UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA INDUSTRIALE

UNIVERSITÀ DEGLI STUDI DI PADOVA

Dipartimento di Ingegneria Industriale DII

Corso di Laurea Magistrale in Ingegneria Aerospaziale

Implementazione di un metodo di riconoscimento e raccolta di campioni rocciosi tramite stereocamera e braccio robotico del rover Morpheus

Relatore: Prof. Stefano Debei
Correlatore: Ing. Sebastiano Chiodini

Marco Ghetti
n° 1204901

Anno Accademico 2020/2021

A tutte le ragazze e i ragazzi che si sentono dire: « è bravo ma non si applica »

Ringraziamenti

Dedico questo spazio a chi, con la propria presenza, mi ha permesso di concludere al meglio il mio percorso di studi contribuendo alla mia formazione di uomo e ingegnere.

In primis voglio ringraziare tutto il team Morpheus. In particolare il mio correlatore Sebastiano Chiodini che, con la sua infinita disponibilità, mi ha seguito in ogni step della realizzazione del progetto di tesi. Inoltre, voglio ringraziare il mio collega Giacomo per aver condiviso gioie e sofferenze in questi ultimi mesi di laboratorio.

Il più grande e sincero ringraziamento lo porgo a Giada, la mia fidanzata, perché in questi anni ha sempre sopportato tutte le mie ansie e paranoie aiutandomi a superarle. Tutte le volte che sono caduto mi ha rialzato o si è sdraiata accanto a me.

Ringrazio i miei amici dell'università: Zeno, Fil, Luca, Fede e Ste perché una grande avventura merita di essere vissuta con grandi compagni di viaggio.

Ringrazio tutta la mia famiglia. I miei genitori per avermi sostenuto e per avermi lasciato la libertà di prendere la mia strada. Le mie sorelle: Maria, Anna e Agnese, e mio fratello Andrea perché nonostante tutto ci vogliamo bene. E ringrazio i miei nonni che mi hanno insegnato il valore dell'istruzione e che ogni lavoro è dignitoso, se fatto con impegno e passione.

Infine, ringrazio tutti i professori che sono riusciti a trasmettermi la passione delle materie studiate e non solo i contenuti. In particolare ringrazio la mia maestra di matematica Ago perché grazie a lei tutto questo ha avuto inizio insegnandomi a contare.

Abstract

Per questo progetto di tesi è stato implementato un algoritmo per il riconoscimento e la raccolta di campioni rocciosi da parte del rover di ateneo Morpheus. Per l'individuazione dell'oggetto si è scelto di utilizzare un sistema di visione formato da una stereo camera. Quest'ultima, fornendo coppie di immagini stereo, ha permesso la costruzione della *point cloud*. Utilizzando l'algoritmo RANSAC è avvenuto il riconoscimento oggetto confrontando i punti 3D e il modello di un cilindro. Grazie alle seguenti operazioni si è stati in grado di identificare la posa del campione roccioso con una precisione dell'ordine del centimetro. Successivamente, si è effettuata la relativa raccolta tramite il braccio robotico. Le traiettorie eseguite dai motori, associati ai giunti del braccio, sono state pianificate tramite il software MoveIt il quale richiede come input la sola posizione obiettivo dell'end effector. Data l'interazione tra il sistema di visione e di raccolta, è stato necessario utilizzare l'algoritmo di calibrazione occhio mano per ottenere la matrice di rototraslazione che lega i due sistemi. I risultati forniti dalla calibrazione presentano un forte errore sistematico dovuto, prevalentemente, ai giochi meccanici presenti nei giunti del braccio. Sfruttando la ripetitività dell'errore è stato possibile ridurlo, effettuando così la raccolta di un campione di roccia in maniera automatizzata. Infine, si precisa che tutto il lavoro di tesi è stato implementato utilizzando il sistema operativo ROS.

Parole chiave: rover, ROS, stereo camera, braccio robotico, riconoscimento oggetto, RANSAC, calibrazione occhio mano.

Indice

Ringraziamenti v
Abstract vii
Lista delle Tabelle xi
Lista delle Figure xiii

1 Introduzione **1**
1.1 Motivazione 1
1.2 Robotica 2
1.3 Progetto studentesco Morpheus 2
1.4 *Outline* della tesi 4

2 Teoria della visione artificiale **5**
2.1 Modello lente sottile 5
2.1.1 Lunghezza focale 6
2.1.2 Formazione dell'immagine 6
2.1.3 Sensore 7
2.2 Modello Pin Hole 8
2.3 Matrice di calibrazione 10
2.4 Calibrazione camera 12
2.4.1 Effetti di distorsione 13
2.5 Sistema di stereo visione 14
2.6 Feature detector 16
2.6.1 Feature matching 17

3 Teoria dei Manipolatori **19**
3.1 Teoria dei manipolatori 20
3.2 Posa di un corpo rigido 23
3.2.1 Matrice di rotazione e Roll Pitch Yaw 25
3.3 Cinematica diretta 26
3.3.1 Convenzione di Denavit-Hartenberg 27
3.4 Cinematica inversa 30

4 Hardware del rover e ROS **33**
4.1 Sistema di stereo visione . . . 33
4.2 Braccio robotico . . . 34
4.2.1 Gripper . . . 37
4.3 ROS . . . 39
4.3.1 Funzionamento . . . 40
4.3.2 Interfaccia grafica . . . 41

5 Implementazione **43**
5.1 Riconoscimento oggetto . . . 43
5.1.1 Acquisizione immagini . . . 44
5.1.2 Calibrazione stereo camera . . . 45
5.1.3 Elaborazione immagini . . . 46
5.1.4 Ricerca oggetto . . . 47
5.2 Calibrazione occhio mano . . . 49
5.2.1 Teoria introduttiva . . . 49
5.2.2 Implementazione . . . 51
5.3 Raccolta e posa . . . 54
5.3.1 Raccolta . . . 55
5.3.2 Gripper . . . 57
5.3.3 Posa . . . 57

6 Risultati **59**
6.1 Risultati riconoscimento oggetto . . . 59
6.2 Calibrazione occhio mano . . . 62
6.3 Presa e posa del campione roccioso . . . 65

7 Conclusioni **69**
7.1 Valutazioni riconoscimento oggetto . . . 69
7.2 Valutazioni calibrazione occhio mano . . . 71
7.3 Valutazioni pick and place . . . 73

Bibliografia **75**

A Risultati della calibrazione **79**

Elenco delle tabelle

2.1 Tabella riassuntiva delle caratteristiche dei vari feature detector 16

4.1 Parametri fisici [23] 34
4.2 Parametri Video [23] 34
4.3 Parametri link 35
4.4 Parametri motore giunto 1 35
4.5 Parametri motore giunto 2 e 3 36
4.6 Parametri motore giunto 4 36
4.7 Ingombri 2F-85 [27] 38
4.8 Specifiche gripper 2F-85 [27] 38

6.1 Risultati riconoscimento oggetto 62
6.2 Caratteristiche tag utilizzati 62
6.3 Risultati calibrazione occhio mano con il primo tag 64
6.4 Risultati calibrazione occhio mano con il secondo tag 64
6.5 Intervalli di confidenza prima prova (tag1) 64
6.6 Intervalli di confidenza seconda prova (tag2) 65
6.7 Parametri fisici 65

Elenco delle figure

1.1 Esempi di robot [5]–[7] 2
1.2 Rover Rosalind Franklin,ESA [8] 3

2.1 Lente sottile [15] 6
2.2 Propagazione raggi attraverso lente sottile 7
2.3 Varie aperture del diaframma [16] 8
2.4 Rappresentazione della profondità di campo [17] 8
2.5 sistema di riferimento CCS e S-2 D 9
2.6 Proiezione X-Z 9
2.7 Proiezione Y-Z 10
2.8 Sistema di riferimento camera (CCD), sensore, world 10
2.9 Sistema di riferimento sensore discretizzato in pixel e decentrato 11
2.10 Processo di calibrazione [18] 13
2.11 Proiezione prospettica [19] 14
2.12 Confronto tra immagine reale, distorsione a barile e a cuscino 14
2.13 Modello stereo camera 15
2.14 Modello stereo camera con camere allineate 15
2.15 schematizzazione modello stereo camera 15
2.16 Esempio di utilizzo del detector SURF, foto ESA [20] 17
2.17 Feature matching 17
2.18 Esempio di utilizzo del filtraggio RANSAC 18

3.1 Esempio di manipolatore [21] 19
3.2 Esempio schematico di manipolatore 20
3.3 Esempio di catena cinematica 20
3.4 Tipi di joints 21
3.5 Manipolatore cartesiano e il suo workspace [22] 21
3.6 Manipolatore cilindrico e il suo workspace [22] 22
3.7 Manipolatore sferico e il suo workspace [22] 22
3.8 Manipolatore SCARA e il suo workspace [22] 22
3.9 Manipolatore antropomorfico e il suo workspace [22] 22

3.10 Sistema di riferimento O_{XYZ} e proiezione del punto P 23
3.11 Determinazione posa di un corpo rigido 24
3.12 Sistema di riferimento fisso e Body 24
3.13 Rappresentazione angoli di Eulero [22] 25
3.14 Rappresentazione angoli di Roll-Pitch-Yaw [22] 25
3.15 Posizione e orientamento dell'EE rispetto alla base [22] 27
3.16 Manipolatore generico [22] 27
3.17 Esempio di utilizzo di DH su un robot SCARA [22] 28
3.18 Esempio di soluzione multipla [22] 30
3.19 Punto p non raggiungibile dall'end-effector 30

4.1 Stereo camera ZED [23] 33
4.2 Dimensioni zed [23] 34
4.3 Braccio Morpheus e Sistema di riferimento fisso 35
4.4 Pinza modello 2F-85 della Robotiq [27] 36
4.5 Braccio robotico a 4 gradi di libertà con il gripper 2F-85 della Robotiq [4] 37
4.6 messa in tavola gripper in posizione aperta 37
4.7 messa in tavola gripper in posizione chiusa 38
4.8 Raccolta di un campione roccioso tramite gripper 2F-85 39
4.9 Logo ROS (Robot Operating System))[28] 39
4.10 Loghi di Noetic e Ubuntu [28], [29] 40
4.11 Rappresentazione figurata di un middleware [30] 40
4.12 Esempio funzionamento nodo *Usb Cam*, *Image View* e *topic Usb Image Raw* 41
4.13 Modifica sistema di visione 41
4.14 Pannello di controllo mostrante più GUI [31] 42
4.15 Pannello di controllo di Rviz mostrante braccio del rover Morpheus [32] 42

5.1 Nodo Usb Cam 44
5.2 Nodo Usb Cam e Splitter 44
5.3 Sistema di stereovisione sincronizzato 45
5.4 GUI di calibrazione stereo [41] 45
5.5 Nodo *Stereo Image Proc* 46
5.6 GUI rqt reconfigure 47
5.7 Sistema di riferimento point cloud 47
5.8 Point cloud e area di interesse [43] 48
5.9 Identificazione oggetto a partire da una *point cloud* [43] 48
5.10 Calibrazione occhio mano 49
5.11 Diverse configurazione dell'*eye* [44], [45] 49
5.12 Posa i-esima e j-esima con relative trasformazioni [45] 50
5.13 *Plug-in EyeCalibration* disponibile per Rviz [46] 51

5.14 Finestra *Calibrate* del Plug-in *EyeCalibration* disponibile per Rviz [46] 52
5.15 Finestra *Context* del Plug-in *EyeCalibration* disponibile per Rviz [46] 53
5.16 Distribuzione *t di Student* 54
5.17 Controllo dei giunti 55
5.18 Progetto lavoro futuro 55
5.19 Fase di avvicinamento 56
5.20 Allineamento End Effector 56
5.21 Raggiungimento obiettivo 56
5.22 GUI rqt controllo gripper 57
5.23 Rotazione dell'*end effector* 57
5.24 Posa campione roccioso in centro di raccolta 58

6.1 Esempio di raccolta di un campione roccioso 59
6.2 Coppia immagine stereo in output dal nodo *splitter* 60
6.3 Rettifica e correzione immagini 60
6.4 *Disparity map* ottenuta da una coppia di immagini rettificate non distorte 60
6.5 *Disparity map* ottenuta con parametri riconfigurati 61
6.6 Riconoscimento oggetto in diversi casi di accuratezza 61
6.7 Diverse pose del tag 63
6.8 Confronto tra sistema reale e sistema simulato 63
6.9 Rilevamento tag 65
6.10 Avvicinamento al campione roccioso 66
6.11 Presa del campione roccioso 67
6.12 Rotazione dei giunti 67
6.13 Posa del campione in un ipotetico centro di raccolta 67

7.1 Risultati della calibrazione 70
7.2 Confronto tra un riconoscimento impreciso dell'oggetto e uno preciso 71
7.3 Calibrazione occhio mano e riconoscimento *tag* 71
7.4 Rappresentazione della media dei risultati ottenuti tramite l'utilizzo dei 2 tag e dalla misurazione laser 72
7.5 Raccolta campione roccioso tramite braccio robotico 73

Capitolo 1

Introduzione

In questo primo capitolo vengono esposte le motivazioni principali per le quali è stato scelto di intraprendere questo progetto di tesi. Successivamente viene introdotto il concetto di robotica e definito il concetto di robot. Nella penultima sezione viene presentato il progetto studentesco Morpheus parlando della sua storia, dei suoi obiettivi e dei suoi traguardi. Infine, viene presentata l'*outline* del documento, una breve descrizione di cosa verrà trattato in ogni capitolo.

1.1 Motivazione

Nell'ultimo secolo l'uomo ha utilizzato sempre più la tecnologia per l'esplorazione spaziale. Dal primo lancio di un satellite artificiale, lo Sputnik1, a oggi, le innovazioni hanno permesso di portare l'uomo sulla Luna e i rover su Marte. In particolare, negli ultimi anni, il suolo marziano è stato raggiunto diverse volte permettendo così l'esplorazione del pianeta. A tal proposito, l'ultimo obiettivo della NASA è stato raggiunto mediante il rover *Perseverance* e il drone *Ingenuity* appartenenti alla missione *Mars2020* atterrati sul pianeta rosso il 18 febbraio 2021. Lo scopo principale della missione è lo studio dell'abitabilità del pianeta e la ricerca di eventuale vita biologica. Inoltre, è previsto il raccoglimento di campioni rocciosi per consentire alla prossima missione *Mars Sample Return* di portarli sulla Terra ed eseguirne un'accurata analisi.

Nonostante i traguardi importanti raggiunti, la ricerca di nuove soluzioni o il miglioramento di quelle esistenti è fortemente incentivata. Infatti, a livello europeo esistono diverse competizioni in ambito robotico che promuovono la realizzazione dei rover, ad esempio l' *European Rover Challange*.
L'università di Padova contribuisce al processo di miglioramento tecnologico, in ambito rover, tramite il progetto Morpheus. Nella cornice di tale progetto prende luogo il lavoro di tesi, che nasce dalla necessità di rendere il veicolo completamente autonomo. A tale scopo l'intenzione è quella di integrare il sistema di raccolta con il sistema di visione stereo. A seguito di numerosi studi, compiuti da precedenti membri del team, nell'ottica finale di rendere il rover un sistema in grado di muoversi nello spazio ed eseguire raccolte di campioni in modo autonomo [1]–[4], la tesi si propone di dimostrare il processo di test e validazione in laboratorio dell'integrazione del sistema stereo combinato con il manipolatore.

1.2 Robotica

La robotica è la branca dell'ingegneria che studia e implementa algoritmi affinché i robot possano eseguire mansioni, più o meno complesse, in supporto all'uomo. L'etimologia della parola *robot* deriva dal Ceco e significa *lavoro pesante* perciò, in generale, per robot si intende un qualsiasi dispositivo in grado di svolgere in maniera totalmente autonoma, o parziale, lavori ritenuti fisicamente e/o psicologicamente pesanti come sollevare carichi o eseguire azioni ripetitive per lunghi periodi di tempo. Nell'immaginario collettivo, il classico esempio di robot è l'androide, ovvero una macchina con sembianze umane, ma lo sono anche gli elettrodomestici capaci di effettuare pulizie in modo autonomo oppure i veicoli dotati di guida autonoma, di cui si riportano degli esempi nella Fig. 1.1.

(a) Androide

(b) Robot per la pulizia del pavimento

(c) UGV: Unmanned Ground Vehicle s

Figura 1.1: Esempi di robot [5]–[7]

I problemi che la robotica si trova ad affrontare, come muoversi, afferrare un oggetto o riconoscerlo, sono spesso banali per l'uomo, ma per un robot richiedono algoritmi complessi e l'interazione tra sensori, attuatori e schede di controllo. Questo è il motivo per cui questa disciplina necessita di conoscenze approfondite di molte materie scientifiche: matematica, fisica, elettronica, informatica e meccanica.

