

**Ricostruzione e calcolo del volume di oggetti 3D in  
ambiente controllato**

**Relatore:** Prof. *Simone Bianco*

**Co-relatore:** Prof. *Paolo Napoletano*

**Relazione della prova finale di:**

*Samuele Ventura*

*Matricola 793060*

**Anno Accademico 2016-2017**

# Sommario

L'idea iniziale proposta dai tutor sulla quale si basa questa tesi è quella di costruire un dispositivo fisico in grado di automatizzare l'acquisizione di immagini in ambiente controllato, con l'obiettivo di implementare una pipeline di preprocessing delle immagini e, attraverso una ricostruzione 3d dell'oggetto, calcolarne il volume.

Nella tesi sarà mostrato come il lavoro sia stato diviso in tre parti, due delle quali svolte da miei colleghi, e come insieme formino un flusso di lavoro completo, che integra hardware e software.

Nello specifico saranno analizzate le principali tecnologie di ricostruzione 3d, le motivazioni delle scelte effettuate e gli algoritmi implementati.

Nella tesi sarà esposta la tecnica di ricostruzione basata sulla visione stereo, quindi i modelli matematici e geometrici su cui si basa, come per esempio la geometria epipolare o i parametri che caratterizzano uno strumento di acquisizione digitale, nel nostro caso una camera. Ogni passaggio intermedio sarà rappresentato anche da immagini che mostrano lo stato del sistema in ogni istante dell'elaborazione.

# Indice

<b>1</b>	<b>Introduzione</b>	<b>1</b>
1.1	Scopo del progetto . . . . .	2
1.2	Struttura del dispositivo . . . . .	3
1.3	Divisione dei compiti . . . . .	4
1.3.1	Configurazione hardware . . . . .	4
1.3.2	Preprocessing immagini . . . . .	7
1.3.3	Ricostruzione 3D . . . . .	9
1.4	Caso di studio . . . . .	9
1.5	Organizzazione tesi . . . . .	10
<b>2</b>	<b>Stato dell'arte</b>	<b>12</b>
2.1	Tecnologie a contatto . . . . .	13
2.2	Tecnologie senza contatto attive . . . . .	14
2.2.1	Distanziometri . . . . .	15
2.2.2	Sistemi a triangolazione ottica . . . . .	16
2.2.3	Sistemi a luce strutturata . . . . .	17
2.3	Tecnologie senza contatto passive . . . . .	17
2.3.1	Fotogrammetria e stereo vision . . . . .	18
2.3.2	Shape from silhouette . . . . .	18
2.3.3	Photometric stereo . . . . .	20
2.4	Tabella riassuntiva . . . . .	21
<b>3</b>	<b>Calcolo del volume</b>	<b>22</b>
3.1	Architettura e componenti . . . . .	22
3.2	Concetti preliminari . . . . .	23
3.2.1	Basi ottica . . . . .	23
3.2.2	Basi visione binoculare umana . . . . .	27

3.3	Stereo vision . . . . .	29
3.3.1	Posizione camere . . . . .	29
3.3.2	Calibrazione camere . . . . .	29
3.3.3	Risoluzione modello geometrico . . . . .	39
3.3.4	Triangolazione . . . . .	43
3.3.5	Calibrazione stereo . . . . .	45
3.3.6	Rettificazione . . . . .	46
3.3.7	Stereo correspondence . . . . .	52
3.3.8	Point cloud . . . . .	57
3.4	Calcolo volume . . . . .	59
<b>4</b>	<b>Risultati e possibili miglioramenti</b>	<b>67</b>
4.1	Risultati . . . . .	67
4.2	Possibili miglioramenti . . . . .	74
4.2.1	Jetson TX1 . . . . .	74
4.2.2	Camere . . . . .	75
<b>5</b>	<b>Conclusioni</b>	<b>77</b>
<b>Bibliografia</b>		<b>78</b>
<b>Ringraziamenti</b>		<b>81</b>

# Capitolo 1

## Introduzione

La tesi nasce dall’esperienza maturata nei mesi di stage presso il laboratorio di IoT, Internet of Things, dell’università degli Studi di Milano-Bicocca.

Nel mondo moderno, grazie alle invenzioni tecnologiche sia sul lato hardware che sul lato software, in particolare sull’efficienza con cui si possono processare molti dati, stiamo assistendo ad un fenomeno in cui oggetti inanimati prendono “vita” e comunicano attraverso sensori, si riconoscono e raccolgono dati prendendo talvolta decisioni per noi.

In questo progetto si è deciso di sfruttare la raccolta di dati da parte di due sensori, per automatizzare l’avvio della pipeline software successivamente implementata.

### 1.1 Scopo del progetto

L’idea iniziale, proposta dai tutor, era quella di costruire un dispositivo fisico in grado di automatizzare l’acquisizione di immagini in ambiente controllato, con l’obiettivo di implementare una pipeline di preprocessing delle immagini e, attraverso una ricostruzione 3D dell’oggetto, calcolarne il volume.

Il progetto completo quindi comprende tre blocchi principali :

1. la configurazione dell’hardware necessario;
2. la fase di preprocessing delle immagini acquisite;
3. l’uso di algoritmi per la ricostruzione 3d al fine di calcolare il volume.

Personalmente mi sono occupato del terzo punto, che sarà esposto nel dettaglio nel capitolo 3. I miei due colleghi hanno sviluppato i punti 1 e 2, che descriverò brevemente nella prossima sezione, con alcune immagini esplicative. In figura 1 è mostrato il flow chart completo del progetto.

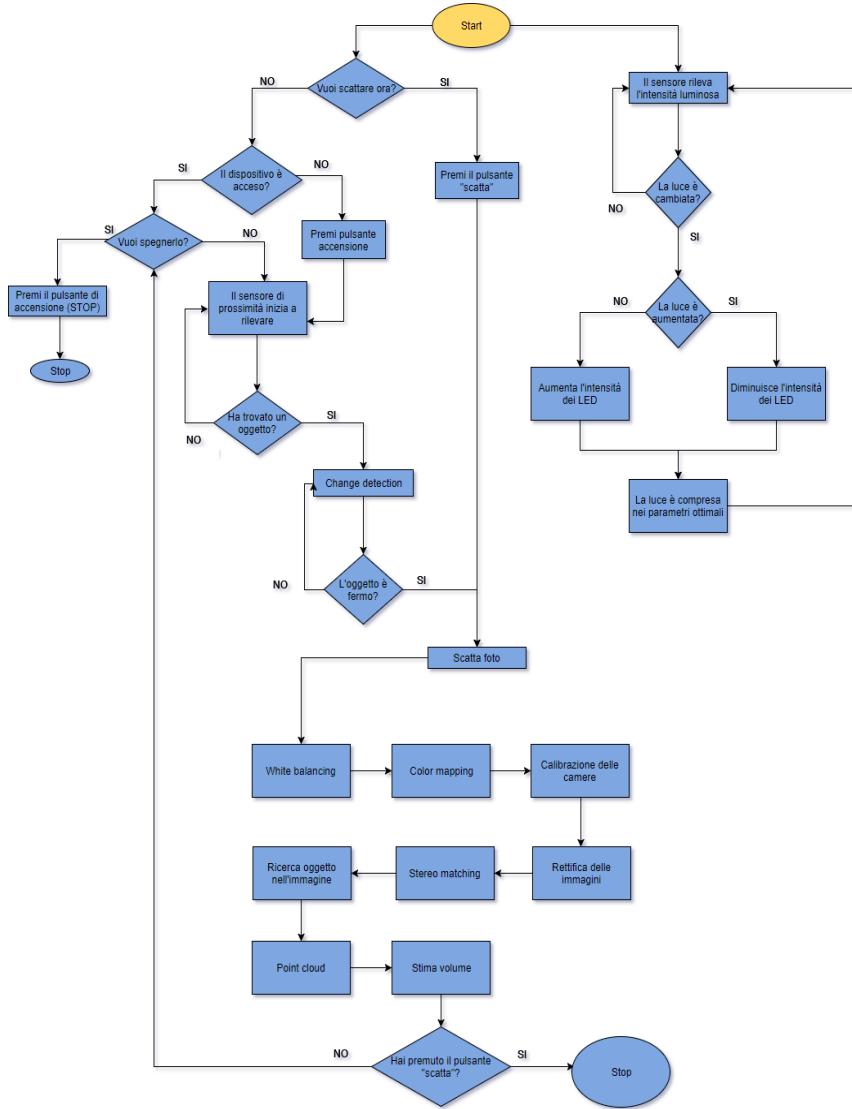


Figura 1: Flow chart delle tre parti.

## 1.2 Struttura del dispositivo

Prima di ogni cosa è stato necessario pensare allo “scheletro” del dispositivo, cioè al materiale e come sarebbe stato costruito. L’idea, consigliata dai tutor, era di comprare una struttura che potesse essere malleabile per le nostre esigenze. Abbiamo optato per l’IVAR dell’Ikea, mostrato in figura 2, una struttura di legno che avremmo potuto adattare a seconda delle necessità e sulla quale iniziare a montare i pezzi del dispositivo. Abbiamo scelto questo genere di struttura aperta, perché ci permetteva una ampia libertà di movimento e inoltre esisteva la possibilità sia di ricevere luce naturale dall’ambiente circostante, sia di modificare le condizioni di luce in modo artificiale. Queste possibilità permettevano di studiare tutte le possibili condizioni, in modo tale da ricreare i presupposti di luce da noi concordati precedentemente.

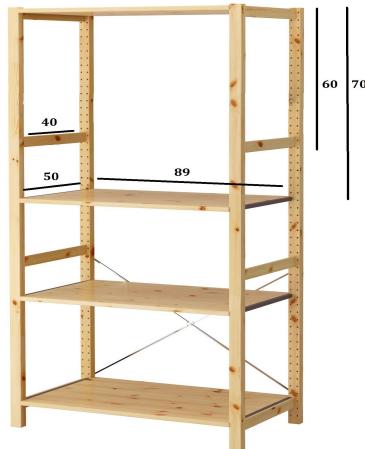


Figura 2: Struttura dispositivo

## 1.3 Divisione dei compiti

### 1.3.1 Configurazione hardware

Questo blocco del progetto si sviluppa su 3 punti principali :

- **Connessione dei componenti ad un PC centrale**

Per gestire e sincronizzare le diverse parti hardware e software si è scelto di utilizzare la Jetson TK1[1], mostrata in figura 3, sviluppata dall'NVIDIA, sostanzialmente un piccolo computer on board.

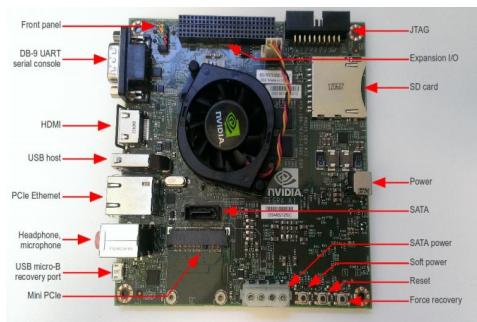


Figura 3: Dettaglio Jetson TK1.

- **Rilevazione automatica dell'oggetto**

Per quanto riguarda questo punto il mio collega aveva il compito di configurare un hardware in grado di rilevare automaticamente la presenza di oggetti posti all'interno del dispositivo di acquisizione.

Questo obiettivo è stato raggiunto grazie all'uso di un microcontrollore, nel caso specifico è stato impiegato Arduino Uno[2], e di un sensore di prossimità ad ultrasuoni.

Il sensore rileva la presenza di oggetti se posti davanti ad esso, in quanto l'impulso viene riflesso sul ricevitore quando colpisce gli oggetti stessi. La distanza dalla quale è possibile rilevare un corpo può essere impostato in base all'utilizzo necessario. Se il sensore rileva un oggetto, sfruttando le camere e un algoritmo di change detection, siamo in grado di automatizzare lo scatto, se e solo se, è presente un nuovo oggetto, o lo stesso in posizione diversa, ed è fermo (per evitare immagini sfocate o indesiderate).

- **Adattamento condizioni di luce**

Uno dei vincoli del progetto è di avere un ambiente controllato, obiettivo raggiunto grazie al controllo e adattamento delle condizioni di luce. Sfruttando un fotoresistore, si rileva l'intensità luminosa presente all'interno nel dispositivo (nella zona dove saranno posti gli oggetti), con uno spettrofotometro si rileva la luce ambientale in lux. Il secondo dispositivo è servito inizialmente per definire le condizioni da raggiungere in real time; il fotoresistore serve per controllare in ogni momento che le condizioni siano rispettate. Per modificare le condizioni di luce si è sfruttata una striscia di led RGB, quindi programmabili sia in colore che intensità. L'ambiente controllato è visibile in figura 4.



Figura 4: Dispositivo di acquisizione con led accessi, per il controllo dell'ambiente.

Di seguito, figura 5, è mostrato il posizionamento dei vari dispositivi hardware utilizzati.

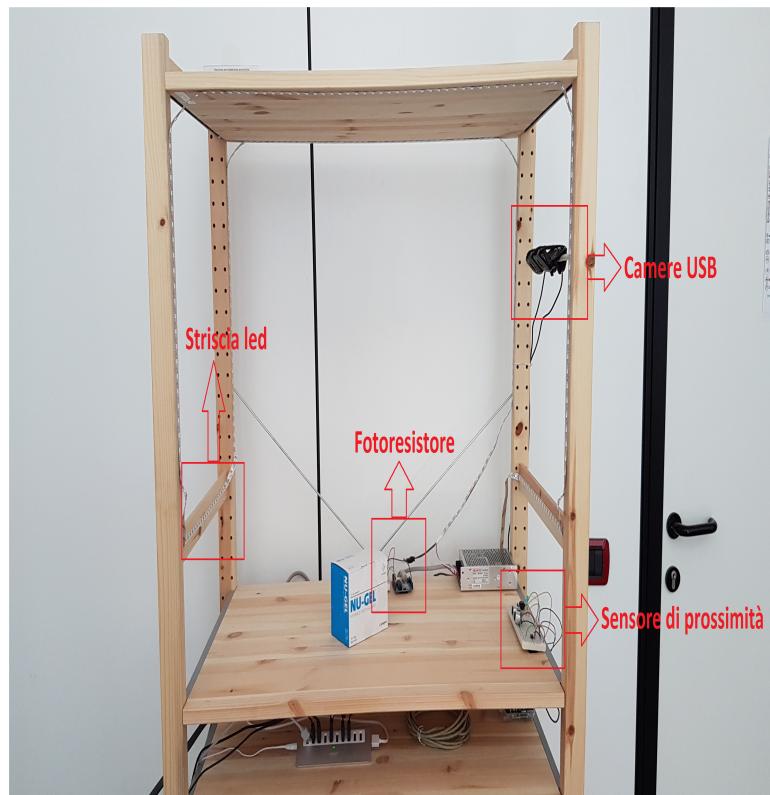


Figura 5: Dispositivo di acquisizione, in cui sono evidenziate le posizioni dei vari hardware utilizzati.

### 1.3.2 Preprocessing immagini

Questo blocco implementa tre punti di una pipeline di preprocessing di una camera digitale:

#### 1. Demosaicing

Un algoritmo di demosaicing permette di ricostruire la rappresentazione a colori di un'immagine partendo dai dati grezzi ottenuti dal sensore di una fotocamera digitale che utilizza un color filter array(CFA), come mostrato in figura 6.

Molte camere permettono anche di salvare le immagini in formato RAW, sulle quali è possibile applicare la tecnica di demosaicing che si vuole utilizzare al posto dell'algoritmo proprietario della fotocamera.

Un formato RAW rappresenta un'immagine in cui per ogni pixel è presente solo l'informazione di uno dei tre canali con i quali è possibile rappresentare il colore reale, solitamente con una tripletta RGB, questo perché su ogni fotocamera digitale è presente un sensore che è in grado di catturare in ogni pixel uno solo dei tre canali, attraverso un Bayer pattern, gli altri dati sono ottenuti interpolando i valori disponibili. Non sono usati tre sensori, uno per ogni canale, per un fattore di dimensione e delicatezza del setup di allineamento.



(a) Immagine RAW



(b) Immagine RGB

Figura 6: Esempio di algoritmo di demosaicing

## 2. Color constancy

Implementa un algoritmo in grado di mantenere il colore percepito degli oggetti costante relativamente a diverse illuminazioni. Si cerca di individuare il colore dell'illuminante nella scena, regolando di conseguenza gli altri colori. In figura 7 è mostrato un esempio di algoritmo di color constancy.

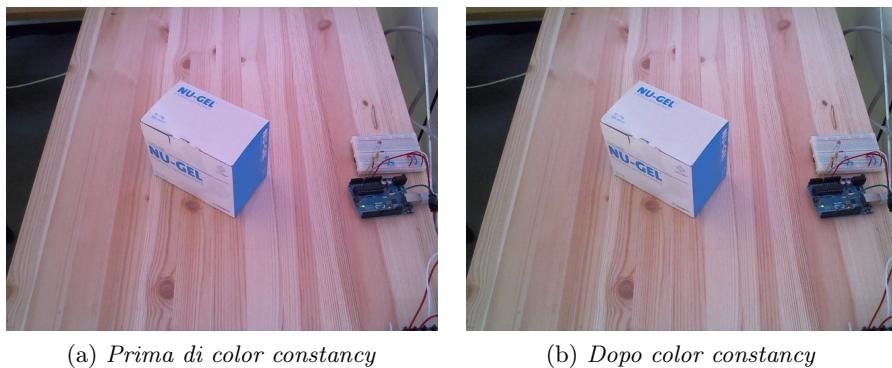


Figura 7: Esempio di algoritmo di color constancy

## 3. Color mapping

Dopo il processo di color constancy i colori acquisiti non sono ancora come quelli percepiti dall'osservatore. Questo è dovuto al fatto che le curve di trasmittanza del sensore non sono identiche a quelle dell'occhio umano, e la risposta del sensore al variare della quantità di luce acquisita non è lineare.

Per rendere i colori acquisiti dalla camera il più simile possibile a quelli percepiti dall'osservatore, si utilizza una matrice  $3 \times 3$  da moltiplicare all'immagine.

Questa matrice viene stimata utilizzando una Color Checker, che consiste in 24 tasselli colorati i cui valori sono noti in vari spazi colori.

La Color Checker viene fotografata con la camera, e i valori medi di ogni tassello colorato vengono estratti. La matrice  $3 \times 3$  viene stimata numericamente facendo una regressione tra i valori acquisiti dalla camera e i valori noti nello spazio colore di riferimento.

