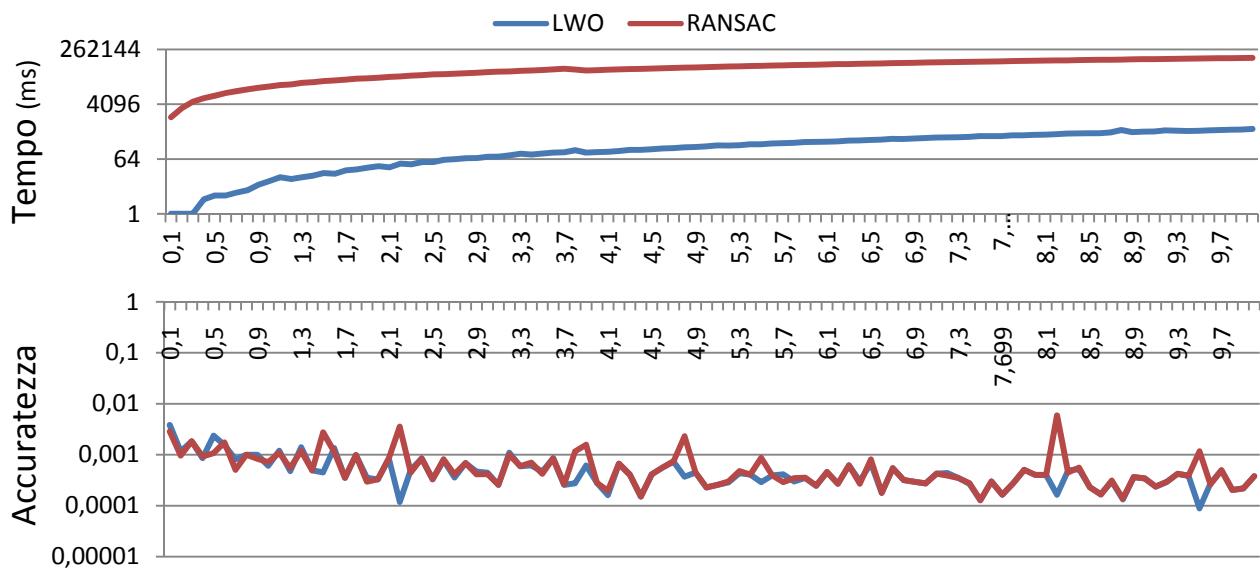


Figura 3.3.4 – Confronto LWO-RANSAC (10% di inlier)

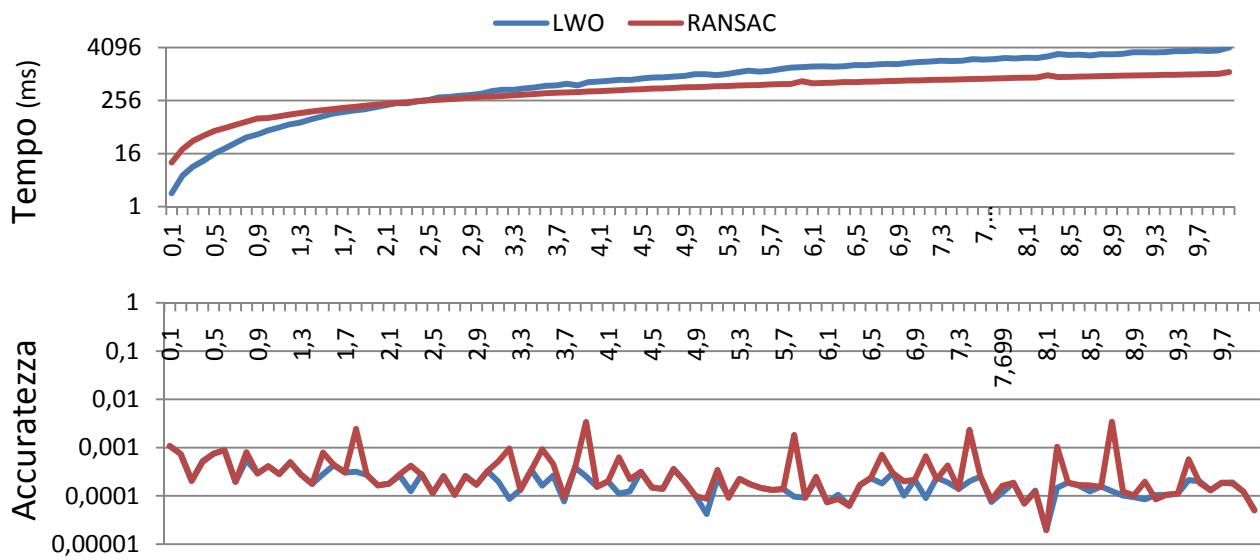


Figura 3.3.5 – Confronto LWO-RANSAC (50% di inlier)

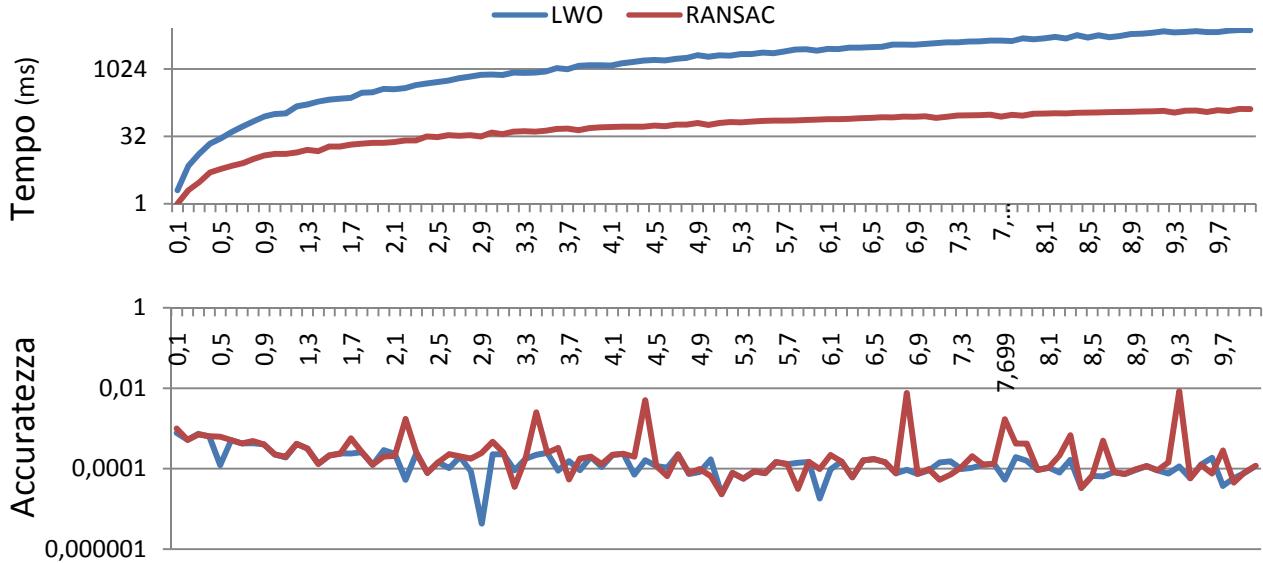


Figura 3.3.6 – Confronto LWO-RANSAC (90% di *inlier*)

Dai risultati ottenuti si deduce che RANSAC lavora bene con molti *inlier* in quanto necessita di poche iterazioni, mentre diventa molto costoso a livello computazionale con basse percentuali di *inlier* (con p fissa al 99%, col 90% di *inlier* servono 4 iterazioni, col 50% ne servono 35 mentre col 10% ne servono 4603). D'altra parte LWO lavora bene con pochi *inlier* in quanto costa poco tagliare grossi blocchi di punti. Il caso peggiore in LWO si ha quindi con $N - 1$ *inlier* su N punti totali. RANSAC lavora in termini probabilistici, il che comporta sempre una determinata probabilità di fallimento di registrazione mentre LWO arriva sempre ad una soluzione. Per quanto riguarda la precisione LWO risponde meglio in presenza di rumore Gaussiano, trovando in tutti i casi la soluzione corretta.

3.3.2 Riproiezione dei punti

Dalla stima robusta del moto ottenuta dalle corrispondenze SIFT 3D si ottiene una trasformazione con una discreta precisione. Tuttavia può non essere ancora sufficiente per una buona registrazione, soprattutto quando si ricostruiscono molteplici mappe. Per migliorare ulteriormente la registrazione è stato sviluppato ed implementato un algoritmo di riproiezione originale.

La stima di trasformazione attuale tra una mappa e la successiva, chiamate I_k e I_{k+1} , viene utilizzata per proiettare un punto di I_k su I_{k+1} , e tramite l'informazione di disparità costruire una nuova corrispondenza (Figura 3.3.7). La nuova corrispondenza si considera valida se la distanza d tra i punti è minore della soglia s utilizzata per la stima robusta del moto, in questo modo si evita l'inclusione di potenziali *outlier*. La riproiezione viene effettuata su tutti i punti del modello n volte in cui ad ogni iterazione si generano nuove corrispondenze. Con le nuove corrispondenze trovate si stima una nuova trasformazione diminuendo progressivamente la soglia s .

Per ottenere una precisione migliore durante la riproiezione, il punto 3D proiettato si calcola mediante l'interpolazione bilineare dei suoi 4 vicini. In Figura 3.3.8 è mostrato un esempio di interpolazione bilineare di P_{proj} dai suoi vicini P_1, P_2, P_3 e P_4 .

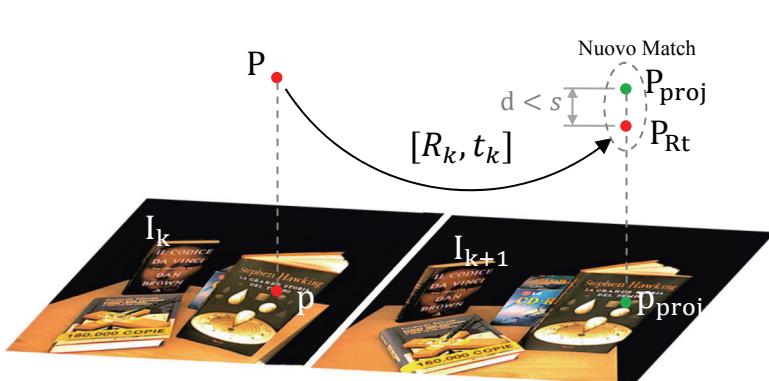


Figura 3.3.7 – Esempio di riproiezione di un punto

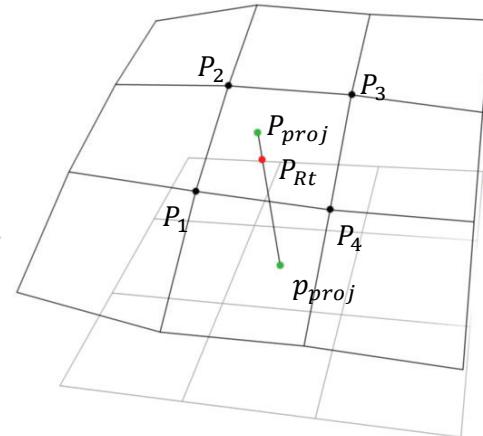


Figura 3.3.8 – Interpolazione bilineare

```

Per  $i = (1..n)$ 
 $[R_k, t_k] \leftarrow$  Stima robusta con soglia  $s$ 
 $\forall$  punto  $p$  in  $I_k$ 
     $P \leftarrow$  coordinate 3D di  $p$ 
     $P_{Rt} \leftarrow$  trasformazione  $[R_k, t_k]$  di  $P$ 
     $p_{proj} \leftarrow$  proiezione su  $I_{k+1}$  di  $P_{Rt}$ 
     $P_{proj} \leftarrow$  coordinate 3D di  $p_{proj}$  con interpolazione bilineare
     $d \leftarrow$  distanza( $P_{Rt}, P_{proj}$ )
    Se ( $d < s$ )
        crea nuovo match tra  $P_{Rt}$  e  $P_{proj}$ 
     $s \leftarrow s/c$ 

```

Figura 3.3.9 – Pseudocodice dell'algoritmo di riproiezione

In Figura 3.3.9 è riportato lo pseudocodice dell'algoritmo che effettua n volte la riproiezione dei punti dell'immagine I_k sull'immagine I_{k+1} , c è il fattore di riduzione della soglia.

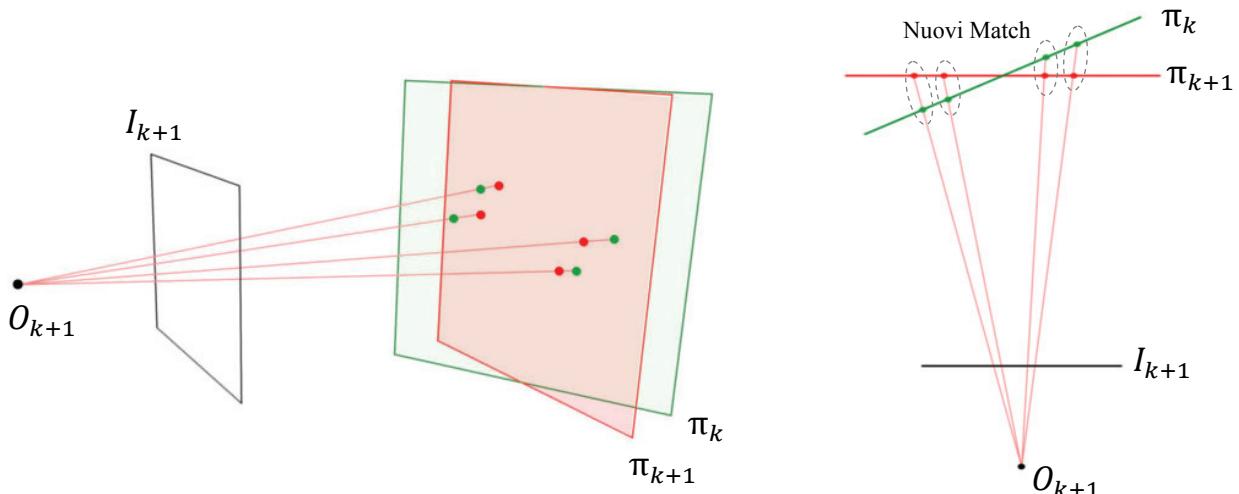


Figura 3.3.9 – Riproiezione dei punti di un piano

In Figura 3.3.9 è mostrato un esempio di riproiezione di alcuni punti di un piano. I punti del piano non correttamente registrato π_k sono proiettati sul piano π_{k+1} . Le nuove corrispondenze trovate sono utilizzate per il calcolo di una nuova stima, al fine di ottenere la convergenza dell'allineamento.