1.3 Progetto studentesco Morpheus

Il progetto studentesco Morpheus nasce nel 2014 come progetto all'avanguardia tecnologica per la realizzazione di un rover, i.e., un robot per l'esplorazione extraterrestre, finalizzato al suolo marziano. L'obiettivo è stato, e rimane tutt'ora, quello di sviluppare una piattaforma di supporto agli astronauti

nell'esplorazione di aree di interesse scientifico. In particolare dovrà essere in grado di risolvere 3 task fondamentali:

- *Science Task*: ausilio alla selezione e documentazione di campioni rocciosi presenti nel suolo e/o sottosuolo
- *Terrain Traversal Task*: navigazione autonoma
- *Maintenance/Assistance Task*: trasporto oggetti e supporto all'operatore sito di lavoro (interazione uomo/macchina)

Per la concretizzazione di questi obiettivi sono stati progettati e poi realizzati diversi sottosistemi. Il principale è il PC di bordo composto dalla scheda *nVidia Jetson TX2* con la funzione di controllare i sottosistemi secondari fornendo ad essi i corretti input ed elaborando le informazioni ricevute. I sistemi meccanici principali sono due: il sistema *locomotion* e il braccio robotico. Entrambi stanno subendo notevoli trasformazioni dettate dall'avanzamento tecnologico e dalle conoscenze acquisite da parte del team. Il sistema *locomotion* è composto da sei ruote motrici che conferiscono estrema mobilità al rover. Ogni ruota è azionata da un motore il quale viene controllato a basso livello dal PC di bordo che a sua volta riceve informazioni ad alto livello da un operatore tramite *joypad*. Lo sviluppo imminente riguarda la forma delle ruote. Si vuole passare dal primo modello, classiche ruote a camera d'aria, a un modello molto avanzato emulando *Rosalind Franklin*, il rover di *Exomars 2022*.

Figura 1.2: Rover Rosalind Franklin,ESA [8]

Anche il braccio robotico ha subito notevoli miglioramenti. Il primo prototipo utilizzava una serie di cinematismi che permettevano un'attuazione lineare. Negli ultimi anni, invece, è stato progettato e realizzato un braccio robotico a 4 gradi di libertà con giunti rotoidali e link in fibra di carbonio a cui viene applicato un gripper.

Oltre ai sistemi meccanici e di controllo sono presenti dei sensori in grado di raccogliere informazioni sull'ambiente circostante. Viene utilizzata una *stereo-camera* per raccogliere immagini del percorso effettuato e un *Lidar* (Light Detection and Ranging) per la mappatura di quest'ultimo. Le informazioni così raccolte possono essere utilizzate per eseguire tecniche di slam, riconoscimento oggetti o solo per raccogliere informazioni riguardati il sito di lavoro.

Questo progetto di tesi è uno degli ultimi sviluppi tecnologici del team. L'obiettivo è di unire il sistema di visione al sistema del braccio robotico implementando un algoritmo per l'individuazione di un campione di roccia e relativa raccolta.

1.4 *Outline* della tesi

Il progetto di tesi consiste nell'implementazione di un algoritmo che permetta al rover Morpheus l'identificazione di un campione roccioso tramite il sistema di visione costituito dalla stereo camera ZED, di effettuarne la raccolta mediante il braccio robotico e infine di posizionarlo in un punto prestabilito. Brevemente ogni capitolo può essere riassunto come segue:

- Teoria della visione artificiale: vengono introdotti i concetti principali della *visual theory* con l'obiettivo di spiegare, in maniere semplificata, i principi fisici e matematici che permettono l'acquisizione di un immagine da parte di un sistema ottico. Inoltre, viene introdotto il sistema stereo camera mostrandone la capacità di ricostruire mappe 3D di punti, dette *point cloud*, attraverso l'analisi delle immagini 2D.

- Teoria dei manipolatori: riporta i concetti basi della meccanica applicati alla robotica. Viene introdotto il concetto di manipolatore portandone degli esempi e mostrandone le componenti principali, in seguito si espongono i concetti base della cinematica diretta e della cinematica inversa.

- Hardware del rover e ROS: vengono presentati i principali sottosistemi utilizzati del rover Morpheus cioè la stereo camera Zed e il braccio robotico a quattro gradi di libertà specificando le caratteristiche fisiche di ogni componente. Infine, viene descritto il sistema operativo ROS focalizzando l'attenzione sul suo funzionamento.

- Implementazione: contiene l'implementazione teorica della tesi spiegando quali processi sono stati eseguiti e mostrando come sono stati riprodotti. Nella prima parte viene trattato lo sviluppo dell'algoritmo di riconoscimento oggetto. Successivamente, si mostra la necessità di effettuare il processo di calibrazione occhio mano e come questo è stato implementato tramite ROS. Concludendo, viene mostrato il funzionamento del braccio robotico.

- Risultati: vengono mostrati i risultati ottenuti in laboratorio applicando i procedimenti spiegati nel capitolo precedente. In particolare vengono riportati i dati relativi alla calibrazione occhio mano e viene mostrata l'effettiva realizzazione del progetto di tesi.

- Conclusioni: vengono valutati i processi implementati, spiegandone i punti di forza e i punti deboli. Inoltre, eseguendo un'analisi dei risultati, vengono proposti degli sviluppi futuri per migliorare il sistema di riconoscimento e raccolta oggetti del rover Morpheus, implementato in questo progetto di tesi.

Capitolo 2

Teoria della visione artificiale

Con il progredire della tecnologia, le camere, intese come sensori in grado di catturare un' immagine, hanno trovato sempre più spazio per le applicazioni di tipo ingegneristico. Gli utilizzi possibili delle videocamere sono innumerevoli, il più comune è indubbiamente la videosorveglianza. Mentre, un esempio più ingegneristico, potrebbe essere la ricerca oggetti (*object detection*). Al contrario della videosorveglianza dove, solitamente, le immagini vengono osservate da un operatore, la ricerca oggetti è un processo autonomo eseguito da una macchina. Un esempio importante in ambito spaziale è la *visual odometry*, in particolare per esplorazioni extraterrestri svolte dai rover, in quanto verifica dell'effettivo moto del robot. Un veicolo, basandosi sulla rotazione delle ruote, può stimare l'avanzamento che ha compiuto, però la non perfetta aderenza delle ruote col suolo induce errori che col tempo possono divergere. Nasce così la necessità di un sistema di verifica: in ambito terrestre si utilizza il sistema GPS mentre per rover Marziani, o in generale dove non ci sia la possibilità di utilizzare il GPS, si utilizza la *visual odometry*.

In questo capitolo sono riportati, in forma semplificata, i principi fisici e matematici relativi alla computer vision necessari per comprendere il Capitolo 5 in cui si parlerà dell'implementazione del lavoro svolto. In primo luogo è presentato il modello *lente sottile* come introduzione al comportamento fisico di questi strumenti. Successivamente viene proposto il *modello PinHole* cioè l'evoluzione matematica del modello precedentemente spiegato. In seguito, il sotto capitolo relativo alla determinazione dei parametri intrinseci ed estrinseci: la *calibrazione*. Infine, viene trattato il sistema *stereo camera* e la possibilità, grazie ad esse, di effettuare la triangolazione dei punti 3D e la costruzione così della *point cloud*.

La trattazione presentata si basa sull'elaborazione di metodi presenti in letteratura [9]–[14].

2.1 Modello lente sottile

In ottica, una lente si definisce sottile se lo spessore d è trascurabile rispetto ai raggi di curvatura delle superfici R_1 e R_2, alla lunghezza focale f e alle distanze dell'immagine e dell'oggetto.

$$d << R_1, R_2, f \tag{2.1}$$

In maniera speculare, le lenti il cui spessore non è trascurabile sono chiamate lenti spesse.

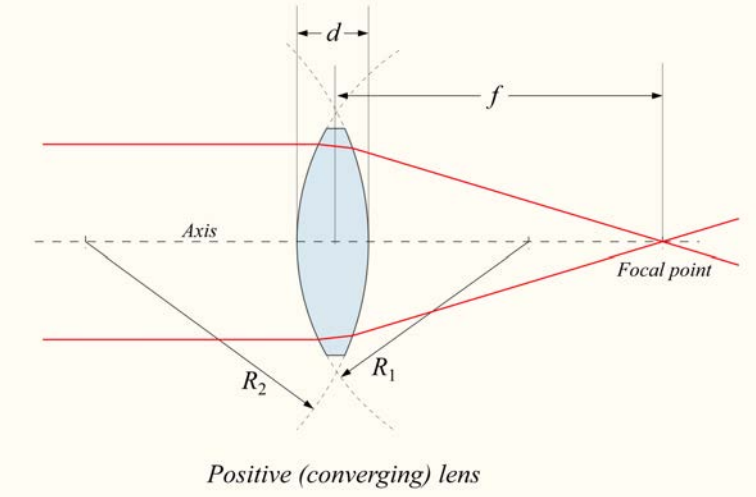


Figura 2.1: Lente sottile [15]

2.1.1 Lunghezza focale

Esistono vari parametri che identificano il comportamento di una lente tra cui la **lunghezza focale**. La lunghezza focale (f) di una lente, circondata dalla miscela di gas aria, è data dall'equazione:

$$\frac{1}{f} = (n-1)\left[\frac{1}{R_1} - \frac{1}{R_2} + \frac{(n-1)d}{nR_1R_2}\right] \tag{2.2}$$

Dove n è l'indice di rifrazione del materiale della lente, R_1 e R_2 sono i raggi della curvatura delle superfici. Per una lente sottile il prodotto R_1R_2 è molto maggiore del diametro d perciò l'ultimo termine diventa trascurabile e la lunghezza focale di una lente sottile in aria può essere approssimata a :

$$\frac{1}{f} \approx (n-1)\left[\frac{1}{R_1} - \frac{1}{R_2}\right] \tag{2.3}$$

R_1 è considerato positivo se la prima superficie è convessa e negativo se la superficie è concava. I segni sono invertiti per la superficie posteriore della lente: R_2 è positivo se la superficie è concava e negativo se convessa. La convenzione di segno è arbitraria.

2.1.2 Formazione dell'immagine

I raggi che attraversano una lente sottile seguono le seguenti regole:

1. Qualsiasi raggio che entra parallelo all'asse da un lato della lente procede verso il punto focale f dell'altro lato;

2. Qualsiasi raggio che arriva alla lente dopo aver attraversato il punto focale sul lato anteriore, esce parallelo all'asse sull'altro lato;

3. Qualsiasi raggio che passa attraverso il centro della lente non cambia direzione.

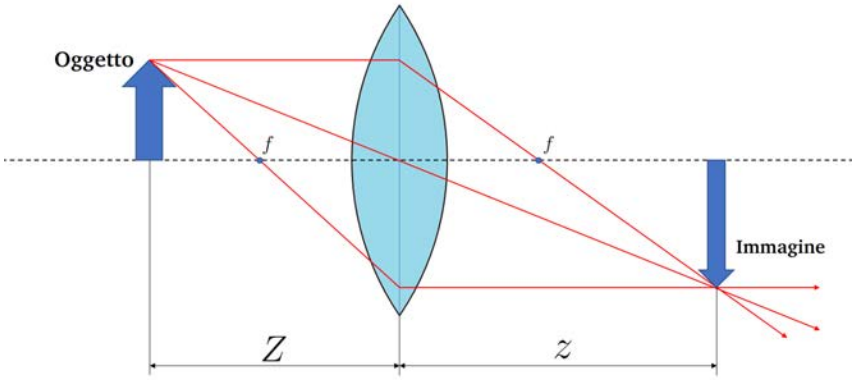


Figura 2.2: Propagazione raggi attraverso lente sottile

L'equazione della **lente sottile** lega la distanza dell'oggetto Z e la distanza dell'immagine z e può essere espressa nella forma:

$$\frac{1}{Z} + \frac{1}{z} = \frac{1}{f} \tag{2.4}$$

2.1.3 Sensore

Per catturare l'immagine è necessario aggiungere alla lente sottile un sensore in grado di convertire l'informazione luminosa in una grandezza elettrica. Esistono diverse famiglie di sensori:

- Sensore **CMOS** (Metallo-Ossido-Semiconduttore Complementare)
- **CCD** (Charge-Coupled Device)

Brevemente, il *CMOS* è un sensore a pixel attivi, ovvero in ogni cella è presente un fotodiodo e uno o più transistor attivi. Il *CCD*, invece è un sensore composto da una griglia di elementi semiconduttori in grado di accumulare carica elettrica in maniera proporzionale all'intensità luminosa incidente. Inoltre sono in grado di trasferire la proprio carica all'elemento adiacente se stimolati con un segnale elettrico adeguato. In questo modo se si fornisce in input a un CCD un treno di impulsi temporizzato, l'output è un segnale elettrico sufficiente per ricostruire i pixel dell'immagine.
In base alla luminosità dell'ambiente circostante e al tipo di scena osservata, oggetto in movimento o in quiete, è possibile migliorare la qualità delle immagini ottenibili variando tre parametri:

1. Apertura del diaframma
2. Tempo di esposizione
3. ISO (guadagno)

Il **diaframma** è un meccanismo usato nell'ottica per regolare la quantità della luce che attraversa un obiettivo, in maniera analoga all'iride dell'occhio umano.

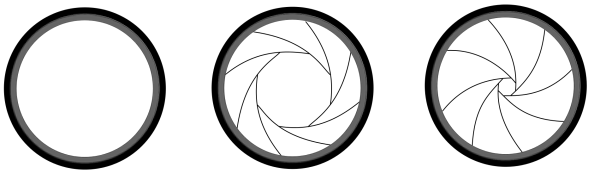


Figura 2.3: Varie aperture del diaframma [16]

La riduzione dell'apertura del diaframma influisce positivamente sulla sfocatura dell'immagine e sulla profondità di campo mentre lo svantaggio principale è la riduzione della quantità di luce utilizzabile. Il **tempo di esposizione** è il tempo durante il quale l'otturatore della macchina fotografica rimane aperto per permettere alla luce di raggiungere il sensore. Lunghi tempi di esposizione determinano un vantaggio in termini di quantità di luce disponibile per la lettura dell'immagine, ma uno svantaggio in termini di *immagini mosse* nel caso in cui la camera o il soggetto si muovano. Infine, l' **ISO** è il fattore di amplificazione usato nella conversione tra carica e tensione. Regolando l'ISO si ottengono immagini più o meno chiare. Come effetto secondario si ha un aumento del rumore.

2.2 Modello Pin Hole

L'evoluzione del modello a lente sottile è il **PinHole model** il quale prevede che l'apertura d tenda idealmente a zero. Questo permette di considerare solo i raggi entranti passanti per il centro ottico andando così a semplificare la trattazione matematica. Questo modello è valido solo per punti appartenenti alla zona di profondità di campo cioè la zona in cui gli oggetti nell'immagine appaiono ancora nitidi e focalizzati (zona di messa a fuoco)

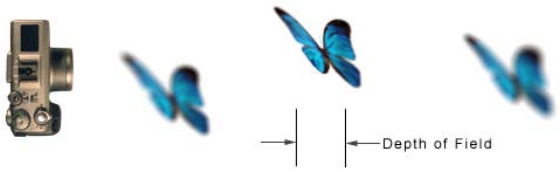


Figura 2.4: Rappresentazione della profondità di campo [17]

Consideriamo un sistema di riferimento 3D con asse X, Y e Z chiamato Camera Coordinates System (CCS), con origine in O, chiamato centro di proiezione.

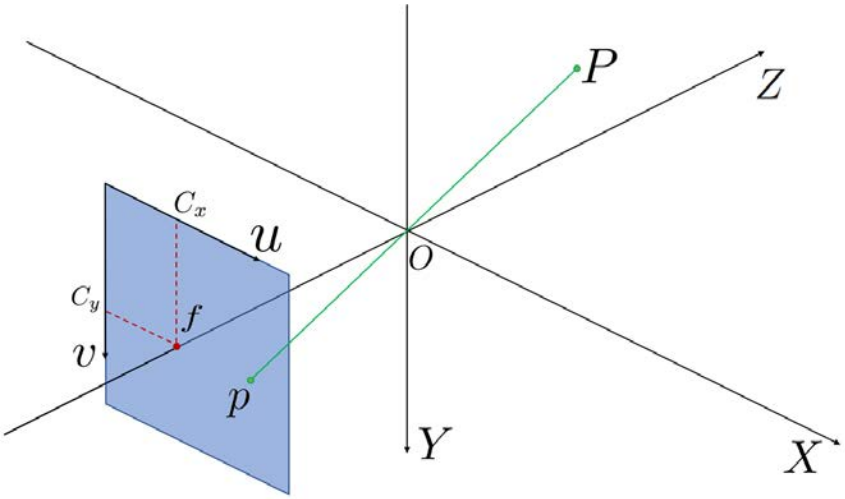


Figura 2.5: sistema di riferimento CCS e S-2 D

Il piano, parallelo al piano X-Y e che interseca l'asse Z alla cordinata f (punto focale), viene chiamato *image plane* S (or sensor plane S). Consideriamo inoltre il sistema di riferimento 2D associato al sensore chiamato S-2D orientato come mostrato in Fig. 2.5. La trasformazione di coordinate tra i sistemi di riferimento è esprimibile nella forma:

$$\begin{cases} u = X + C_x \\ v = Y + C_y \end{cases} \tag{2.5}$$

Dove C_x e C_y sono le coordinate del punto di intersezione tra l'asse Z e il piano del sensore espresse nel sistema di riferimento Sensore. I punti del sistema 2D vengono chiamati pixel, sono espressi tramite le coordinate p= [u,v] e sono ottenuti dell'intersezione dei raggi che collegano il centro di proiezione O con tutti i punti P della scena espressi dalle coordinate P=[X, Y, Z]. La relazione tra P e p è chiamata proiezione prospettica.