### 1.3.3 Ricostruzione 3D

In questa tesi ci si è occupati di scegliere la posizione delle camere e di sfruttare algoritmi di ricostruzione 3d per calcolare il volume degli oggetti. Scelte, metodi usati e motivazioni saranno analizzati esaustivamente nel capitolo 3.

### 1.4 Caso di studio

Dopo la divisione dei compiti e la scelta della struttura del dispositivo, siamo giunti alla decisione di usare, come oggetti di test e studio, delle scatole di medicinali, come quelle in figura 8, o scatole di dimensioni simili. Esse possono essere di qualsiasi colore, caratteristica che non influisce sull'esecuzione del flow implementato. Al contrario la forma di un parallelepipedo retto è un vincolo che l'oggetto deve avere, per motivi che saranno spiegati nel capitolo 3.



Figura 8: Oggetti usati come caso di studio

## **1.5 Organizzazione della tesi**

Nel capitolo 2 saranno descritte velocemente alcune delle tecniche principali per quanto riguarda la ricostruzione 3D, cercando di riassumere brevemente i pro e contro di queste, e motivando le scelte che hanno portato all’implementazione di una in particolare. Questa tecnica sarà descritta esaurientemente nel capitolo 3. Risultati e possibili miglioramenti saranno analizzati nel capitolo 4. Infine nel capitolo 5 sarà riassunto il lavoro svolto e verranno indicati alcuni possibili lavori futuri.

# Capitolo 2

## Stato dell'arte

Per un essere umano è semplice intuire la distanza o la forma di un oggetto, mentre la percezione nel contesto lavorativo per una macchina è molto più complessa. Per facilitare questa comprensione si integrano sensori, strumenti in grado di catturare immagini, suoni, o altre forme di radiazione per poi inviare i dati ad una macchina, che eseguendo complicati algoritmi è in grado di estrarre informazioni contenute in quei dati.

In questo capitolo verranno mostrate le tecnologie principali utilizzate per l'analisi di oggetti fisici, al fine di ottenere informazioni 3D su di essi. Saranno inoltre riassunti vantaggi e svantaggi di ognuna, per motivare le scelte fatte nel progetto.

Nonostante le diverse tecniche esistenti l'obiettivo di ognuna è acquisire dei dati, sotto diverse forme, per poi computare una nuvola di punti (Point Cloud), nella quale ogni elemento è costituito da una terna di coordinate (x, y, z) e nel loro insieme rappresentano la superficie dell'oggetto studiato.

Il processo di acquisizione può essere sviluppato sfruttando tecnologie molto differenti tra di loro, scelte in base al contesto d'uso: si va da strumenti ottici ad acustici, da laser a radar, e infine a tecniche di contatto. Questi dati possono essere utilizzati per diversi scopi: analisi, controllo qualità, animazione, reverse engineering, reverse modeling e molti altri. In questo progetto vengono analizzati per calcolare le dimensioni delle scatole prese in esame e calcolarne il volume.

Le diverse tecnologie si possono dividere in due categorie, come si vede dal grafico in figura 9: le tecniche di contatto e quelle di non contatto. A loro

volta quelle di non contatto si dividono in due aree: attivo e passivo, le prime sfruttano l'emissione di radiazione luminose artificiali, le seconde lavorano sulle radiazioni ambientali naturali[3].

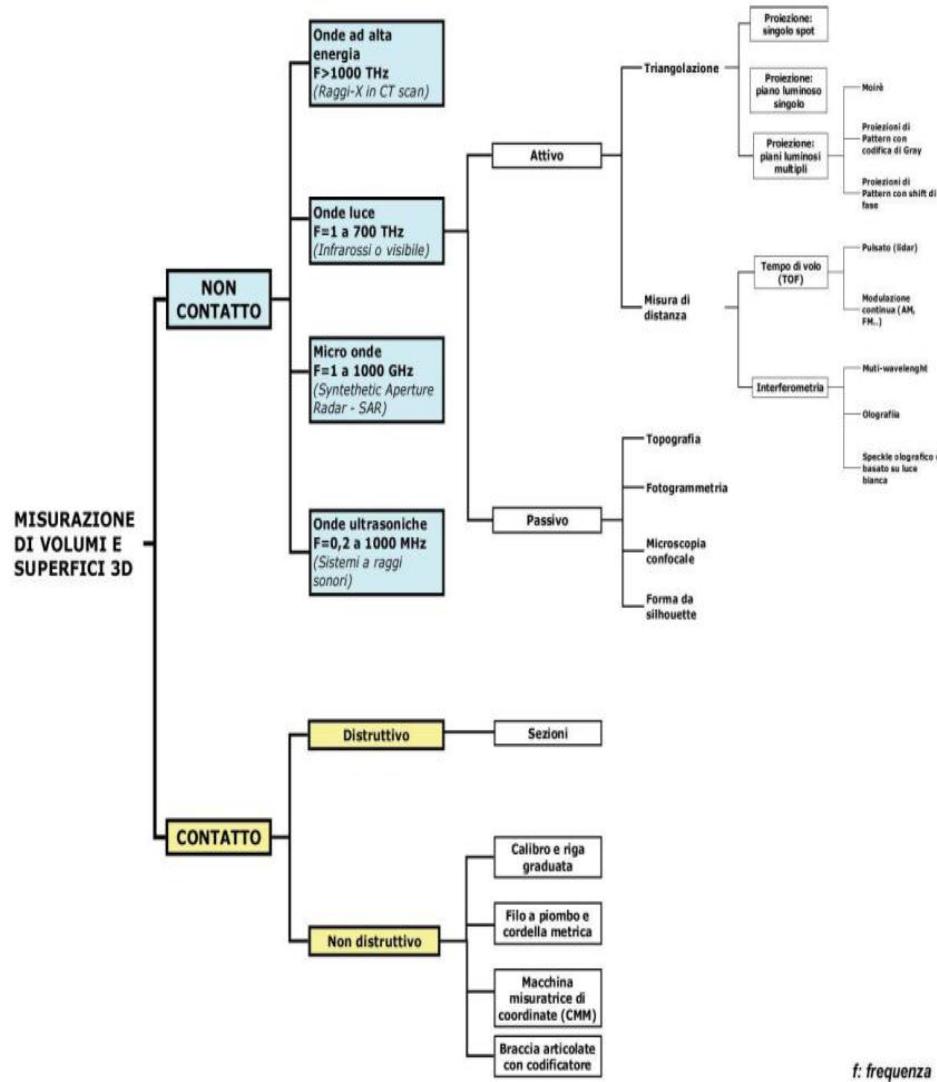


Figura 9: Classificazione tecniche di misurazione 3D.

## 2.1 Tecnologie a contatto

Gli strumenti che sfruttano il contatto con l'oggetto di studio si chiamano CMM(Coordinate Measuring Machine), e come sensore di acquisizione usano una sonda montata su un braccio rigido, come mostrato in figura 10, oppure articolato, in grado di seguire in modo automatico la superficie dell'oggetto[4].

Partendo dall'informazione sulla posizione della sonda nello spazio si è in grado di recuperare i dati tridimensionali dell'oggetto con grande accuratezza, nell'ordine di 1 um. Altro vantaggio di questa tecnologia è l'indipendenza dal tipo di superficie, principalmente non si riscontrano problemi di riflessione della luce. A svantaggio si ha un elevato tempo di elaborazione e, se usata su oggetti fragili, il rischio che la superficie venga danneggiata. Per questi motivi sono tecnologie usate soprattutto a livello industriale.



Figura 10: Esempio di CMM.

## 2.2 Tecnologie senza contatto attive

Le tecniche attive si basano sull'uso di una coppia sorgente-sensore[5]. La sorgente emette una radiazione (luce, ultrasuoni, raggi X) e il sensore acquisisce il segnale di ritorno riflesso dalla superficie dell'oggetto per determinarne la forma; questa tecnologia prende il nome di scanner.

Uno scanner può essere definito come uno strumento in grado di registrare le coordinate tridimensionali dei punti di una porzione di superficie, in modo automatico e con elevata densità.

Oltre a questo strumento ci sono tecniche che non prevedono l'uso di uno scanner, ma solamente una regolazione della luce artificiale emessa dalla sorgente.

Una possibile classificazione, per quanto riguarda la classe degli scanner, si può basare sul principio della misura della distanza, distinguendo tra distanziometri, sistemi a triangolazione ottica e a luce strutturata.

Queste tre classi saranno descritte brevemente nelle prossime 3 sotto-sezioni.

### 2.2.1 Distanziometri

In questi sistemi possiamo distinguere due principi di funzionamento: a tempo di volo e a misura di fase[6]. La differenza fondamentale è come le due tecniche sfruttano una radiazione per ricavare la distanza di un oggetto.

- Il tempo di volo (TOF, dall'inglese Time Of Flight) indica la misura del tempo impiegato da un oggetto, una particella o onda elettromagnetica per percorrere una certa distanza in un mezzo determinato.

Nell'ambito della misura, il sensore usato è di tipo ottico e sfrutta la luce per riuscire ad estrapolare l'informazione della distanza di un oggetto della scena osservata.

Il sensore emette una serie di impulsi luminosi e rileva quelli riflessi dall'oggetto target. Tali impulsi vengono poi convertiti in segnali elettrici e quindi elaborati per individuare la differenza temporale fra luce emessa e quella riflessa, come mostrato in figura 11, quindi il tempo che impiega la radiazione a percorre due volte la distanza sensore-oggetto.

Mediante tale parametro si deduce la distanza dell'oggetto.

Infatti nota la velocità con cui la radiazione si propaga nel mezzo di trasmissione la distanza è:

$$d = \frac{t*c}{2}$$

Dove t indica il tempo e c la velocità.

Il vantaggio di questa tecnologia è la possibilità di acquisire ampie superfici, ad una distanza elevata. Si presta quindi per misurare la distanza di edifici, siti archeologici, e qualsiasi oggetto di grande dimensioni.

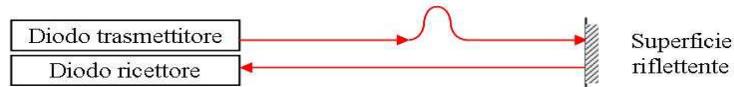


Figura 11: Principio sistemi a tempo di volo.

- Nei sistemi di misura di fase lo scanner emette un'onda modulata e la distanza è calcolata confrontando la fase dell'onda emessa con quella ricevuta dopo la riflessione sulla superficie dell'oggetto, figura 12. Questa classe di strumenti ha generalmente una portata più limitata di quelli a tempo di volo, ma una velocità di scansione nettamente superiore. Per questi motivi alcuni scanner a differenza di fase sono utilizzati in applicazioni dinamiche, montati su piattaforme in movimento quali auto o treni per esempio.

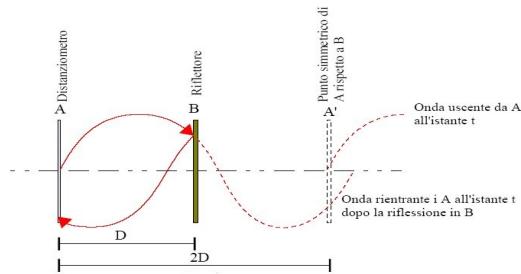


Figura 12: Principio sistemi a differenza di fase.

## 2.2.2 Sistemi a triangolazione ottica

I sistemi a triangolazione ottica[7] sono costituiti da un emettitore di luce, tipicamente un piano laser, proiettato sull'oggetto così da estrarne il profilo. A questo emettitore viene connesso rigidamente una telecamera, a distanza nota, e con un angolo tra il piano di luce e l'asse ottico noto. A seconda della distanza reciproca a cui sono posti l'oggetto e lo scanner la linea riflessa sul sensore assumerà una posizione differente.

Il sistema è posto in maniera che la sorgente, il centro di proiezione sul sensore e il punto luminoso riflesso dalla superficie vengano a formare un triangolo, figura 13.

Attraverso la conoscenza di una serie di parametri noti a priori (baseline  $b$ , angolo di incidenza del laser sullo specchio e lunghezza focale) e grazie ad alcune relazioni trigonometriche è possibile risalire alla distanza  $z$  tra lo strumento e l'oggetto.

Per poter ricostruire esattamente la superficie dell'oggetto è necessario muovere l'emettitore del laser o facendo ruotare il sistema intero intorno ad un asse.

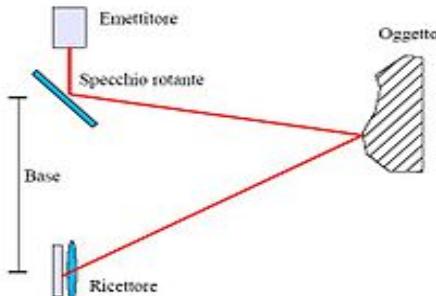


Figura 13: Posizione relativa laser-sensore-oggetto nella triangolazione.

### **2.2.3 Sistemi a luce strutturata**

I sistemi a luce strutturata[8] si basano su sensori che condividono il principio di triangolazione sopra menzionato, ma invece di scansionare la superficie proiettano schemi bidimensionali di luce sugli oggetti di studio. Solitamente questi schemi sono rappresentati da bande di luce che producono linee luminose sul corpo colpito. Queste bande in base all'inclinazione del punto colpito saranno più o meno distorte, se osservate da prospettive diverse. In base alle distorsioni del pattern si è in grado di ricavare la distanza del punto.

## **2.3 Tecnologie senza contatto passive**

Questo gruppo di tecniche prende il nome di metodi passivi perché non si interagisce attivamente con l'oggetto analizzato, proiettando fasci di luce per esempio, ma si sfrutta l'illuminazione naturale presente nell'ambiente e riflessa sull'oggetto, modificandola alcune volte. Tipicamente il sensore utilizzato in queste tecniche è una fotocamera e in input si hanno una, o più, immagini digitali o un video[5].

### **2.3.1 Fotogrammetria e stereo vision**

La fotogrammetria[9] ha come obiettivo quello di recuperare informazioni sulla posizione, l'orientazione, la forma e le dimensioni di oggetti o scene, senza entrare fisicamente in contatto con essi. Solitamente la tecnica principale consiste nell'analizzare una o più immagini scattate in condizioni particolari. Per ottenere l'informazione sulla profondità dei punti nella scena è necessario stabilire una relazione geometrica fra i punti immagine e la scena reale, con cui poter successivamente triangolare il punto fisico e la sua proiezione nelle immagini. Nel caso si utilizzino due camere per acquisire le immagini si parla di Binocular Stereo Vision.

I passaggi principali per ottenere una nuvola di punti con questa tecnica sono:

1. L'acquisizione delle immagini in modo sincronizzato
2. La calibrazione interna ed esterna delle camere
3. La risoluzione del problema della corrispondenza
4. Triangolazione per determinare la profondità dei punti

I vari passaggi saranno spiegati in dettaglio nel capitolo 3.

### 2.3.2 Shape from silhouette

#### Shape from texture

L'idea è di sfruttare l'informazione sulla disposizione delle texture dell'oggetto, per ricostruire l'orientazione della superficie[10]. Una texture è il modo in cui si dispongo pattern spaziali su di una superficie. Gli elementi che si ripetono, i texel, devono essere abbastanza piccoli da non poter essere distinguibili come oggetti separati. Questi elementi modificano in modo prospettico le proprie dimensioni e forme, in base all'orientazione e alla forma della superficie. Per esempio dato un texel circolare, su di una superficie planare, questo può avvicinarsi ad un'ellisse, in base all'orientazione del piano.

#### Shape from shading

Shape from shading [11] richiede l'acquisizione dell'oggetto da un singolo angolo di visione, con la variazione della posizione della sorgente luminosa, che provoca un cambiamento nell'ombreggiatura sulla superficie del target. Le informazioni sulla distanza dei punti sono ottenute analizzando una mappa, che indica quanto un punto della superficie riflette la luce incidente, ottenuta dalle diverse immagini raccolte. Il problema consiste quindi nel ricostruire la forma di un oggetto tridimensionale a partire da più fotografie a livelli di grigio, figura 14. Per farlo si devono estrapolare le informazioni nascoste nei valori delle immagini: punti più scuri indicano una superficie inclinata rispetto alla sorgente di luce, in quanto riflette meno luce, al contrario punti più chiari indicano parti meno inclinate rispetto la fonte luminosa. Combinando queste informazioni si riesce ad associare ad ogni punto una

altezza, in base all'intensità del pixel.

Per poter matematizzare il problema è necessario basarsi sui seguenti presupposti: l'oggetto deve essere di un solo colore, o comunque molto uniforme, bisogna conoscere con esattezza la posizione dell'unica fonte luminosa e infine bisogna supporre che il sensore di acquisizione sia posto sufficientemente lontano dall'oggetto.

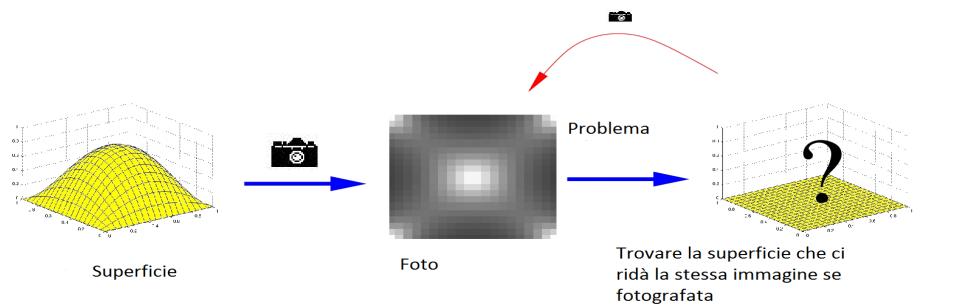


Figura 14: Problema shape from shading.

### Shape from shadow

Shape from shadows è una variante sei sistemi a luce strutturata. Il modello 3D dell'oggetto è costruito catturando l'ombra di un oggetto conosciuto e proiettato sul target, mentre la luce che lo colpisce è in movimento.

### 2.3.3 Photometric stereo

Photometric stereo [12] può essere vista come un'evoluzione della tecnica shape from shading. In questo caso il punto di osservazione è sempre il medesimo e ciò presenta un grosso vantaggio sia in termini di costo, perché in questo caso è sufficiente una sola fotocamera, ma anche computazionali nella misura in cui non sussiste il problema della corrispondenza tipico della fotometria binoculare, come mostrato in figura 15. Ogni singolo frame viene infatti acquisito illuminando gli oggetti che si vogliono ricostruire da diverse angolazioni. In altri termini in questa circostanza è la sorgente luminosa che viene spostata intorno all'oggetto. La finalità è dunque quella di studiare il comportamento fotometrico dell'oggetto, eccitato in condizioni differenti, ricostruendo una mappa dei gradienti e una mappa delle normali che ci consentono di stabilire l'orientazione punto per punto dell'oggetto che può essere poi in un secondo momento utilizzata per ricostruire la superficie stessa.