A seguito dell'iterazione finale di riproiezione i punti accoppiati contengono informazioni ridondanti per il modello in quanto classificati come rappresentanti degli stessi punti della scena. La ridondanza viene eliminata scartando un punto per coppia, un esempio è riportato in Figura 3.3.10.

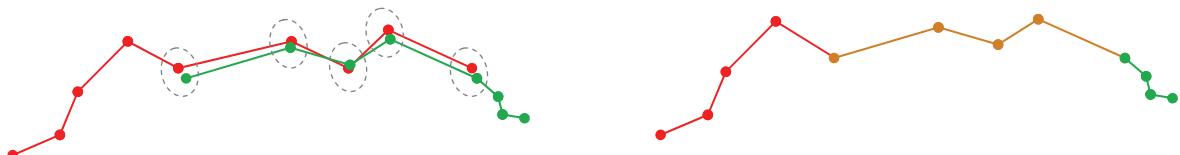


Figura 3.3.10 – Rimozione dell'informazione ridondante nella riproiezione

In Figura 3.3.11 è mostrato un esempio del miglioramento ottenuto dall'applicazione di questo algoritmo.

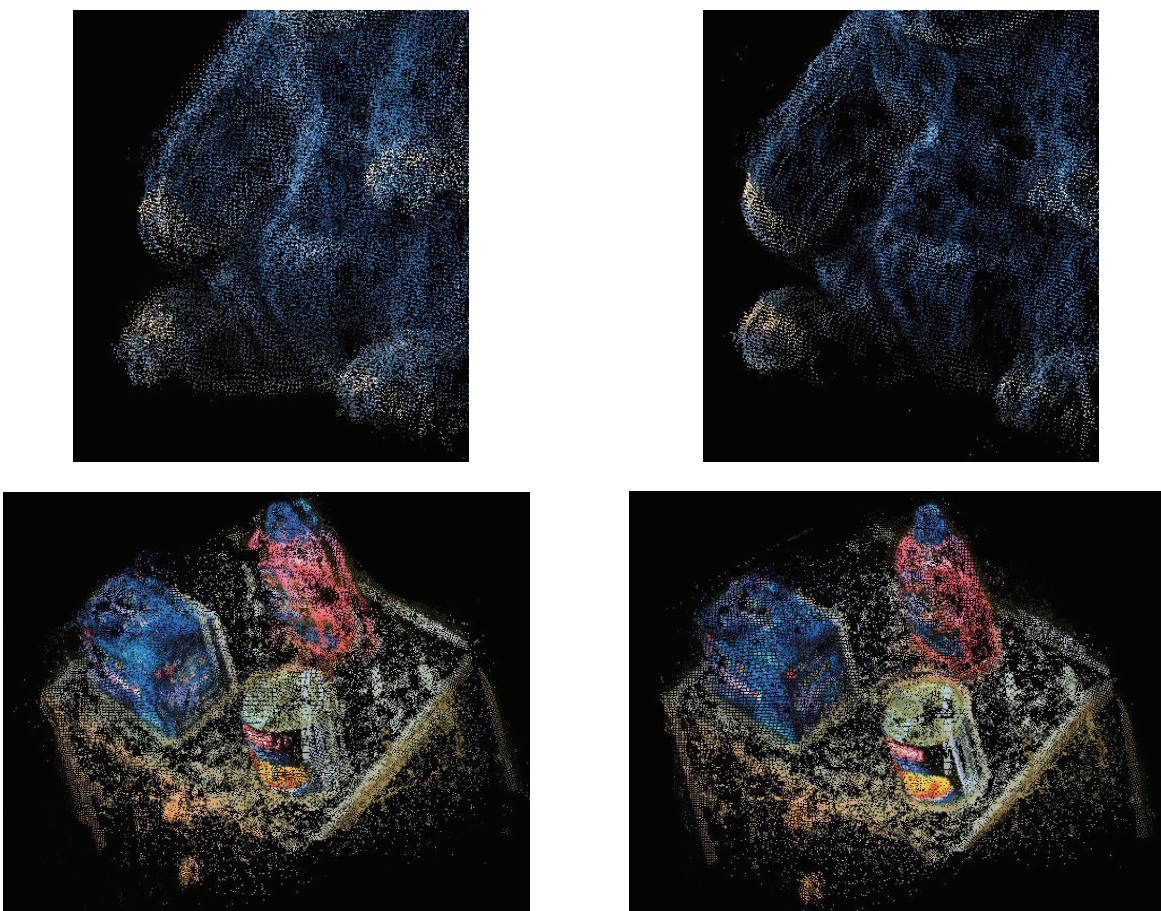


Figura 3.3.11 – Riproiezione assente (sinistra) e presente (destra)

Nelle immagini di sinistra (senza riproiezione) sono ben visibili alcuni errori di allineamento che creano l'effetto di "sdoppiamento" degli oggetti, a destra l'algoritmo di

riproiezione è attivo. Dai risultati sperimentali si è visto che 1 o 2 iterazioni di riproiezione sono sufficienti per stimare correttamente il moto.

L'algoritmo di riproiezione è quindi molto utile per raffinare una stima in quanto estende le corrispondenze al maggior numero possibile di punti del modello (convergenza della stima), ma risulta inefficace in caso di eccessivo errore nella stima iniziale.

3.4 Quantizzazione e filtraggio

Nell'ultima fase del processo di ricostruzione si provvede alla semplificazione della nuvola di punti e all'eliminazione degli *outlier*. Per eseguire queste operazioni in modo efficiente si ricorre all'utilizzo degli Octree.

3.4.1 Octree

Gli Octree sono strutture ad albero in cui ogni nodo ha esattamente 8 figli. Risultano molto comodi nella partizione di spazi tridimensionali. A tal fine si divide l'Octree ricorsivamente in 8 ottanti (Figura 3.4.1) in cui la radice dell'albero è il cubo che contiene la scena. Le foglie (ramo senza ulteriori figli) sono classificate come piene o vuote nel caso contengano o meno dei punti.

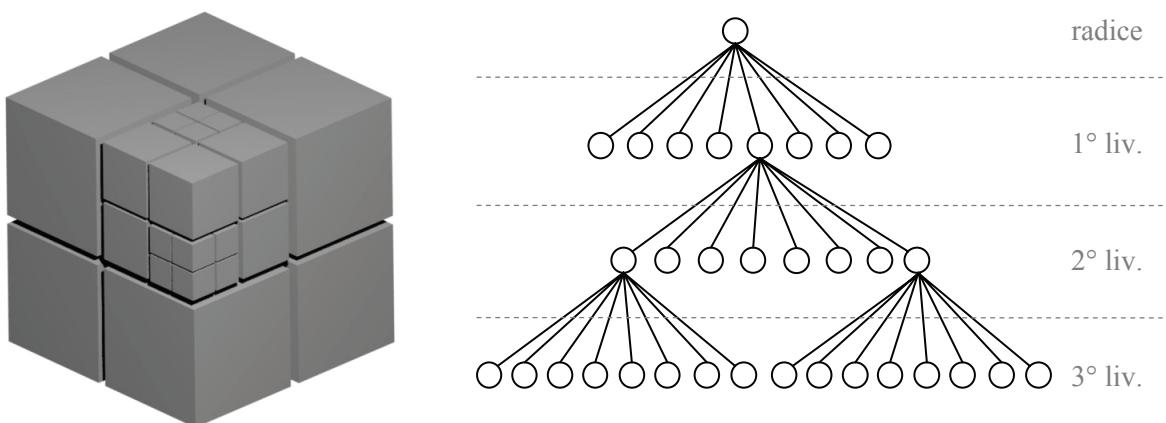


Figura 3.4.1 – Suddivisione di un Octree (sinistra) e relativo albero (destra)

Una struttura di questo tipo contiene tutti i punti del modello e consente di effettuare ricerche al suo interno in modo efficiente. In

Figura 3.4.2 è mostrato per semplicità un esempio di ricerca su un'area circolare di un Quadtree (l'equivalente bidimensionale di un Octree). Si può notare come la divisione in quadranti (equivalenti agli ottanti) permetta una significativa riduzione dell'area di ricerca (meno di 1 / 10 dell'area totale nell'esempio in figura).

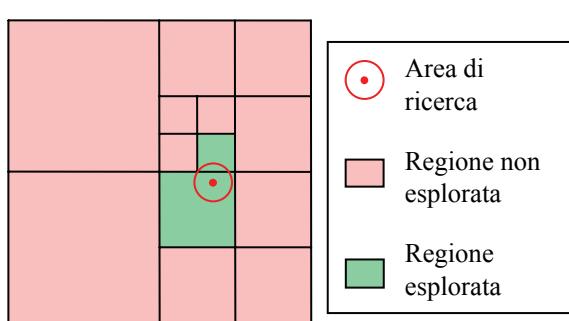


Figura 3.4.2 – Esempio di esplorazione di un Quadtree

3.4.2 Quantizzazione

La prima operazione che viene effettuata mediante l'Octree generato è la quantizzazione spaziale dei punti al fine di semplificare la nuvola 3D. Il modello ricostruito, a questo stadio, contiene diverse regioni in cui sono concentrati molti punti in poco spazio, e spesso contengono informazioni ridondanti. L'obiettivo è quindi uniformare la densità spaziale del modello mediante la quantizzazione dei punti. In Figura 3.4.5 è mostrato per semplicità un esempio di quantizzazione nello spazio 2D, l'estensione al caso tridimensionale è di facile intuizione. Questo processo permette una riduzione del numero di punti, con conseguente vantaggio a livello computazionale per le operazioni successive (filtraggio, meshing, visualizzazione, etc.) e definisce la risoluzione del modello (parametro utile per l'algoritmo di meshing).

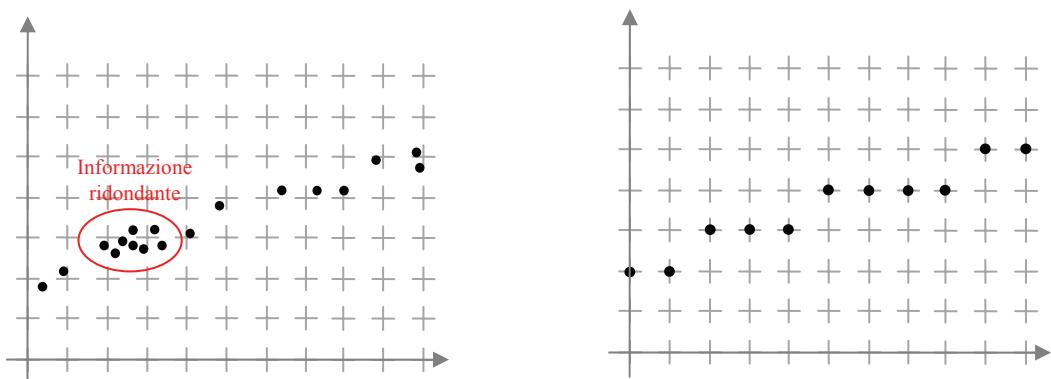


Figura 3.4.5 – Quantizzazione dei punti in uno spazio 2D

3.4.3 Filtraggio

Nel modello, ora quantizzato, sono presenti numerosi *outlier* derivanti dal rumore presente nelle mappe di disparità. Per filtrare gli *outlier* si è implementato un algoritmo che per ogni punto determina la densità in un suo intorno sferico. L'intorno di un punto con densità al di sotto di una certa soglia classifica il punto come *outlier* e viene filtrato. La ricerca mediante Octree risulta particolarmente efficiente perché consente di elidere grosse porzioni di ricerca. In Figura 3.4.6 e Figura 3.4.7 è mostrato un esempio di rimozione degli *outlier*.



Figura 3.4.6 – Fase di filtraggio degli outlier (vista globale). A sinistra la nuvola di punti da filtrare, a destra il risultato

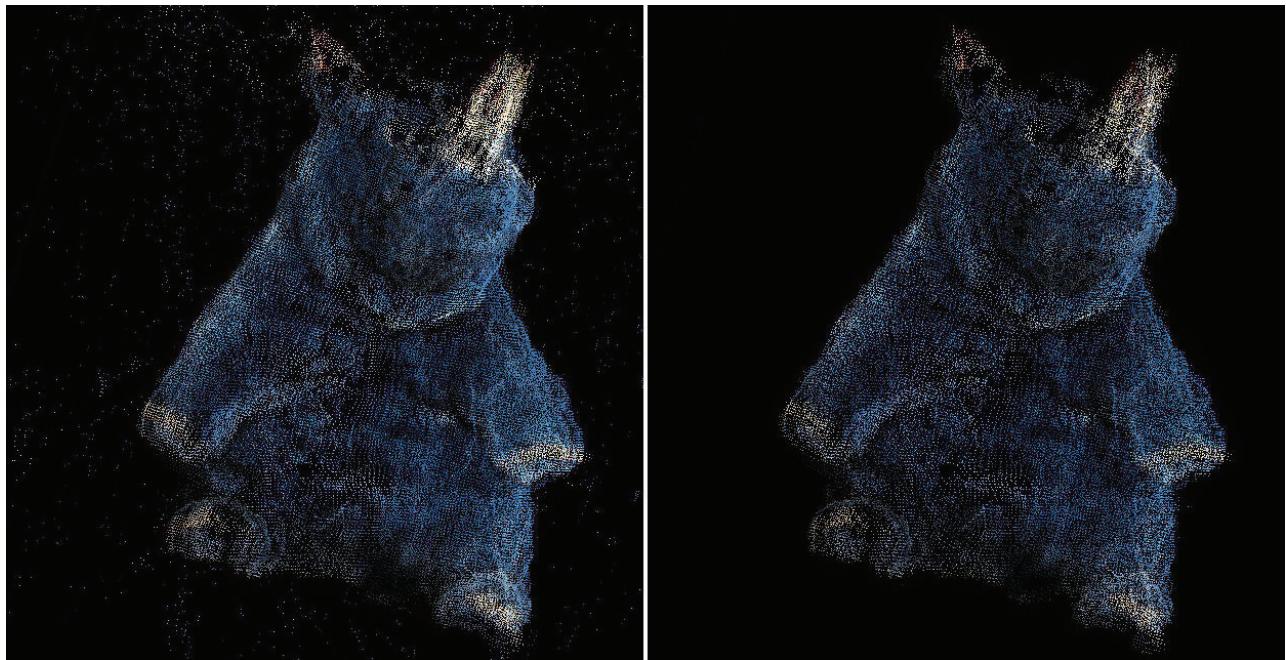


Figura 3.4.7 – Fase di filtraggio degli outlier (vista dettaglio). A sinistra la nuvola di punti da filtrare, a destra il risultato

4 - Risultati sperimentali

In questo capitolo sono presentati alcuni risultati ottenuti dal nostro sistema di ricostruzione 3D. I risultati sono divisi in 2 categorie a seconda della modalità di acquisizione delle immagini:

- Immagini reali
- Immagini sintetiche

Le immagini reali (paragrafo 4.1) sono state acquisite con 2 webcam a basso costo (risoluzione 1.3 MP, 640x480) montate su un treppiede, come mostrato in Figura 3.4.1. La ricostruzione di un singolo oggetto (eliminando il resto della scena) si è ottenuta ponendo l'oggetto su uno sfondo a colore uniforme. Il colore uniforme è identificato dall'algoritmo stereo e successivamente viene rimosso. In Figura 3.4.2 è mostrata l'acquisizione di un oggetto (v. Rhino Dataset) ponendolo su sfondo nero. I parametri intrinseci ed estrinseci sono calcolati tramite le funzioni di calibrazione della telecamera in OpenCV.