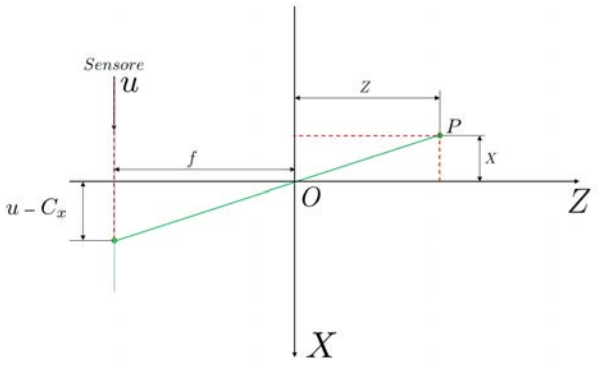


Figura 2.6: Proiezione X-Z

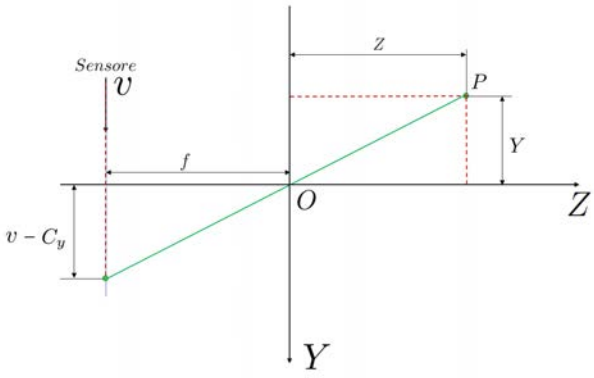


Figura 2.7: Proiezione Y-Z

Grazie alle proprietà della similitudine tra triangoli è possibile scrivere l'equazione:

$$\begin{cases} u - C_x = f\frac{X}{Z} \\ v - C_y = f\frac{Y}{Z} \end{cases} \tag{2.6}$$

Dove f è la lunghezza focale ed ha valore negativo dato l'orientamento degli assi.

2.3 Matrice di calibrazione

Sviluppando algebricamente le equazioni di proiezione si può ottenere una relazione che lega i punti 2D (*pixel*) con i punti 3D (*world*) tramite una matrice K (3x3) chiamata **matrice di calibrazione**. Consideriamo un sistema composto da una camera osservante lo spazio 3D. Si avranno i seguenti sistemi di riferimento:

- CCD: relativo alla camera
- S-2D: relativo al sensore
- World: sistema di riferimento assoluto rispetto al quale sono espresse le coordinate dei punti nello spazio

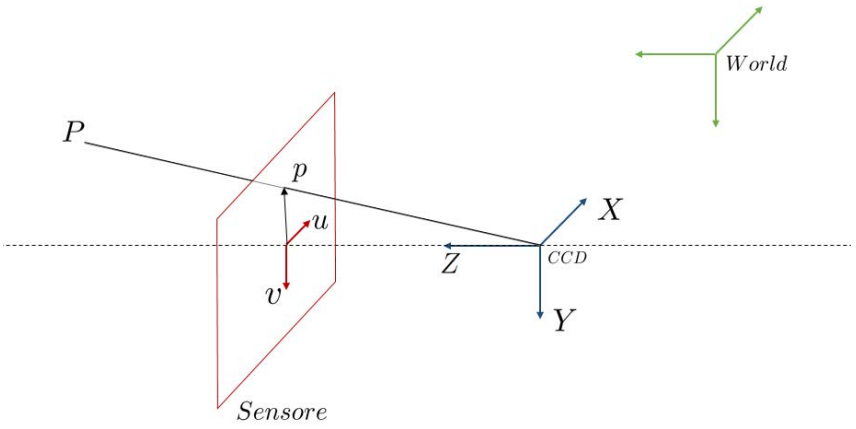


Figura 2.8: Sistema di riferimento camera (CCD), sensore, world

Per passare dal sistema *world* a *CCD* è sufficiente effettuare una rototraslazione che leghi il sistema di coordinate assolute a quello camera:

$$[X] = [R][W] + [t] \tag{2.7}$$

Dove X sono le coordinate del punto P espresse rispetto al *CCD*, W sono le coordinate dello stesso punto però espresse nel sistema di riferimento assoluto *World*, R e t sono rispettivamente la matrice di rotazione e il vettore traslazione che legano i sistemi di riferimento.

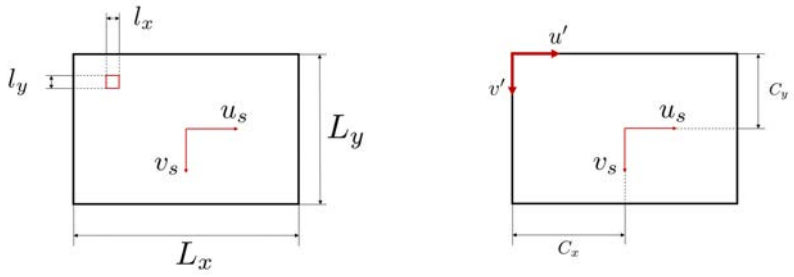
Data l'equazione prospettica Eq. (2.6) scritta in coordinate omogenee e con $C_x = C_y = 0$, cioè con il centro del S-2D posizionato lungo l'asse Z, è possibile scrivere la seguente uguaglianza:

$$\lambda[p] = [k_f][\pi_0][P] \tag{2.8}$$

Dove λ è la **profondità** cioè la distanza del punto 3D lungo l'asse Z espressa nel sistema *CCD*. π_0 è la matrice di proiezione standard canonica, X sono le coordinate dei punti 3D espresse rispetto alla CCD mentre x sono le coordinate dei punti proiettati sul piano sensore. Infine K_f è così composta:

$$[k_f] = \begin{bmatrix} \mathrm{f} & 0 & 0 & 0 \\ 0 & \mathrm{f} & 0 & 0 \\ 0 & 0 & \mathrm{f} & 0 \end{bmatrix} \tag{2.9}$$

Ora è necessario discretizzare il piano sensore in quanto formato da pixel. Inoltre l'origine viene spostata nel vertice in alto a sinistra per convenzione.



(a) Discretizzazione **(b)** Decentramento

Figura 2.9: Sistema di riferimento sensore discretizzato in pixel e decentrato

Come si nota dalla figura Fig. 2.10, il sensore di dimensione $L_x \times L_y$ viene suddiviso in una scacchiera di N pixel, N_x lungo l'asse u e N_y lungo l'asse v. Gli assi del sistema di riferimento discretizzato sono u_s e v_s che decentrati diventano u' e v'. Matematicamente questa trasformazione si traduce con:

$$\begin{bmatrix} u_s \\ v_s \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \tag{2.10}$$

S_x e S_y prendono il nome di *fattori di scala* e permettono la conversione da unità metriche a unità discrete (pixel), la matrice 2×2 che li contiene viene chiamata *matrice di scala*. Geometricamente i loro valori

sono ottenibile come segue:

$$\begin{cases} L_x = N_x l_x \\ L_y = N_y l_y \\ S_x = \frac{N_x}{L_x} = \frac{1}{l_x} \\ S_y = \frac{N_y}{L_y} = \frac{1}{l_y} \end{cases} \tag{2.11}$$

Combinando anche la traslazione del sistema di riferimento si ottiene:

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = [K_s] \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{2.12}$$

Dove la matrice K_s racchiude entrambe le trasformazioni:

$$[K_s] = \begin{bmatrix} S_x & 0 & C_x \\ 0 & S_y & C_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

Perciò, dato il cambio di coordinate, l'equazione Eq. (2.8) può essere riscritta nella forma:

$$\lambda[p'] = [k_s][k_f][\pi_0][P] \tag{2.14}$$

Dove p' esprime le coordinate del punto p nel nuovo sistema di riferimento sensore (discretizzato e decentrato). Infine, il prodotto tra le matrici k_s e k_f prende il nome di **matrice di calibrazione [K]**:

$$[K] = \begin{bmatrix} S_x f & S_\theta f & C_x \\ 0 & S_y f & C_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.15}$$

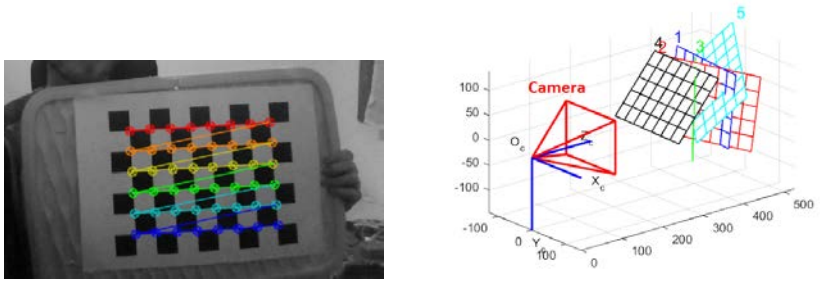
Il termine S_θ è chiamato *Skew Factor*. Questo parametro rappresenta il leggero disallineamento degli assi u e v, nei modelli teorici idealizzati è nullo, ma nelle applicazioni in generale assume valori diversi da zero.

L'Eq. (2.14) lega i punti di un'immagine 2D ai punti della scena 3D.

2.4 Calibrazione camera

La calibrazione di un sistema di visione consiste nella stima, la più accurata possibile, dei parametri che definiscono il modello delle camere presenti. Nel caso di una singola camera il processo di calibrazione permette di trovare i parametri contenuti dalla matrice $[K]$, detti anche parametri intrinseci, e la matrice di rototraslazione che lega il CCD al sistema World (Eq. (2.7)), cioè i parametri estrinseci. Se invece si attua la calibrazione a un sistema con 2 camere, detto stereo camera, si ottengono anche i parametri dalla

matrice di rototraslazione che lega i sistemi di riferimento delle due camere. Uno dei primi metodi di calibrazione è stato sviluppato da Zhang nel 1998 ed è anche noto come metodo a scacchiera. In generale ogni metodo di calibrazione è basato sull'osservazione di un pattern planare noto posto in diversi punti dello spazio 3D e con diverse angolazioni.



(a) Scacchiera su foglio A3 e relativo riconoscimento mediante software

(b) Pose scacchiera

Figura 2.10: Processo di calibrazione [18]

L'algoritmo eseguito durante ogni calibrazione è il seguente:

1. Si acquisiscono n immagini di una scacchiera in diverse pose e si ottengono così i punti x_{ij} cioè le posizioni dei vertici della scacchiera espressi nel sistema di riferimento camera. Il pedice j varia da 1 a m e indica i vertici riconosciuti per scacchiera, mentre il pedice i varia da 1 a n ed è il numero della posa;

2. Si calcola l'errore e_{ij} come differenza tra punti 2D misurati sull'immagine e quelli noti;

3. Si elabora una funzione costo g complessiva:

$$g = \sum_{i=0}^{n} \sum_{j=0}^{m} ||e_{ij}|| \tag{2.16}$$

4. Si procede con la minimizzazione dell'errore e così facendo si trova il modello di proiezione: una stima di $[K]$, $[R]$ e $[t]$.

Per evitare che l'algoritmo converga a un minimo locale in letteratura è consigliato effettuare una linearizzazione del sistema e poi procedere con una ottimizzazione non lineare. In questo modo i parametri calcolati appartengono al minimo globale.

2.4.1 Effetti di distorsione

Sebbene durante la realizzazione delle lenti i disturbi e le distorsioni vengano resi marginali, il modello reale si discosta (spesso anche di molto) dal PinHole model. Questo scostamento viene modellato tramite due distorsioni:

- **distorsioni radiali**: dovute alla curvatura delle lenti;

- **distorsioni tangenziali**: effetti secondari dovuti al decentramento tra le varie lenti e da eventuali difetti di produzione.

Questi tipi di distorsione, radiale e tangenziale, portano ad avere le immagini acquisite alterate. Nello specifico si parla di **distorsione a barile** se l'immagine presenta un' inarcatura verso l'esterno, mentre si parla di **distorsione a cuscino** se l'inarcatura è rivolta verso il centro.

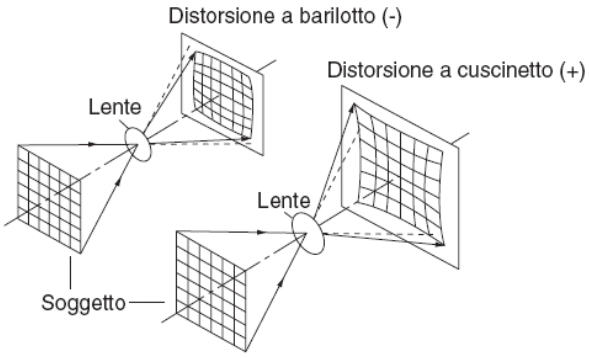
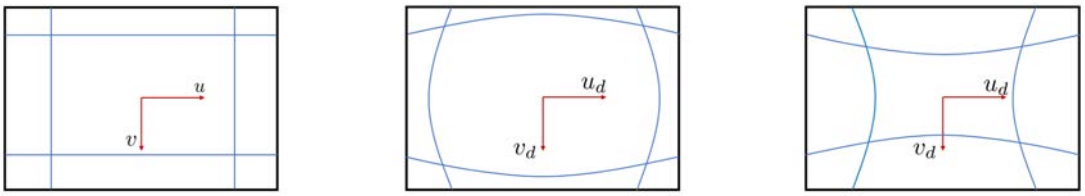


Figura 2.11: Proiezione prospettica [19]



(a) Immagine reale **(b)** Distorsione a barile **(c)** Distorsione a cuscino

Figura 2.12: Confronto tra immagine reale, distorsione a barile e a cuscino

Solitamente questo fenomeno viene modellato con relazioni non lineare tra i punti proiettati sul piano immagine. In letteratura si trovano diversi polinomi di distorsione, un esempio è il seguente tratto da *OpenCV library*:

$$\begin{cases} u_d = u[1 + k_1(u^2 + v^2) + k_2(u^2 + v^2)^2 + k_5(u^2 + v^2)^3] + 2k_3uv + k_4(3u^3 + v^2) \\ v_d = v[1 + k_1(u^2 + v^2) + k_2(u^2 + v^2)^2 + k_5(u^2 + v^2)^3] + 2k_3(u^2 + 3v^2) + 2k_4uv \end{cases} \tag{2.17}$$

Dove k è un coefficiente dei polinomi e, in generale, un modello è tanto più accurato quanto più alto è il grado delle equazioni che lo descrivono. Grazie al processo di calibrazione è possibile correggere le distorsioni presenti nelle immagini.

2.5 Sistema di stereo visione

Una stereo camera è un sistema formato da 2 mono-camere le quali sono disposte in maniera tale da poter osservare la stessa scena 3D, ma con 2 differenti punti di vista. Grazie a questa configurazione è

possibile, partendo dalle immagini fornite dal sistema, calcolare la distanza di un punto dal sistema di visione.
Si consideri, inzialmente, due camere poste una fianco all'altra.

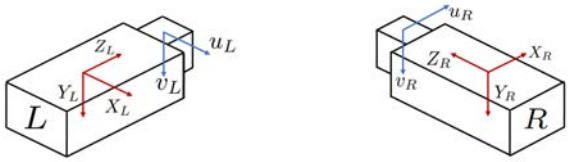


Figura 2.13: Modello stereo camera

Per ognuna si consideri il sistema di riferimento CCS e il sistema di riferimento sensore 2D, per semplicità la camera di sinistra viene chiamata *Left* (con coordinate 3D X_L, Y_L, Z_L e $2Du_L, v_L$) e quella di destra *Right* (con coordinate 3D X_R, Y_R, Z_R e 2D u_R, v_R).Grazie al processo di calibrazione stereo è possibile ottenere la matrice di trasformazione che lega i sistemi di riferimento CCD-Left e CCD-Right. Data questa conoscenza è possibile effettuare la rettifica software delle immagini ed è possibile semplificare la trattazione matematica andando a considerare le due camere perfettamente allineate, poste alla stessa altezza e a una determinata distanza tra loro.

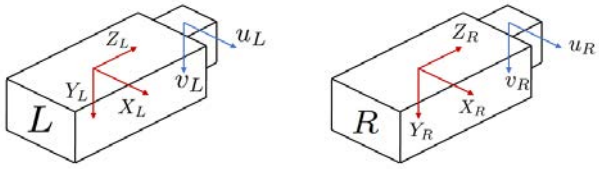


Figura 2.14: Modello stereo camera con camere allineate

Perciò, il modello stereo camera si può schematizzare come segue:

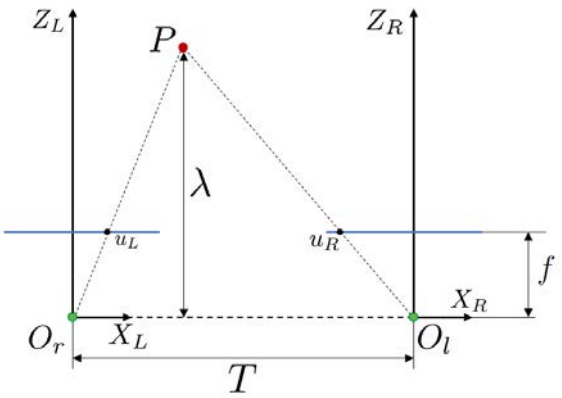


Figura 2.15: schematizzazione modello stereo camera

Dove O_R e O_L sono rispettivamente l'origine del CCS di Right and Left, T è la distanza nota tra O_R e O_L, f è la distanza focale, P è un punto nello spazio 3D e la sua proiezione nel piano sensore è rappresentata

dai punti p_L e p_R avente coordinate [u_L, v_L] e [u_R, v_R]. Dato che i CCS sono allineati e sono posti alla stessa altezza, si nota che v_R=v_L=V . Tramite l'ottica geometrica è possibile trovare la relazione che lega la profondità λ del punto P ad altri parametri noti:

$$\lambda = \frac{fT}{u_L - u_R} \tag{2.18}$$

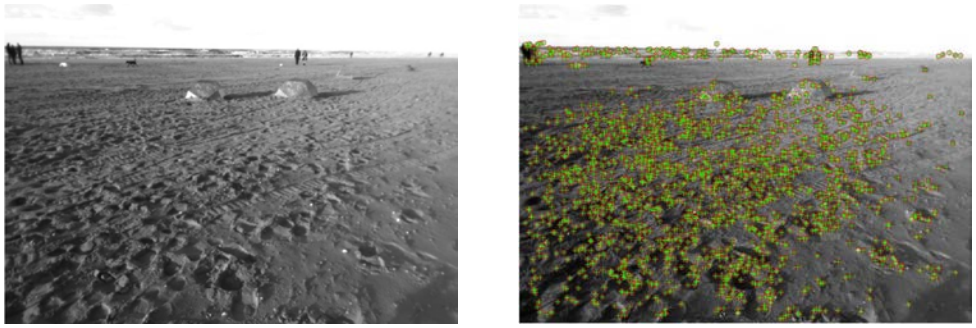
La differenza tra u_R e u_L è detta disparità ed indica quanto sono vicini le proiezioni degli stessi punti 3D nei vari sistemi di riferimento sensore: quando è molto bassa significa che il punto è molto lontano, viceversa se ha valore elevato significa che il punto è vicino alla stereo camera.