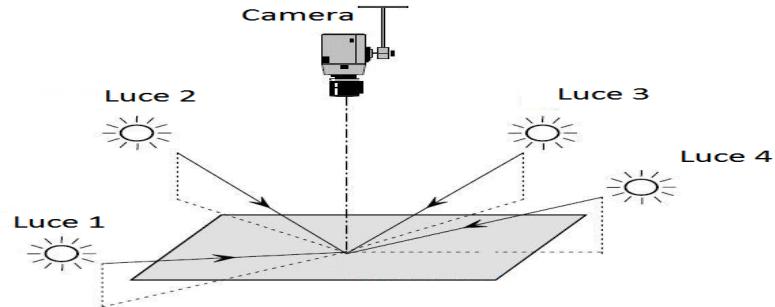


Figura 15: Problema shape from shading.

## 2.4 Tabella riassuntiva

Nella figura 16 è rappresentata la tabella che riassumo vantaggi e svantaggi di ogni tecnica.

	Vantaggi	Svantaggi
Sistemi a contatto	Grande accuratezza Nell'ordine di 1um	Lentezza di elaborazione Costo Rischio di danneggiare superfici
Sistemi distanziometri	Grandi range di misurazione Buon rate di acquisizione Indipendente dalle condizioni di luce	Costo Accuratezza inferiore a triangolazioni a distanze ravvicinate
Sistemi a triangolazione ottica	Semplicità Perfomance indipendenti dalle condizioni di luce Alto tasso di acquisizione	Mancanza di dati in corrispondenza di occlusioni e ombre Gamma e volume di misurazioni limitati Costo
Sistemi a luce strutturata	Perfomance indipendenti dalle condizioni di luce Alto tasso di acquisizione	Mancanza di dati in corrispondenza di occlusioni e ombre Costo
Shape from shading	Semplice e poco costoso	Bassa accuratezza
Shape from shadows	Poco costoso	Bassa accuratezza
Shape from texture	Semplice e poco costoso	Bassa accuratezza
Photometric stereo	Semplice e poco costoso	Bassa accuratezza
Stereo vision	Semplice e poco costoso Grande accuratezza con target ben definiti	Computazione esigente Limitato a scene ben definite Basso rate di acquisizione

Figura 16: Vantaggi e svantaggi delle tecniche di ricostruzione.

# Capitolo 3

## Calcolo del volume

### 3.1 Architettura e componenti

Inizialmente l'obiettivo era quello di sfruttare due camere dalle quali poter ottenere il formato RAW dell'immagine, questo non è stato possibile, quindi sono state utilizzate due fotocamere USB, mostrate in figura 17, semplici da collegare e posizionare sulla struttura del dispositivo.



Figura 17: Camere USB.

Il codice è stato scritto in Python e sono state utilizzate le seguenti librerie:

- **OpenCV** (Open Source Computer Vision Library) è una libreria software multi-piattaforma nell'ambito della visione artificiale in tempo reale. È un software libero, originariamente sviluppato da Intel. OpenCV è stato progettato per l'efficienza computazionale e con un forte focus sulle applicazioni in tempo reale. La libreria può trarre vantaggio dall'elaborazione multi-core. Il linguaggio di programmazione principalmente utilizzato per sviluppare con questa libreria è il C++, ma è possibile interfacciarsi anche attraverso il C, Python e Java. Noi per comodità abbiamo deciso di utilizzare OpenCV 3.0, utilizzando la libreria tramite comandi Python.
- **NumPy** è un'estensione open source del linguaggio di programmazione Python, che aggiunge supporto per vettori e matrici multidimensionali, e funzioni matematiche di alto livello con cui operare.
- **Matplotlib** è una libreria per la creazione di grafici per il linguaggio di programmazione Python e la libreria matematica NumPy, e permette di inserire grafici all'interno di applicativi.

## 3.2 Concetti preliminari

Prima di spiegare nei dettagli la tecnica implementata è utile avere un'idea generale delle proprietà su cui si basano le lenti di una camera digitale, quindi dei concetti di ottica che saranno ripresi nel corso del testo, e anche del funzionamento molto semplificato del sistema visivo umano.

Unendo questi diversi concetti sarà più facile comprendere l'idea generale sulla quale si fonda la tecnica sfruttata.

### 3.2.1 Basi ottica

Una **lente**, fondamentale in uno strumento di acquisizione ottico, è un dispositivo di trasmissione che concentra o disperde un raggio di luce attraverso il fenomeno della **rifrazione**.

La rifrazione è il cambiamento di direzione nella propagazione di un'onda attraverso due diversi mezzi di trasporto, che differiscono per differenti indici di rifrazione.

Questo fenomeno segue la legge di Snell:

$$\frac{\sin \theta_1}{\sin \theta_2} = \frac{v_1}{v_2} = \frac{n_1}{n_2} = \sqrt{\frac{e_2 u_2}{e_1 u_1}}$$

Dove:

- $\theta_1$  è l'angolo di incidenza;
- $\theta_2$  è l'angolo di rifrazione;
- $v_1$  e  $v_2$  sono le due velocità di fase nei due materiali;
- $n_1$  e  $n_2$  sono i due indici di rifrazione dei due materiali;
- $e_1$  ed  $e_2$  indicano la costante dielettrica nei due mezzi di trasporto;
- $u_1$  ed  $u_2$  indicano il momento magnetico nei due mezzi di trasporto;

Queste proprietà sono mostrate in figura 18.

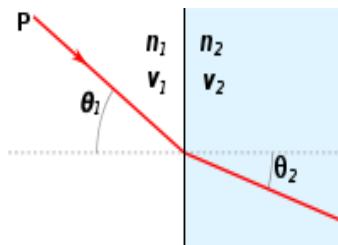


Figura 18: Schema rifrazione.

Caratteristica fondamentale di una lente è la **lunghezza focale**, che può essere definita come la misura di quanto il sistema converge o diverge i raggi di luce. Per un sistema ottico reale è la distanza del **punto focale**, in cui i raggi collimano (convergono), dalla lente, come mostrato in figura 19.

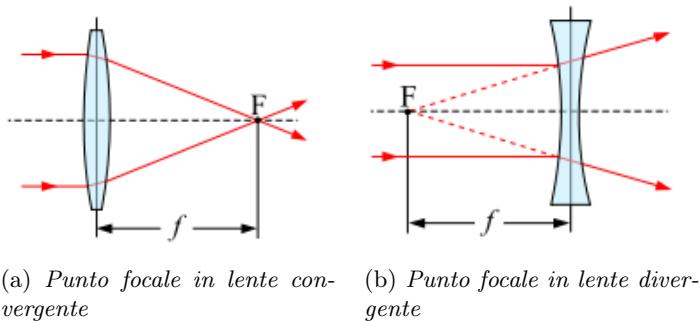


Figura 19: Schematizzazione punto focale

Quando una lente è usata per formare un'immagine la distanza  $u$  dell'oggetto dalla lente, la distanza  $v$  della lente dal piano su cui si forma l'immagine e la lunghezza focale  $f$  sono correlate dalla seguente formula:

$$\frac{1}{f} = \frac{1}{u} + \frac{1}{v}$$

In una lente singola l'**asse ottico** è la linea coincidente con l'unica traiettoria dell'unico raggio di luce che non subisce deviazioni, come mostrato in figura 20. In sistemi di lenti passa per i centri di curvatura di tutti gli elementi di cui è composto il sistema.

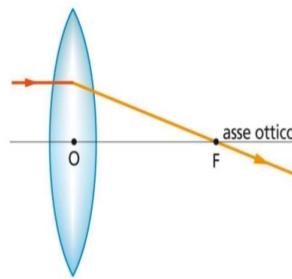


Figura 20: Schema asse ottico.

Alcune volte possono essere presenti delle **distorsioni geometriche** nelle immagini risultanti, nel momento in cui ai punti del piano oggetto di una certa figura corrispondono immagini che non costituiscono una scena simile. Questo fenomeno è dovuto in buona parte al fatto che le superfici esterne delle lenti sono sferiche e quindi le linee possono apparire curve nell'immagine, quando in realtà non lo sono. Esistono due tipi principali di distorsioni ottiche, figura21:

- **Barrel:** tipico di obiettivi grandangolari, avviene perché il campo di vista è maggiore dello spazio sul sensore e quindi deve essere ridimensionato. Gli effetti sono maggiori vicino ai bordi del frame, mentre al centro le linee non subiscono alcuna distorsione.
- **Pincushion:** è l'esatto opposto della barrel distortion, questa volta il campo di vista è minore del sensore e deve essere stirato.

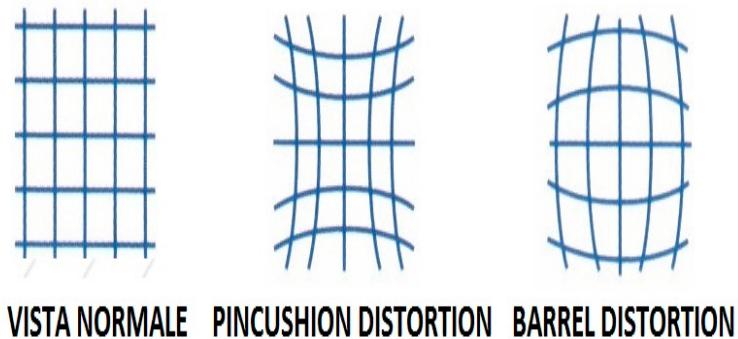


Figura 21: Esempio di distorsione ottica.

### 3.2.2 Basi visione binoculare umana

Per ogni essere vivente è fondamentale riuscire ad avere una concezione tridimensionale della scena osservata, elaborando le immagini 2d acquisite dagli occhi. L'uomo, e altri animali, dispongono di una visione binoculare, nella quale le informazioni vengono acquisite da due occhi e permettono un'efficiente stima della profondità.

Esistono molte caratteristiche naturali che ci permettono di stimare la distanza degli oggetti: texture, colore, ombre, l'esperienza nella conoscenza delle dimensioni e molte altre. Un meccanismo molto importante però è la stereopsi, cioè la capacità di analizzare e fondere le informazioni contenute in due immagini della stessa scena, prese da punti di vista differenti.

Gli occhi dell'uomo sono, in media, distanti 6.5 centimetri circa, e quindi hanno due punti di vista differenti. Questa particolarità fisionomica produce una certa disparità orizzontale nelle due immagini risultanti. Questa caratteristica è evidente studiando il fenomeno della **parallasse**, mostrata in figura 22, dove si nota come un oggetto fermo visto da due diverse angolazioni produce un movimento relativo con gli altri oggetti della scena, dando un effetto di movimento che non esiste. Inoltre questa sensazione di movimento relativo aumenta per oggetti più vicini al punto di vista. Quest'informazione sarà fondamentale per poter ricostruire l'oggetto in 3d.

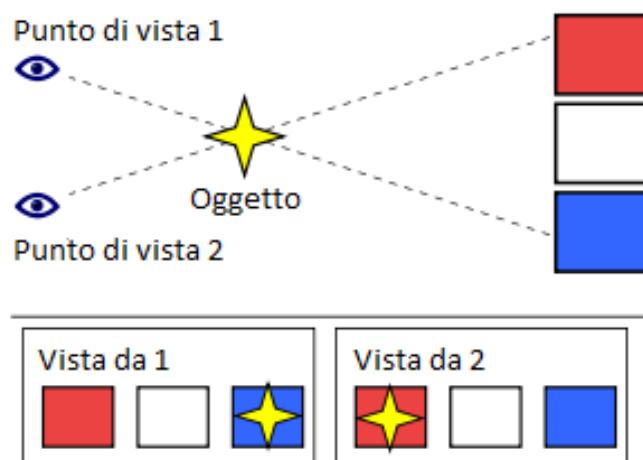


Figura 22: Schematizzazione della parallasse.

Rifacendoci alla figura 23 è possibile spiegare come il nostro cervello stima la distanza di tutti i punti presenti in una scena.

Supponiamo che un osservatore fissi il punto P, che Q sia un altro punto nello spazio, alla stessa distanza di P secondo l'osservatore e che QL, QR siano le immagini di Q sulle retine sinistra e destra. QL e QR sono punti corrispondenti sulle due retine, perché rappresentano lo stesso punto della scena ma proiettata da due angolazioni di vista differenti. Grazie a questa corrispondenza il nostro cervello elabora tutti i punti di una scena, identificando i diversi punti corrispondenti e creando delle superfici immaginarie, **horopter**[13], su cui posizionare tutti i punti con la stessa profondità.

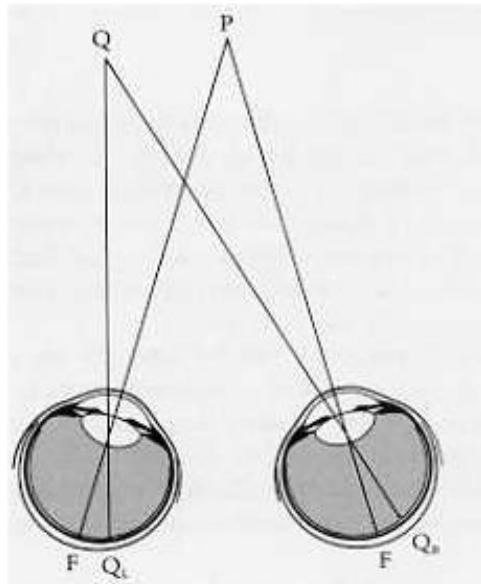


Figura 23: Schematizzazione della proiezione di un punto sulla retina dei due occhi.

Questa tecnica può essere simulata come vedremo in seguito, basandosi su delle considerazioni geometriche.

### 3.3 Stereo vision

In questa sezione saranno analizzati i passaggi principali eseguiti per ottenere una nuvola di punti partendo da una coppia di immagini[14], scattate istantaneamente da un sistema stereo di fotocamere.

Gli step necessari sono :

1. scegliere la posizione delle camere, per ottenere un sistema il più semplice possibile;
2. calibrare ogni singola camera del sistema;
3. eliminare la distorsione nelle immagini se presente;
4. rettificare le immagini;
5. risolvere il problema della corrispondenza tra le due acquisizioni;
6. tramite la triangolazione proiettare i punti nello spazio 3D.

#### 3.3.1 Posizione camere

La posizione delle camere è stata scelta basandosi su diverse considerazioni, che saranno spiegate più dettagliatamente nel proseguo del capitolo:

- devono essere sufficientemente vicine agli oggetti da ricostruire;
- devono essere il più possibile vicine orizzontalmente e parallele;
- devono avere un'inquadratura tale da acquisire la maggior parte possibile della superficie dell'oggetto.

#### 3.3.2 Calibrazione camere

Il primo passaggio nella nostra pipeline è la calibrazione delle camere. Durante questo step l'obiettivo è trovare una relazione geometrica tra i punti fisici, quindi i punti della scena osservata espressi nelle coordinate del sistema di riferimento che chiameremo mondo, e i corrispettivi punti nell'immagine acquisita, per essere in grado di conoscere la posizione di un punto nell'immagine e nella scena, avendo solamente le coordinate di uno dei due.

Per definire questa relazione dobbiamo essere in grado di calcolare alcuni parametri che rappresentano il nostro sistema:

1. **I parametri estrinseci** delle camere, che rappresentano una trasformazione rigida dalle coordinate 3D del sistema mondo alle coordinate 3D del sistema di riferimento camera.
2. **I parametri intrinseci** delle camere, che legano le coordinate 3D nel sistema camera alle coordinate 2D dell'immagine.

Prima di spiegare nel dettaglio i vari passaggi è utile analizzare la geometria su cui si basa l'acquisizione della luce in una camera, quindi sarà descritto un modello ideale di camera, la **pinhole**[15].

Rifacendoci alla figura 24, la pinhole camera è rappresentata come un muro immaginario, con un piccolo foro al centro. Tutti i raggi di luce riflettono contro la superficie a parte un unico raggio che attraversa il foro. Non rappresenta la reale acquisizione di un'immagine ma è utile per capire come si comporta ogni raggio appartenente ad un fascio di luce. L'unico raggio che attraversa il foro è poi proiettato su un altro piano, detto piano-immagine o piano di proiezione.

In questo sistema le dimensioni dell'immagine in relazione alla distanza dell'oggetto sono definite da un singolo parametro: la **lunghezza focale**, definita nel capitolo 3.2.1. Per il nostro modello ideale la distanza dal foro al piano immagine corrisponde alla lunghezza focale.

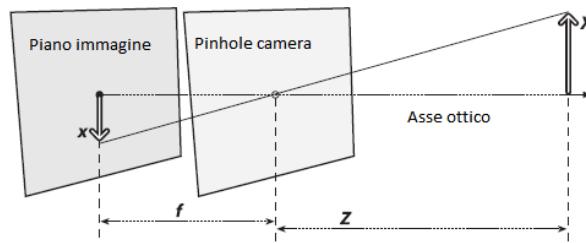


Figura 24: Schematizzazione di una pinhole camera.

Nell'immagine 24 , $f$  è la lunghezza focale,  $Z$  è la distanza della camera dall'oggetto,  $X$  è l'altezza dell'oggetto e  $x$  è lo stesso proiettato sul piano immagine. Confrontando i due triangoli simili si ha:

$$-\frac{x}{f} = \frac{X}{Z} \text{ da cui } -x = f \frac{X}{Z}$$

Ora riorganizziamo il nostro modello in una forma equivalente ma con una matematica più semplice e una struttura più simile ad una camera reale, come mostrato in figura 25. In questo sistema invertiamo la posizione della pinhole camera e del piano immagine, ora il punto proiettato appare in alto sul piano. Il foro della pinhole camera, ora, è rappresentato del centro di proiezione, che rappresenta il punto per cui tutti i raggi convergono. Il punto di intersezione del piano immagine con l'asse ottico prende il nome di **punto principale**, le dimensioni dell'oggetto sul piano immagine sono le stesse del vecchio modello e il piano immagine è esattamente ad una distanza  $f$  dal centro di proiezione. L'unica differenza sta nel modello matematico, non trovando più il segno negativo sulla  $x$ :

$$x = f \frac{X}{Z}$$

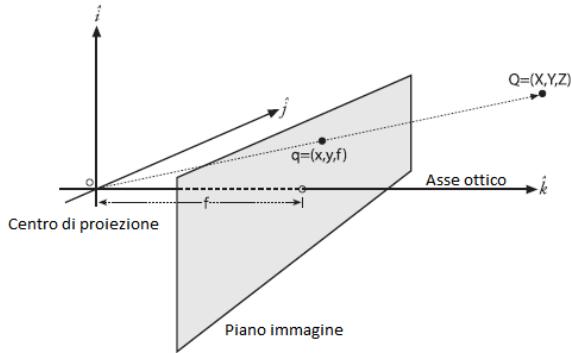


Figura 25: Schematizzazione inversione pinhole.