Figura 3.4.1 – Sistema di acquisizione

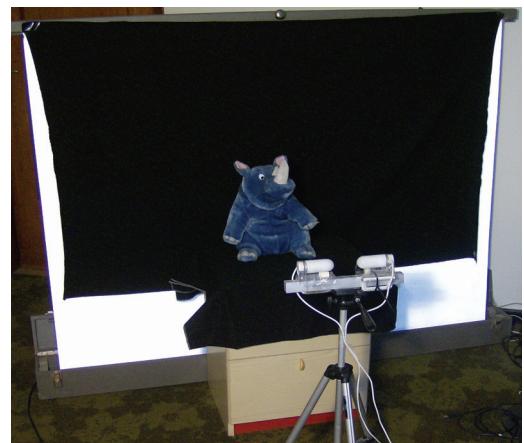


Figura 3.4.2 – Acquisizione di immagini reali (Rhino Dataset)

Le immagini sintetiche sono state prodotte con Blender¹ mediante il rendering di un modello importato. Nella scena da renderizzare sono posizionate 2 telecamere per ottenere le 2 viste, Figura 3.4.1. Le camere sono perfettamente allineate, in questo modo è possibile

¹ Applicazione open source di grafica 3D, <http://www.blender.org/>

saltare la fase di rettificazione mentre parametri intrinseci ed estrinseci sono noti per costruzione.

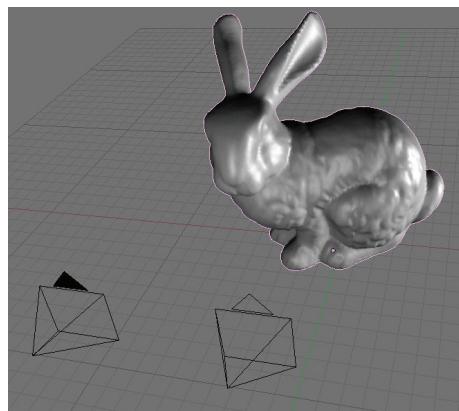


Figura 3.4.1 – Acquisizione di immagini sintetiche con Blender (Stanford Bunny Dataset)

È evidente che il secondo approccio porta ad acquisizioni qualitativamente superiori in quanto esenti dalle seguenti problematiche che sono presenti nelle immagini reali:

- **Limitata qualità dell'ottica/elettronica:** porta ad un conseguente limite della risoluzione e della definizione dell'immagine.
- **Condizioni di illuminazione non ottimali:** possono rendere alcune parti dell'oggetto non chiaramente visibili perché scarsamente illuminate o perché eccessivamente illuminate.
- **Incertezza nei parametri di calibrazione:** induce errori nel posizionamento spaziale dei punti e inficia l'intero processo di ricostruzione (a partire dalla rettificazione).

Di seguito sono riportati i risultati di 5 ricostruzioni ottenute a partire da immagini reali e 5 a partire da immagini sintetiche. Per ogni ricostruzione sono mostrate alcune viste della nuvola di punti 3D. I tempi di calcolo si riferiscono all'esecuzione su un Pentium 4 a 2.66 GHz con 1,00 GB di RAM.

4.1 Immagini reali

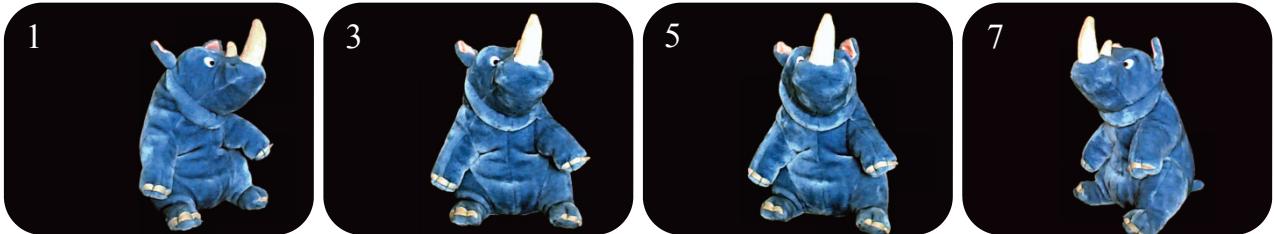
In questo paragrafo sono riportate le seguenti 5 ricostruzioni di oggetti reali:

- Rhino Dataset – Rinoceronte di peluche
- Books Dataset – Tre libri e una scatola di cd sopra un mobile
- Shoe Dataset – Scarpa da ginnastica
- Bunny Dataset – Coniglio di peluche
- Bear Dataset – Orso di peluche

L'algoritmo di stima robusta è stato scelto in base alla quantità di punti e al rapporto tra *inlier* e *outlier*, in base a quanto emerso dai confronti degli algoritmi nel paragrafo 2.3.5. Nei dataset a immagini reali è generalmente preferibile LWO in quanto l'incertezza sulla stima dei parametri intrinseci ed il rumore nelle mappe di disparità produce una consistente percentuale di *outlier*. Nel dataset Rhino, ad esempio, la stima robusta rileva una presenza di *inlier* con percentuali che vanno dal 10% al 25%.

Rhino Dataset	
Risoluzione immagini	640x480
N. immagini	7
Algoritmo stereo	SMP
Algoritmo di stima robusta	LWO
N. iterazioni riproiezione	2

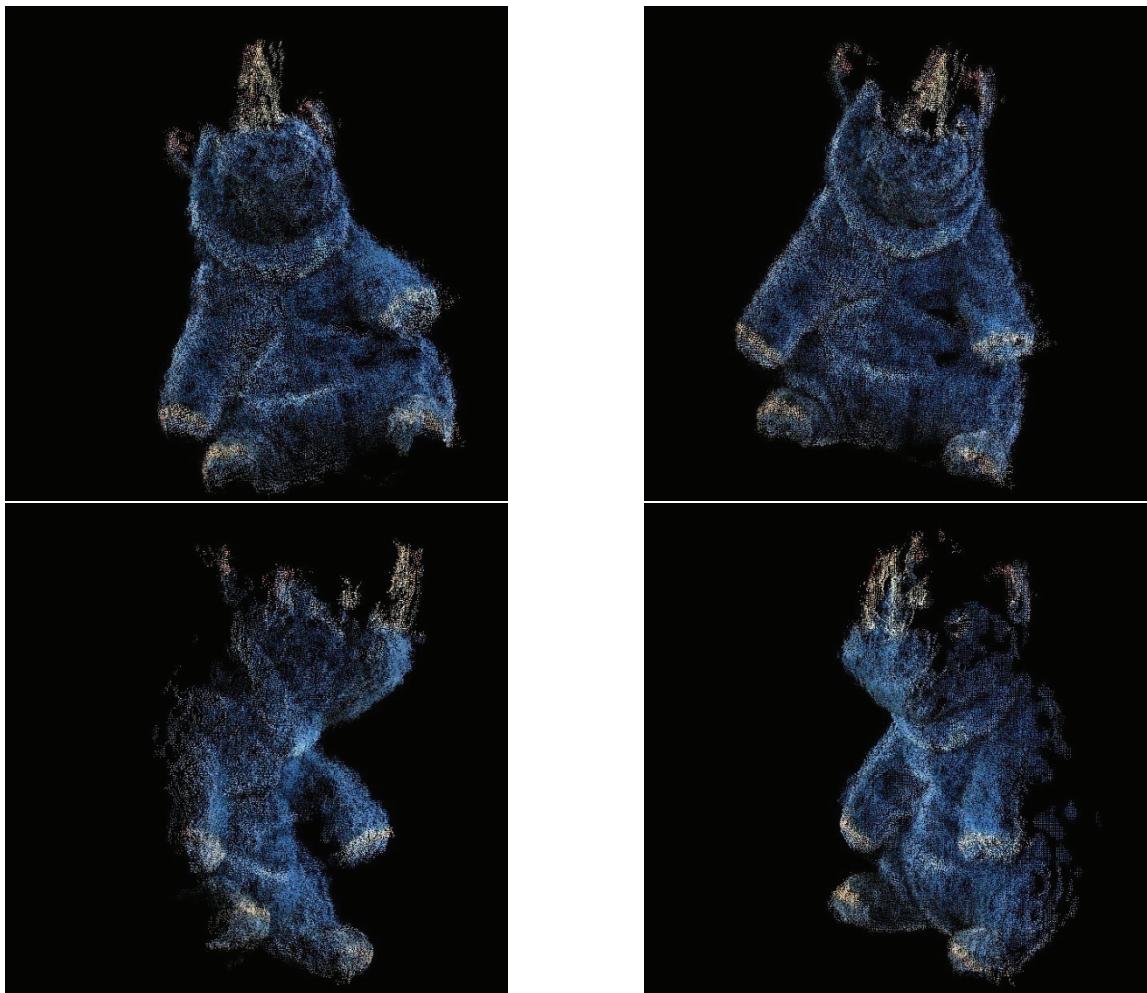
Immagini acquisite

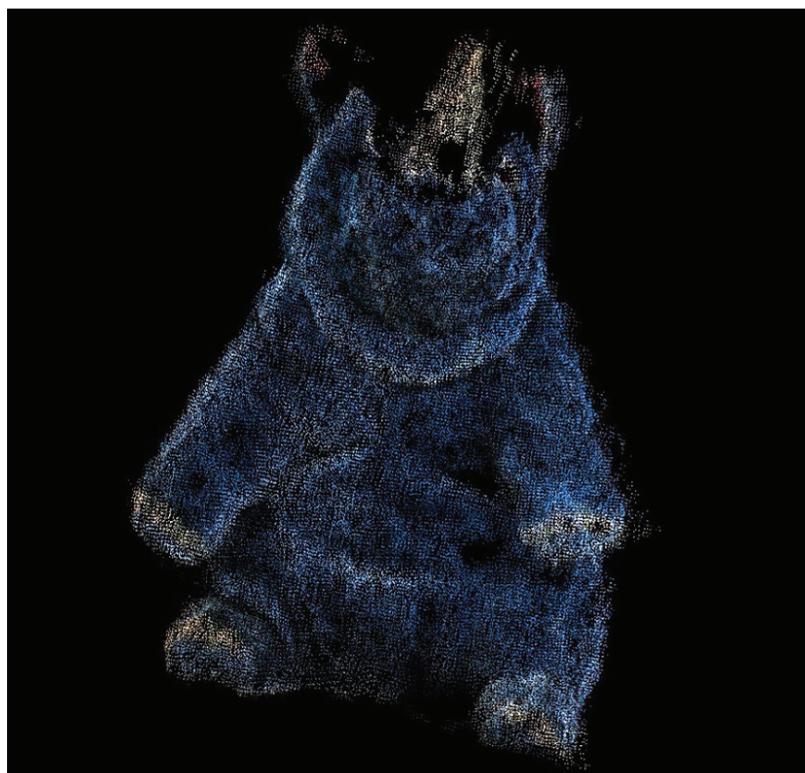
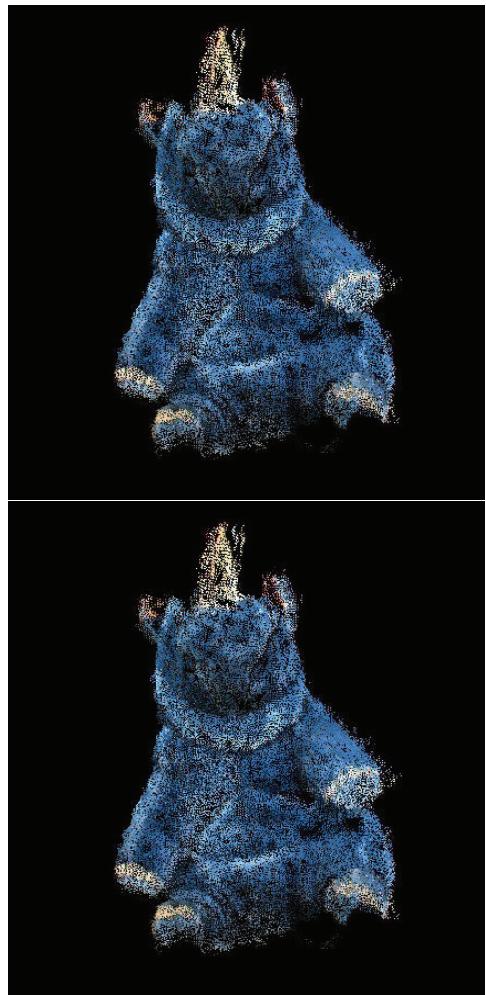
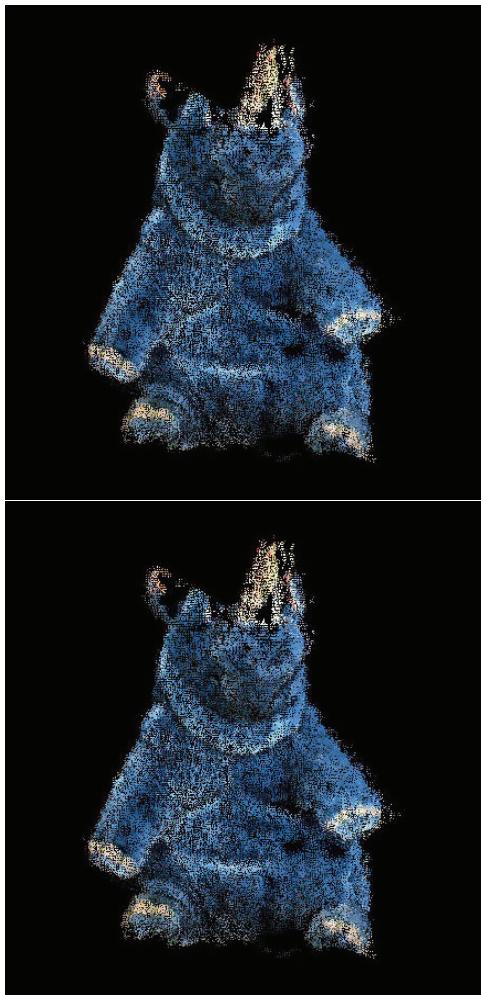