2.6 Feature detector

Per il riconoscimento di uno stesso punto nella coppia di immagini disponibili dalla camera *Left* e *Right* si utilizzano degli algoritmi chiamati *feature detector*. Una *feature* consiste in un *pattern* locale dell'immagine che differisce dal contorno in termini di intensità, colore e texture. Esistono diverse tipologie di feature detector, distinguibili in due grandi famiglie: i *blob detector* (e.g. SIFT, SURF, CENSURE) e i *corner detector* (e.g. Harris, Shi-Tomasi, FAST). Solitamente i corner detector sono più veloci ma tendono a soffrire della scala e delle variazioni di rotazione dell'immagine. Al contrario i *blob detector* non hanno quest'ultimo problema, ma richiedono un tempo di individuazione delle *feature* maggiore. Infine, a ogni *feature* è associato un *descriptor* che descrive l'intorno del punto trovato.

Tabella 2.1: Tabella riassuntiva delle caratteristiche dei vari feature detector

	Descriptor Angoli	Descriptor Blob	Invariante alla rotazione	Invariante alla scala	Invariante ai sistemi affini	Ripetibilità	Accuratezza	Robustezza	Efficienza
Harris	✓		✓			⋆⋆⋆	⋆⋆⋆	⋆⋆	⋆⋆
Shi-Tommasi	✓		✓			⋆⋆⋆	⋆⋆⋆	⋆⋆	⋆⋆
FAST	✓		✓	✓		⋆⋆	⋆⋆	⋆⋆	⋆⋆⋆
SIFT		✓	✓	✓	✓	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆	⋆
SURF		✓	✓	✓	✓	⋆⋆⋆	⋆⋆	⋆⋆	⋆⋆
CENSURE		✓	✓	✓	✓	⋆⋆⋆	⋆⋆	⋆⋆⋆	⋆⋆⋆



(a) Spiaggia di Katwijk in Olanda, immagine ESA **(b)** Detection Feature tramite SURF

Figura 2.16: Esempio di utilizzo del detector SURF, foto ESA [20]

2.6.1 Feature matching

Grazie ai detector vengono identificate le *features* nelle immagini, ma non si ha ancora la corrispondenza tra feature presenti nell'immagine di *Left* e *Right*. Per risolvere questo problema si effettua la procedura di *matching*.

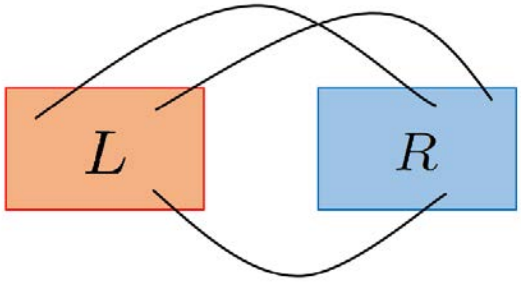


Figura 2.17: Feature matching

Perché avvenga il matching devono essere soddisfatti due vincoli:

1. Similitudine: i descrittori delle feature devono essere simili
2. Vincolo epipolare: richiede che il punto coniugato a un punto p_1 appartenente all'immagine Left giaccia su una linea retta nell'immagine Right chiamata linea epipolare di p_1.

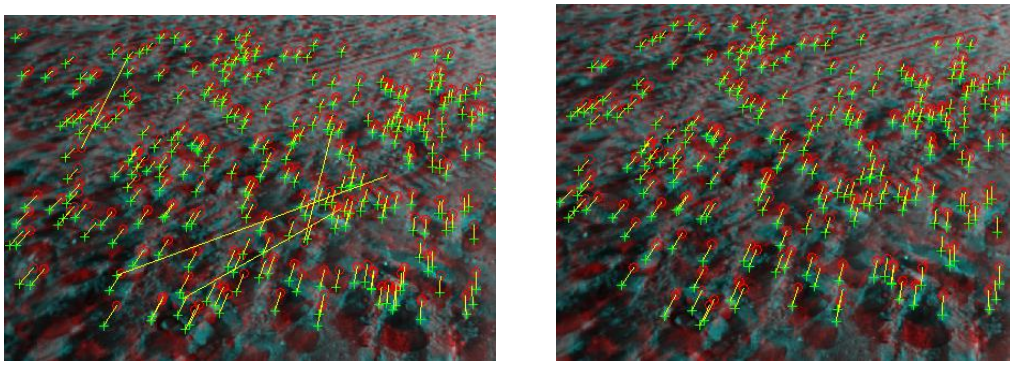
Il primo vincolo non è sufficiente in quanto se in una scena 3D sono presenti oggetti simili i loro descrittori saranno ugualmente simili, causando un errore nell'algoritmo di matching. Quindi è necessario introdurre un vincolo geometrico: il vincolo epipolare. Per soddisfare le restrizioni è sufficiente valutare la distanza tra i descrittori e applicare un filtraggio di tipo RANSAC (RANdom SAmple Consensus).La distanza può essere valutata come **SDA** (**S**omma delle **D**ifferenze **A**ssolute) oppure come **SDQ** (**S**omma delle **D**ifferenze al **Q**uadrato).

$$SDA = \sum_{i=1}^{n} |f_1(i) - f_2(i)| \tag{2.19}$$

$$SDQ = \sum_{i=1}^{n} [f_1(i) - f_2(i)]^2 \tag{2.20}$$

Per ogni descrittore di feature nel fotogramma *Left* è calcolata la distanza da tutti i descrittori del fotogramma *Right*. Se la disparità di una coppia è inferiore al limite di soglia impostato allora la corrispondenza viene considerata valida. Questo approccio è rapido e diretto, ma poco flessibile. Infatti i cambiamenti di scala, la rotazione, illuminazione ed effetti delle distorsioni ottiche inducono variazioni numeriche nel descrittore.

Il filtraggio RANSAC è un algoritmo iterativo efficace per trovare il modello di interpolazione migliore dato un insieme denso di valori. In questo modo è possibile eliminare i falsi *matching* andando a rimuovere gli *outlier*, ovvero i dati che non sono rappresentati dal modello matematico.



(a) *Matching* di *feature* tra immagine destra e sinistra contenente sia *inlier* che *outlier*

(b) Matching di feature filtrato col metodo RANSAC

Figura 2.18: Esempio di utilizzo del filtraggio RANSAC

Capitolo 3

Teoria dei Manipolatori

L'obiettivo primario di un robot è di interagire con l'ambiente circostante e spesso questa azione si traduce nel dover muovere degli oggetti. Risulta quindi ovvia la necessità di effettuare uno studio sui manipolatori. In questo capitolo sono riportati i concetti principali della meccanica riferita alla robotica. Nello specifico: si introdurrà il concetto di manipolatore, rappresentato in Fig. 3.1. In seguito si esporranno i concetti base della cinematica diretta fino ad arrivare alla presentazione della cinematica inversa. Lo scopo è di riportare le basi teoriche, relative al braccio robotico, su cui poggia lo sviluppo della tesi avanzato al Capitolo 5.

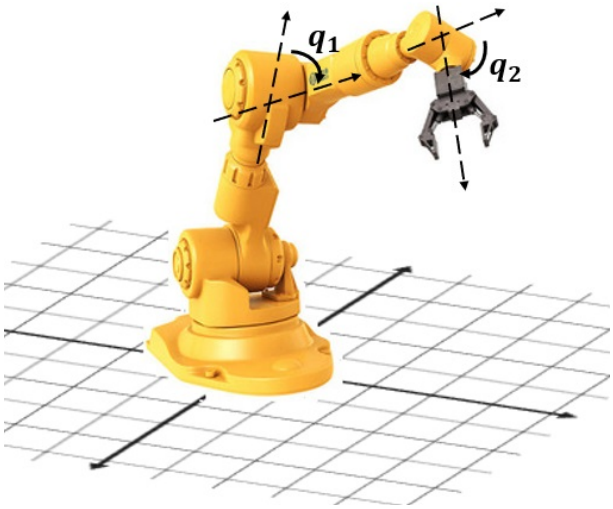


Figura 3.1: Esempio di manipolatore [21]

Il lavoro presentato è una rielaborazione della trattazione presente nel libro di testo fondamentale per la robotica scritto dal professor Siciliano [22].

3.1 Teoria dei manipolatori

Un modello standard di manipolatore non esiste, però in ognuno si può identificare una struttura meccanica precisa la quale consiste in una sequenza di corpi rigidi (*links*) interconnessi tramite delle articolazioni (*joints*), come in Fig. 3.2.

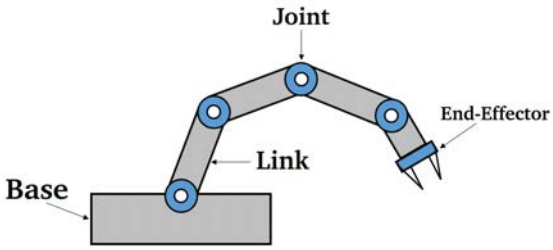
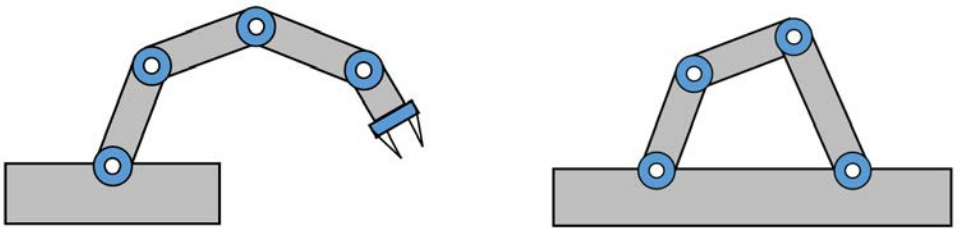


Figura 3.2: Esempio schematico di manipolatore

In questa struttura possono essere individuati 3 sottogruppi:

- arm: assicura mobilità al sistema
- wrist: conferisce destrezza
- end-effector (EE): parte del sistema che interagisce con l'ambiente, es. pinza

Esistono diverse classificazioni dei manipolatori, una tra queste è in base al tipo di catena cinematica che possiede. Catena cinematica aperta (*open kinematic chain*), dal punto di vista topologico, si definisce quando c'è solo una sequenza di link che collegano le due estremità della catena (EE e base). Al contrario, quando una sequenza di *links* forma un loop si parla di catena cinematica chiusa (*closed kinematic chain*).



(a) Catena cinematica aperta

(b) Catena cinematica chiusa: quadrilatero articolato

Figura 3.3: Esempio di catena cinematica

La caratteristica principale di un manipolatore è la possibilità di muoversi che è garantita dalla presenza dei *joints*. Questi ultimi possono essere descritti da due forme primitive: i giunti prismatici e i giunti rotoidali che rispettivamente permettono una traslazione tra due *link* successivi e una rotazione. Nelle catene cinematiche aperte, ogni giunto fornisce alla struttura un grado di libertà aggiuntivo (*degree of freedom* o DOF).

(a) Giunto rotoidale **(b)** Giunto prismatico

Figura 3.4: Tipi di joints

Al fine di ottenere un manipolare con un'elevata destrezza è necessario effettuare un'adeguata progettazione sul posizionamento e sul tipo di giunti da utilizzare. Per avere la totale mobilità, in termini di traslazione e rotazione, sono necessari un minimo di 6 gradi di libertà (disposti in maniera congrua) che permettono il posizionamento 3D dell'*end effector* e il suo orientamento rispetto ad ogni asse. Per applicazioni complesse si è soliti inserire dei gradi di libertà ulteriori detti ridondanti i quali forniscono maggiore destrezza al sistema.

Lo spazio di lavoro è la parte dell'ambiente raggiungibile dall'end-effector. La sua conoscenza è fondamentale per poter pianificare, in maniera consapevole, le operazioni che il manipolatore dovrà eseguire e la forma è strettamente correlata alla struttura di quest'ultimo. Data l'enorme importanza che hanno, il tipo di giunti e la loro disposizione definiscono un'ulteriore classificazione dei manipolatori:

- *Catesian*: è realizzato da 3 giunti prismatici con assi, tipicamente, mutualmente ortogonali tra loro

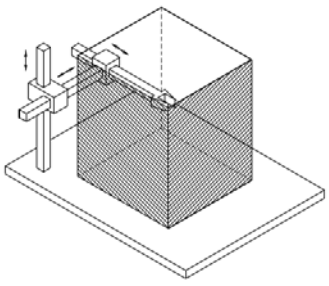


Figura 3.5: Manipolatore cartesiano e il suo workspace [22]

- *Cylindrical geometry*: differisce dal cartesiano in quanto il primo giunto è di rotazione anziché prismatico

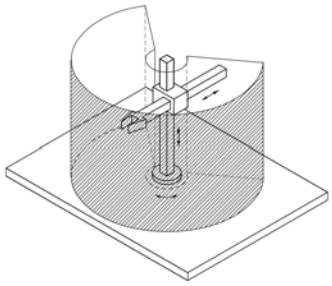


Figura 3.6: Manipolatore cilindrico e il suo workspace [22]

- *Spherical*: differisce dal precedente in quanto anche il secondo giunto è di rotazione

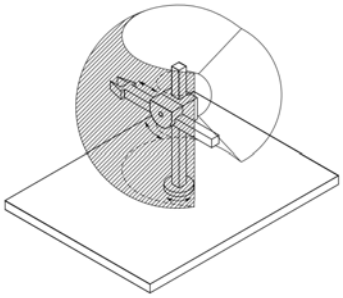


Figura 3.7: Manipolatore sferico e il suo workspace [22]

- *SCARA* (Selective Compliance Assembly Robot Arm) : realizzabile disponendo 2 giunti di rotazione e uno prismatico in modo tale che tutti gli assi di movimento siano paralleli

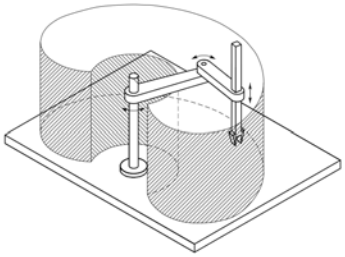


Figura 3.8: Manipolatore SCARA e il suo workspace [22]

- *Anthropomorphic*: realizzato da 3 giunti rotoidali dove l'asse di rotazione del primo giunto è ortogonale all'asse degli altri 2 che sono paralleli

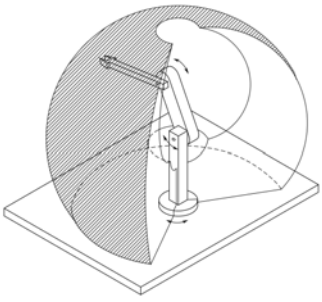


Figura 3.9: Manipolatore antropomorfico e il suo workspace [22]

Componente fondamentale per un manipolatore è l'end-effector (EE) che ne definisce l'impiego. Nel caso sia necessario interagire con oggetti, ad esempio spostandoli, l'EE potrebbe essere un gripper di dimensioni e forma adeguate. Mentre per altre applicazioni, come il carotaggio del terreno, si potrebbe utilizzare una trivella. L'utilizzo di un EE adeguato è fondamentale per il completamento delle varie task, ma restano altrettanto fondamentali i requisiti di progetto come l'accuratezza nel posizionamento, prestazioni dinamiche e dimensioni workspace.

3.2 Posa di un corpo rigido

Per descrivere la posa di un corpo rigido nello spazio è necessario conoscere la sua posizione e il suo orientamento rispetto a un sistema di riferimento. Consideriamo un sistema di riferimento ortonormale con centro O e assi X, Y, Z.

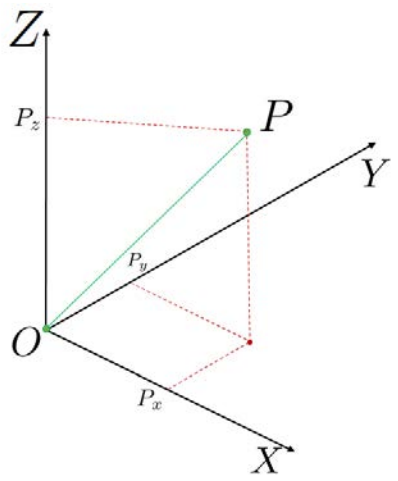
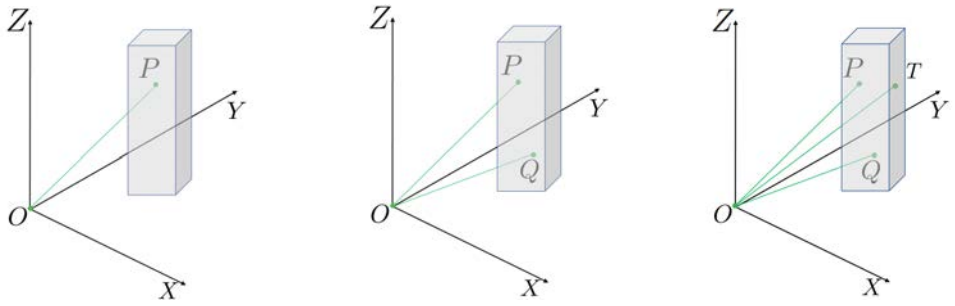


Figura 3.10: Sistema di riferimento O_{XYZ} e proiezione del punto P

La posizione di un generico punto P è esprimibile tramite il vettore:

$$[P] = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{3.1}$$

La conoscenza di un solo punto di un corpo rigido non ne definisce in maniera univoca l'orientamento nello spazio, è quindi necessario avere altri due punti non allineati, T e Q .