Tornando alla struttura del modello si potrebbe pensare che il punto principale equivalga al centro dell'immagine, sarebbe vero se si potesse attaccare il sensore con grande accuratezza, ma in realtà non è così, e quindi bisogna introdurre due nuovi parametri,  $c_x$  e  $c_y$ , che rappresentano un possibile scostamento del centro del piano immagine dall'asse ottico. Quindi, unendo la relazione precedentemente trovata e questi due nuovi parametri, sappiamo che un punto nello spazio, di coordinate  $(X,Y,Z)$  viene proiettato sul piano immagine, in coordinate  $x_{screen}, y_{screen}$  secondo queste due equazioni:

$$x_{screen} = f_x \frac{X}{Z} + c_x$$

$$y_{screen} = f_y \frac{Y}{Z} + c_y$$

Nelle due nuove relazioni compaiono due differenti lunghezze focali,  $f_x$  e  $f_y$  perché realmente i pixel non sono quadrati ma leggermente rettangolari e quindi questi due differenti parametri sono ottenuti dividendo la lunghezza focale  $f$  per le dimensioni dei pixel,  $s_x$  e  $s_y$ :

$$f_x = \frac{f}{s_x}$$

$$f_y = \frac{f}{s_y}$$

### Parametri intrinseci

I parametri intrinseci sono dei valori che permettono di mappare un punto  $Q$  di coordinate  $(X, Y, Z)$ , in coordinate del sistema camera, nel piano immagine di coordinate  $(x, y)$ . Questa relazione è una trasformazione prospettica, il cui punto di fuga è il centro di proiezione della camera, come mostrato in figura 25.

Quando si lavora con queste trasformazioni è utile sfruttare le coordinate omogenee, perché così possono essere calcolate come semplici moltiplicazioni tra matrici. Una coordinata omogenea rappresenta un punto di uno spazio a  $n$  dimensioni con un vettore di  $n + 1$  elementi[16].

Nel nostro caso il piano immagine è lo spazio proiettivo e ha due dimensioni, quindi rappresenteremo i punti sul piano con un vettore  $q = (q_1, q_2, q_3)$  da cui è possibile tornare alle coordinate non omogenee dividendo per  $q_3$ . Questo ci permette di riunire i nostri parametri in un'unica matrice e rappresentare la proiezione del punto dal mondo al piano immagine secondo la seguente equazione:

$$q = MQ \text{ dove } q = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, M = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, Q = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Riassumendo, i parametri intrinseci sono la lunghezza focale e il centro ottico ,che rimangono fissi per la camera per ogni configurazione scelta.

## Parametri estrinseci

I parametri estrinseci permettono di stabilire una relazione geometrica tra i punti 3D della scena nel sistema mondo e gli stessi punti ma nel sistema della camera. L'obiettivo è quindi quello di trovare una matrice di rototraslazione che permetta di passare tra questi due sistemi di riferimento come mostrato in figura 26.

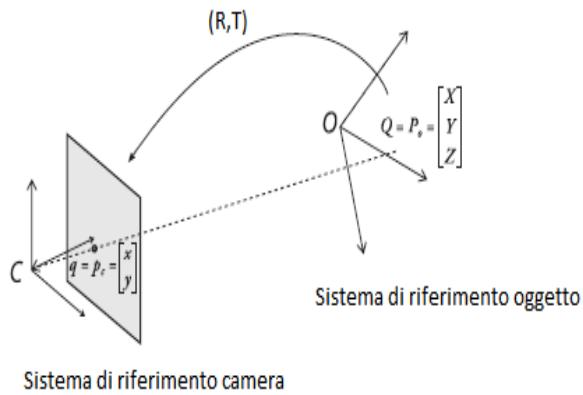


Figura 26: Schematizzazione parametri estrinseci.

Analizziamo nel dettaglio come avviene una rototraslazione da due sistemi di riferimento. Per ogni immagine di un oggetto possiamo descrivere la posa di questo rispetto al sistema di coordinate camera in termini di rotazione e traslazione. In generale una rotazione può essere descritta come una moltiplicazione di un punto, quindi un vettore di coordinate, per una matrice quadrata dalle dimensioni appropriate, grazie alla quale possiamo descrivere la posizione di un punto secondo un diverso sistema di coordinate. Ruotare un sistema di riferimento di un angolo  $\theta$  equivale a ruotare il nostro punto dello stesso angolo intorno all'origine del sistema. Una rotazione in tre dimensioni può essere decomposta in tre rotazioni bidimensionali, mostrata in figura 27, una per ogni asse di riferimento.

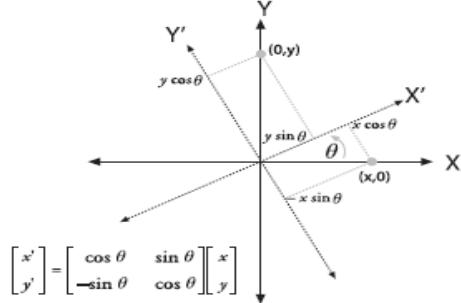


Figura 27: Rotazione bidimensionale.

Se ruotiamo intorno all'asse x,y,z in sequenza secondo degli angoli  $\alpha, \beta$  e  $\theta$ , il risultato è una matrice di rotazione R data dal prodotto delle tre matrici  $R_x(\alpha), R_y(\beta)$  e  $R_z(\theta)$ , dove:

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & \sin \alpha \\ 0 & -\sin \alpha & \cos \alpha \end{bmatrix},$$

$$R_y(\beta) = \begin{bmatrix} \cos \beta & 0 & -\sin \beta \\ 0 & 1 & 0 \\ \sin \beta & 0 & \cos \beta \end{bmatrix},$$

$$R_z(\theta) = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Quindi  $R = R_x(\alpha)R_y(\beta)R_z(\theta)$ .

Il vettore di traslazione invece rappresenta lo spostamento tra i due sistemi di riferimento rappresentato dalla differenza tra le due origini:

$$T = O_{oggetto} - O_{camera}.$$

Quindi le coordinate di un punto  $P_{oggetto}$  sono espresse nel sistema di riferimento camera secondo la seguente equazione:

$$P_{camera} = R(P_{oggetto} - T).$$

## Pattern di calibrazione

Per stimare queste matrici è necessario acquisire le immagini di un pattern noto, cioè del quale conosciamo dimensioni e orientamento, e in cui siano facilmente identificabili alcuni punti, nel nostro caso identificheremo i corners interni della scacchiera, figura 28. Guardando il pattern da diversi punti di vista è possibile computare la posizione e l'orientamento della camera, rispetto l'oggetto, e i parametri intrinseci.

Ecco alcuni esempi delle immagini sfruttate per la calibrazione delle camere:



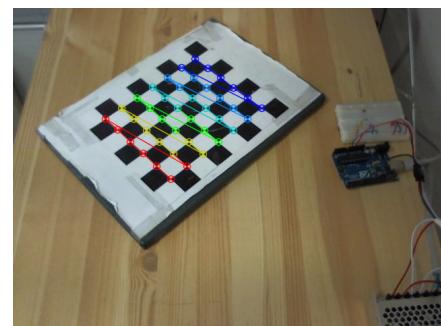
(a) *Vista sinistra*



(b) *Vista destra*



(c) *Vista sinistra*



(d) *Vista destra*

Figura 28: Identificazione corners interni della scacchiera nelle due viste.

## Omografia planare

Usando un oggetto di calibrazione planare, come la scacchiera, i punti del piano sono soggetti ad una trasformazione prospettica quando osservati da una lente e mappati su un altro piano, nel nostro caso il piano immagine. Questa trasformazione in computer vision è definita **omografia planare**, ed unisce l'insieme dei parametri, intrinseci ed estrinseci, per calibrare una camera. Usando le coordinate omogenee è possibile esprimere questa omografia in termini di moltiplicazioni di matrici.

Dato un punto  $Q = \begin{bmatrix} X & Y & Z & 1 \end{bmatrix}^T$ , nel sistema di riferimento mondo, e il punto corrispondente  $q = \begin{bmatrix} x & y & 1 \end{bmatrix}^T$ , sul piano immagine, la relazione è:

$$q = sHQ$$

Viene introdotto il parametro  $s$  perché la precedente equazione vale a meno di un fattore di scala. Nella figura 29, è mostrata graficamente l'omografia, che porta i punti dal sistema mondo al piano immagine.

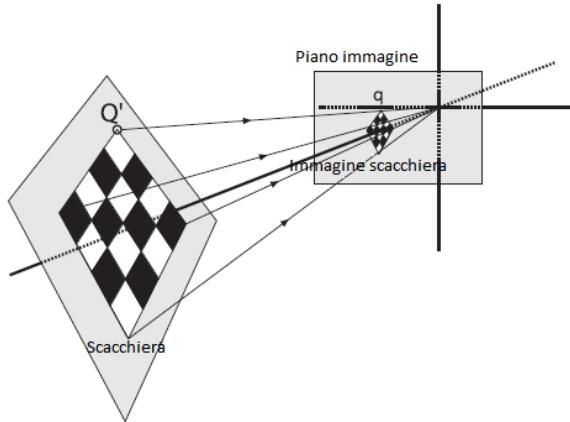


Figura 29: Schematizzazione omografia scacchiera.

La matrice  $H$  è composta da due parti, quella che rappresenta la proiezione e quella che rappresenta la trasformazione fisica dell'oggetto:

$$q = sMWQ, \text{ dove } W = \begin{bmatrix} R & t \end{bmatrix}, M = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

Senza perdita di generalità imponiamo che  $Z = 0$ , quindi riscrivendo la matrice di rotazione come composizione di vettori colonna,  $R = \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix}$ , la matrice di omografia diventa una  $3 \times 3$  perché il terzo vettore colonna sarebbe annullato.

Usando questa equazione e diverse immagini dello stesso pattern è possibile computare i parametri intrinseci ed estrinseci delle camere.

### Distorsione immagini

Nel modello ideale della pinhole camera non esistono distorsioni nelle immagini, ma in realtà la luce acquisita è molto maggiore grazie all'uso delle lenti, che però introducono delle distorsioni. Esistono due principali tipi di distorsione: radiale e tangenziale.

- La distorsione radiale è causata dalla forma sferica delle lenti e l'effetto principale è quello della barrel distortion, mostrato in figura 30.

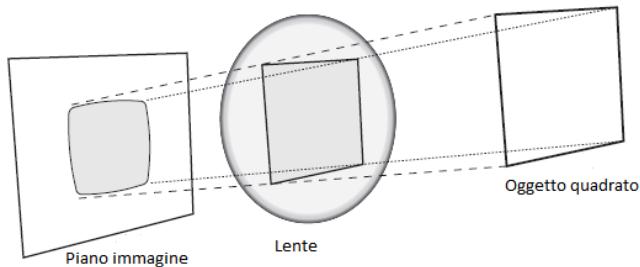


Figura 30: Distorsione radiale.

La distorsione è 0 nel centro ottico dell'immagine e aumenta verso i bordi. La correzione è descritta dalla seguente formula:

$$x_{corretto} = x(1 + k_1r^2 + k_2r^4 + k_3r^6)$$

$$y_{corretto} = y(1 + k_1r^2 + k_2r^4 + k_3r^6)$$

- La distorsione tangenziale è causata da errori di assemblaggio della camera nella quale le lenti non sono esattamente parallele al piano immagine, come mostrato in figura 31. Questa distorsione è caratterizzata da due parametri aggiuntivi  $p_1$  e  $o_2$ , secondo le seguenti equazioni:

$$x_{corretto} = x + [2p_1y + p_2(r^2 + 2x^2)]$$

$$y_{corretto} = y + [p_1(r^2 + 2y^2) + 2p_2x]$$

In totale abbiamo bisogno di cinque coefficienti per descrivere ed eliminare le distorsioni presenti nelle immagini.

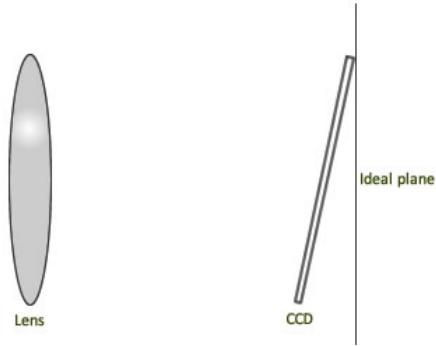


Figura 31: Causa distorsione tangenziale.

### 3.3.3 Risoluzione modello geometrico

La soluzione adottata implementa il metodo di Zhang [17], che analizzeremo di seguito.

#### Calibrazione di Zhang

In questo metodo sono noti:

- Il numero di corner interni della scacchiera.
- La lunghezza del lato dei quadrati che la compongono.
- L'orientamento della scacchiera che è rettangolare, quindi gli assi x e y sono facilmente riconoscibili.

Inizialmente imponiamo che nelle camere non sia presente alcun tipo di distorsione, mentre si risolve il modello rispetto gli altri parametri. Per ogni vista della scacchiera otteniamo una matrice di omografia  $H$ , discussa precedentemente. Riscrivendo la matrice come un vettore di colonne,  $H = [h_1 \ h_2 \ h_3]$ , possiamo porre:

$$H = [h_1 \ h_2 \ h_3] = sM [r_1 \ r_2 \ t]$$

Dove  $s$  è il fattore di scala,  $M$  è la matrice camera dei parametri intrinseci,  $r_1$  e  $r_2$  sono le colonne di rotazione e  $t$  è il vettore di traslazione.

Svolgendo l'equazione abbiamo:

$$h_1 = sMr_1 \text{ o } r_1 = \lambda M^{-1}h_1$$

$$h_2 = sMr_2 \text{ o } r_2 = \lambda M^{-1}h_2$$

$$h_3 = sMt \text{ o } t = \lambda M^{-1}h_3$$

Dove  $\lambda = \frac{1}{s}$ .

I vettori di rotazione,  $r_1$  e  $r_2$  sono ortogonali per costruzione, e a meno di un fattore di scala sono ortonormali. Questa proprietà implica due vincoli:

- Il prodotto dei due vettori è uguale a 0, quindi  $r_1^T r_2 = 0$ . Sapendo che dati due vettori  $a$  e  $b$  si ha:  $(ab)^T = b^T a^T$ , sostituendo  $r_1$  e  $r_2$  deriviamo il nostro primo vincolo:

$$h_1^T M^{-T} M^{-1} h_2 = 0$$

- La magnitudine dei due vettori di rotazione è uguale, quindi:

$$\|r_1\| = \|r_2\| \text{ o } r_1^T r_1 = r_2^T r_2$$

Sostituendo per  $r_1$  e  $r_2$  otteniamo il nostro secondo vincolo:

$$h_1^T M^{-T} M^{-1} h_1 = h_2^T M^{-T} M^{-1} h_2$$

Per semplificare le cose poniamo  $B = M^{-T} M^{-1}$ , e quindi:

$$B = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix}$$

Da cui:

$$B = \begin{bmatrix} \frac{1}{f_x^2} & 0 & \frac{-c_x}{f_x^2} \\ 0 & \frac{1}{f_y^2} & \frac{-c_y}{f_y^2} \\ \frac{-c_x}{f_x^2} & \frac{-c_y}{f_y^2} & \frac{c_x^2}{f_x^2} + \frac{c_y^2}{f_y^2} + 1 \end{bmatrix}$$

Usando la matrice  $B$ , i due vincoli precedentemente trovati hanno la forma generale  $h_i^T B h_j$ . Dato che  $B$  è simmetrica, può essere riscritta come un vettore  $b$  di dimensioni 1x6 e esplicitamente si ottiene:

$$h_i^T B h_j = v_{ij}^T b = \begin{bmatrix} h_{i1} h_{j1} \\ h_{i1} h_{j2} + h_{i2} h_{j1} \\ h_{i2} h_{j2} \\ h_{i3} h_{j1} + h_{i1} h_{j3} \\ h_{i3} h_{j2} + h_{i2} h_{j3} \\ h_{i3} h_{j3} \end{bmatrix}^T \begin{bmatrix} B_{11} \\ B_{12} \\ B_{22} \\ B_{13} \\ B_{23} \\ B_{33} \end{bmatrix}$$

Usando questa nuova definizione i due vincoli possono essere riscritti come:

$$\begin{bmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{bmatrix} b = 0$$

Collezionando  $n$  immagini della nostra scacchiera otteniamo  $n$  equazioni che possiamo unire:

$$Vb = 0$$

$V$  è una matrice  $2nx6$ . Se  $n >= 2$  questa equazione può essere risolta per  $b$  e per trovare i nostri parametri intrinseci ed estrinseci, ponendo  $\lambda = BB_{33} - (B_{13}^2 + C_Y(B_{12}B_{13} - B_{11}B_{23}))/B_{11}$ , si ha:

$$f_x = \sqrt{\lambda/B_{11}}$$

$$f_y = \sqrt{\lambda B_{11}/(B_{11}B_{22} - B_{12}^2)}$$

$$c_x = -B_{13}f_x^2/\lambda$$

$$c_y = (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2)$$

$$r_1 = \lambda M^{-1}h_1$$

$$r_2 = \lambda M^{-1}h_2$$

$$r_3 = r_1 X r_2$$

$$t = \lambda M^{-1}h_3$$

In questo modello non abbiamo considerato le possibili distorsioni introdotte a causa delle lenti, in particolare le distorsioni radiali. Poniamo che  $(u, v)$  siano le coordinate ideali del pixel, privo di distorsioni, e  $(u', v')$  i punti reali corrispondenti osservati nell'immagine con distorsione. I punti ideali si basano sulla proiezione in una pin-hole camera, aggiungendo la distorsione si ha:

$$u' = u + (u - c_x)[k_1r^2 + k_2r^4]$$

$$v' = v + (v - c_y)[k_1r^2 + k_2r^4]$$

Da queste due equazioni l'obiettivo è stimare i due parametri di distorsione radiale  $k_1$  e  $k_2$ , ed è possibile riscrivendo le due equazioni precedenti nella seguente forma:

$$\begin{bmatrix} (u - c_x)r^2 & (u - c_x)r^4 \\ (v - c_y)r^2 & (v - c_y)r^4 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} u' - u \\ v' - v \end{bmatrix}$$

Dati  $m$  punti in  $n$  immagini usate per calibrare le camere, possiamo unire queste equazioni e avere una  $2mn$  equazioni, che nella forma matriciale corrispondono a  $Dk = d$ , dove  $k = [k_1, k_2]^T$ , e risolvendo attraverso la tecnica dei minimi quadrati data da:

$$k = (D^T D)^{-1} D^T d$$

Trovati i due parametri di distorsione è necessario raffinare la stima degli altri parametri, secondo la maximum likelihood estimation, minimizzando la seguente funzione:

$$\sum_{i=1}^n \sum_{j=1}^m \|m'_{ij} - m'(A, k_1, k_2, R_i, t_i, M_j)\|^2,$$

dove  $m'(A, k_1, k_2, R_i, t_i, M_j)$  è la proiezione del punto  $M_j$  nell'immagine  $i$ , secondo la matrice di omografia e i parametri di distorsione trovati precedentemente.