Risultato elaborazione

N. Punti: 73.464

Tempo: 1m



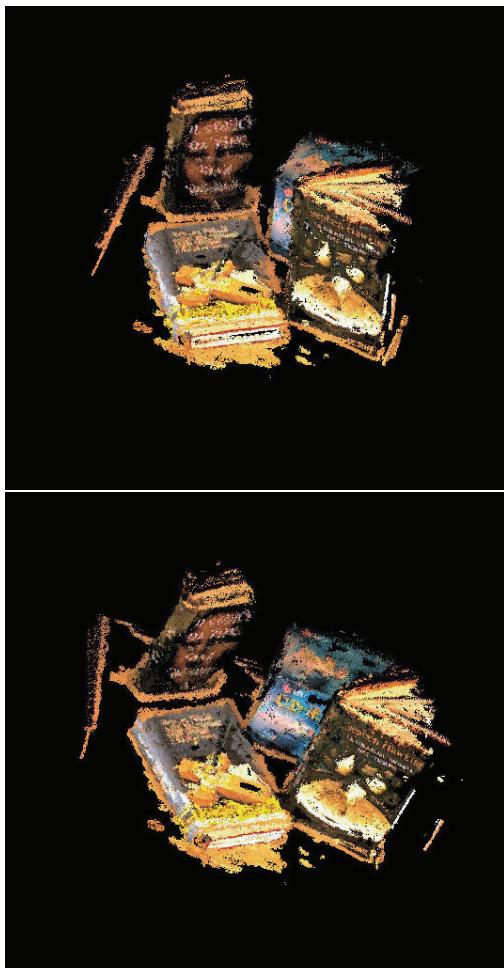


Books Dataset	
Risoluzione immagini	640x480
N. immagini	12
Algoritmo stereo	SMP
Algoritmo di stima robusta	RANSAC
N. iterazioni riproiezione	2

Immagini acquisite

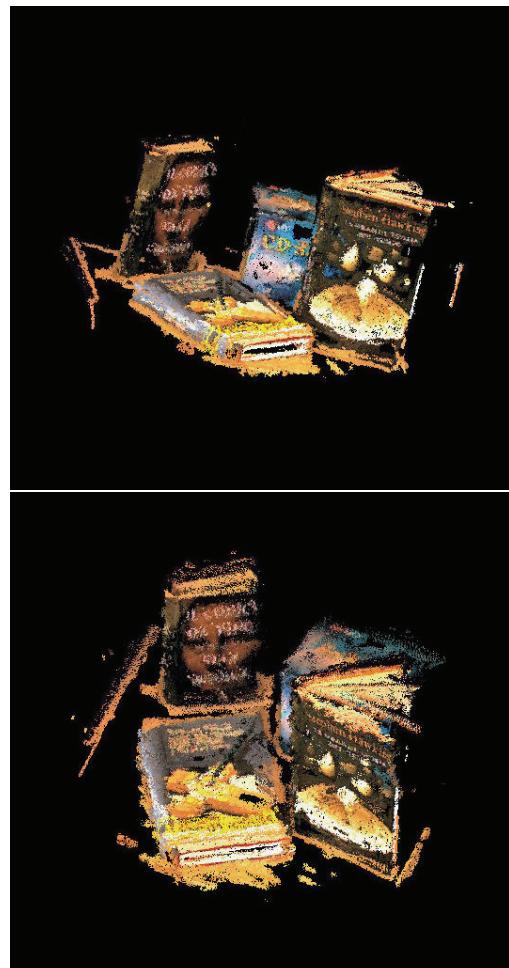


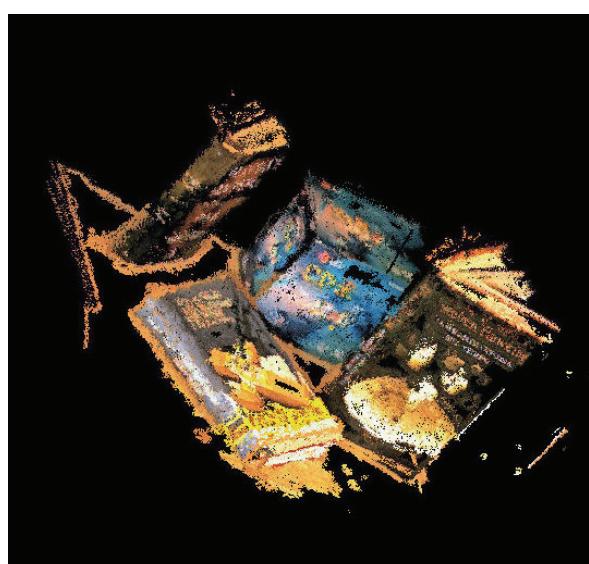
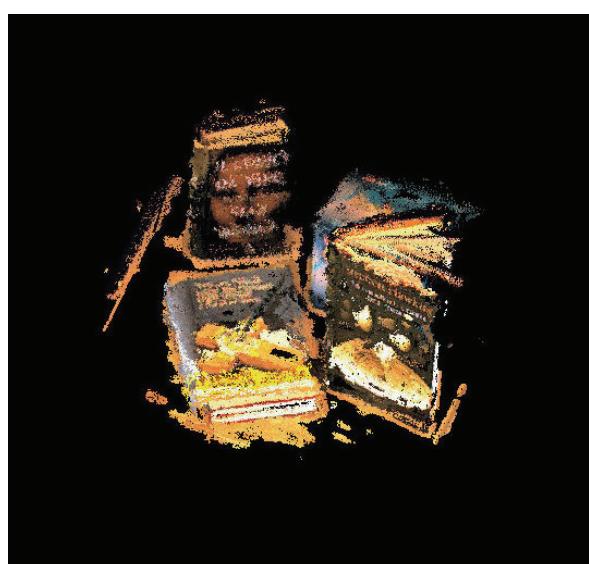
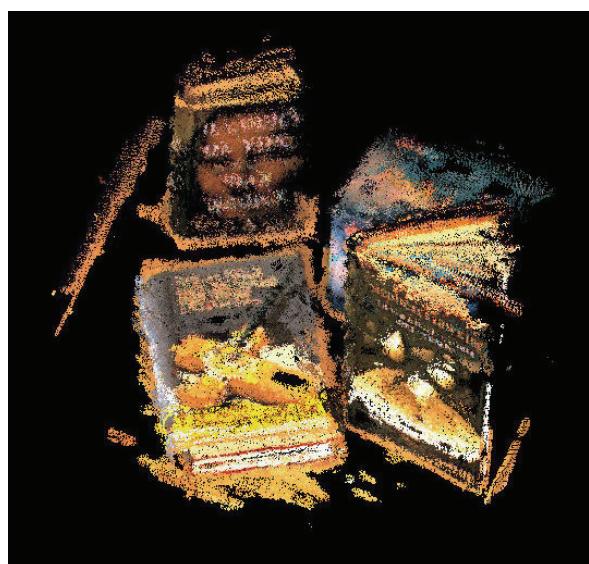
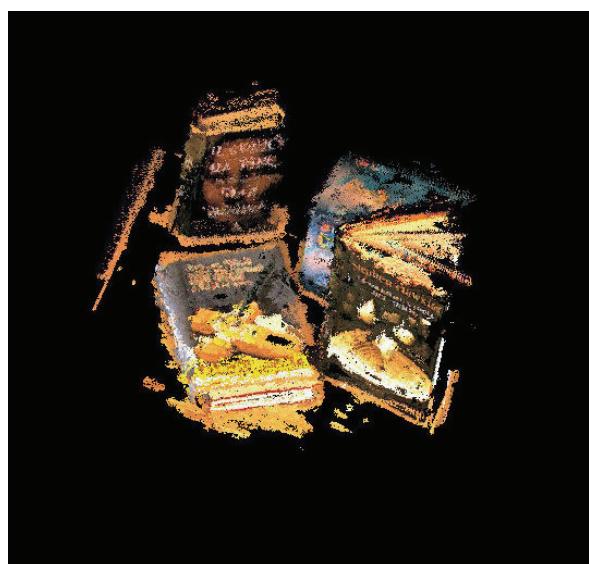
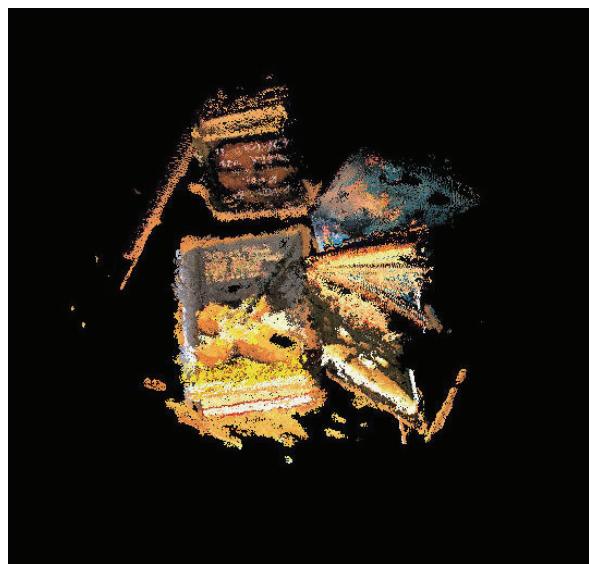
Risultato elaborazione



N. Punti: 92.679

Tempo: 2m





Shoe Dataset	
Risoluzione immagini	640x480
N. immagini	13
Algoritmo stereo	SMP
Algoritmo di stima robusta	LWO
N. iterazioni riproiezione	1

Immagini acquisite

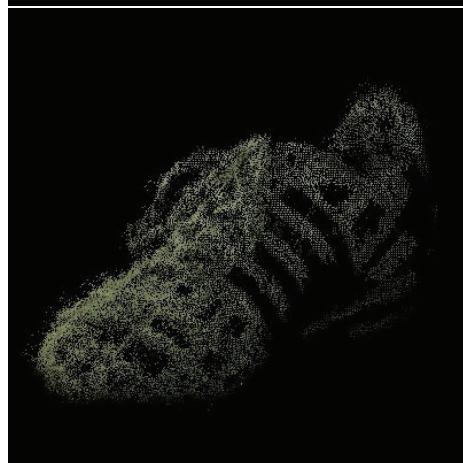
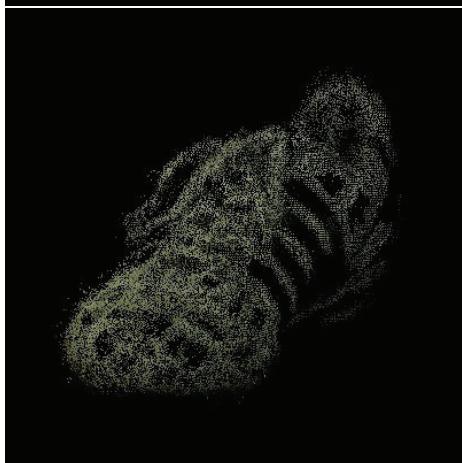
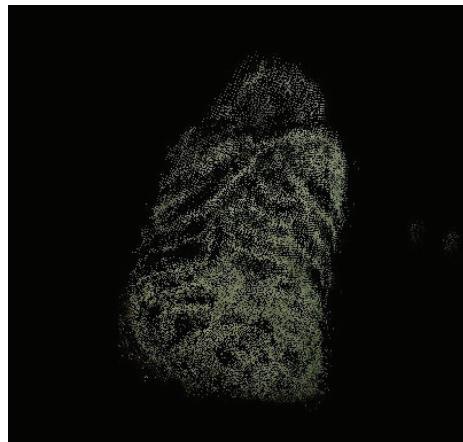