(a) Posa indeterminata **(b)** Posa indeterminata **(c)** Posa determinata

Figura 3.11: Determinazione posa di un corpo rigido

In alternativa all'utilizzo dei tre punti, per identificare la posa di un corpo rigido è consuetudine utilizzare un sistema di riferimento solidale al corpo detto **sistema di riferimento Body** con centro O_B e assi X_B, Y_B e Z_B. Esprimendo la posizione del centro del sistema Body e l'orientamento dei suoi assi rispetto alla terna fissa O_{XYZ} si identifica in maniera univoca la posa del corpo.

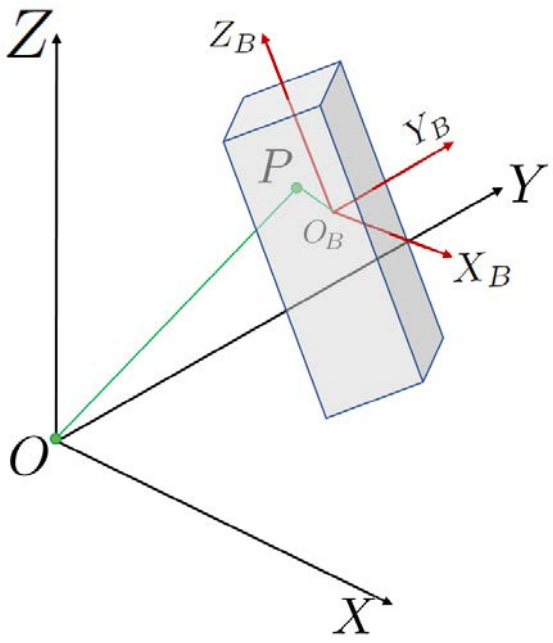


Figura 3.12: Sistema di riferimento fisso e Body

Per esprimere le coordinate del punto generico P rispetto al sistema di riferimento fisso O_{XYZ} conoscendo le coordinate nel sistema di riferimento Body è necessario applicare una rototraslazione

$$[P_F] = [R][P_B] + [t] \tag{3.2}$$

Dove P_F sono le coordinate espresse nel sistema fisso, P_B sono le coordinate espresse nel sistema Body, t e R sono rispettivamente il vettore traslazione e la matrice di rotazione che legano i due sistemi di riferimento.

3.2.1 Matrice di rotazione e Roll Pitch Yaw

La matrice di rotazione fornisce l'orientamento relativo tra due sistemi di riferimento. E' una matrice 3×3, ma non tutti gli elementi sono indipendenti dati i sei vincoli di ortogonalità e questo implica che siano sufficienti tre parametri per descrivere l'orientamento di un corpo rigido. Spesso si utilizza una sequenza di tre angoli, detti angoli di Eulero:

$$\phi = \begin{bmatrix} \varphi \\ \theta \\ \psi \end{bmatrix} \tag{3.3}$$

Combinando tre rotazioni elementari, garantendo che due successive non siano attorno allo stesso asse, è possibile ottenere la matrice di rotazione. Perciò una generica rotazione può essere ottenuta combinando 3 rotazioni adatte.

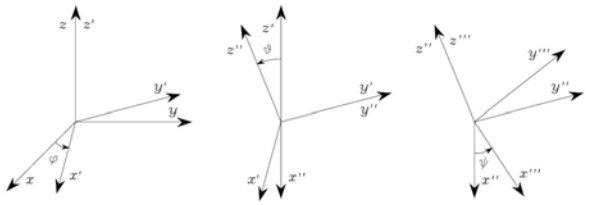


Figura 3.13: Rappresentazione angoli di Eulero [22]

Una terna di angoli di Eulero molto usata nel settore aerospaziale per indicare l'assetto di un corpo è la terna **Roll-Pitch-Yaw**.

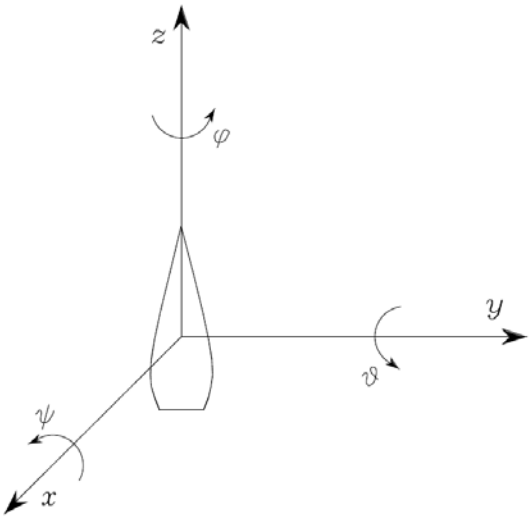


Figura 3.14: Rappresentazione angoli di Roll-Pitch-Yaw [22]

In questo caso, considerando la terna Body fissa al centro di massa del corpo, come in Fig. 3.14, la rotazione del corpo definita dalla terna Eq. (3.3) è ottenibile mediante questo procedimento:

- ruotare il sistema di riferimento di un angolo ψ attorno all'asse x (yaw)

- ruotare il sistema di riferimento di un angolo θ attorno all'asse y (pitch)
- ruotare il sistema di riferimento di un angolo φ attorno all'asse z (roll)

L'orientamento finale del sistema è ottenuto moltiplicando in maniera opportuna le matrici di rotazione elementari:

$$\begin{aligned} R(\phi) &= R_z(\varphi)R_y(\theta)R_x(\psi) \\ &= \begin{bmatrix} c_\varphi c_\theta & c_\varphi s_\theta s_\psi - s_\varphi c_\psi & c_\varphi s_\theta c_\psi + s_\varphi s_\psi \\ s_\varphi c_\theta & s_\varphi s_\theta s_\psi + c_\varphi c_\psi & s_\varphi s_\theta c_\psi - c_\varphi s_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix} \end{aligned} \tag{3.4}$$

Viceversa, nel caso sia necessario passare da una matrice di rotazione R ai corrispondenti angoli di Roll-Pitch-Yaw è sufficiente confrontarla con la matrice Eq. (3.4) per ottenere un legame tra elementi di R ed angoli di Eulero.
Se θ appartiene all'intervallo $(\pi/2, \pi/2)$ e indicando con r_{ij} l'elemento alla riga i e alla colonna j della matrice di rotazione, la soluzione è:

$$\begin{aligned} \varphi &= \arctan 2(r_{21}, r_{11}) \\ \theta &= \arctan 2(-r_{31}, \sqrt{r_{32}^2 + r_{33}^2}) \\ \theta &= \arctan 2(r_{32}, r_{33}) \end{aligned} \tag{3.5}$$

Mentre se θ appartiene all'intervallo $(\pi/2, 3\pi/2)$:

$$\begin{aligned} \varphi &= \arctan 2(-r_{21}, -r_{11}) \\ \theta &= \arctan 2(-r_{31}, -\sqrt{r_{32}^2 + r_{33}^2}) \\ \theta &= \arctan 2(-r_{32}, -r_{33}) \end{aligned} \tag{3.6}$$

Quando $c_\theta = 0$ Eq. (3.5) e Eq. (3.6) divergono e non è possibile calcolare φ e ψ separatamente, ma solo la loro somma (o differenza).

3.3 Cinematica diretta

Come mostrato nei paragrafi precedenti, un manipolatore può essere schematizzato usando una catena cinematica di corpi rigidi (*links*) connessi da giunti rotoidali o prismatici. Un'estremità della catena è vincolata alla base mentre l'*end effector* è posto sull'altra, nel caso di una catena cinematica aperta. Nell'ottica di dover manipolare un oggetto nello spazio, è necessario descrivere la posizione dell'end-effector rispetto al sistema di riferimento fisso detto base e ciò si ottiene combinando il moto relativo tra un link e il successivo. La cinematica diretta si pone lo scopo di risolvere il problema matematico appena descritto: determinare la posizione dell'End Effector rispetto alla base partendo dalla conoscenza della posizione relativa tra i link.

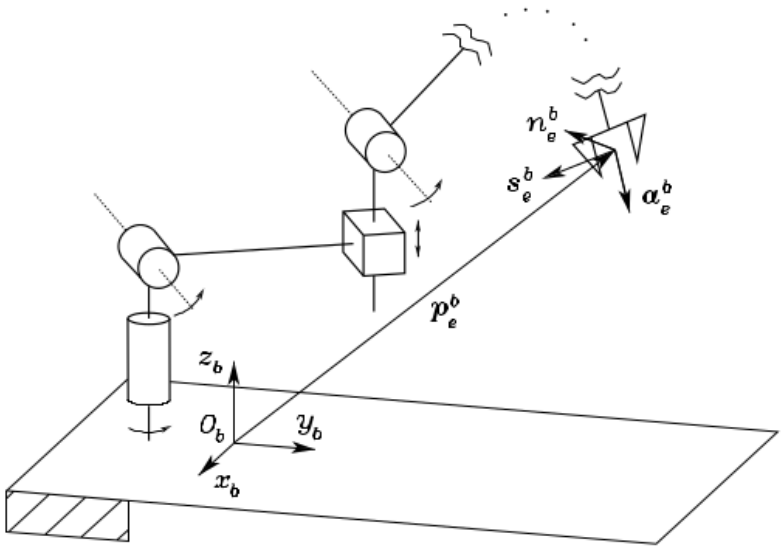


Figura 3.15: Posizione e orientamento dell'EE rispetto alla base [22]

3.3.1 Convenzione di Denavit-Hartenberg

Uno dei metodi più utilizzati per risolvere il problema della cinematica diretta consiste nel associare ad ogni link un sistema di riferimento, costruire le matrici di rototraslazione che legano ogni sistema di riferimento al successivo e moltiplicarle, in maniera opportuna, tra loro. Per semplificare questa procedura, matematicamente parlando, è possibile utilizzare la **convenzione di Denavit-Hartenberg** (DH) la quale introduce alcune regole per la posa dei sistemi di riferimento solidali ad ogni link. Si consideri, in maniera del tutto generale, il seguente manipolatore:

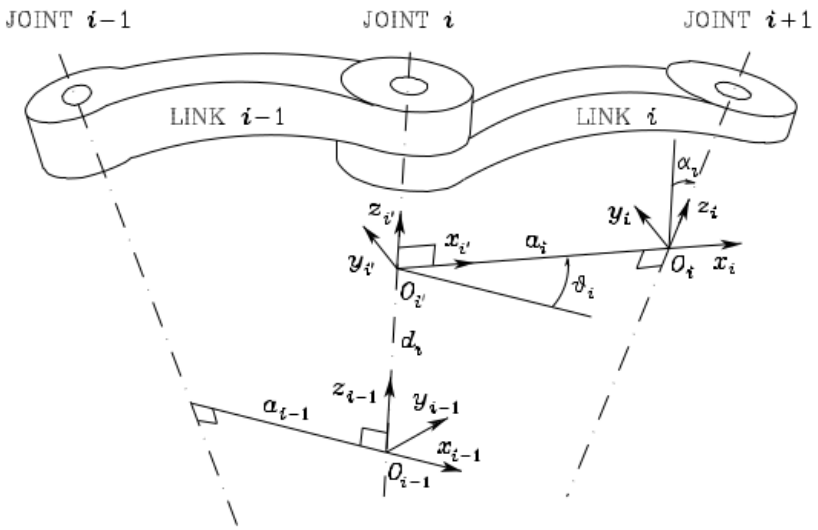


Figura 3.16: Manipolatore generico [22]

La posa del frame i-esimo viene costruita secondo questi vincoli:

- Si posiziona l'asse z_i lungo asse di rotazione del giunto $i + 1$

- L'origine O_i giace nel punto di intersezione dell'asse z_i con la normale congiungente l'asse z_i con z_{i-1} (cioè l'asse z della terna $i-1$)
- Si posiziona il centro o_i' nell'intersezione della normale congiungente asse z_{i-1} e z_i con asse z_{i-1}
- L'asse x_i viene posto lungo la normale congiungente asse z_{i-1} e z_i con direzione positiva dal giunto i al $i+1$
- Asse y_i completa la terna rispettando la convenzione della mano destra

L'architettura del manipolatore in alcuni casi può essere tale da non rendere univoca la costruzione delle terne di riferimento con la convenzione DH:

- Se due giunti consecutivi hanno asse di rotazione parallelo allora la normale congiungente non è unica
- Se due giunti consecutivi hanno assi di rotazione incidenti allora la direzione di x_i è arbitraria
- Quando il giunto i-esimo è prismatico, la direzione z_{i-1} è arbitraria.

Inoltre, anche per la terna 0-esima ed n-esima si hanno delle mancanze di unicità:

- Per il frame 0 solo la direzione z_0 è specificata (o_0 e x_0 possono essere scelti arbitrariamente)
- Per il frame N, dato che è l'ultimo non esiste il frame $N+1$ perciò z_n non è univocamente definito

Le situazioni di indeterminazione vengono utilizzate per semplificare ulteriormente la trattazione matematica.

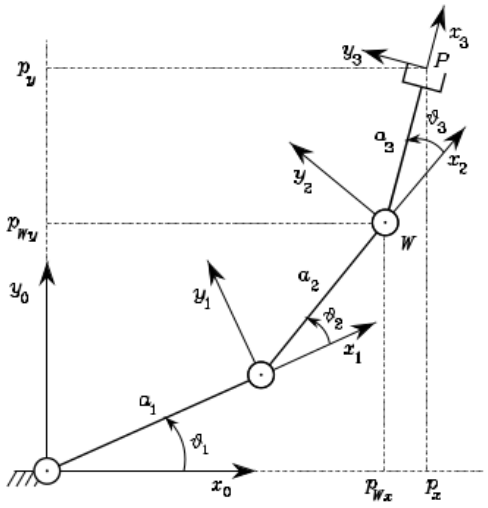


Figura 3.17: Esempio di utilizzo di DH su un robot SCARA [22]

Grazie alla convenzione D-H la posizione e orientamento della terna i rispetto alla $i-1$-esima è completamente definita da 4 parametri:

1. a_i: distanza tra 0_i and $0_i'$
2. d_i coordinata di $0_i'$ lungo z_{i-1}

3. α_i angolo tra asse z_{i-1} e z_i attorno all'asse x (considerato positivo quando la rotazione è antioraria)

4. θ_i angolo tra asse x_{i-1} e x_i attorno all'asse z_{i-1} (considerato positivo quando antiorario)

I parametri a_i e α_i sono sempre costanti in quanto dipendono unicamente dalla geometria del manipolatore mentre θ e d sono variabili:

- Se il giunto i-esimo è di rivoluzione allora θ_i è variabile
- Se il giunto i-esimo è prismatico la variabile è d_i

Per la costruzione della **matrice di rototraslazione** si procede allineando le terne di riferimento eseguendo una serie di rotazioni e traslazioni elementari.
Considerando la terna $i-1$ è necessario traslarla di d_i lungo l'asse z_{i-1} e ruotarla di θ_i lungo l'asse z_{i-1}. Grazie a questa sequenza la terna $i-1$ viene allineata alla terna i'. In forma matriciale è riassumibile come segue:

$$A_{i'}^{i-1} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

Si procede traslando il frame $i-1$ di a_i lungo l'asse x_i' e ruotandolo di α_i attorno all'asse x_i. Ora la terna $i-1$ è stata allineata con la terna i-esima.

$$A_i^{i'} = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.8}$$

La matrice di trasformazione complessiva si ottiene moltiplicando tra loro ogni trasformazione elementare:

$$\begin{aligned} A_i^{i-1}(q_i) &= A_{i'}^{i-1} A_i^{i'} \\ &= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} c_{\alpha_i} & s_{\theta_i} s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i} c_{\alpha_i} & -c_{\theta_i} s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{3.9}$$

Si pone l'attenzione sul fatto che la matrice espressa da Eq. (3.9) è funzione della sola variabile q_i, relativa al medesimo giunto. Questa variabile è θ_i nel caso in cui il giunto sia di rivoluzione, mentre nel caso di giunto prismatico la variabile è d_i.

La seguente procedura può essere applicata a qualsiasi manipolatore avente una catena cinematica aperta.

3.4 Cinematica inversa

La cinematica inversa, in maniera speculare a quella diretta, si pone l'obiettivo di determinare la posizione relativa tra i vari link nota la posizione dell'*end effector* rispetto al sistema di riferimento fisso. La soluzione di questo problema è di fondamentale importanza nell'ottica di far eseguire all'*end effector* un movimento in quanto, solitamente, si desidera spostarlo da un punto generico A al successivo B espressi nel sistema di riferimento base. Risulta quindi necessario poter calcolare gli spostamenti relativi da far eseguire ai giunti per compiere questo spostamento. Purtroppo, rispetto alla cinematica diretta, il problema della cinematica inversa è più complesso da risolvere per le seguenti ragioni:

- solitamente la soluzione non esiste in forma chiusa data la presenza di equazioni non lineari
- possono esistere soluzioni multiple, come in Fig. 3.18
- nel caso ci siano componenti ridondanti le soluzioni possono essere infinite
- esistono soluzioni non realizzabili per la geometria del manipolatore, come quella in Fig. 3.19

Il problema delle soluzioni multiple è favorito dall'elevato numero di gradi di libertà e da un alto numero di parametri non nulli nella convenzione D-H.