### 3.3.4 Triangolazione

Sarà analizzato prima l'ultimo passaggio del processo per ottenere una nuvola di punti, per capire il motivo dei passaggi precedenti.

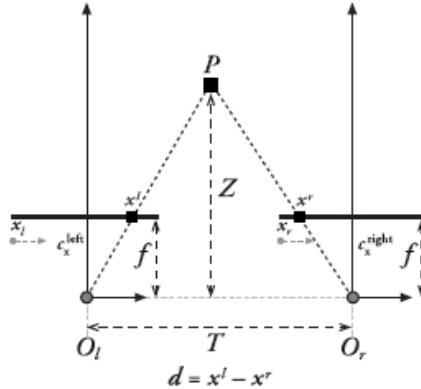


Figura 32: Modello ideale per triangolazione.

Assumiamo di avere un sistema come quello in figura 32, basato sulle seguenti supposizioni:

- le due immagini sono prive di distorsioni;
- le due camere sono posizionate in modo che i rispettivi piani immagine siano complanari, quindi con gli assi ottici paralleli;
- è conosciuta la distanza tra le camere  $T$ , chiamata baseline;
- le due camere hanno uguale lunghezza focale,  $f = f_{left} = f_{right}$ ;
- i punti principali  $c_x^{left}$  e  $c_x^{right}$  hanno le stesse coordinate in pixel nelle due immagini;
- le due immagini sono allineate per righe, quindi ogni punto della scena ha la medesima ordinata nelle due immagini;
- è possibile per ogni punto  $P$  della scena identificare le posizioni dei punti corrispondenti,  $p_{left}$  e  $p_{right}$ , nelle due immagini, con coordinate orizzontali  $x_{left}$  e  $x_{right}$ .

In questo modello ideale si nota come la profondità di un punto fisico P sia inversamente proporzionale alla disparità tra le proiezioni del punto nelle due immagini, dove con disparità si intende la distanza tra le coordinate orizzontali dei due punti immagine:

$$d = x^{left} - x^{right}.$$

Sfruttando quest'informazione è possibile ricavare la profondità del punto P, basandosi sulla seguente triangolazione:

$$\frac{T - (x^{left} - x^{right})}{Z - f} = \frac{T}{Z} \text{ da cui } Z = \frac{fT}{x^{left} - x^{right}}.$$

Dalla precedente equazione si capisce che la profondità Z e la disparità d non sono legati da una relazione lineare, come mostrato in figura 33.

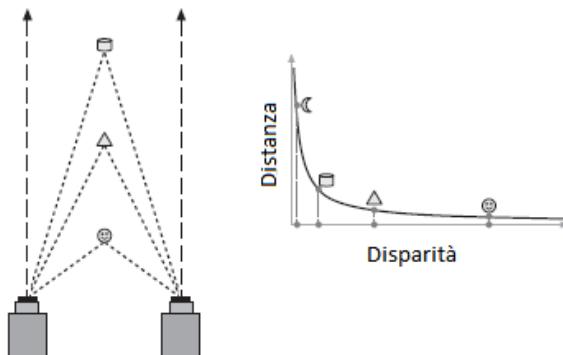


Figura 33: Legame tra disparità e profondità.

Quindi differenze di disparità hanno conseguenze diverse a seconda della distanza a cui avvengono:

- se i punti considerati sono vicini alle camere allora piccole differenze di disparità provocano grandi cambiamenti di profondità dei punti;
- al contrario se i punti considerati sono lontani dalle camere allora piccole differenze di disparità non provocano grandi cambiamenti di profondità dei punti.

La conseguenza principale è che nei sistemi stereo si ha una maggiore risoluzione della profondità con oggetti più vicini alle camere, dove con risoluzione si intende l'accuratezza della misura effettuata.

Sotto le assunzioni fatte inizialmente è relativamente semplice risolvere l'equazione per ricavare la profondità di ogni punto, ma nella realtà due camere non sono mai posizionate esattamente in quella configurazione, è quindi necessario trovare una relazione geometrica per mappare un set up reale nel nostro caso ideale. Questa relazione si trova attraverso una calibrazione stereo delle camere e la rettificazione successiva delle immagini.

### 3.3.5 Calibrazione stereo

L'obiettivo della calibrazione stereo è di trovare una matrice di rotazione  $R$  e un vettore di traslazione  $T$  che leggi la posizione delle due camere, come mostrato in figura 34, per ottenere due piani immagini complanari.

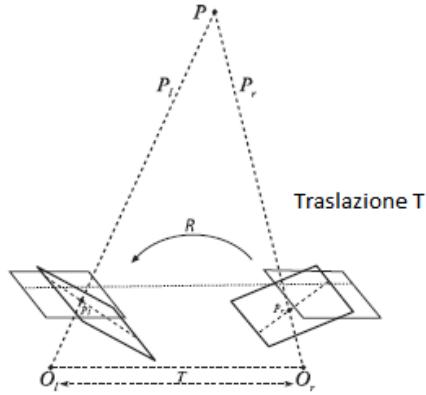


Figura 34: Schematizzazione della relazione tra le due camere.

Per ogni punto  $P$ , di coordinate omogenee  $(x,y,z,w)$  nel sistema mondo, possiamo sfruttare le matrici  $R_{left}$ ,  $R_{right}$ ,  $t_{left}$ ,  $t_{right}$  che legano i punti mondo al sistema di riferimento delle camere, secondo queste equazioni:

$$\begin{aligned} P_{left} &= R_{left}(P - T_{left}) \\ P_{right} &= R_{right}(P - T_{right}) \end{aligned}$$

Inoltre sempre dalla figura 34 si deduce che le due proiezioni di  $P$  sono legate dalla seguente relazione:

$$P_{left} = R^T(P_{right} - T)$$

dove  $R$  e  $T$  sono la matrice di rotazione e il vettore di traslazione tra le due camere.

Con queste tre equazioni, risolvendo per R e T, si ottengono le seguenti relazioni:

$$R = R_{right}(R_{left})^T$$

$$T = T_{right} - RT_{left}$$

dove gli unici parametri sconosciuti sono quelli cercati.

A causa di rumore nelle immagini ed errori di arrotondamento i valori di R e T saranno leggermente differenti per ogni coppia di immagini del pattern di calibrazione. Quindi inizialmente si prende il valore mediano tra quelli ottenuti e successivamente con algoritmo iterativo si cerca il l'errore minimo di proiezione dei corners per entrambe le camere.

### 3.3.6 Rettificazione

L'ultimo passaggio prima di cercare i punti corrispondenti nelle due immagini è la rettificazione delle immagini. L'obiettivo è proiettare i due piani immagine nello stesso piano, in modo che siano allineati per righe e complanari, ottenendo così il modello ideale mostrato in figura 35. Prima di descrivere l'algoritmo, è necessario analizzare la geometria che lega le due immagini acquisite da due punti di vista differenti, chiamata geometria epipolare.

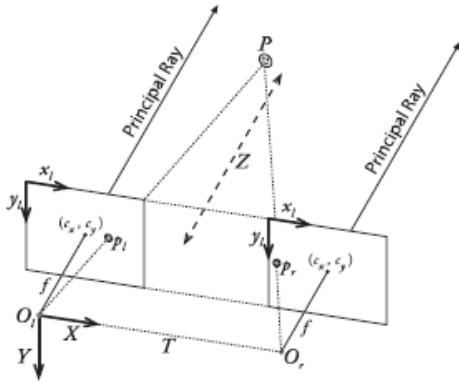


Figura 35: Set-up ideale fotocamere.

## Geometria epipolare

La geometria epipolare lega due modelli ideali di camere, pihole camera, attraverso dei punti chiamati **epipoli**. Riferendosi alla figura 36, definiamo gli elementi che caratterizzano questa geometria:

- per ogni camera si ha un diverso centro di proiezione,  $O_{left}$  e  $O_{right}$ , e due piani di proiezione,  $\Pi_{left}$  e  $\Pi_{right}$ ;
- il punto P ha una proiezione nei due piani,  $p_{left}$  e  $p_{right}$ ;
- $e_{left}$  e  $e_{right}$  sono gli epipoli, definiti come la proiezione di  $O_{right}$  e  $O_{left}$  nei piani immagine dell'altra camera:

$$e_{left} = P_{left}O_{right}$$

$$e_{right} = P_{right}O_{left}$$

dove  $P_{left}, P_{right}$  sono le matrici di proiezione delle due camere;

- il piano formato dal punto P e i due epipoli è chiamato **piano epipolare**;
- le linee  $p_{left}e_{left}$  e  $p_{right}e_{right}$  sono chiamate **linee epipolari**.

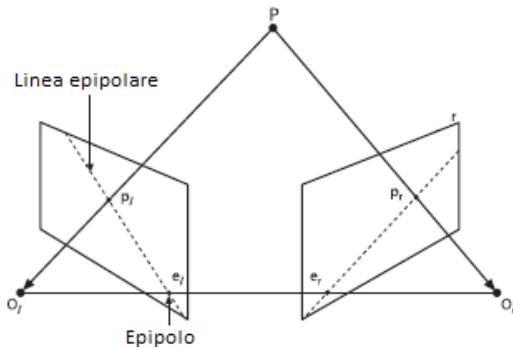


Figura 36: Rappresentazione geometria epipolare.

Per capire l'importanza di questi nuovi concetti è utile precisare un fenomeno che avviene quando si acquisisce un'immagine in un set-up stereo, riferendoci all'immagine 37.

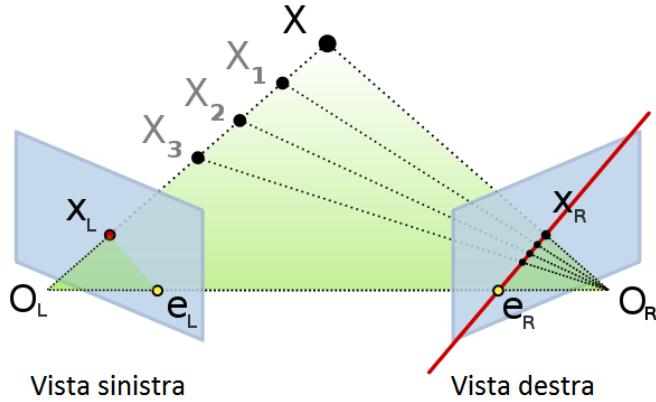


Figura 37: Proiezione punto basata sulla geometria epipolare.

Il punto  $X$  viene proiettato sul piano immagine sinistro nella posizione  $X_L$ , e dal punto di vista sinistro non siamo in grado di capire a quale profondità si trovi il punto osservato. Questo perché in realtà  $X$  potrebbe trovarsi al qualsiasi distanza compresa tra  $X_L$  e  $X$ , nello specifico ( $X_1, X_2, X_3, \dots$ ). L'aspetto interessante è che questa linea di punti,  $X_L X$ , corrisponde alla linea epipolare della camera destra, sulla quale ogni punto appartenente alla linea è proiettato in una posizione differente. Quindi traiamo le seguenti conclusioni:

- ogni punto 3D è contenuto in un piano epipolare che interseca ogni immagine in una linea epipolare;
- data un punto in un'immagine il suo corrispondente nell'altra immagine deve essere sulla corrispondente linea epipolare, questo è il **vincolo epipolare**;
- il vincolo implica che la ricerca dei punti corrispondenti si riduce da un problema a due dimensioni, in tutta l'immagine, ad una dimensione, sulla singola linea;

- in ogni configurazione stereo l'ordine dei punti è mantenuto . Se A e B sono proiettati in entrambe le immagini, allora mantengono lo stesso ordine orizzontale in entrambe.

Tornando alla rettificazione delle immagini, concludiamo che l'obiettivo di allineare le immagini per righe è raggiunto ponendo gli epipoli all'infinito, quindi in una configurazione in cui non ci sia intersezione con il piano immagine dell'altra camera.

### **Algoritmo di rettificazione**

In OpenCV si può scegliere tra due algoritmi di rettificazione:

- l'algoritmo di Hartley[18], consigliato se si vuole stimare la struttura usando solo una camera, perché c'è il rischio di produrre forti distorsioni nelle immagini rettificate;
- l'algoritmo di Bouguet[19], di cui non esiste una pubblicazione ma solo un'implementazione nel Toolbox di Matlab, che riduce le distorsioni nel caso si possano usare dei modelli di calibrazione.

Confrontando i due algoritmi ho scelto di usare quello di Bouguet, perché l'obiettivo è stimare la profondità dei punti sfruttando due camere, non una, e disponiamo di un modello di calibrazione. Date R e T, matrici di traslazione e rotazione tra le due camere, l'algoritmo minimizza l'errore di proiezione dei punti acquisiti, massimizzando l'area vista da entrambe le camere.

Per minimizzare l'errore di proiezione la matrice R, che ruota il piano immagine destro nel piano immagine sinistro, è divisa in due,  $r_{left}$  e  $r_{right}$ , che rappresentano mezza rotazione per ogni camera. Questa operazione permette di avere i due piani immagine complanari, ma non ancora allineati per righe. Per computare la matrice  $R_{rect}$ , che pone gli epipoli all'infinito e allinea le linee epipolari delle due immagini, troviamo tre vettori  $(e_1, e_2, e_3)$ . Partendo da  $e_{left}$  e prendendo il punto principale  $(c_x, c_y)$  come origine dell'immagine sinistra, la direzione dell'epipolo è diretta lungo il vettore di traslazione  $T$  tra i due centri di proiezione delle camere:

$$e_1 = \frac{T}{\|T\|}.$$

Il vettore  $e_2$  deve essere ortogonale a  $e_1$ , e inoltre scegliamo che sia anche ortogonale al raggio principale della camera. Quindi è ottenuto dal prodotto vettoriale tra  $e_1$  e la direzione del raggio principale, normalizzato per ottenere un vettore unitario:

$$e_2 = \frac{\begin{bmatrix} -T_y & T_x & 0 \end{bmatrix}^T}{\sqrt{T_x^2 + T_y^2}}.$$

Il terzo vettore deve essere ortogonale a  $e_1$  e  $e_2$  per costruzione, quindi:

$$e_3 = e_1 x e_2.$$

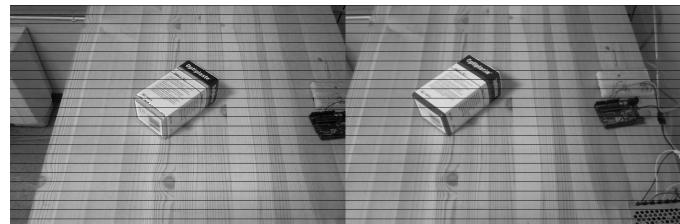
La matrice che ruota la camera sinistra intorno al centro di proiezione così che le linee epipolare siano orizzontali e gli epipoli posti all'infinito è:

$$R_{rect} = \begin{bmatrix} (e_1)^T \\ (e_2)^T \\ (e_3)^T \end{bmatrix}$$

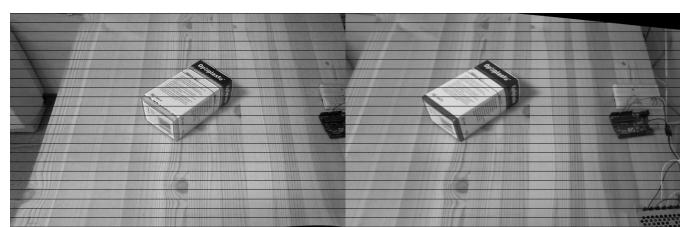
infine combinando la matrice  $R_{rect}$  e la mezza rotazione iniziale si computa una nuova matrice finale, che permette di allineare le righe delle due immagini:

$$\begin{aligned} R_l &= R_{rect} r_l \\ R_r &= R_{rect} r_r \end{aligned}$$

Di seguito sono mostrati due esempi di immagini rettificate del progetto, figura 38.



(a) *Non rettificate*



(b) *Rettificate*



(c) *Non rettificate*



(d) *Rettificate*

Figura 38: Esempi di rettificazione.

### 3.3.7 Stereo correspondence

Ora che abbiamo calibrato entrambe le camere, trovato le matrici che legano la posizione relativa delle due camere e rettificato le immagini, abbiamo quindi un set-up ideale, possiamo risolvere il problema della corrispondenza tra le due viste.

La corrispondenza stereo, che consiste nel trovare la proiezione di un punto 3D nelle due immagini per calcolare la disparità tra le due, può essere ricercata solamente nelle aree dove le due viste si sovrappongono. Questo è uno dei motivi per cui è utile scegliere di posizionare le camere il più possibile vicine e parallele tra di loro.

Riferendosi a [20], analizzeremo il problema generale della corrispondenza, descrivendo le differenze tra due correnti diverse di risoluzione.

#### Problema generale

L'obiettivo principale di un algoritmo di stereo matching, che risolve il problema della corrispondenza, è date in input due immagini della stessa scena, vista da diverse angolazioni, computare una mappa di disparità rispetto ad una delle due immagini presa come riferimento. In generale si possono distinguere due concetti di algoritmi di stereo matching:

- **dense reconstruction**[20]: in questo caso la disparità è calcolata per ogni punto dell'immagine, anche in presenza di occlusioni o punti privi di texture;
- **sparse reconstruction**[20]: al contrario questo gruppo di algoritmi cerca di ricostruire punti dell'immagine basandosi su particolari feature, quindi si ha una mappa di disparità con valori solo per alcuni pixel.

Il caso di studio del nostro progetto non è caratterizzato da particolari features o texture, quindi ho scelto di operare un ricostruzione densa dell'immagine.

Nel nostro caso il concetto di disparità è inteso come interpretazione inversa della profondità, quindi l'output di un algoritmo di stereo matching è una mappa di disparità, un'immagine al livelli di grigi in cui pixel con intensità maggiori corrispondono a punti della scena a profondità minori.