Risultato elaborazione



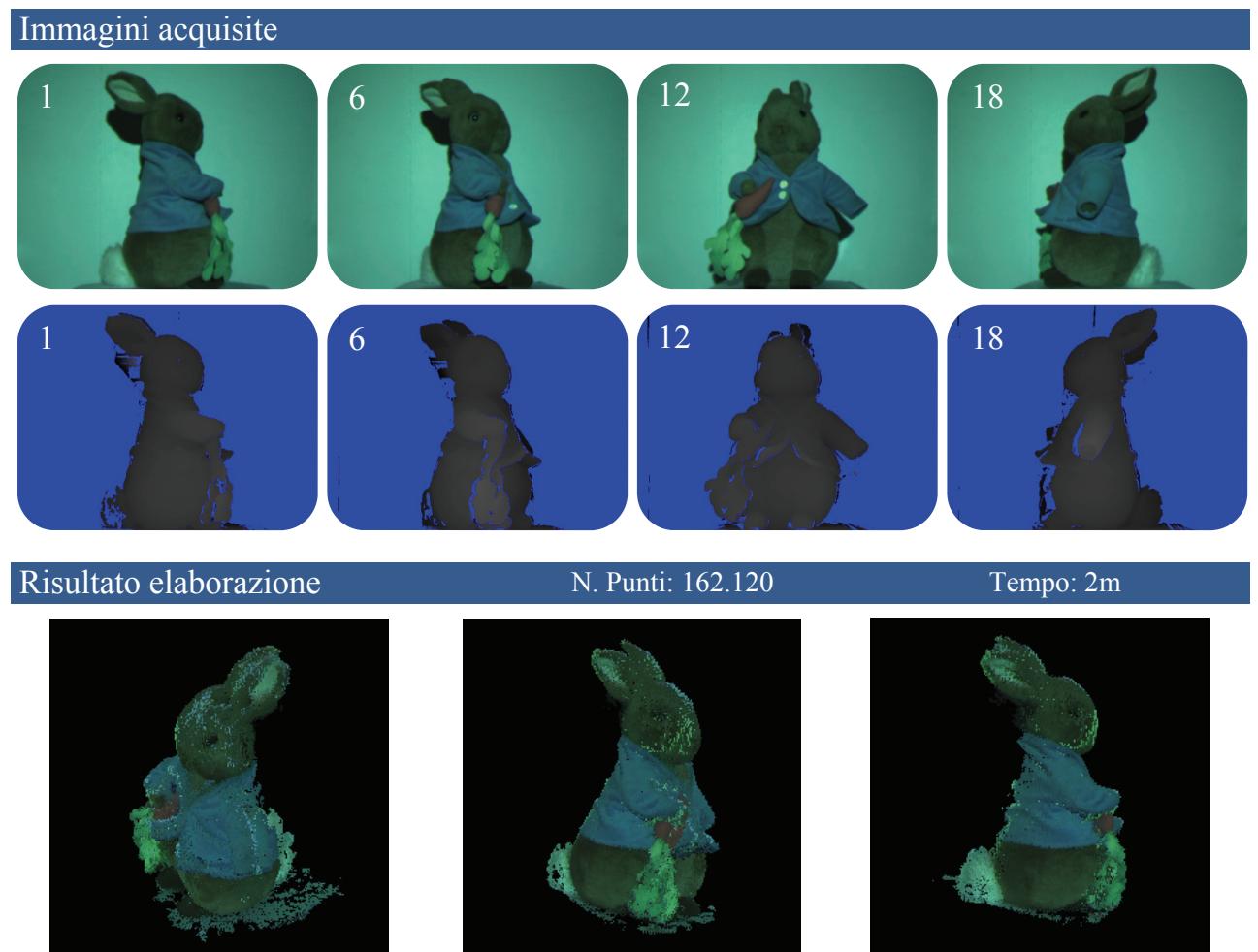
N. Punti: 61.746

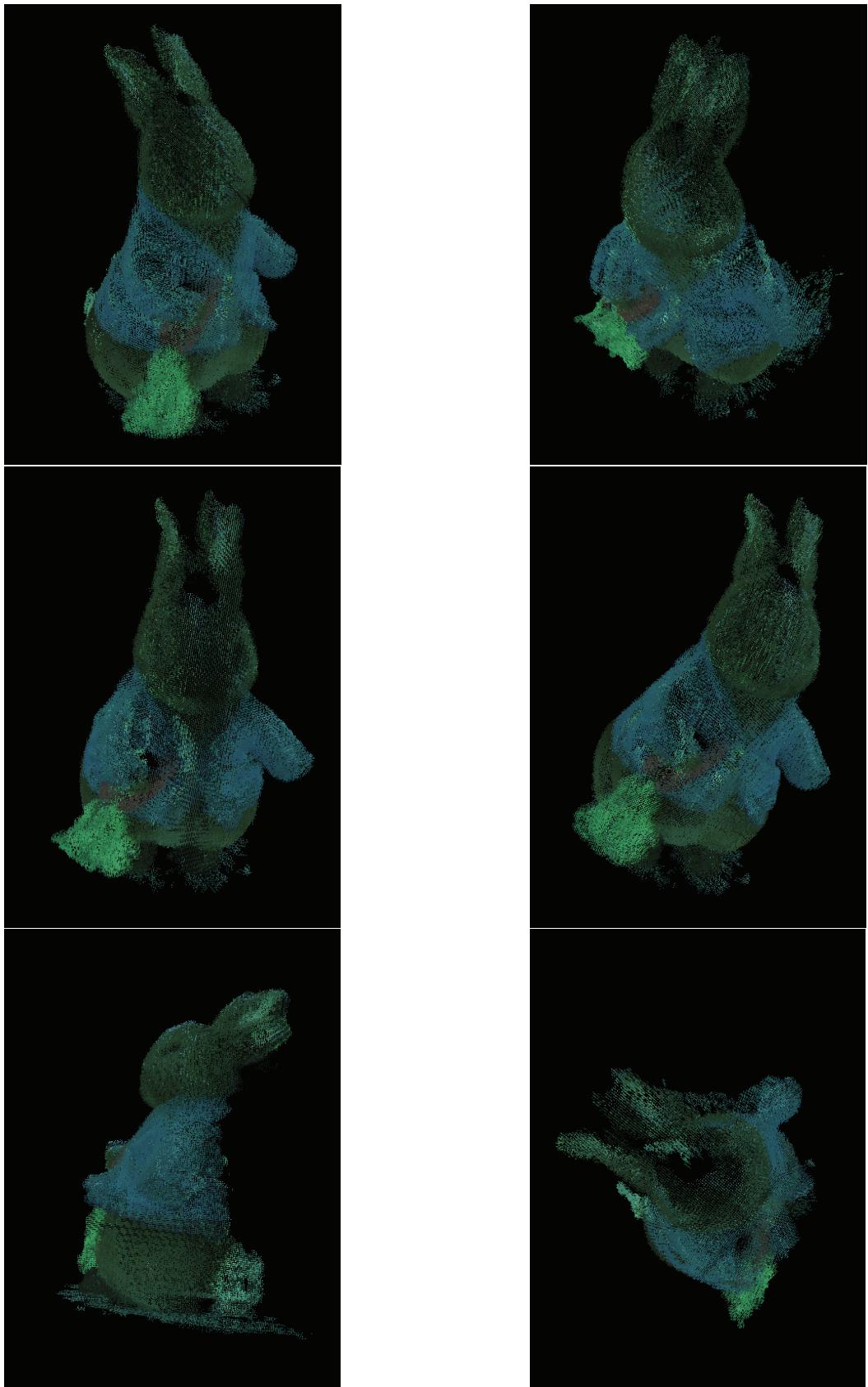
Tempo: 2m



Bunny Dataset		
Risoluzione immagini	537x403	
N. immagini	18	
Algoritmo stereo	SpaceTime	
Algoritmo di stima robusta	RANSAC	
N. iterazioni riproiezione	1	

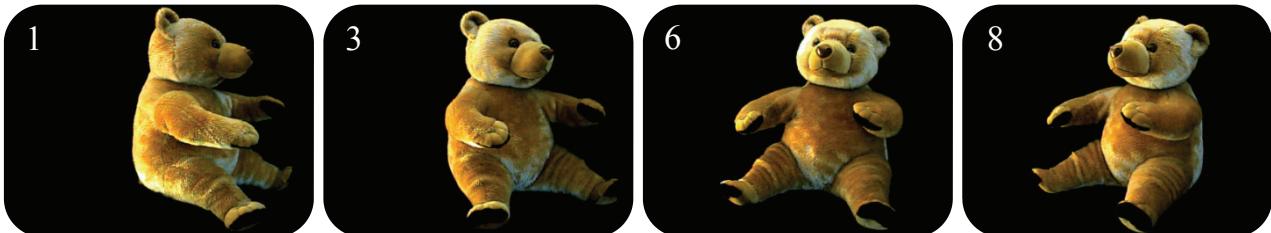
Nell'implementazione del sistema è prevista la possibilità di effettuare la ricostruzione caricando direttamente le mappe di intensità e disparità. In questo modo si svincola il sistema dagli algoritmi di corrispondenza stereo implementati, permettendo di effettuare ricostruzioni con mappe calcolate da qualunque algoritmo. In questa particolare ricostruzione l'algoritmo stereo utilizzato è SpaceTime Stereo (Zhang et al. [21]).





Bear Dataset	
Risoluzione immagini	640x480
N. immagini	8
Algoritmo stereo	SMP
Algoritmo di stima robusta	LWO
N. iterazioni riproiezione	1

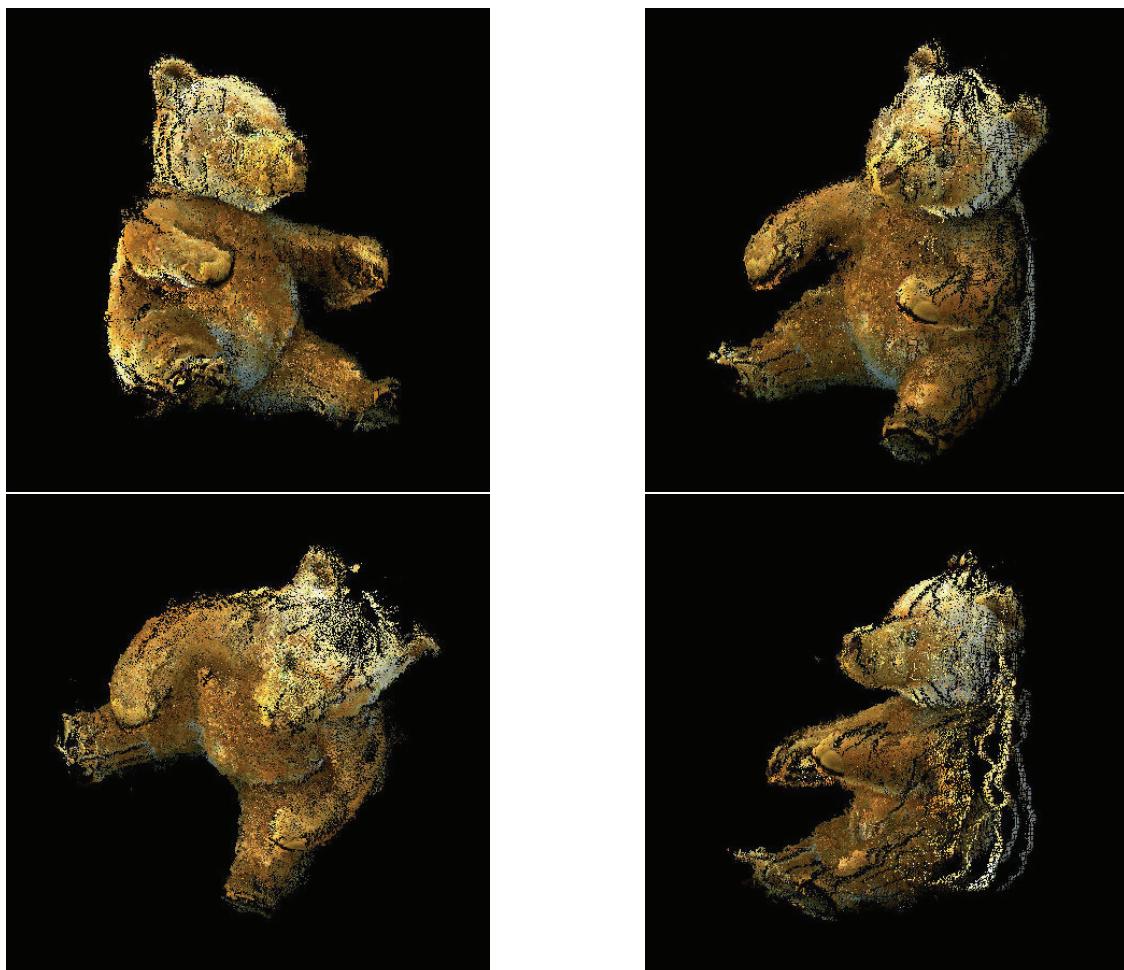
Immagini acquisite

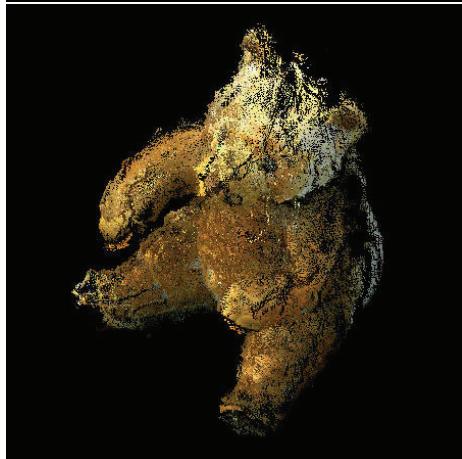


Risultato elaborazione

N. Punti: 135.685

Tempo: 2m





Le ricostruzioni ottenute mostrano un discreto grado di accuratezza, anche a fronte di alcune incertezze nei dati in ingresso. Il rumore ad alta frequenza indotto dall'algoritmo che risolve le corrispondenze non crea grossi problemi ai metodi di stima robusta e viene in buona parte eliminato nella fase di filtraggio. Il rumore a bassa frequenza, invece, introduce un errore più difficile da individuare in quanto i punti si mantengono prossimi al modello. Questo rumore provoca l'effetto di blurring ai bordi ed è ben visibile nei dataset Rhino e Shoe. Meno presente è nel dataset Bunny le cui mappe sono calcolate con un algoritmo più preciso. Si può notare come in questo dataset i bordi siano più definiti, pur essendo una ricostruzione calcolata a partire da immagini con risoluzione e qualità inferiore.

Un altro fattore che grava sull'accuratezza della ricostruzione è la stima dei parametri intrinseci. In particolare una forte incertezza sul calcolo della lunghezza focale induce forti ripercussioni sul modello. Una stima per difetto provoca lo schiacciamento della forma in direzione del punto di vista, viceversa una stima per eccesso ne provoca la dilatazione. In entrambi i casi si presenta una distorsione della forma che introduce rumore nella stima del moto e nel modello finale. L'incertezza nella lunghezza focale genera nuvole 3D non perfettamente allineabili in quanto a forme diverse, con un errore crescente all'aumentare della disparità. Un esempio è ben visibile nel dataset Rhino. Si noti come il corno del rinoceronte presenti effetti di sdoppiamento. Le singole nuvole di punti vengono allineate sui soli punti del corpo che presenta un numero maggiore di corrispondenze SIFT. L'errore finale sarà quindi trascurabile sui punti accoppiati (sul corpo) e maggiormente visibile sui punti a differente disparità (sul corno).

4.2 Immagini sintetiche

In questo paragrafo sono riportate le seguenti 5 ricostruzioni di modelli virtuali:

- Stanford Bunny Dataset – Modello di un coniglio
- Guardian Dataset – Modello di una statua
- Temple Dataset – Modello di un tempio
- Robot Dataset – Modello di un robot
- Stanford Dragon Dataset – Modello di un drago

L'algoritmo di stima robusta generalmente preferibile per questi dataset è RANSAC in virtù della maggior precisione dei dati di partenza. Anche il numero di punti per mappa diventa un fattore determinante, nei dataset a risoluzione più alta come Robot (800x600) e Stanford Dragon (1920x1440) l'utilizzo di LWO si è rivelato inefficiente a causa dell'alta percentuale di *inlier* (dal 70% al 95%) e della quantità di punti (fino a centinaia di migliaia di punti nel dataset Standford Dragon). È opportuno ricordare che RANSAC lavora in modo parzialmente invariante rispetto al numero di punti ed è condizionato solo dal rapporto *inlier/outlier*.