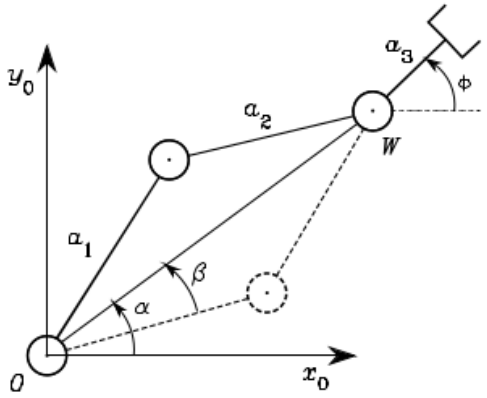


Figura 3.18: Esempio di soluzione multipla [22]

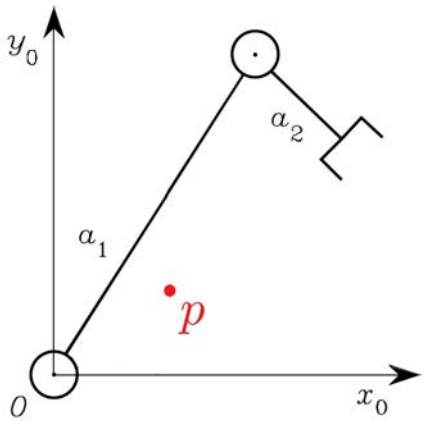


Figura 3.19: Punto p non raggiungibile dall'end-effector

Ovviamente, l'esistenza di una soluzione è garantita esclusivamente se la posizione obiettivo si trova nello spazio destro del manipolatore.

Per le ragioni sopra descritte è difficile elaborare un algoritmo in grado di risolvere la cinematica inversa in forma chiusa. Tuttavia, in alcuni casi la geometria del manipolatore può suggerire punti privilegiati rispetto a cui effettuare i calcoli. In tutte le altre situazioni si utilizzano dei metodi numerici che richiedono tempi e potenze di calcolo spesso superiori ma garantiscono una soluzione, se possibile.

Capitolo 4

Hardware del rover e ROS

In questo capitolo verrà presentata una panoramica sull'hardware del rover Morpheus indirizzando il focus sui sottosistemi utilizzati in questo progetto di tesi. Nel dettaglio verranno introdotti due sottogruppi:

- Sistema di visione
- Braccio robotico

Per ogni sottogruppo verranno mostrati i dati tecnici forniti dall'azienda costruttrice oppure i dati di progetto nel caso di componenti realizzati direttamente dal team. Infine si illustrerà **ROS** (Robotic Operation System), il sistema operativo utilizzato per la comunicazione dei vari sottosistemi.

4.1 Sistema di stereo visione

Un rover necessita di essere il più autonomo possibile perciò deve essere in grado di percepire l'ambiente circostante e trarne le dovute informazioni: presenza o meno di ostacoli lungo il percorso preventivato, avvallamenti o dune, campioni da raccogliere oppure immagini da inviare alla stazione terrestre. Per questo si utilizza un sistema di visione. Al momento attuale il sistema di visione di Morpheus conta unicamente di un stereo camera, nella fattispecie la stereo camera ZED, e di un Lidar.

Figura 4.1: Stereo camera ZED [23]

La ZED è un prodotto di STEREOLABS, azienda leader nella realizzazione di sensori e software per il rilevamento della profondità e del movimento 3D.

La Zed utilizzata è il primo modello relativo alla sua famiglia di prodotti, ma offre comunque prestazioni elevate garantendo precisione e affidabilità. Le sue ridotte dimensioni, visibili nella Tabella 6.7, e le alte prestazioni, mostrate nella Tabella 4.2, la rendono ideale per l'utilizzo sui rover. Inoltre, l'azienda produttrice mette a disposizione degli *SDK* (software development kit) molto accurati e con tutorial ben strutturati per insegnare all'utente come utilizzare i propri prodotti al meglio; l'unico inconveniente è che necessitano di un hardware del PC specifico, in particolare la scheda grafica NVIDIA.

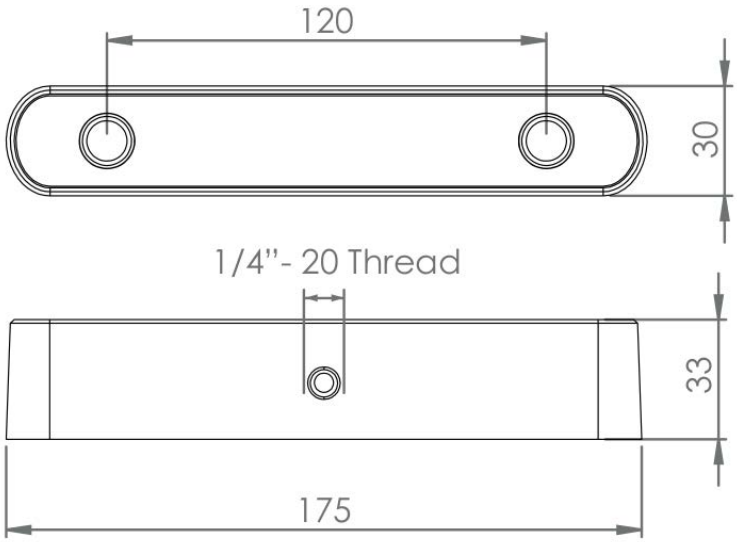


Figura 4.2: Dimensioni zed [23]

Tabella 4.1: Parametri fisici [23]

Dimensioni	175x30x33mm
Peso	170g
Alimentazione	380mA / 5V (USB Powered)
Operating Temperature	0°C ÷ 45°C

Tabella 4.2: Parametri Video [23]

Risoluzione	2x (1920x1080) @30fps
Campo di vista	Max. 90°(H) x 60°(V) x 100°(D)
Lunghezza focale	2.8mm (0.11)
Interfaccia	USB 3.0

4.2 Braccio robotico

In un robot, il braccio robotico probabilmente è il sottosistema più importante perché conferisce alla macchina la possibilità concreta di interagire con l'ambiente. In un rover, la mansione più comune è di raccogliere campioni rocciosi, ma in futuro ci si aspetta che possano essere utilizzati anche come aiutanti per astronauti. Il braccio di Morpheus è dotato di quattro giunti rotoidali che gli conferiscono altrettanti

gradi di libertà. Due sono posti alla base permettendo la rotazione rispetto all'asse Z (giunto 1) e all'asse Y (giunto 2), come rappresentato nella Fig. 4.3, il terzo giunto permette il movimento del secondo link mentre l'ultimo permette la rotazione dell'end-effector. I link presenti sono due, di sezione cilindrica cava.

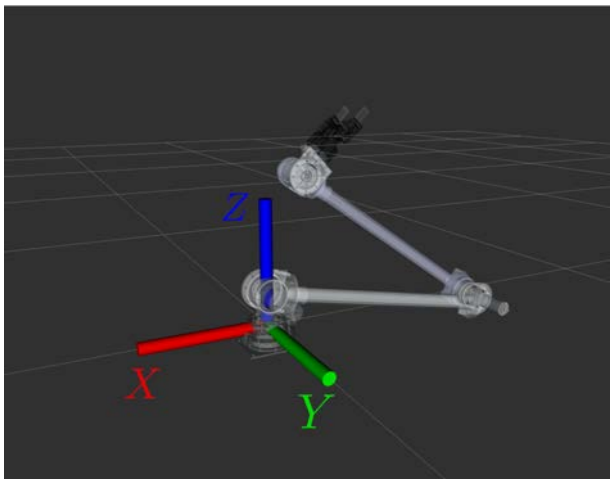


Figura 4.3: Braccio Morpheus e Sistema di riferimento fisso

Tabella 4.3: Parametri link

Sezione	Circolare cava
Lunghezza	570mm
Diametro maggiore	40mm
Diametro minore	37mm
Materiale	Carbonio
peso	0.16kg

I motori utilizzati sono prodotti dall'azienda *Maxon motor* e presentano le seguenti caratteristiche [24]–[26]:

Tabella 4.4: Parametri motore giunto 1

Tipo	Brushless
Alimentazione	24V
Coppia Nominale	107 mNm
Corrente Nominale	3.64 A
Coppia di stallo	1800 mNm
Corrente di stallo	63.1 A
Riduttore esterno	103:1
Peso motore e riduttore	430g

Tabella 4.5: Parametri motore giunto 2 e 3

Tipo	Brushless
Alimentazione	24V
Coppia Nominale	434 mNm
Corrente Nominale	8.96 A
Coppia di stallo	12200 mNm
Corrente di stallo	253 A
Riduttore esterno	81:1
Peso motore e riduttore	1590g

Tabella 4.6: Parametri motore giunto 4

Tipo	Brushless
Alimentazione	24V
Coppia Nominale	128 mNm
Corrente Nominale	3.21 A
Coppia di stallo	1460 mNm
Corrente di stallo	39.5 A
Riduttore esterno	156:1
Peso motore e riduttore	600g

Al fine di raccogliere oggetti, come end-effector è stato scelto di utilizzare un gripper con 2 fingers, in particolare il modello fornito dalla casa produttrice *Robotiq*: 2F-85

Figura 4.4: Pinza modello 2F-85 della Robotiq [27]

L'unione del braccio con l'end-effector restituisce il seguente risultato:

Figura 4.5: Braccio robotico a 4 gradi di libertà con il gripper 2F-85 della Robotiq [4]

4.2.1 Gripper

Il *gripper* utilizzato è il modello 2F-85 della Robotiq di Fig. 4.4[27]. La casa produttrice, come spesso accade, mette a disposizione dell'acquirente innumerevoli pacchetti software per semplificarne l'utilizzo. Nel caso del modello 2F-85 sono disponibili programmi di configurazione per *Windows* e grazie alla community di *GitHub* anche i pacchetti riguardanti ROS. Il nome del modello è esplicativo del numero di dita (*fingers*) e della distanza tra un dito e l'altro nel momento in cui la pinza ha la massima apertura: 85 mm.

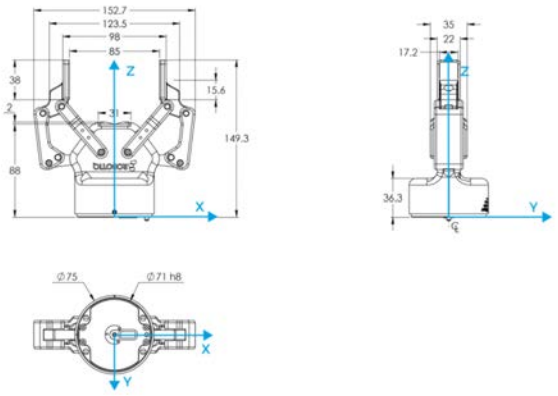


Figura 4.6: messa in tavola gripper in posizione aperta

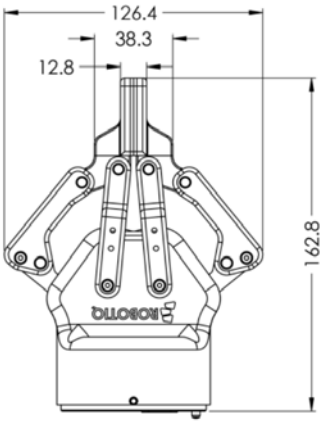


Figura 4.7: messa in tavola gripper in posizione chiusa

Nella Tabella 4.7 vengono riassunti gli ingombri:

Tabella 4.7: Ingombri 2F-85 [27]

Apertura massima pinza	85mm	
Altezza	Max 162.8mm	min 149.3
Larghezza	Max 152.7mm	min 149.3
Spessore	75mm	

L'interfaccia grafica permette di controllarne l'apertura, la velocità di esecuzione e il feedback in forza.

Tabella 4.8: Specifiche gripper 2F-85 [27]

Corsa regolabile	0-85 mm
Forza grip	20-235 N
Massa pinza	0.9 kg
Precisione posizionamento	0.4 mm
Velocità avanzamento	20-150 mm/s
IP (ingress protection)	IP40
Alimentazione	24V

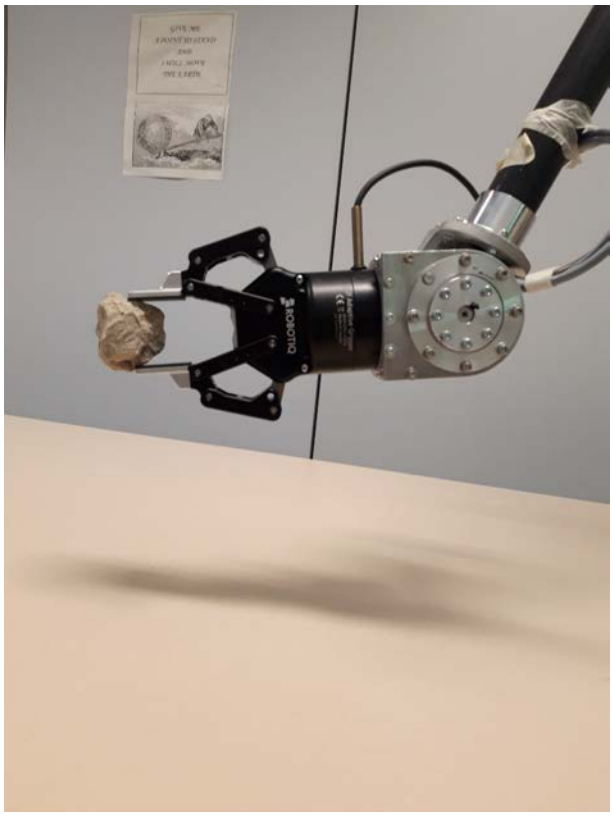


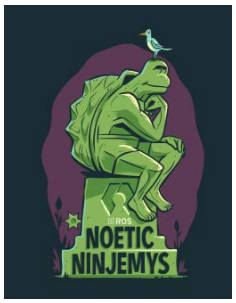
Figura 4.8: Raccolta di un campione roccioso tramite gripper 2F-85

4.3 ROS

Per applicazioni complesse, in cui esistono vari sottosistemi che spesso devono interagire tra loro, è necessario utilizzare un sistema operativo che coordini le operazioni. Nell'ambito della robotica ne esiste uno dedicato allo sviluppo di tali sistemi: **ROS** (**R**obot **O**perating **S**ystem) [28], il cui logo è in Fig. 4.9. ROS è un framework molto flessibile utilizzato per la scrittura di software dedicati ai robot. La sua architettura è molto complessa e conta numerose librerie, tools e convenzioni utili a semplificare il processo di implementazione software. Inoltre, essendo open-source la documentazione è facilmente reperibile e grazie a una community molto presente il livello di avanzamento tecnologico è molto alto; alcune aziende produttrici di sistemi hardware sviluppano anche software compatibili con ROS per facilitarne l'utilizzo, come StereoLabs Sezione 4.1.

Figura 4.9: Logo ROS (Robot Operating System))[28]

Esistono molte release di ROS, le ultime sono Kinetic, Melodic e Noetic. Ogni release è sviluppata per un determinato sistema operativo, in generale Linux. In particolare la versione Kinetic è supportata da Ubuntu 16, la Melodic da Ubuntu 18 e la Noetic (in Fig. 4.10a) da Ubuntu 20 (in Fig. 4.10b) e non solo.



(a) Logo Noetic

(b) Logo Ubuntu

Figura 4.10: Loghi di Noetic e Ubuntu [28], [29]

4.3.1 Funzionamento

Grazie alla sua capacità di intermediario tra diverse applicazioni ROS prende il nome di middleware. Un singolo software che svolge un'azione elementare (o più di esse) è chiamato nodo. I nodi tra loro comunicano tramite topic. Un topic è un canale di comunicazione tra nodi tramite cui si scambiano messaggi, ogni topic è identificato da un nome (deciso dall'utente) e dal proprio tipo che definisce il genere del messaggio che viene scambiato.

Figura 4.11: Rappresentazione figurata di un middleware [30]

Una volta installata la propria versione di ROS, è necessario costruire il *workspace* in cui si svilupperanno i vari programmi. All'interno del *workspace* sono presenti tre cartelle: *build*, *devel* e *src*; ognuna di esse contiene file dedicati al funzionamento dell'intero sistema. All'interno della cartella src l'utente potrà inserire i vari pacchetti, insieme di file come driver o algoritmi, e dopo aver compilato l'ambiente di lavoro tramite gli appositi comandi, se non si presentano errori, i software potranno essere utilizzati.

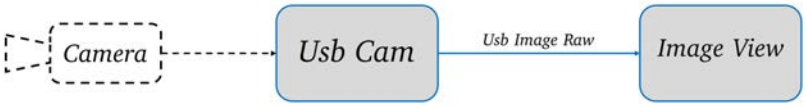


Figura 4.12: Esempio funzionamento nodo *Usb Cam*, *Image View* e *topic Usb Image Raw*

Nella Fig. 4.12 sono riportati due nodi: *usb cam* e *image view* i quali si occupano rispettivamente della gestione dei driver della camera usb e dell'elaborazione dell'immagine ricevuta in input mostrandola a schermo. Come è facilmente intuibile, il sistema acquisisce le immagini tramite la camera usb, tramite il topic *usb image raw* le comunica al nodo image view il quale le pubblica a schermo.