Questa caratteristica è una diretta conseguenza della relazione non lineare che lega disparità e profondità:

$$Z = \frac{fT}{x^l - x^r}.$$

Quindi per ogni coordinata  $(x, y)$ , dell'immagine di riferimento, il pixel corrispondente  $(x', y')$ , nell'altra immagine, è legato dalla seguente formula:

$$x' = x + sd(x, y), \text{ con } y' = y$$

dove  $s = \pm 1$ , così la disparità è sempre positiva.

In accordo con la tassonomia proposta in [20], la maggior parte degli algoritmi di stereo matching seguono questi step:

1. calcolo dei costi di matching;
2. aggregazione dei costi;
3. computazione disparità;
4. ottimizzazione/miglioramento disparità.

Il più semplice approccio, utile solo per capire l'obiettivo, è il seguente: dato un pixel nell'immagine destra e un range di disparità  $[d_{min}, d_{max}]$ , si ricerca il corrispondente di uno scostamento massimo  $d_{max}$  a destra nell'immagine sinistra, come mostrato in figura 39, calcolando per ogni posizione la differenza in valore assoluto tra le intensità dei singoli pixel,  $C(x, y, d) = |I_R(x, y) - I_L(x + d, y)|$ .

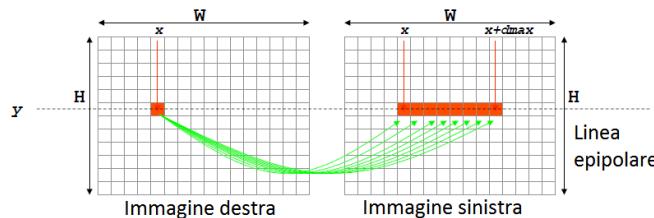


Figura 39: Ricerca base punti corrispondenti.

Dove  $C(x, y, d)$  rappresenta lo spazio tridimensionale di disparità (DSI) di dimensioni  $WxWx(d_{max} - d_{min})$ , dove ogni coordinata è rappresentata da  $(x, y, d)$ , come mostrato in figura 40.

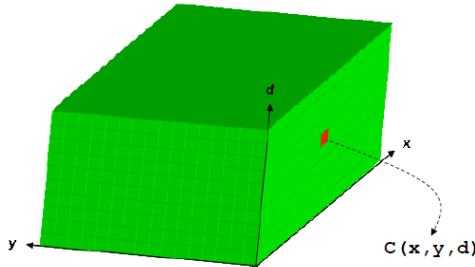


Figura 40: Spazio di disparità.

Salvando tutti i costi di matching trovati si computa quello finale secondo un WTA(Winner Takes All), figura 41, quindi si tiene la disparità corrispondente per quel pixel.

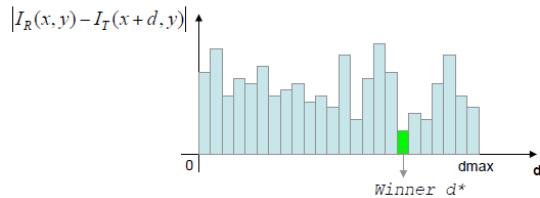


Figura 41: Metodo di selezione Winner Takes All.

Gli step elencati precedentemente non sono sempre rispettati, dipende dall'algoritmo specifico. Per esempio, negli **algoritmi locali**[21] basati su una finestra di ricerca, dove il valore della disparità dipende dal valore del costo nella finestra, solitamente si eseguono gli step 1,2,3. Per esempio il tradizionale algoritmo dove si usa la somma delle differenze dei quadrati (SSD) può essere descritto così:

1. il costo di matching è la differenza dei quadrati ad una data disparità;
2. l'aggregazione dei costi si basa sulla somma dei costi nella finestra di ricerca;
3. la disparità è computata selezionando il minimo valore di aggregazione per ogni pixel.

Dall'altra parte invece ci sono gli **algoritmi globali**[22], che cercano di risolvere un problema di ottimizzazione. Solitamente questi algoritmi non eseguono l'aggregazione dei costi, ma invece computano un valore di disparità che minimizza una funzione globale, ottenuta combinando i costi di matching con una funzione di filtraggio. L'obiettivo è trovare una funzione di disparità  $d$  che minimizza la seguente funzione:

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d)$$

Il termine  $E_{data}(d)$  è una misura di quanto la funzione di disparità  $d$  sia in accordo con il paio di immagini in input. Basandosi sullo spazio di disparità si ha:

$$E_{data}(d) = \sum_{(x,y)} C(x, y, d(x, y)),$$

dove  $C$  è il costo di matching nel DSI.

Il termine  $E_{smooth}(d)$  indica l'assunzione di filtraggio fatta dall'algoritmo. Solitamente è la differenza tra le disparità dei pixel vicini:

$$E_{smooth}(d) = \sum_{(x,y)} p(d(x, y) - d(x + 1, y)) + p(d(x, y) - d(x, y + 1)),$$

dove  $p$  è una funzione monotonamente crescente, che lega una differenza di disparità. Ma può anche dipendere dalla differenze di intensità dei pixel:

$$E_{smooth}(d) = p_d(d(x, y) - d(x + 1, y))p_I(||I(x, y) - I(x + 1, y)||),$$

dove  $p_I$  è una funzione decrescente che lega una differenza di intensità.

La libreria di OpenCV mette a disposizione l'implementazione di un algoritmo locale e uno semi-globale. Il primo sfrutta la somma delle differenze in valore assoluto (SAD) come aggregazione dei costi, ed è poco efficiente su scene con poche texture, quindi con superfici molto omogenee, che è il nostro caso. Per questo motivo ho scelto di sfruttare l'algoritmo semi-globale[23], ottenendo risultati migliori.

Di seguito sono mostrate due mappe di disparità ottenuto con il metodo[23], figura 42.

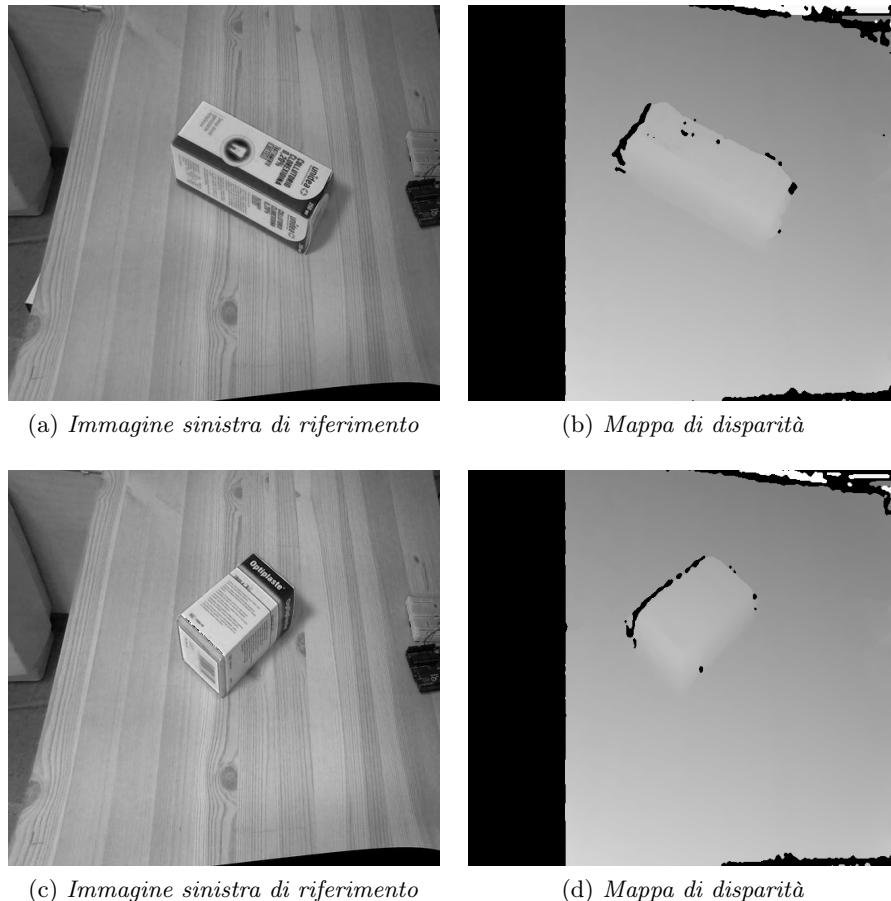


Figura 42: Esempi di mappe di disparità ottenute.

### 3.3.8 Point cloud

Una volta computata una mappa di disparità siamo in grado di proiettare ogni punto dell'immagine in uno spazio 3D secondo la seguente formula:

$$Q \begin{bmatrix} x \\ y \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix},$$

dove Q è una matrice di proiezione 4x4 ottenuta grazie ai parametri trovati durante la rettificazione delle immagini:

$$Q = \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 0 & f \\ 0 & 0 & -1/T_x & (c_x - c_x^r)/T_x \end{bmatrix}.$$

$c_x^r$  rappresenta la coordinata x del punto principale dell'immagine destra, tutti gli altri parametri sono dell'immagine sinistra.

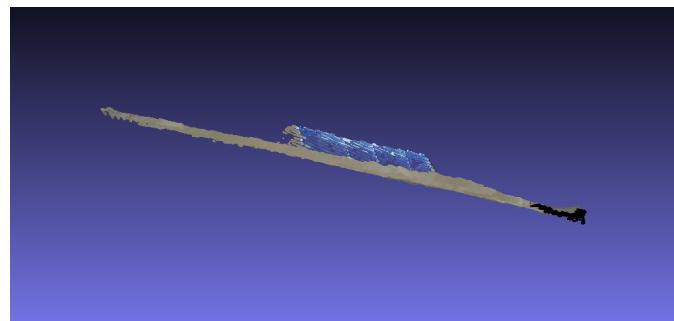
Le coordinate 3D finali si ottengono dividendo per W:

$$\begin{aligned} x_{3d} &= X/W \\ y_{3d} &= Y/W \\ z_{3d} &= Z/W \end{aligned}$$

Di seguito sono mostrati alcuni esempi di point cloud della scena completa, ottenuti con i metodi citati precedentemente.



(a)



(b)



(c)

Figura 43: Esempi di point cloud ottenuti.

Come mostrato nell'immagine 43, figura (c), la nuvola di punti ottenuta non è sempre completa in ogni suo punto. Le motivazioni principali sono due:

- la differenza di visuale tra le due camere implica che alcuni punti catturati in un'immagine non siano visibili nell'altra, e quindi nel risolvere

il problema della corrispondenza non si ottiene un valore di disparità per quel punto;

- le regioni molto omogenee non permettono una netta distinzione dei pixel, e quindi si può incorrere in errori di calcolo della disparità, associando un valore di profondità errato.

### 3.4 Calcolo volume

Per potere calcolare il volume della scatola presa in esame è necessario ottenere solo la nuvola di punti dell'oggetto, eliminando i punti superflui intorno. Questo è possibile segmentando la figura della scatola nell'immagine.

#### Segmentazione scatola

La segmentazione di un'immagine consiste nel partizionamento in regioni omogenee, sulla base di un certo criterio di appartenenza dei pixel ad una regione[24]. L'obiettivo è individuare o riconoscere gli oggetti che compongono la scena. Per segmentare la scatola ci si è basati sul presupposto che sia l'unico oggetto presente sulla scena, quindi non è stato necessario riconoscere effettivamente l'oggetto ma solo evidenziare l'area occupata. I nostri occhi riescono a distinguere i contorni di una figura presente in un'immagine grazie ai salti tra un colore e l'altro. Simulando questa tecnica si è scelto di operare una edge detection, per riconoscere queste variazioni di colore che corrispondono al contorno dell'oggetto. Per evidenziare gli edge deve essere applicata una particolare funzione matematica all'immagine.

Per applicare funzioni su un'immagine, e modificarne i valori, si esegue l'operazione di filtraggio spaziale chiamata convoluzione. L'applicazione della convoluzione è la seguente:

- in input abbiamo un'immagine e una matrice di dimensioni minori o uguali, chiamata kernel o maschera. Solitamente la maschera ha dimensioni dispari, per identificare facilmente il centro;
- il punto centrale del kernel viene fatto coincidere con un pixel dell'immagine;
- si moltiplicano i valori del kernel con i valori dei pixel corrispondenti nell'immagine e si sommano i risultati;

- il valore ottenuto è il nuovo valore del pixel centrale, che viene salvato in un'altra matrice per non modificare le operazioni successive, come mostrato in figura 44;
- l'operazione viene ripetuta spostando il kernel su tutti i pixel dell'immagine secondo la seguente formula:

$$g(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x - s, y - t)$$

dove  $a = \frac{M-1}{2}$ ,  $b = \frac{N-1}{2}$ ,  $w(s, t)$  è il valore del kernel e la funzione  $f$  è il valore del pixel nell'immagine.

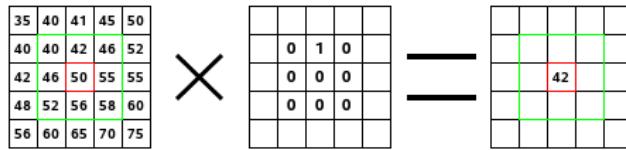


Figura 44: Esempio di convoluzione. Il riquadro rosso indica il centro di applicazione del kernel e il riquadro verde l'area su cui viene calcolato il risultato.

La funzione matematica applicata, per evidenziare gli edge di un'immagine, è la derivata, che indica quanto rapidamente varia una funzione come mostrato in figura 45.

Nel progetto ci si è sfruttato l'algoritmo di Canny[25], che si basa su 4 step principali:

1. riduzione del rumore;
2. calcolo del gradiente della luminosità per ogni pixel dell'immagine;
3. soppressione dei non-massimi;
4. doppia soglia e isteresi.

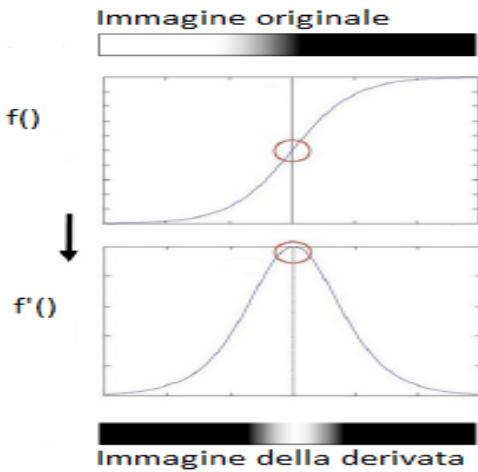


Figura 45: Esempio di come la derivata prima evidenzia il cambio di colore.

Il primo passaggio è fondamentale per evitare di individuare falsi contorni, e si ottiene sfocando l'immagine attraverso la convoluzione con un filtro Gaussiano, che rispetta la seguente formula:

$$H_{i,j} = \frac{1}{2\pi\sigma^2} \left( -\frac{(i-(k+1))^2 + (j-(k+1))^2}{2\sigma^2} \right)$$

Un edge in un'immagine può avere varie direzioni, quindi si filtra l'immagine con due kernel, in modo da evidenziare gli edges verticali e orizzontali. Per ciascun pixel si calcola il valore del gradiente e la direzione secondo le seguenti formule:

$$G = \sqrt{G_x^2 + G_y^2}$$

$$\theta = \text{atan2}(G_y, G_x)$$

La direzione poi è arrotondata tra 4 possibili valori:  $0^\circ, 45^\circ, 90^\circ$  e  $135^\circ$ , per comprendere anche le due possibili direzioni diagonali.

La mappa dei gradienti indica con forte probabilità se un pixel appartiene ad un contorno oppure no, ma non è sufficiente. Solo i punti corrispondenti a dei massimi locali sono considerati come appartenenti ad un contorno, e saranno presi in considerazione dai successivi step di elaborazione. Un massimo locale si ha nei punti in cui la derivata del gradiente si annulla.

I punti rimasti sono ulteriormente raffinati attraverso una doppia soglia e la tecnica dell'isteresi, nel quale vengono definiti due valori di thresh A e B, con  $A > B$ .

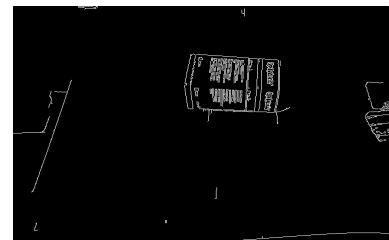
I casi possibili sono i seguenti:

1. se il gradiente di un pixel è maggiore di A allora è marcato come pixel forte;
2. se il gradiente di un pixel è minore di A e maggiore di B allora è marcato come pixel debole;
3. se è minore di B è soppresso dalla mappa dei gradienti.

I pixel deboli sono accettati come edge solo se connessi ad un pixel forte. Di seguito sono mostrati alcuni esempi di output dell'applicazione dell'algoritmo di Canny alle immagini scattate durante il progetto.



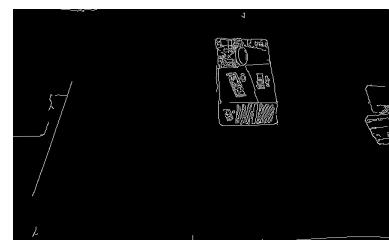
(a)



(b)



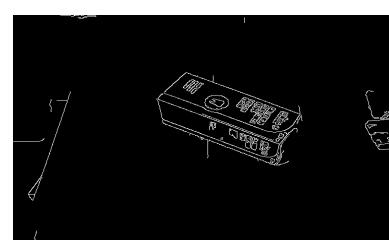
(c)



(d)



(e)



(f)

Figura 46: Esempi di output ottenuti con l'algoritmo di Canny per l'edge detection.

Nei precedenti esempi è mostrato come non vengano evidenziati solamente gli edge della scatola, è quindi necessario migliorare ulteriormente la scelta dei pixel di edge. Quest'operazione è fatta scegliendo il contorno di area maggiore tra quelli evidenziati. Successivamente il valore dei pixel interni al contorno viene posto a 1, per creare la maschera finale, con cui segmentare la mappa di disparità e l'immagine della scatola, come mostrato in figura 47.