Stanford Bunny Dataset

Fonte modello: <http://wwwgraphics.stanford.edu/data/3Dscanrep/>

Risoluzione immagini	640x480
N. immagini	17
Algoritmo stereo	SMP
Algoritmo di stima robusta	RANSAC
N. iterazioni riproiezione	2

Immagini acquisite



Risultato elaborazione

N. Punti: 102.065

Tempo: 2m



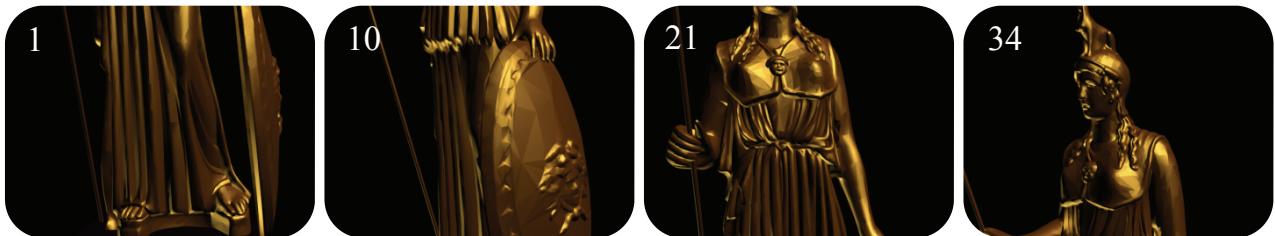


Guardian Dataset

Fonte modello: <http://www.archive3d.net/>

Risoluzione immagini	640x480
N. immagini	34
Algoritmo stereo	SMP
Algoritmo di stima robusta	RANSAC
N. iterazioni riproiezione	2

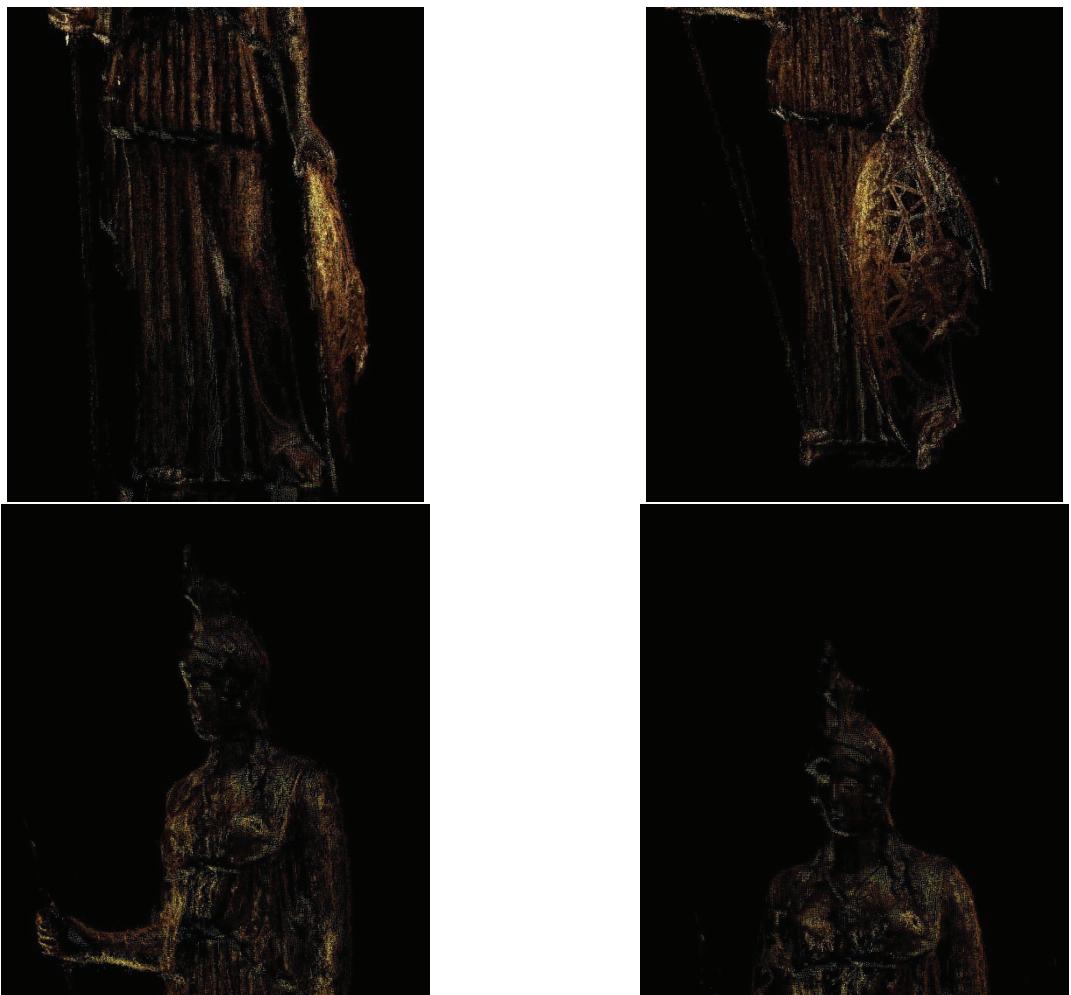
Immagini acquisite



Risultato elaborazione

N. Punti: 216.733

Tempo: 5m





Temple Dataset

Fonte modello: <http://spazioinwind.libero.it/grafica3d/models/models.htm>

Risoluzione immagini	640x480
N. immagini	19
Algoritmo stereo	SMP
Algoritmo di stima robusta	LWO
N. iterazioni riproiezione	2

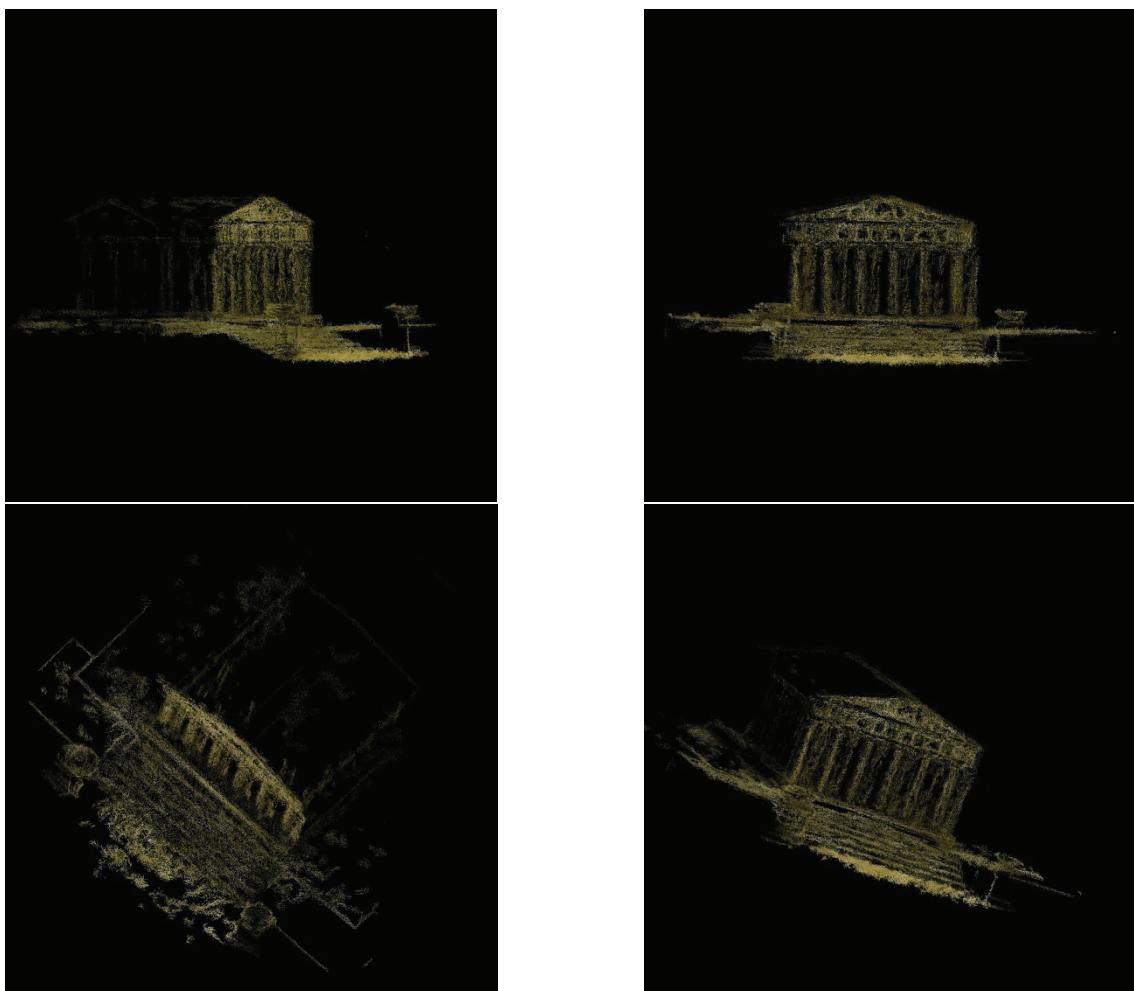
Immagini acquisite

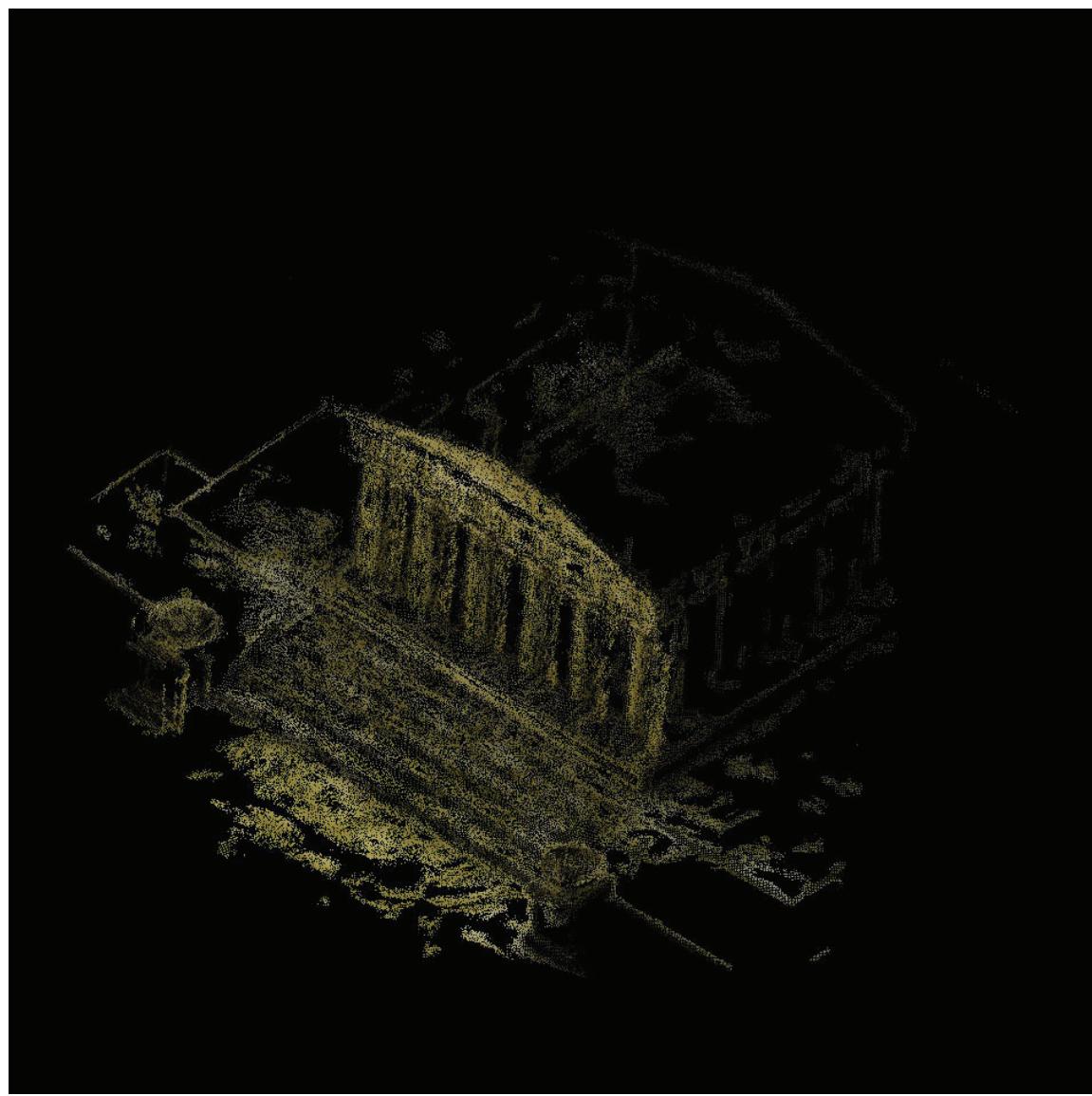
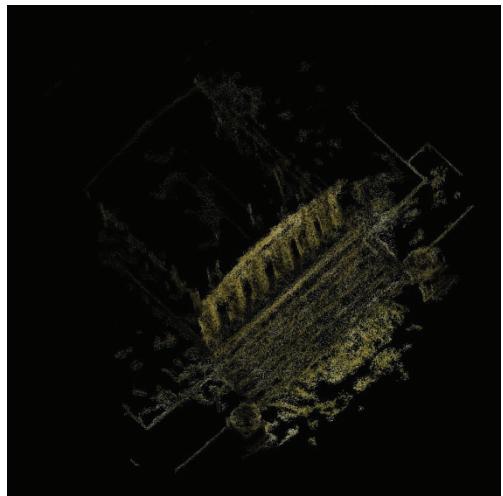
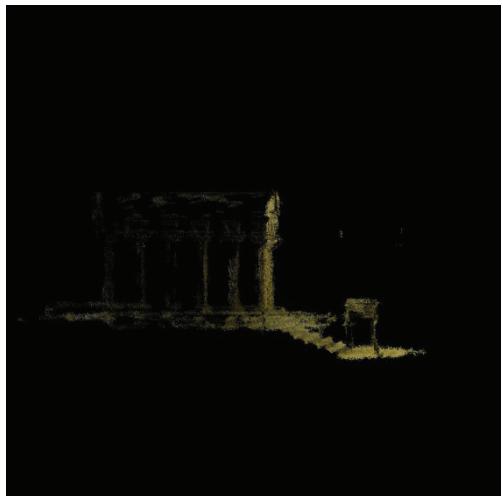


Risultato elaborazione

N. Punti: 139.884

Tempo: 3m





Robot Dataset

Fonte modello: http://artist-3d.com/free_3d_models/

Risoluzione immagini	800x600
N. immagini	18
Algoritmo stereo	SMP
Algoritmo di stima robusta	RANSAC
N. iterazioni riproiezione	1