Grazie a questa organizzazione l'ambiente ROS è un ambiente modulare e ciò comporta un enorme vantaggio: ogni nodo è autonomo sia durante il funzionamento sia per quanto riguarda l'organizzazione dei file all'interno del PC. Perciò, nel caso si debbano apportare delle modifiche al robot, è possibile sostituire un nodo con l'equivalente nuovo rispettando la nomenclatura dei *topic*.

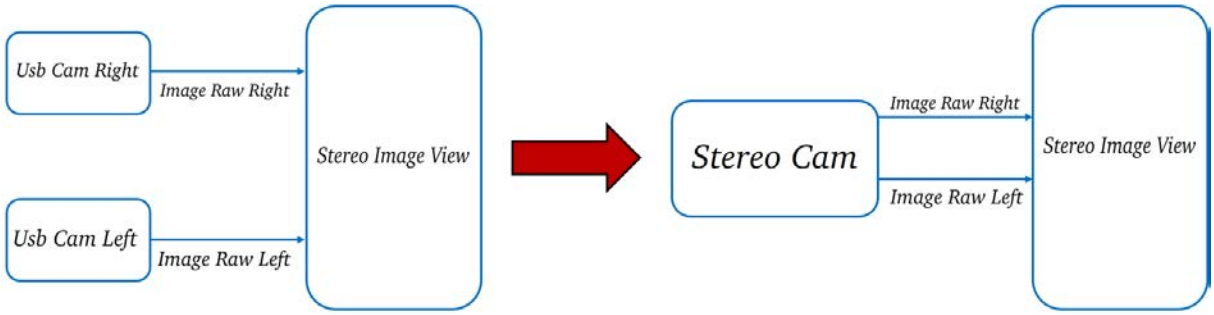


Figura 4.13: Modifica sistema di visione

I linguaggi di programmazione supportati sono:

- C++
- Python
- XML

I primi due sono utilizzati per la scrittura dei nodi mentre l'XML viene utilizzato principalmente per la stesura dei file **Launch**. Un file Launch contiene l'elenco dei vari nodi che ROS dovrà eseguire, consente di rimappare i topic e di definire il valore dei parametri usati nei topic.

4.3.2 Interfaccia grafica

Dato lo scopo ultimo di ROS, cioè controllare un robot, esso non necessita di un'interfaccia grafica (GUI), tuttavia, per semplificarne l'utilizzo da parte dei programmatori, ne esistono diverse. Le principali sono

rqt e **Rviz**. Grazie ad esse è possibile monitorare nel dettaglio il funzionamento del robot: cosa percepiscono i sensori, come vengono interpretate le informazioni e quali comunicazioni avvengono e quali no. Perciò sono il principale strumento di debugging. Rqt è un insieme di GUI che possono essere lanciate separatamente e ognuna esegue una funzione elementare. Spesso vengono utilizzate per monitorare l'esecuzione dei nodi oppure per modificare i parametri presenti nei messaggi, altre mostrano le frequenze di pubblicazione di quest'ultimi e così via.

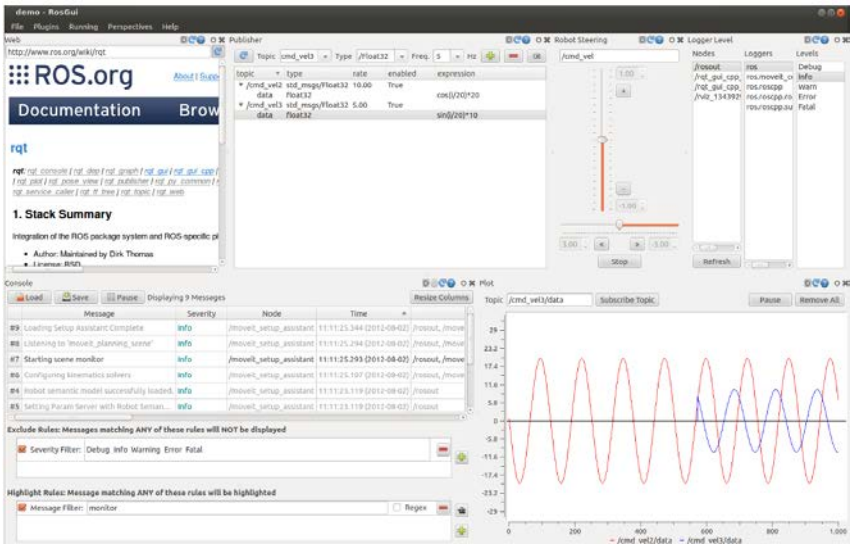


Figura 4.14: Pannello di controllo mostrante più GUI [31]

Rviz è un ambiente di visualizzazione 3D. Permette la rappresentazione tridimensionale del robot ed è in grado mostrare a video le informazioni raccolte dai sensori.

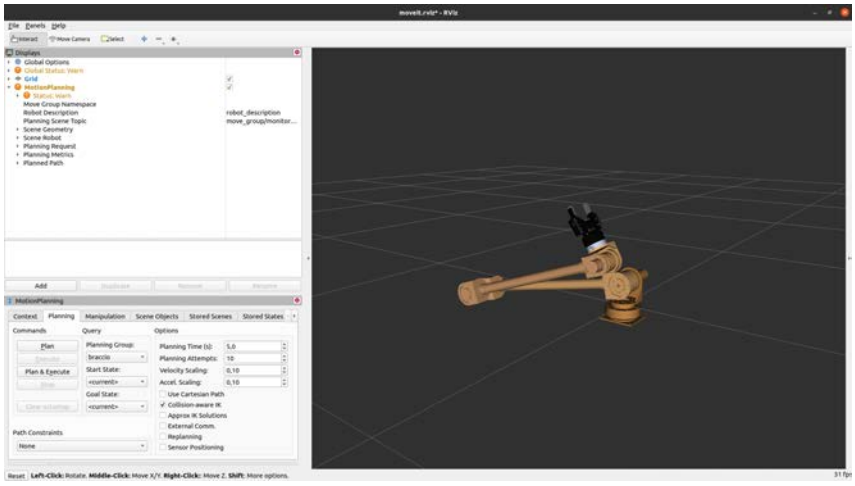


Figura 4.15: Pannello di controllo di Rviz mostrante braccio del rover Morpheus [32]

Capitolo 5

Implementazione

L'obiettivo del progetto di tesi è l'implementazione di un algoritmo in grado di riconoscere un campione roccioso mediante stereo camera e di raccoglierlo tramite il braccio robotico. In questo capitolo verrà presentata la linea guida del progetto soffermandosi su come sono state sviluppate le varie sezioni. Il lavoro svolto può essere suddiviso in tre macro argomenti:

- Riconoscimento oggetto
- Calibrazione occhio-mano
- Raccolta e posa

La prima sezione implementa un metodo per la localizzazione dell'oggetto trovando la sua posa nello spazio $3D$, con la calibrazione occhio-mano si individua la matrice di rototraslazione che lega il sistema di visione a quello di raccolta: braccio robotico. Infine, tramite il manipolatore si procede con la raccolta e la posa del campione in un punto prestabilito. Per la realizzazione sono stati utilizzati i sistemi hardware presenti nelle Sezioni 4.2 e 4.1, le versioni di Ubuntu e ROS impiegate sono state rispettivamente la 20.04 e la release Noetic.

La seguente esposizione è frutto dell'elaborazione delle conoscenze presenti in letteratura [33]–[39].

5.1 Riconoscimento oggetto

Nella prima prima parte del progetto veniva richiesto di determinare la posa di un campione di roccia nello spazio $3D$ mediante l'utilizzo di una stereo camera (Sezione 4.1). Grazie allo studio di una coppia di immagini stereo è stato possibile ricostruire la *pointcloud* della scena osservata, la quale è stata opportunamente filtrata e analizzata col fine di estrapolarne l'oggetto.
L'algoritmo è sintetizzabile tramite i seguenti passaggi:

1. Acquisizione immagini
2. Calibrazione stereo camera
3. Elaborazione immagini

4. Ricerca oggetto

5.1.1 Acquisizione immagini

Per l'acquisizione immagini si è scelto di utilizzare il pacchetto *usb cam* [40], fornito dalla *community* di ROS, il quale permette il dialogo tra il sistema di visione fisico, cioè la stereo camera ZED (Sezione 4.1), e gli altri software. I parametri richiesti per il corretto funzionamento del nodo sono i seguenti:

- Risoluzione: 2208x1242 pixel
- Frame rate: 30fps
- Formato Pixel: yuyv

Figura 5.1: Nodo Usb Cam

Il nodo usb cam fornisce un solo topic contenente entrambe le immagini, per dividerle è sufficiente introdurre un nuovo pacchetto chiamato *Splitter*, realizzato dai componenti del team negli anni scorsi.

Figura 5.2: Nodo Usb Cam e Splitter

Il nodo Splitter, come mostrato dall'immagine Fig. 5.2 , fornisce in output due topic distinti contenenti l'immagine di destra e di sinistra. A valle dell'ultimo nodo è consigliato inserire un sincronizzatore di topic: nodo *Synch*. Il suo funzionamento è elementare: aspetta l'arrivo dei topic selezionati come input e una volta che possiede le informazioni di ognuno li pubblica con lo stesso time-step.

Figura 5.3: Sistema di stereovisione sincronizzato

5.1.2 Calibrazione stereo camera

In generale le immagini fornite da una stereo camera sono affette da distorsioni e, come mostrato nella Sezione 2.4, per eliminarle è necessario eseguire il processo di calibrazione. E' stato scelto di eseguirla su ROS utilizzando il pacchetto *camera calibration* [41] che è fornito di tutorial esaustivi per il suo utilizzo. Il funzionamento è analogo a quello descritto in Sezione 2.4: necessita di una scacchiera di dimensioni note osservata in diverse pose. Inoltre, il pacchetto è provvisto di una GUI la quale mostra in tempo reale lo stato di avanzamento della calibrazione.

Figura 5.4: GUI di calibrazione stereo [41]

In accordo con la teoria è necessario effettuare diverse pose per ottimizzare il processo:

1. Traslazione del pattern lungo asse X (da sinistra verso destra o viceversa)
2. Traslazione del pattern lungo asse Y (dall'alto verso il basso o viceversa)
3. Traslazione del pattern lungo asse Z (da vicino a lontano, in modo da riempire il campo di vista)
4. Rotazione attorno asse Z del pattern con vari angoli rispetto alla camera
5. Inclinazione del pattern verso destra, sinistra, alto e basso.

Eseguite con successo le pose richieste il software avrà abbastanza dati per ricostruire il modello camera e fornire i parametri intrinseci ed estrinseci del sistema stereo camera.

5.1.3 Elaborazione immagini

Grazie al sistema di visione schematizzato in Fig. 5.3, si hanno a disposizioni le immagini raw della camera destra e sinistra, combinandole con i parametri ottenuti dalla calibrazione è possibile ottenere le immagini rettificate e non distorte. ROS fornisce un pacchetto adatto a questo tipo di operazione: **Stereo Image Proc** [42]. Dandogli in input i quattro *topic* contenenti l'immagine di sinistra, di destra e i parametri della camera di sinistra e destra, fornisce in output le immagini rettificate e non distorte. Inoltre, lo stesso pacchetto restituisce la mappa delle disparità e la *point cloud*, come mostrato dall'immagine seguente:

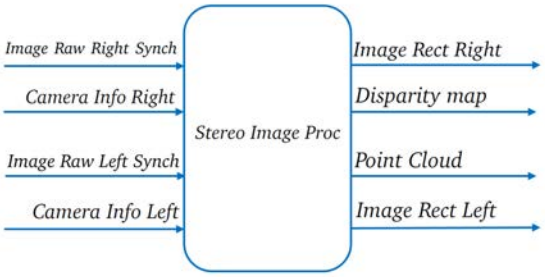


Figura 5.5: Nodo *Stereo Image Proc*

La costruzione della *disparity map*, e quindi della *point cloud*, è eseguita tramite *OpenCV's block matching* ed è funzione dei seguenti parametri:

1. Correlatio windowsize (int, default: 15): dimensione del lato della finestra di matching espresso in pixel

2. Min disparity (int, default: 0): soglia della disparità minima. Impostando un valore positivo verranno identificati punti vicini alla camera (oggetti distanti non verranno trovati)

3. Disparity range (int, default: 64): dimensione della finestra di ricerca per la mappa di disparità espressa in pixels. Insieme a min disparity definisce il volume dello spazio $3D$ su cui viene effettuata la mappa delle disparità

4. Texturethreshold (int, default: 10): soglia minima di texture necessaria a una feature per essere considerata valida

5. Speckle size (int, default: 100): numero minimo di pixel che una regione deve avere per non essere scartata

6. Speckle range (int, default: 4) : regioni diverse della mappa di disparità vengono raggruppate se la loro distanza è minore di questo valore

É stato possibile effettuare una taratura dinamica dei parametri appena elencati grazie al tool di ROS chiamato *rqt* (Sezione 4.3.2) il quale mette a disposizione dell'utente numerose GUI tra cui: *rqt reconfigure*

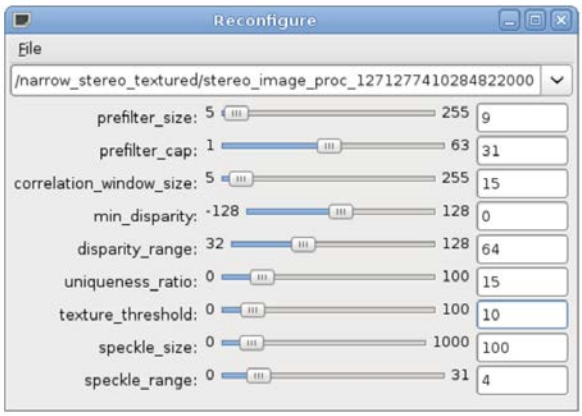


Figura 5.6: GUI rqt reconfigure

Il sistema di riferimento rispetto a cui è definita la point cloud è il CCD relativo alla camera di sinistra.

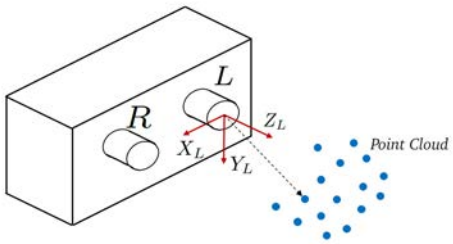


Figura 5.7: Sistema di riferimento point cloud

5.1.4 Ricerca oggetto

La ricerca oggetti è stata sviluppata partendo dalla pipeline di MoveIt chiamata **Perception** [43]. L'algoritmo, dopo aver effettuato i diversi calcoli, è in grado di identificare un oggetto associandolo a un cilindro. Le operazioni eseguite sono le seguenti:

1. Filtraggio point cloud
2. Identificazione piano
3. Costruzioni parametri cilindro
4. Ricostruzione cilindro

Filtrando opportunamente la *point cloud* è stato possibile creare un'area di interesse tridimensionale, riducendo così i tempi di calcolo ed evitando falsi matching.

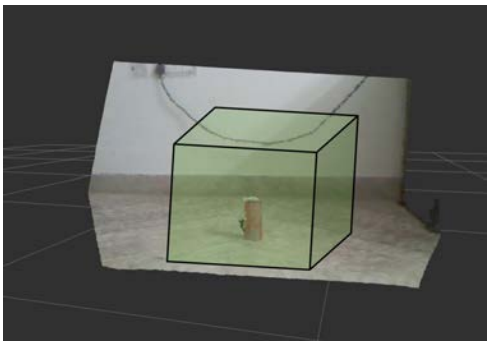
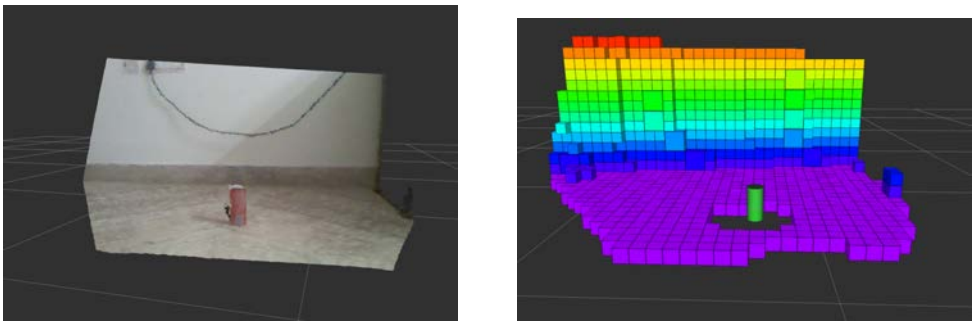


Figura 5.8: Point cloud e area di interesse [43]

Successivamente tramite il metodo RANSAC è stato identificato il piano principale presente nella zona d'interesse. Questa operazione ha permesso di eliminare la superficie su cui poggia l'oggetto dalla ricerca del medesimo. Infine, si identifica l'insieme dei punti assimilabili ad un cilindro, tramite un ulteriore filtraggio RANSAC. Se la *point cloud* risultata paragonabile a un cilindro allora ne vengono identificati i punti di contorno così da definire le dimensioni del cilindro quali l'altezza e il diametro, ma anche l'inclinazione.