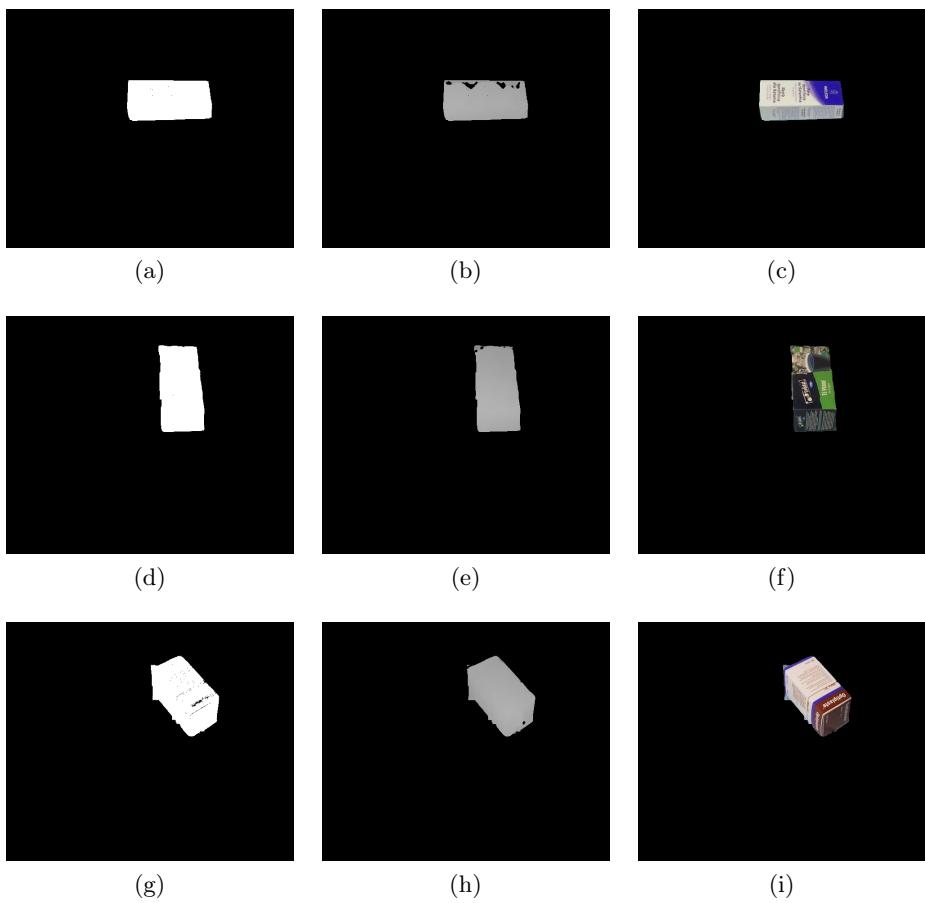
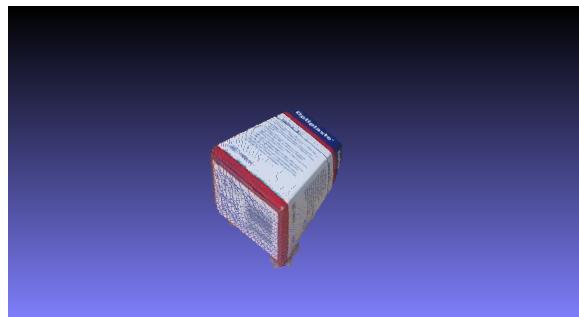


Figura 47: Esempi di segmentazione della mappa di disparità e della scatola nell'immagine. In figura (a),(d),(g) sono mostrate le maschere, in figura (b),(e),(h) le mappe di disparità e in figura (c),(f),(i) le scatole segmentate nell'immagine.

Sfruttando la mappa di disparità segmentata è possibile proiettare in 3d solo i punti della scatola, figura 48, su cui successivamente calcolare il volume.



(a)



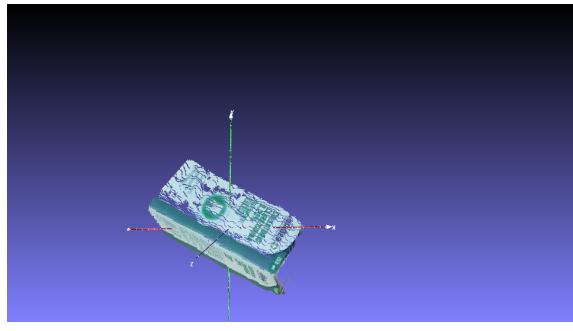
(b)



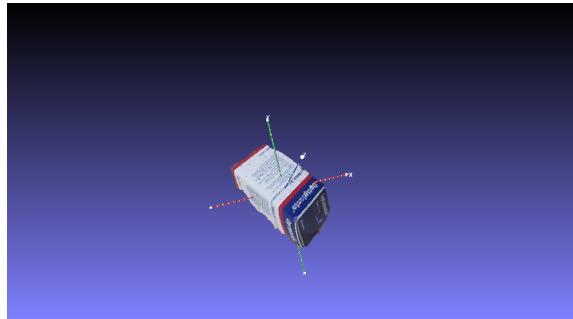
(c)

Figura 48: Esempi di point cloud solo dell'oggetto, ottenuti con la mappa di disparità segmentata.

Per calcolare il volume della scatola si è implementata una funzione che taglia la nuvola di punti attraverso dei piani, paralleli ai tre assi di riferimento x,y,z, e per ognuna delle tre direzioni salva il piano a cui appartengono il maggior numero di punti 3d e ne calcola la dimensione. Così facendo si evitano errori di calcolo in presenza di possibili buchi nella nuvola di punti a causa di mancanza di informazioni sulla profondità. Questa funzione può essere applicata su un point cloud nel quale i piani della scatola sono paralleli agli assi di riferimento. Il point cloud ottenuto dalla proiezione non rispetta questa caratteristica, come si vede dalle immagini in figura 49, quindi è stato necessario ruotare la nuvola di punti per ottenere il modello ideale.



(a)



(b)

Figura 49: Posizione originale del point cloud rispetto agli assi.

La funzione di rotazione implementata ricerca i tre angoli di rotazione cercando di massimizzare la somma dei massimi calcolati sugli istogrammi nelle tre direzioni, x,y,z, secondo la seguente formula:

$$f = \max(h_x) + \max(h_y) + \max(h_z),$$

dove  $h_x, h_y$  e  $h_z$  sono gli istogrammi normalizzati delle tre coordinate.

Di seguito sono mostrate alcune nuvole di punti prima e dopo la rotazione.

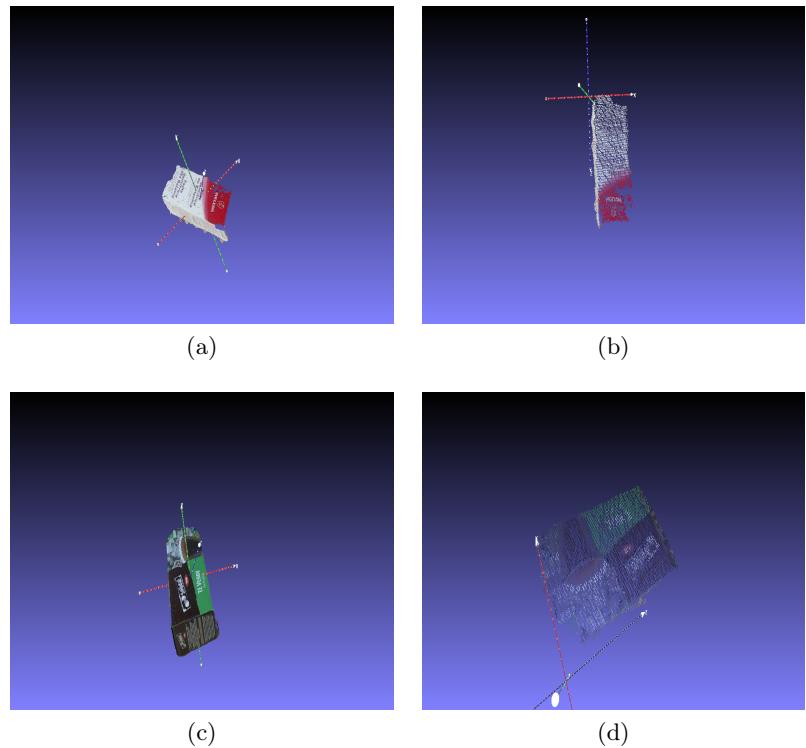


Figura 50: Point cloud prima e dopo la rotazione necessaria per calcolare il volume.

# Capitolo 4

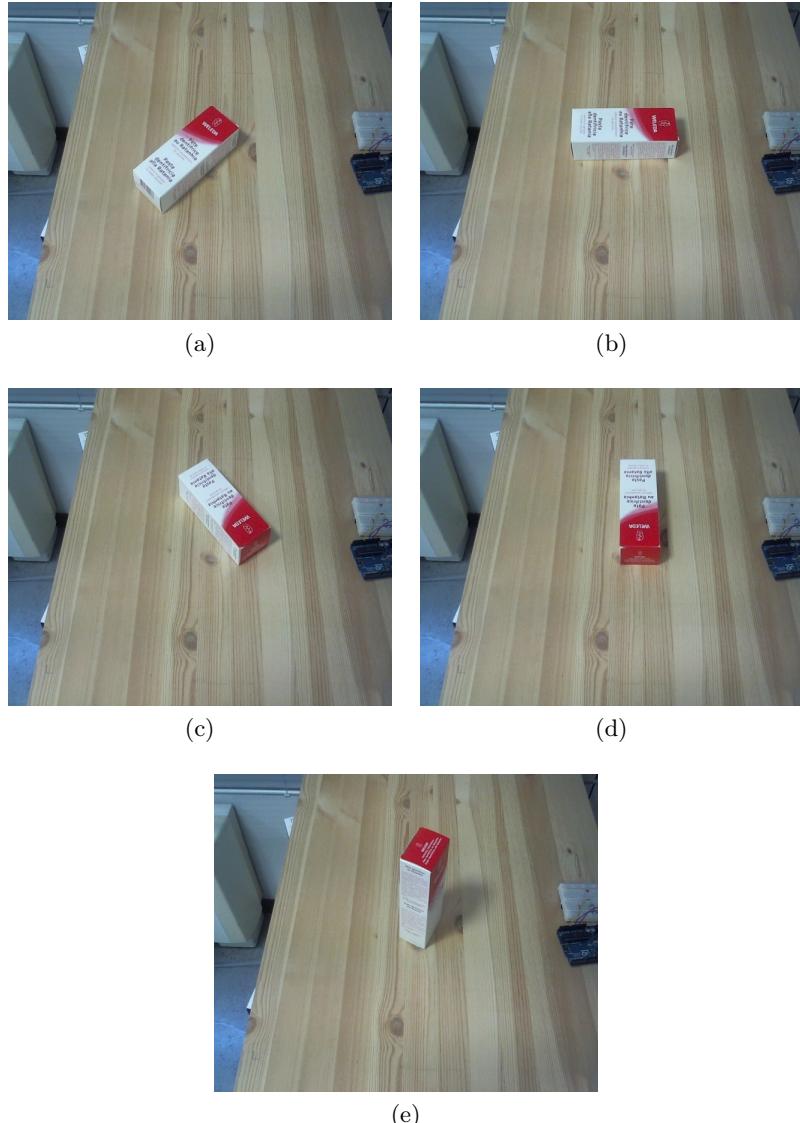
## Risultati e possibili miglioramenti

### 4.1 Risultati

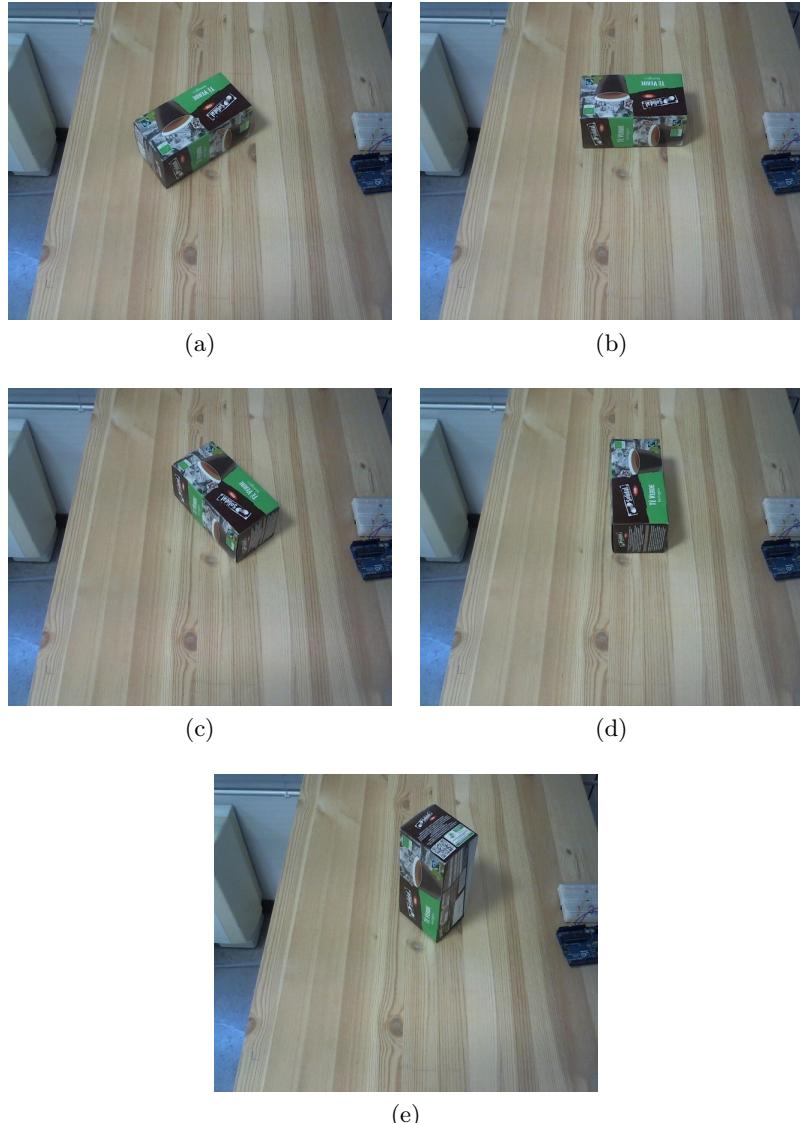
L'algoritmo implementato è stato provato su 5 modelli di scatole, e per ognuno su 5 diverse posizioni.

I risultati, espressi in centimetri, sono mostrati nella seguenti tabelle ognuna preceduta dalle immagini a cui si riferiscono.

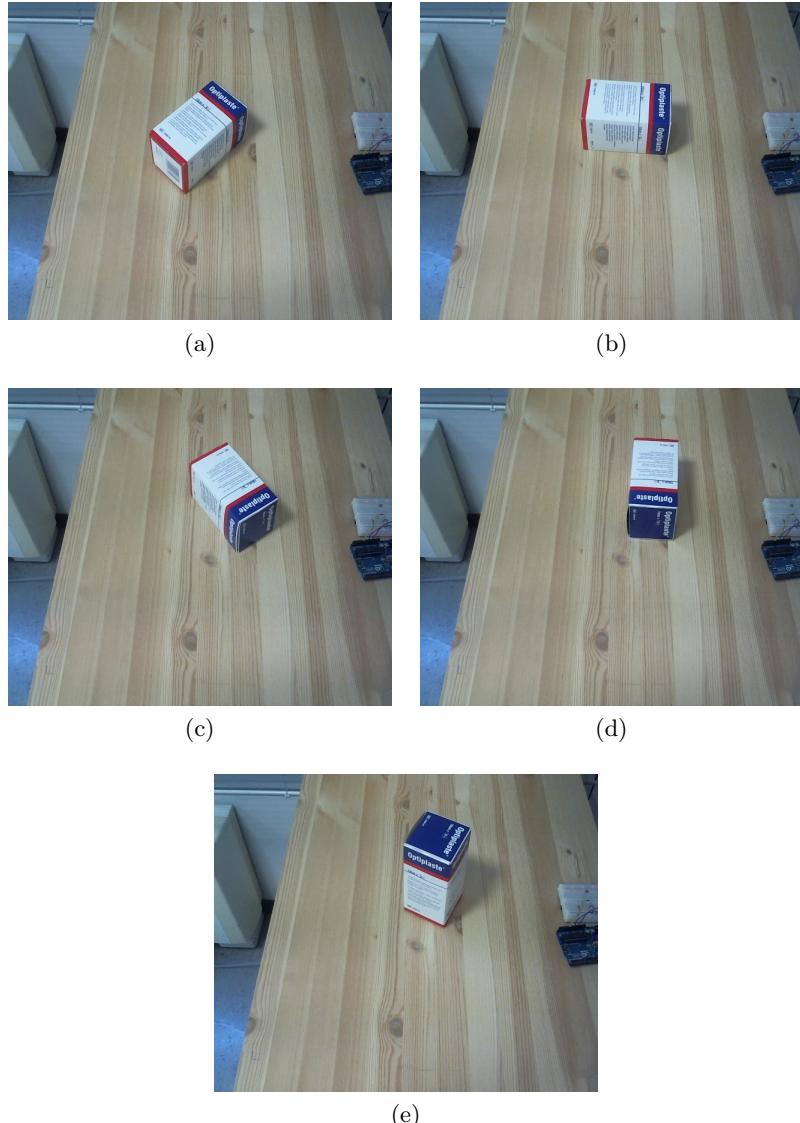
In ogni tabella nella prima riga è presente il riferimento della misura calcolata a mano, composta dalle tre dimensioni e dal volume, mentre nelle altre righe sono contenute le misure calcolate per ogni foto con la tecnica analizzata nel capitolo 3.