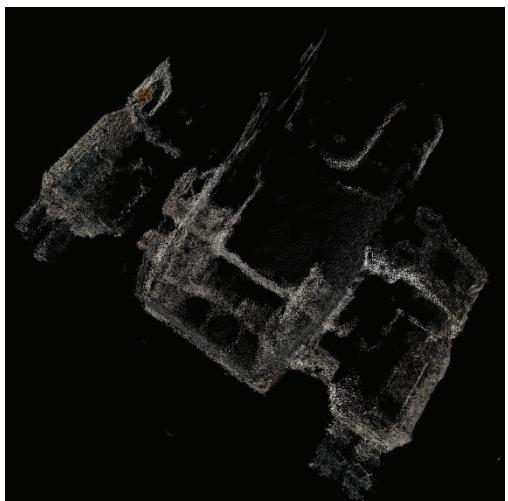
Immagini acquisite



Risultato elaborazione

N. Punti: 206.671

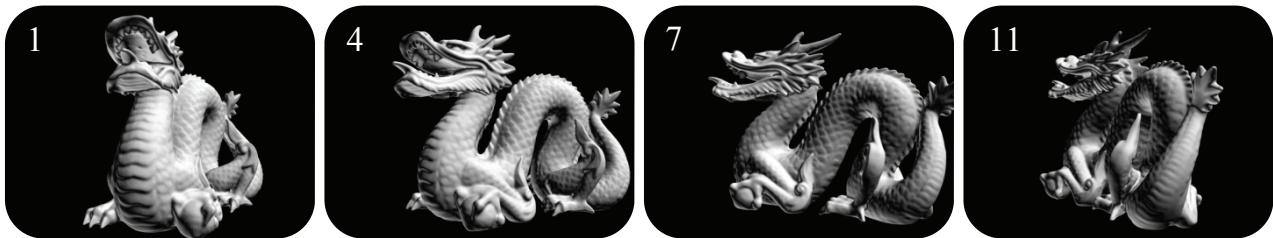
Tempo: 5m





Stanford Dragon Dataset	
Fonte modello: http://wwwgraphics.stanford.edu/data/3Dscanrep/	
Risoluzione immagini	1920x1440
N. immagini	11
Algoritmo stereo	SMP
Algoritmo di stima robusta	RANSAC
N. iterazioni riproiezione	1

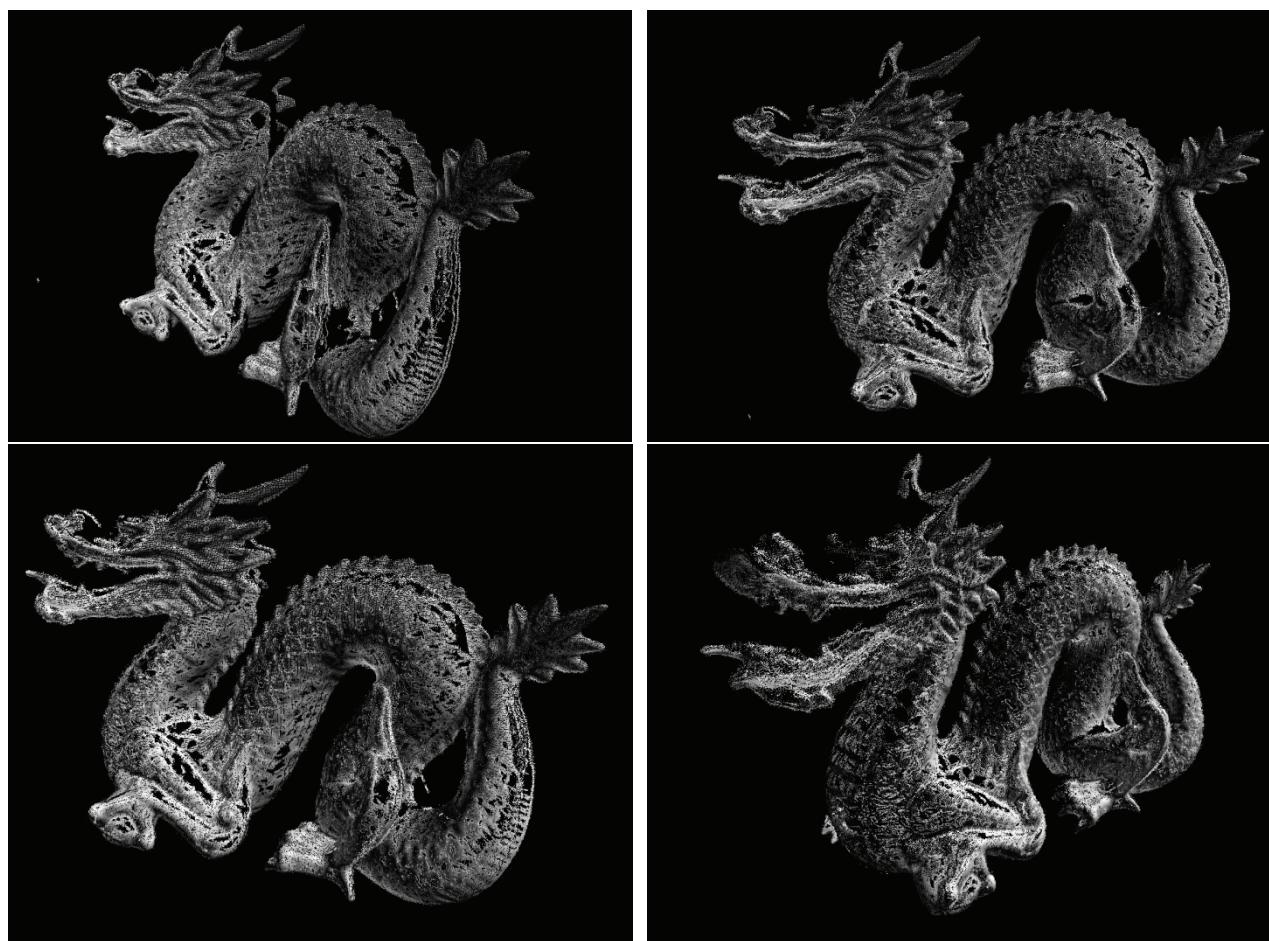
Immagini acquisite

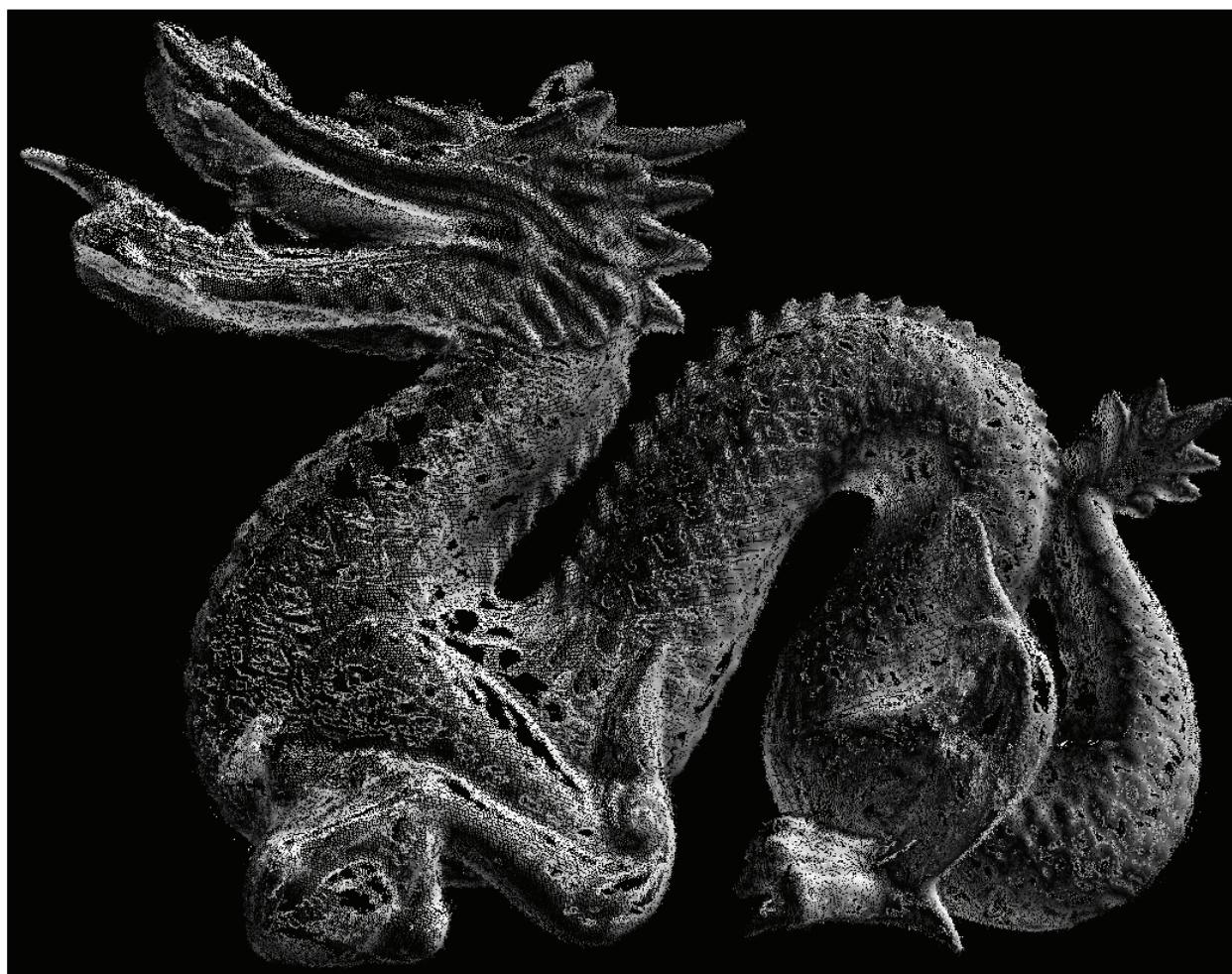
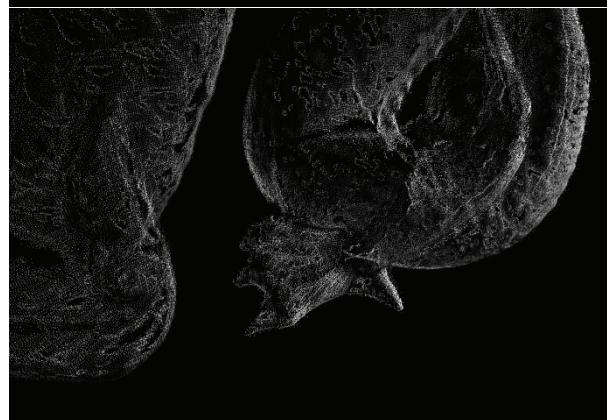
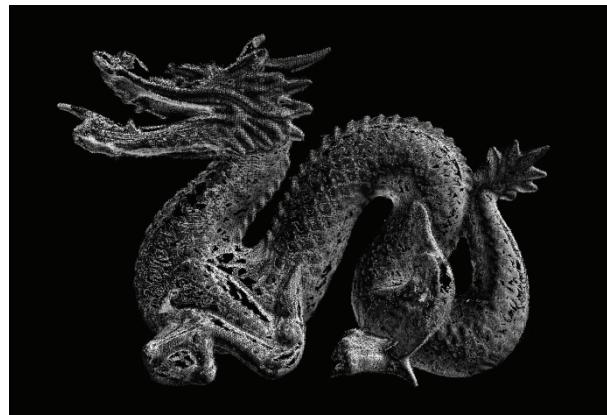


Risultato elaborazione

N. Punti: 483.153

Tempo: 7m





Le ricostruzioni ottenute sono decisamente più accurate, anche nei dataset con immagini alla stessa risoluzione delle immagini reali. Si può notare la maggior definizione dei bordi, dovuta ad una maggior accuratezza delle immagini di partenza (assenza di rumore indotto dall'ottica o dall'elettronica e luminosità regolare). È inoltre assente l'effetto sdoppiamento, in virtù dell'accuratezza dei parametrici di calibrazione.

La risoluzione delle immagini di partenza è ovviamente un parametro determinante. Un'alta risoluzione permette una miglior localizzazione dei punti sia in termini di coordinate X, Y di partenza che in termini di disparità, con conseguente miglioramento nella determinazione del punto 3D. È possibile notare come nel dataset Stanford Dragon (risoluzione 1920x1440) siano presenti dettagli molto accurati come creste e artigli. Gli altri dataset presentano un livello di dettaglio inferiore proprio a causa della risoluzione inferiore.

5 - Conclusioni

In questo capitolo sono riportate le considerazioni finali sul sistema di ricostruzione proposto. Nel paragrafo 5.1 sono presentati e discussi gli obiettivi conseguiti, con una particolare analisi sui punti di forza e sulle principali limitazioni. Nel paragrafo 5.2 sono proposti alcuni eventuali sviluppi futuri che mirano al miglioramento del sistema.

5.1 Considerazioni finali

Il sistema da noi proposto e sviluppato è in grado di effettuare la ricostruzione 3D di una scena mediante immagini catturate da un sistema stereo, in questo modo il sistema risulta essere assolutamente non invasivo. A livello di componenti hardware sono necessari solamente un sistema stereo ed un PC. Il sistema è inoltre indipendente da qualunque vincolo di rigidità, permettendo la scansione mediante il libero spostamento del sistema di acquisizione attorno alla scena. La ricostruzione si dimostra essere piuttosto accurata anche in presenza di acquisizioni a qualità non eccessiva ed in presenza di una certa incertezza nei dati in ingresso. L'impiego di SIFT abbinato agli algoritmi di stima robusta e riproiezione rende il sistema alquanto robusto nella determinazione del moto delle camere. A seguito della fusione delle singole nuvole di punti gli algoritmi di quantizzazione e filtraggio provvedono all'eliminazione di buona parte degli *outlier* presenti. La scena, ricostruita sotto forma di nuvola di punti, è visualizzata a video tramite la libreria OpenGL ed è esportabile nei formati standard PLY¹ e VRML². Il tempo di elaborazione è di pochi minuti, dipendentemente dal volume dei dati in ingresso.

La problematica principale dell'approccio di questo sistema è la forte dipendenza della ricostruzione con la sequenza di mappe in ingresso. Un eventuale errore accumulato durante la ricostruzione tra una coppia di viste si ripercuote inevitabilmente nelle successive, nei casi peggiori inficiando l'intero processo. Un'altra limitazione è l'assenza di controlli fotometrici sulla consistenza dei punti generati, ovvero non si effettuano controlli sulla coerenza dei punti nelle altre viste. Questa limitazione introduce talvolta degli *outlier* non identificabili durante la fase di filtraggio. La stessa limitazione si presenta durante la fase di riproiezione dei punti che essendo proiettati “alla cieca” possono portare in qualche caso a falsi accoppiamenti (in modo particolare nelle occlusioni).

¹ PLY è un formato standard progettato per contenere dati provenienti da scanner 3D.

² VRML (Virtual Reality Modeling Language) è un formato standard di grafica 3D vettoriale impiegato principalmente per il Web.

5.2 Sviluppi futuri

Allo stato attuale il nostro sistema di ricostruzione produce in uscita una nuvola di punti 3D. Per ottenere la ricostruzione completa è necessario generare una mesh poligonale a partire dai punti della nuvola. L'algoritmo Marching Cubes (Lorensen et al. [22]) costruisce la mesh considerando 8 punti vicini alla volta (i vertici di un cubo immaginario) e determinando il poligono (o i poligoni) che attraversano tale cubo. Questo algoritmo può essere ben integrato nel sistema in quanto i punti sono immagazzinati in una struttura dati partizionata in cubi (Octree) e sono a densità costante (a seguito della quantizzazione). Esistono comunque molti altri algoritmi di meshing che forniscono ottimi risultati. In Figura 5.2.1 sono mostrate alcune ricostruzioni effettuate con gli algoritmi Power Crust (Amenta et al. [23]), Hoppe Reconstruction (Hoppe et al. [24]), VRIP (Curless et al. [25]) e Poisson Surface Reconstruction (Kazhdan et al. [26]).

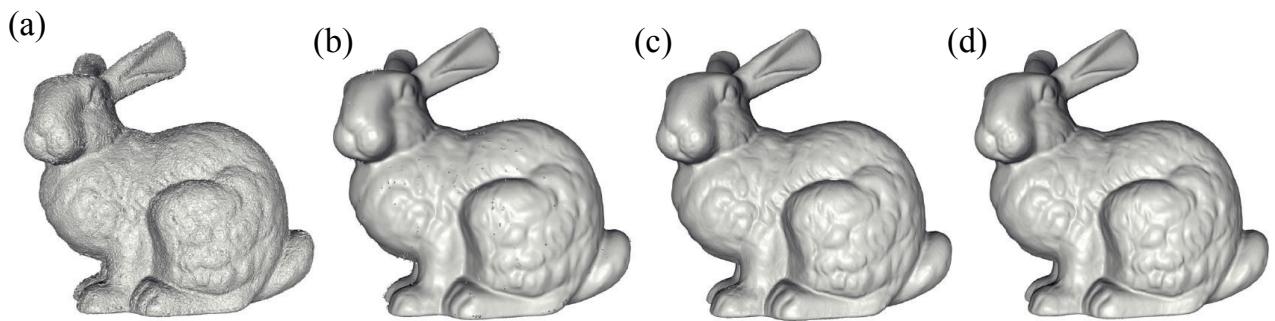


Figura 5.2.1 – Esempio di mesh ottenute con Powercrust (a), Hoppe Reconstruction (b), VRIP (c) e Poisson Surface Reconstruction (d)

Per migliorare ulteriormente la visualizzazione della ricostruzione è possibile applicare una texture ad ogni poligono della mesh. La texture da applicare si può ottenere proiettando il poligono sul piano immagine in cui è meglio visualizzato. Per determinare la vista migliore si può considerare l'area del poligono proiettato (maggiore è l'area proiettata e migliore sarà la vista), oppure considerare l'angolo tra la direzione di vista e la normale del poligono (una normale parallela alla direzione di vista sarà considerata migliore).

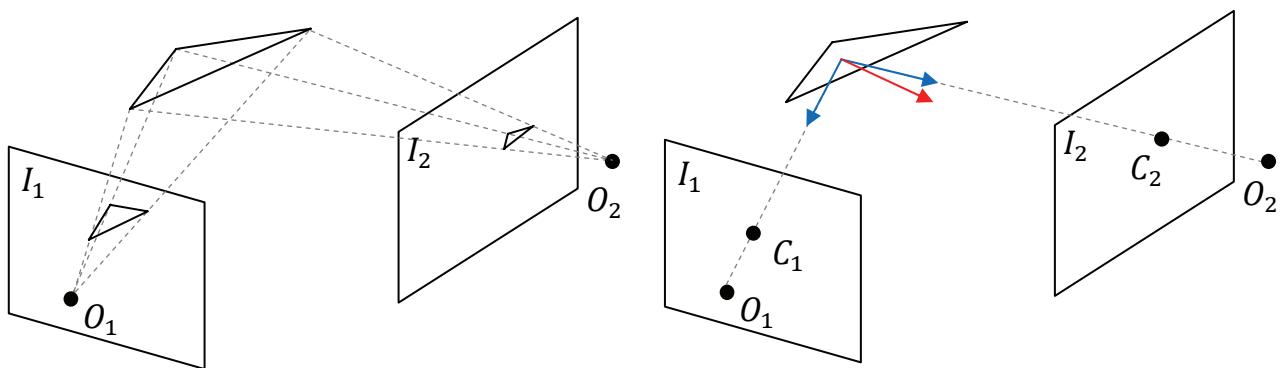


Figura 5.2.2 – Assegnamento della texture per area proiettata (sinistra) e angolo rispetto alla normale del poligono (destra)

In Figura 5.2.2 è mostrato un esempio di assegnamento della texture ad un triangolo. Considerando come criterio di selezione l'area proiettata (a sinistra) verrà selezionata l'immagine I_1 mentre secondo il criterio dell'angolo tra la direzione di vista e la normale del poligono (a destra) verrà selezionata l'immagine I_2 .

Durante la fase di selezione della texture è possibile inserire alcuni controlli di coerenza fotometrica del poligono generato. Un poligono visibile (e quindi non occluso) in diverse viste dovrà necessariamente proiettarsi in modo coerente su tutte le viste. Per questo controllo fotometrico è possibile utilizzare un'opportuna funzione di correlazione. Questa funzione dovrà essere in grado di confrontare aree di forme e dimensioni diverse (distorsioni prospettiche) e dovrà essere robusta ad eventuale cambi di illuminazione (distorsioni fotometriche). Lo stesso tipo di controllo può essere fatto punto per punto prima della fase di meshing in modo da ottimizzarne l'esecuzione (rimozione di ulteriori *outlier* e del numero di punti da processare).

Un ulteriore interessante sviluppo mira a risolvere la limitazione principale del sistema, ovvero la forte dipendenza con la sequenza di immagini in ingresso. Per risolvere questo problema è possibile considerare l'intero blocco di mappe in ingresso e definire un opportuno algoritmo che determini il moto delle camere a livello globale. Essendo già calcolati i keypoint SIFT per ogni immagine è possibile determinare i match tra tutte le coppie possibili (con N mappe si passa da $N - 1$ confronti a $N(N - 1)/2$). Con le corrispondenze trovate e mediante la stima robusta è possibile determinare se tra le coppie di viste esiste o meno una trasformazione coerente. Avendo ora a disposizione una serie di trasformazioni tra coppie di viste si può raffinare la stima minimizzando una funzione di costo globale (Bundle Adjustment, Triggs et al. [27]).

Al momento il contenuto informativo utilizzato durante la ricostruzione è limitato ad una serie di coppie di mappe intensità/disparità. Queste mappe sono calcolate dall'algoritmo stereo a partire dalle immagini sinistra e destra (L e R) su una sola di queste, supponiamo L . Il moto delle camere è poi stimato tra una mappa L e la successiva ($L_1 \rightarrow L_2$). È tuttavia possibile determinare una nuova coppia di mappe intensità/disparità calcolata sull'immagine R . Con questo approccio diventa immediatamente disponibile un filtro per entrambe le mappe che riduce rumore e occlusioni (consistenza Left-Right).

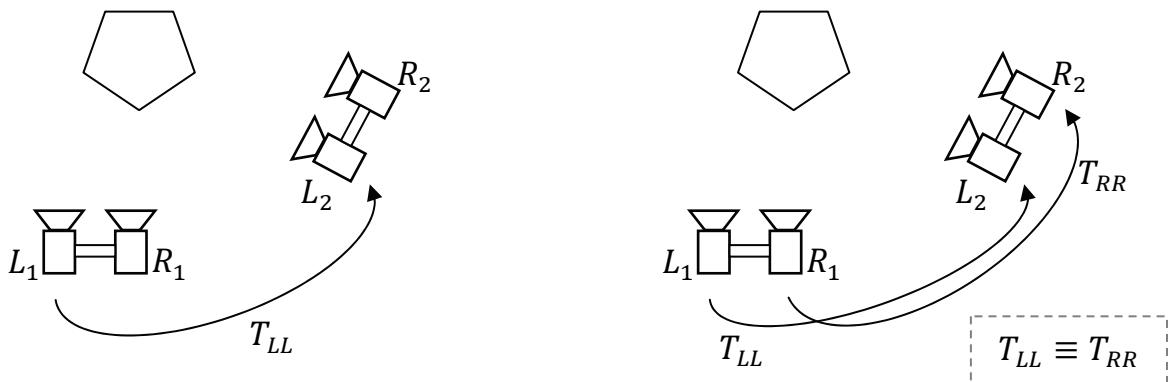


Figura 5.2.3 – Schema di calcolo del moto nel sistema attuale (sinistra) e dello sviluppo proposto (destra)

Inoltre è possibile stimare un nuovo moto delle camere tra una mappa R e la successiva ($R_1 \rightarrow R_2$) che dovrà necessariamente essere coerente con il moto stimato sulle corrispondenti mappe L . In Figura 5.2.3 è mostrato lo schema di calcolo del moto nel sistema attuale (sinistra) e lo schema nello sviluppo proposto (destra). È comunque possibile verificare la coerenza del moto incrociando le mappe sinistra e destra ($L_1 \rightarrow R_2$ e $R_1 \rightarrow L_2$), la trasformazione risultante dovrà essere sempre la medesima. Uno sviluppo di questo tipo introduce una maggiore affidabilità nella stima nel moto. È inoltre possibile determinare corrispondenze SIFT più robuste se calcolate sulle viste più vicine ($R_1 \rightarrow L_2$ in Figura 5.2.3) in quanto meno affette da distorsioni prospettiche e fotometriche.

Bibliografia

- [1] Park S. & Baek J. (2007) Online Registration of Multi-view Range Images using Geometric and Photometric Feature Tracking. *IEEE Computer Society (3DIM07)*, pp. 281-288
- [2] Shi J. & Tomasi C. (1994) Good features to track. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR94), Seattle*
- [3] Bradley D., Boubekeur T. & Heidrich W. (2008) Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR08), Anchorage*
- [4] Furukawa Y. & Ponce J. (2007) Accurate, dense and robust multi-view stereopsis. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR07), Minneapolis*
- [5] Fusiello A., Trucco E. & Verri A. (2000) A compact algorithm for rectification of stereo pairs. *Mach. Vision Appl.*, 12(1), pp. 16–22
- [6] Seitz S. M., Curless B., Diebel J., Scharstein D., and Szeliski R. (2006) A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR06), New York City*
- [7] Hartley R., Zisserman A. (2003) Multiple View Geometry in Computer Vision second edition. Cambridge University Press
- [8] Se S. & Jasiobedzki P (2005) Instant scene modeler for crime scene reconstruction. *IEEE Workshop on Advanced 3D Imaging for Safety and Security (A3DISS)*, San Diego
- [9] Lowe D. G. (2004) Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* 60
- [10] Roth G. and Wibowo E. (1997) An efficient volumetric method for building closed triangular meshes from 3-d image and point data. In *Proceedings of Graphics Interface (GI)*, pp. 173-180, Kelowna, Canada.
- [11] Horn B. K. P. (1987) Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, Vol. 1, No. 3, pp. 629-642

- [12] Harris C. & Stephens M. (1988) A combined corner and edge detector. *In Fourth Alvey Vision Conference*, pp. 147-151 , Manchester
- [13] Scharstein D. & Szeliski R. (2002) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, Vol. 47, pp. 7-42
- [14] Lindeberg, T. (1994) Scale space theory: A basic tool for analyzing structures at different scales. *Journal of Applied Statistics*
- [15] Iocchi L., Konolige K. & Bajracharya M. (2000) Visually Realistic Mapping of a Planar Environment with Stereo. *Proc. of Seventh International Symposium on Experimental Robotics (ISER'00)*, Honolulu
- [16] Yun U. S., Min D. & Sohn K. (2007) 3D scene reconstruction system with hand-held stereo cameras. *3D True Vision Conference*, Kos Island, Greece
- [17] Faugeras O. (1993). Three-Dimensional Computer Vision: A Geometric Viewpoint. *The MIT Press*, Cambridge
- [18] Di Stefano L., Marchionni M., Mattoccia S. & Neri G. (2004) A Fast Area-Based *Stereo Matching Algorithm*. *Image and Vision Computing*, Vol. 22, No. 12, pp. 983-1005, Elsevier
- [19] Tombari F., Mattoccia S., Di Stefano L. & Addimanda E. (2008) Near real-time stereo based on effective cost aggregation. *International Conference on Pattern Recognition (ICPR08)*, Florida
- [20] Fischler M. A., Bolles C. R. (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, Vol. 21, No. 6, pp. 381-395, New York
- [21] Zhang L., Curless B. & Seitz M. S. (2003) Spacetime Stereo: Shape Recovery for Dynamic Scenes. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* , pp. 367-374, Madison, WI
- [22] Lorensen E. W. & Cline E. H. (1987) Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *In Proceedings of Computer Graphics (SIGGRAPH '87)*, Vol. 21, No. 4, pp. 163-169
- [23] Amenta N., Choi S. & Kolluri R. (2001) The power crust, unions of balls, and the medial axis transform. *Computational Geometry: Theory and Applications* 19, pp. 127–153
- [24] Hoppe H., DeRose T., Duchamp T., McDonald J. & Stuetzle W. (1992) Surface reconstruction from unorganized points. *In Proceedings of Computer Graphics (SIGGRAPH '92)*, pp. 71–78

- [25] Curless B. & Levoy M. (1996) A volumetric method for building complex models from range images. *In Proceedings of Computer Graphics (SIGGRAPH '96)*, pp. 303–312
- [26] Kazhdan M., Bolitho M. & Hoppe H. (2006) Poisson surface reconstruction. *In Proceedings of Computer Graphics (SIGGRAPH '06)*, pp. 61–70
- [27] Triggs W., McLauchlan P., Hartley R. & Fitzgibbon A. (2000). Bundle adjustment: A modern synthesis. *In W. Triggs, A. Zisserman & R. Szeliski (eds), Vision Algorithms: Theory and Practice*, LNCS, Springer Verlag