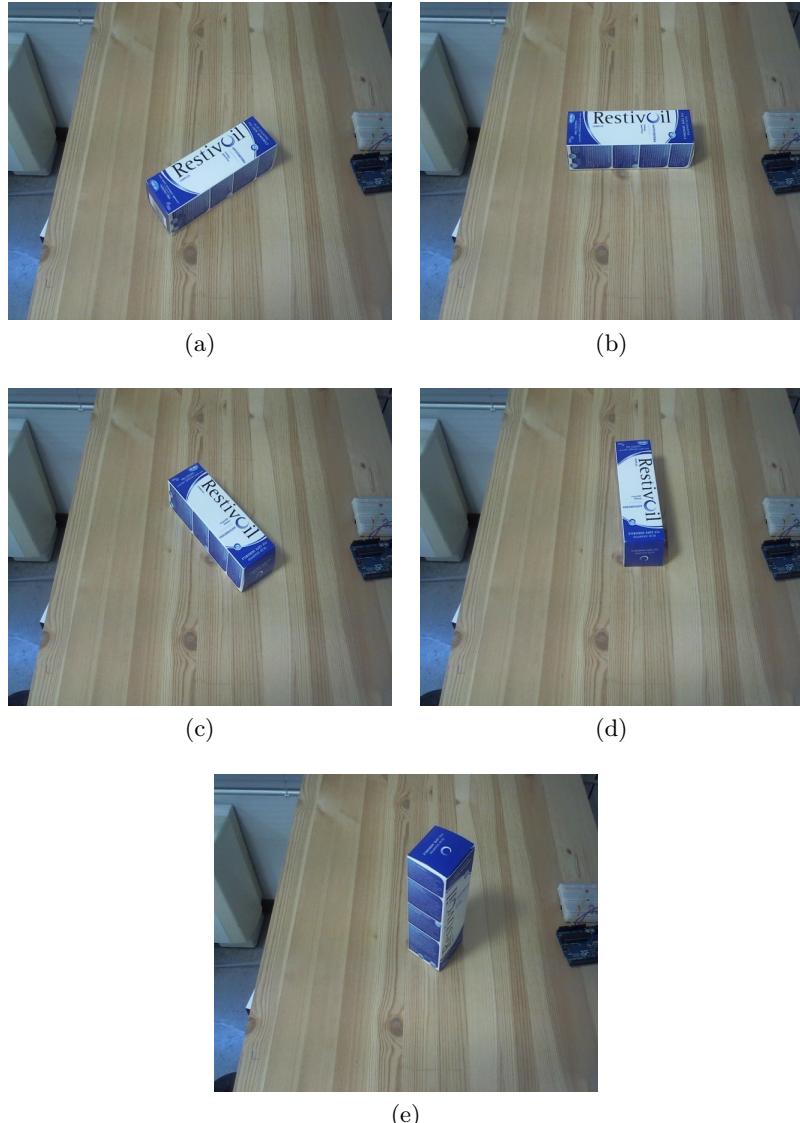
	Dimensione 1	Dimensione 2	Dimensione 3	Volume
<b>Weleda</b>	5	3.8	13.9	264
Foto (a)	5	3.5	14	245
Foto (b)	5	4	14	280
Foto (c)	5	3.6	13.5	243
Foto (d)	errore nella rotazione			
Foto (e)	4.6	4.4	13.5	273



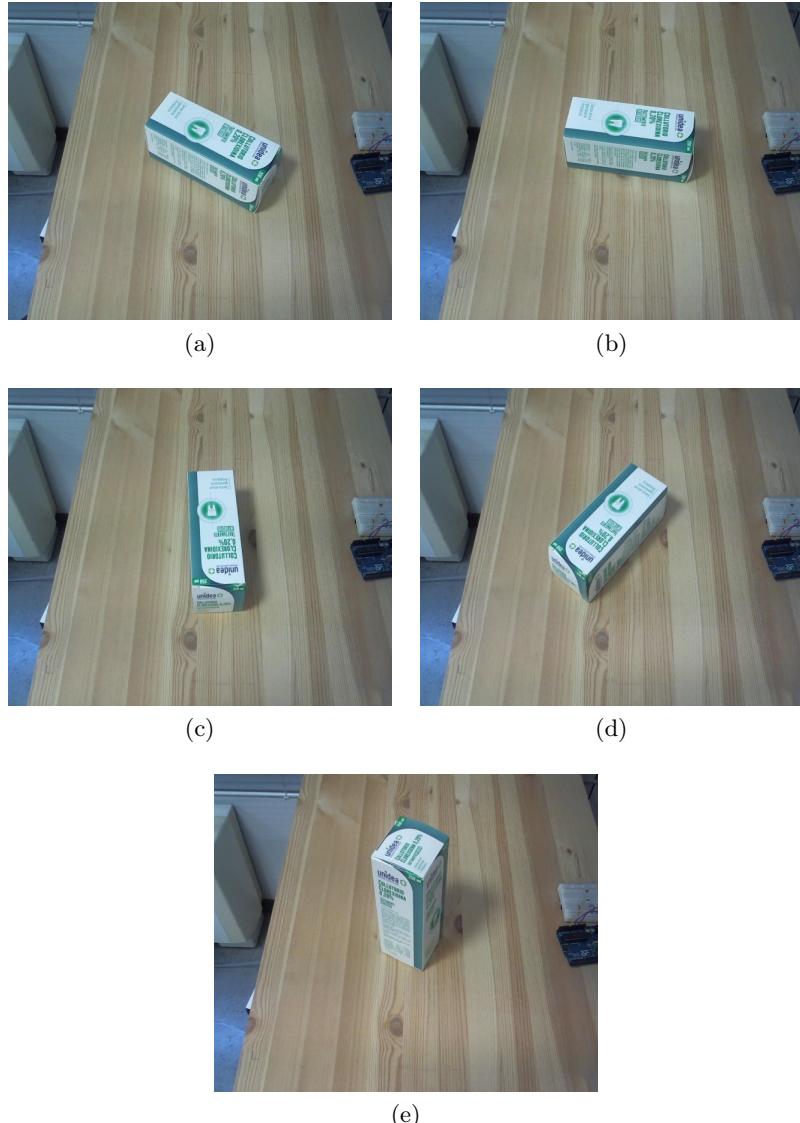
	Dimensione 1	Dimensione 2	Dimensione 3	Volume
<b>Tè verdeù</b>	5.8	6.9	14	560
Foto (a)	6.2	7	14	607
Foto (b)	5.9	6.7	14.2	561
Foto (c)	6	7.2	14.3	617
Foto (d)	6	6.8	14	571
Foto (e)	5.8	6	14.8	515



	Dimensione 1	Dimensione 2	Dimensione 3	Volume
<b>Optiplaste</b>	6.2	6.2	11.5	442
Foto (a)	6	6.6	12	475
Foto (b)	6.5	6	11	429
Foto (c)	5.9	6.3	11	408
Foto (d)	5.8	6	11.3	393
Foto (e)	5.8	6.2	12	431



	Dimensione 1	Dimensione 2	Dimensione 3	Volume
<b>Restivoil</b>	5	5	16.2	405
Foto (a)	4.8	5	16.5	396
Foto (b)	4.7	5	16	376
Foto (c)	4.9	5.8	15.5	440
Foto (d)	errore rotazione			
Foto (e)	5	5.4	15.8	426



	Dimensione 1	Dimensione 2	Dimensione 3	Volume
<b>Unidea collutorio</b>	6	6	16	576
Foto (a)	6.2	5.9	15.5	566
Foto (b)	5.5	6	16.3	537
Foto (c)	5.7	6	15.8	540
Foto (d)	5.4	6.3	16.4	557
Foto (e)	6.3	5.6	15.9	560

## **Analisi dei risultati**

Come si può vedere dalle precedenti tabelle i risultati ottenuti sono di un buon livello.

L'errore massimo nel calcolo di una delle tre dimensioni della scatola, che può rappresentare l'altezza, la profondità o la larghezza in base all'orientazione, non supera  $\pm 1\text{cm}$ . Per quanto riguarda il volume finale ci sono errori anche di  $30\text{-}40 \text{ cm}^3$ .

La principale causa degli errori nelle misurazioni effettuate è la mancanza di punti 3d in alcune zone della nuvola di punti, soprattutto vicino ai bordi della scatola, che spesso risulta non riprodotta perfettamente.

Questa mancanza di informazione sulla profondità di quei punti è causata principalmente per due motivi:

- nelle regioni omogenee la ricerca dei punti corrispondenti non è molto efficace, perché è complicato identificare un esatto pixel in una regione dove hanno tutti un valore molto simile;
- ci sono regioni che compaiono in un'immagine e non nell'altra, a causa delle differente visuale, e quindi il problema della corrispondenza non è risolto per quei punti.

## 4.2 Possibili miglioramenti

Alla fine del progetto, insieme ai miei colleghi, abbiamo pensato a possibili miglioramenti per rendere più efficiente l'elaborazione e ne abbiamo trovati due principalmente: usare il modello di scheda Jetson TX1[26] e usufruire della vista da più di due camere.

### 4.2.1 Jetson TX1

Sfruttare la scheda Jetson TX1, in figura 51, può migliorare la velocità di esecuzione dell'algoritmo implementato grazie alle caratteristiche superiori rispetto alla Jetson TK1.



Figura 51: Jetson TX1.

La tabella in figura 52, mostra un confronto tra le caratteristiche principali delle due schede.

	TX1	TK1
CPU	quad-core ARM Cortex-A57	quad-core ARM Cortex-A15
GPU	256-core Maxwell GPU	NVIDIA Kepler 'GK20a' GPU
Memory	4GB	2GB
Storage	16GB eMMC 5.1	16GB eMMC 4.51
USB	USB 3.0 + USB 2.0	USB 3.0 + USB 2.0
Power	19V DC	12V DC

Figura 52: Scheda di confronto fra la Jetson TX1 e TK1.

#### 4.2.2 Camere

Sfruttando la Jetson TX1 è possibile collegare fino a 6 camere per Raspberry Pi[27] del tipo OV5647[28] contemporaneamente, usando la scheda di collegamento Auvidea J20[29], figura 53.

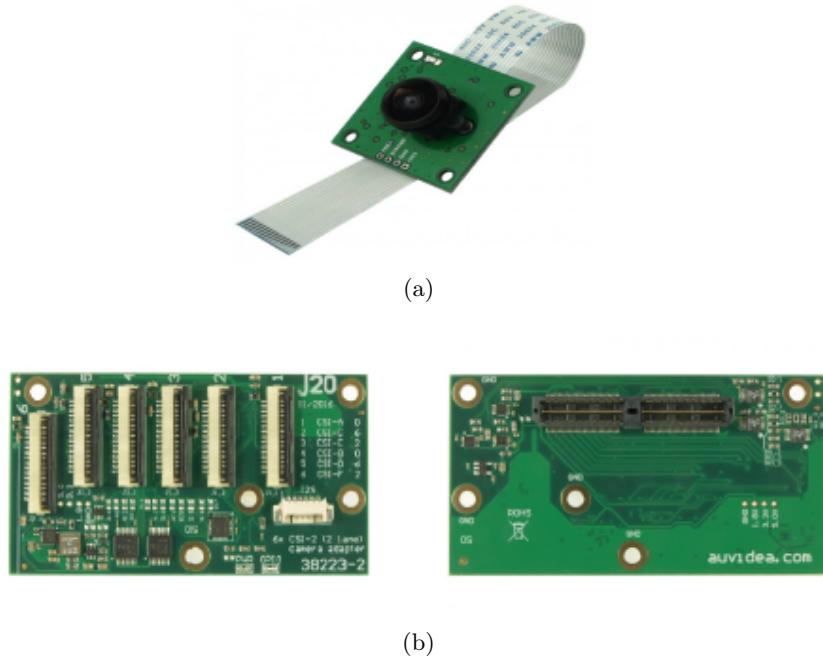


Figura 53: Camera e scheda da collegare alla Jetson TX1

Collegando fino a sei camere è inoltre possibile avere una ricostruzione completa dell'oggetto studiato, in quanto esse possono essere posizionate intorno all'oggetto e catturare ogni sua facciata.

In figura 54 è mostrata la scheda Jetson TX1 a cui sono state collegate le 6 camere.

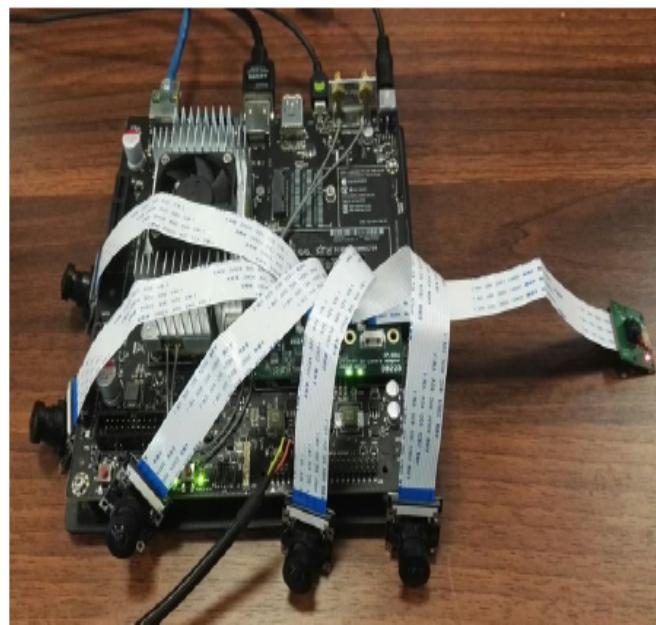


Figura 54: Collegamento di 6 camere alla scheda Jetson TX1.

# Capitolo 5

## Conclusioni

L’obiettivo iniziale della tesi era quello di ricostruire una nuvola di punti dell’oggetto acquisito al fine di stimarne il volume.

La prima parte del progetto si è svolta basandosi su una tecnica di ricostruzione 3d, di non contatto e passiva. Nello specifico si è implementato un’algoritmo di Stereo Vision, basato sull’uso di due camere, in grado di ricreare una nuvola di punti con dimensioni fedeli all’originale.

La stima del volume invece è stata implementata basandosi su due passaggi principali: la rotazione della nuvola di punti per ottenere un modello allineato con gli assi di riferimento e una stima delle dimensioni dell’oggetto attraverso un fit di piani.

Al termine del progetto entrambi gli obiettivi sono stati raggiunti con buoni risultati e anche i miei colleghi hanno terminato le loro parti. Nel complesso si è realizzato un dispositivo in grado di acquisire automaticamente degli oggetti, controllare le condizioni di luce, modificare le immagini attraverso una pipeline di pre-processing e stimarne il volume.

In futuro si potrebbe pensare di adattare l’algoritmo implementato all’uso di più camere per avere una ricostruzione 3d più precisa e completa dell’oggetto, diminuendo l’errore nella stima delle dimensioni e del volume. Inoltre si potrebbe rendere l’algoritmo di stereo matching più robusto in presenza di regioni omogenee.

# Bibliografia

- [1] NVIDIA. <http://www.nvidia.it/object/jetson-tk1-embedded-dev-kit-it.html>.
- [2] Arduino. <https://store.arduino.cc/usa/arduino-uno-rev3>.
- [3] Tamas Varady, Ralph R Martin e Jordan Cox. «Reverse engineering of geometric models—an introduction». In: *Computer-aided design* 29.4 (1997), pp. 255–268.
- [4] Wikipedia. *Coordinate-measuring machine — Wikipedia, L'encyclopédia libera*. [https://en.wikipedia.org/wiki/Coordinate-measuring\\_machine](https://en.wikipedia.org/wiki/Coordinate-measuring_machine).
- [5] Theo Moons, Luc Van Gool, Maarten Vergauwen et al. «3D reconstruction from multiple images part 1: Principles». In: *Foundations and trends® in Computer Graphics and Vision* 4.4 (2010), pp. 287–404.
- [6] Leandro Bornaz. «Principi di funzionamento e tecniche di acquisizione». In: *Pubblicato nel volume F. Crosilla* (2006).
- [7] Michele Russo, Fabio Remondino e Gabriele Guidi. «Principali tecniche e strumenti per il rilievo tridimensionale in ambito archeologico». In: *Archeologia e calcolatori* 22 (2011), pp. 169–198.
- [8] Sandro Barone, Alessandro Curcio e Armando V Razonale. *A structured light stereo system for reverse engineering applications*. Università di Pisa, 2003.
- [9] Fabio Remondino. «Rilievo e modellazione 3D di siti e architetture complesse». In: *Disegnarecon* 4.8 (2011), pp. 90–98.

- [10] Ken-ichi Kanatani e Tsai-Chia Chou. «Shape from texture: General principle». In: *Artificial Intelligence* 38.1 (1989), pp. 1–48. ISSN: 0004-3702. DOI: [https://doi.org/10.1016/0004-3702\(89\)90066-0](https://doi.org/10.1016/0004-3702(89)90066-0). URL: <http://www.sciencedirect.com/science/article/pii/0004370289900660>.
- [11] Emmanuel Prados e Olivier Faugeras. «Shape from shading». In: *Handbook of mathematical models in computer vision* (2006), pp. 375–388.
- [12] Riccardo Dessì. «Algoritmi Ottimizzati per la Photometric Stereo Applicati all’Archeologia». Tesi di laurea. Università degli studi di Cagliari Facoltà di Ingegneria, 2012.
- [13] Adelbert Ames, Kenneth N Ogle e Gordon H Gliddon. «Corresponding retinal points, the horopter and size and shape of ocular images». In: *JOSA* 22.11 (1932), pp. 575–631.
- [14] Gary Bradski e Adrian Kaehler. *Learning OpenCV*. 2008. Cap. 11,12.
- [15] Pinhole. <http://www.pinhole.cz/en/pinholecameras/whatis.html>.
- [16] UniPr. <http://www.ce.unipr.it/~medici/geometry/node7.html>.
- [17] Z. Zhang. «A flexible new technique for camera calibration». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (2000), pp. 1330–1334.
- [18] Richard I Hartley. «Theory and practice of projective rectification». In: *International Journal of Computer Vision* 35.2 (1999), pp. 115–127.
- [19] Jean-Yves Bouguet. *MATLAB calibration tool*. [www.vision.caltech.edu/bouguetj/calib\\_doc/](http://www.vision.caltech.edu/bouguetj/calib_doc/). 2015.
- [20] Daniel Scharstein e Richard Szeliski. «A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms». In: *International Journal of Computer Vision* 47.1 (2002), pp. 7–42. URL: <https://doi.org/10.1023/A:1014573219977>.
- [21] Olivier Faugeras et al. *Real time correlation-based stereo: algorithm, implementations and applications*. Rapp. tecn. Inria, 1993.

- [22] Vladimir Kolmogorov e Ramin Zabih. «Computing visual correspondence with occlusions using graph cuts». In: *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*. Vol. 2. IEEE. 2001, pp. 508–515.
- [23] H. Hirschmüller. «Stereo Processing by Semiglobal Matching and Mutual Information». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.2 (feb. 2008), pp. 328–341. ISSN: 0162-8828.
- [24] L Luccheseyz e SK Mitray. «Color image segmentation: A state-of-the-art survey». In: *Proceedings of the Indian National Science Academy (INSA-A)* 67.2 (2001), pp. 207–221.
- [25] John Canny. «A computational approach to edge detection». In: *Readings in Computer Vision*. Elsevier, 1987, pp. 184–203.
- [26] <http://www.nvidia.it/object/embedded-systems-dev-kits-modules-it.html>.
- [27] <https://www.raspberrypi.org/products/>.
- [28] <http://www.arducam.com/raspberry-pi-camera-rev-c-improves-optical-performance/>.
- [29] <https://auvidea.com/j20/>.

# **Ringraziamenti**

Desidero ringraziare tutti coloro che mi hanno aiutato nella stesura della tesi con suggerimenti, critiche ed osservazioni.

Ringrazio anzitutto il professore Simone Bianco, relatore, ed il professore Paolo Napoletano, co-relatore, per i consigli e gli aiuti ricevuti.

Un ringraziamento particolare va ai colleghi che mi hanno accompagnato in questo percorso universitario.

Vorrei infine ringraziare le persone a me più care: la mia famiglia, i miei amici e la mia fidanzata per avermi sostenuto in qualsiasi momento.