(a) *Point cloud* appartenente al tutorial Perception

(b) Identificazione oggetto

Figura 5.9: Identificazione oggetto a partire da una *point cloud* [43]

La conoscenza dell'incertezza sulla determinazione della posa dell'oggetto è fondamentale, in quanto, se è maggiore della corsa massima del gripper il processo di raccolta difficilmente verrà eseguito con successo. In letteratura sono presenti dei modelli per determinare l'errore lungo l'asse Z che si ha utilizzando una stereo camera ZED della SteroLabs. L'errore è in funzione della risoluzione utilizzata e dalla profondità Z per cui lo si intende calcolare. Come mostrato dall'Eq. (5.1), l'errore è proporzionale alla distanza dalla stereo camera:

$$f(Z) = ae^{bZ} \tag{5.1}$$

Dove a e b sono coefficienti tabulati in funzione della risoluzione scelta. L'errore rispetto agli assi X e Y viene sovrastimato considerandolo uguale a $f(Z)$, in accordo con la documentazione scientifica.

Grazie a questa tecnica di riconoscimento oggetti sarà possibile identificare la posa di un campione roccioso.

5.2 Calibrazione occhio mano

Tramite la ricerca oggetto (Sezione 5.1.4) si conosce la posa del campione roccioso rispetto al sistema di riferimento *camera* (Fig. 5.7), ma è necessario determinarla rispetto al sistema di riferimento *base* (Fig. 3.15) in modo da poter effettuare la cinematica inversa (Sezione 3.4) determinando così la posizione relativa che ogni link deve raggiungere. L'algoritmo utilizzato per effettuare questo cambio di coordinate è noto come **calibrazione occhio mano** e permette di ottenere la rototraslazione che lega il sistema di visione, appunto *occhio*, alla base del manipolatore, detto *mano*. Per un corretto funzionamento, l'algoritmo necessita della posa $3D$ di un oggetto perciò, in maniera analoga alla calibrazione di una camera, viene utilizzato un insieme di QR Code di dimensioni note e tramite un software di *riconoscimento tag* è possibile determinarne la posizione nello spazio.

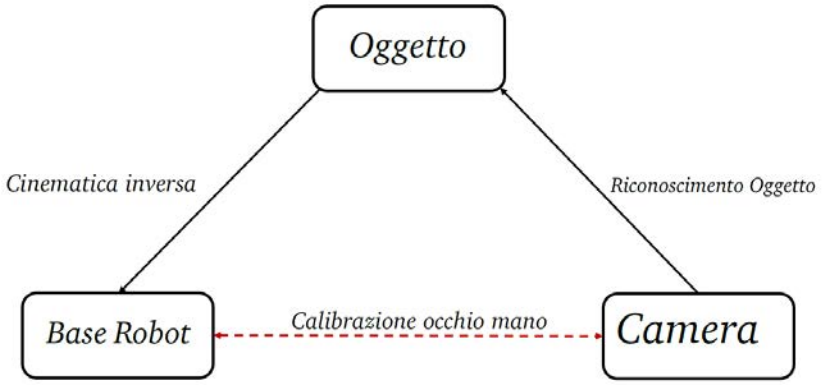
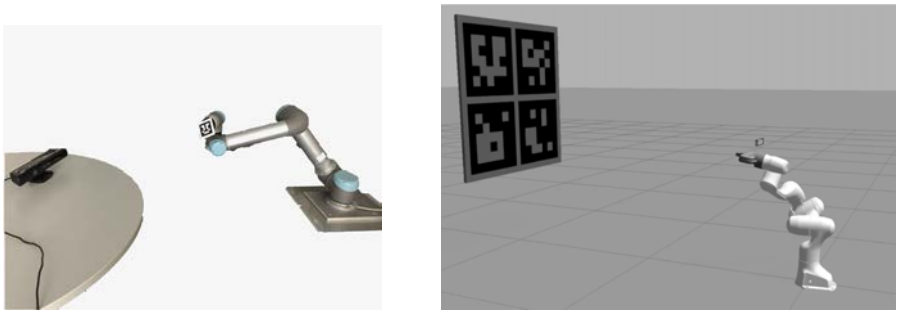


Figura 5.10: Calibrazione occhio mano

5.2.1 Teoria introduttiva

Esistono due tecniche per la calibrazione occhio-mano: la prima è denominata *eye-in-hand* mentre la seconda *eye-on-base*. Come suggeriscono i nomi, esse differiscono per la posizione della camera rispetto al braccio.



(a) Eye on base **(b)** Eye in hand

Figura 5.11: Diverse configurazione dell'*eye* [44], [45]

Il principio matematico alla base è il medesimo per entrambe: conoscendo la distanza tra la camera e il *tag* e la distanza tra *end effector* e base robot in diverse posizioni è possibile calcolare la trasformazione

tra camera e robot risolvendo l'equazione:

$$[A][X] = [X][B] \tag{5.2}$$

Dove, per la calibrazione eye in hand, la matrice A esprime lo spostamento della camera tra una posa e la successiva, la matrice B comunica lo spostamento dell'End Effector tra una posa e la successiva e la matrice incognita X descrive la trasformazione che lega il sistema di visione all'end effector. Se la calibrazione è del tipo eye on base, la matrice A esprime lo spostamento del tag, la matrice B è invariata mentre la matrice X descrive la trasformazione che lega il sistema di visione alla base del robot.

Considerando, in maniere del tutto generale, la calibrazione *eye on hand* (Fig. 5.23b) è possibile identificare quattro trasformazioni per la posa i-esima:

- Camera-tag (matrice nota A_i)
- Base-end effector (matrice nota B_i)
- Camera-end effector (matrice incognita X)
- Base robot- tag (matrice incognita Y)

In maniera analoga è possibile individuarle per una posa generica successiva j-esima. Inoltre si nota che le matrici X e Y sono invarianti al moto del manipolatore perciò $X_i = X_j = X$ e $Y_i = Y_j = Y$.

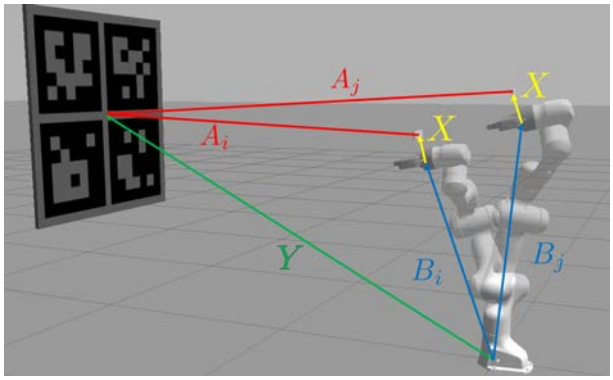


Figura 5.12: Posa i-esima e j-esima con relative trasformazioni [45]

Tramite la Fig. 5.12 è possibile notare come per ogni posa la conoscenza di tre trasformazioni permetta di determinare in maniera univoca la quarta:

$$\begin{aligned}[Y] &= [B_i][X][A_i] \\ [Y] &= [B_j][X][A_j]\end{aligned} \tag{5.3}$$

Dato che la matrice Y è invariante alla posa si può scrivere:

$$[B_i][X][A_i] = [B_j][X][A_j] \tag{5.4}$$

Moltiplicando il primo elemento dell'equazione per l'inverso della matrice B_j e il secondo per l'inverso della matrice A_i si ottiene:

$$[A][X] = [X][B] \tag{5.5}$$

Con:

$$\begin{cases} [A] = [A_j][A_i]^-1 \\ [B] = [B_j]^-1[B_i] \end{cases} \tag{5.6}$$

L'Eq. (5.5) è lineare nell'incognita X perciò le soluzioni ammissibili sono $0, 1$ o ∞. Dato che almeno due soluzioni esistono, per la posa i-esima (o j-esima) e la soluzione triviale $X = 0$, il numero di soluzioni totali è infinito. Un vincolo pone limite al numero di soluzioni: le matrici devono essere di rototraslazione.

5.2.2 Implementazione

Per la versione Noetic di ROS, è disponibile un *plugin* per Rviz (Sezione 4.3.2) [46] il quale permette di eseguire la calibrazione tramite un'interfaccia grafica *user friendly*.

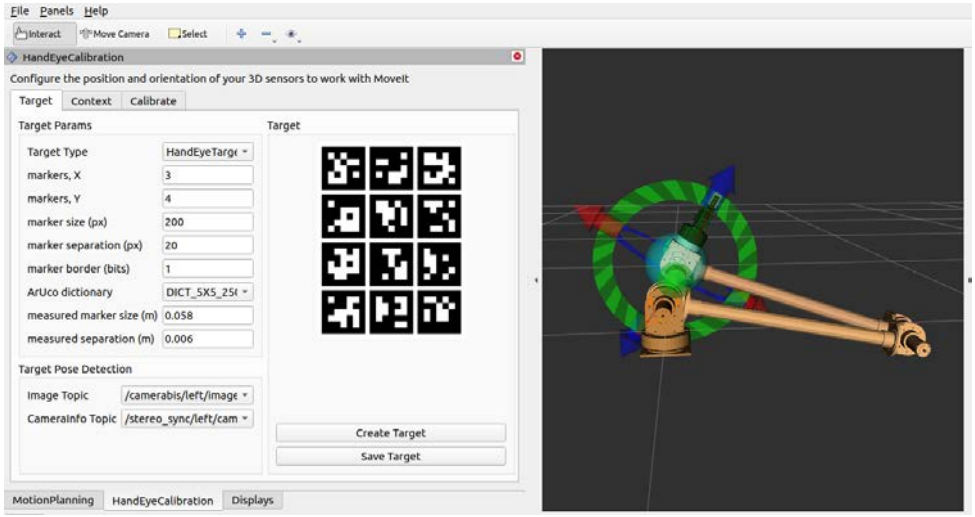


Figura 5.13: *Plug-in EyeCalibration* disponibile per Rviz [46]

Tramite il *plug in* è stato possibile specificare i parametri del target, associare ogni *topic* alla propria funzione e raccogliere i *sample* per effettuare la calibrazione. Come mostrato dalla Fig. 5.13, la prima finestra è dedicata alla definizione dei parametri relativi al target:

- Target Type: tipo di target che si vuole utilizzare
- Markers X: numero di QR code per fila
- Markers Y: numero di QR code per colonna
- Markers size: dimensione del QR code espressa in pixel
- Markers separation: distanza in pixel tra un bordo del QR code e il successivo
- Markers border: numero di bit (quadrati elementari all'interno del QR code) usati come bordo
- Measured marker size: dimensione del lato del QR code utilizzato espresso in metri

- Measured marker size: distanza tra un bordo e il successivo espressa in metri

Grazie alla conoscenza dei parametri, il software è in grado di costruire un *Target* che rispetti le caratteristiche impostate e tramite il pulsante *Save Target* restituisce un file immagine (estensione *png*) che lo contiene. Infine è necessarrio impostare i topic per la determinazione della posa del target: *image topic* e *camera info topic*; entrambi relativi al sistema di visione.

Tramite la seconda finestra (Fig. 5.18) si determina il tipo di calibrazione che si è scelto di eseguire, eye-on-hand oppure eye-to-hand, e i nomi dei sistemi di riferimento utilizzati. Inoltre è possibile dare una stima iniziale della matrice X.

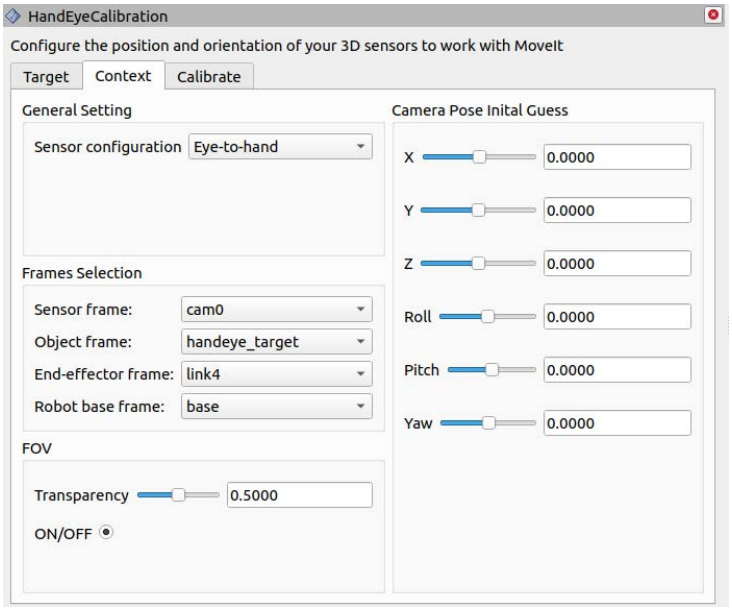


Figura 5.14: Finestra *Calibrate* del Plug-in *EyeCalibration* disponibile per Rviz [46]

Nell'ultima finestra si specifica il tipo si risolutore [47] che si desidera utilizzare per l'Eq. (5.5), il planning group e si esegue la raccolta dei campioni per la calibrazione. Facendo attenzione che il tag sia all'interno del campo di vista della camera, è sufficiente utilizzare il pulsante **Take sample** e l'algoritmo memorizza lo stato dei giunti e la posizione relativa tra camera e tag. Per proseguire con la raccolta dati è necessario muovere il braccio ed acquisire nuovamente un campione.

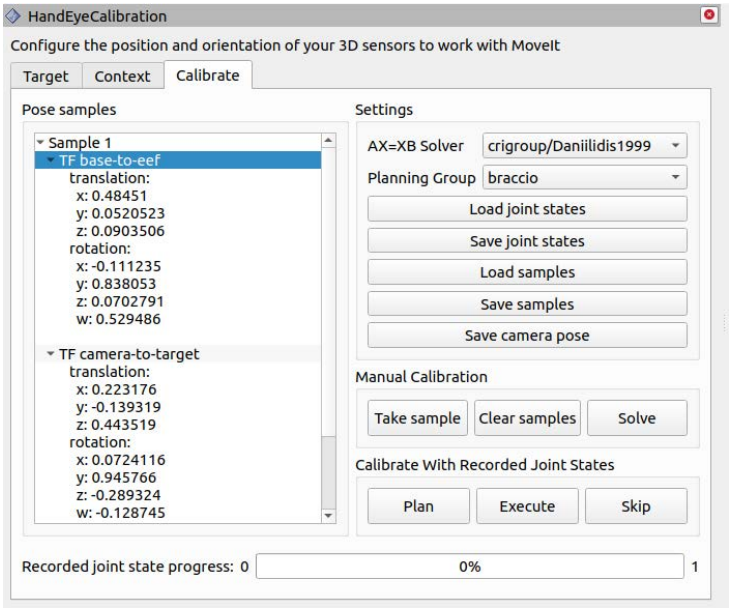


Figura 5.15: Finestra *Context* del Plug-in *EyeCalibration* disponibile per Rviz [46]

La documentazione scientifica fornisce alcuni suggerimenti per massimizzare la precisione dell'algoritmo:

1. Eseguire una rotazione il più ampia possibile tra le pose
2. Minimizzare la distanza dal target e la camera
3. Minimizzare la traslazione tra le varie pose
4. Usare pose ridondanti

Eseguiti i test numerose volte è possibile elaborare i risultati per determinare l'incertezza della calibrazione e l'intervallo di confidenza. Nella circostanza in cui il numero delle prove eseguite sia inferiore a trenta è consigliato usare la distribuzione *t di Student* (Fig. 5.16). Tramite la conoscenza dei dati, numero N e valore x_i, è possibile calcolarne la *media* $\overline{x}$ e la *deviazione standard corretta* (s):

$$\begin{aligned} \overline{x} &= \frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x}) \\ s &= \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2 \end{aligned} \tag{5.7}$$

Per determinare l'intervallo di confidenza è necessario definire un valore di *rischio di errore* (α) e noto il numero di *gradi di libertà del sistema* ($\nu = N - 1$) è possibile, tramite valori tabulari, estrapolare il valore della variabile t. L'intervallo risulta:

$$\overline{x} - \frac{ts}{\sqrt{N}} < \mu < \overline{x} + \frac{ts}{\sqrt{N}} \tag{5.8}$$

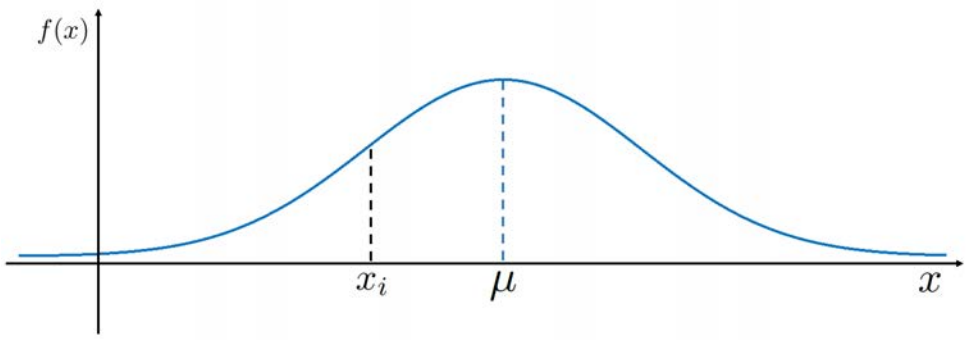


Figura 5.16: Distribuzione *t di Student*

Nota l'incertezza della calibrazione occhio mano è possibile determinare l'incertezza sul posizionamento dell'End Effector. La quale è necessaria per verificare che l'incertezza combinata tra i vari sistemi, stima della posa dell'oggetto e posizionamento End Effector, sia compatibile con la corsa del gripper.

Concludendo, combinando le informazioni ottenute da Sezione 5.1 e Sezione 5.2 si ottengono le coordinate del campione di roccia nel sistema di riferimento braccio.

$$[P_{braccio}] = [t] + [R][P_{camera}] \tag{5.9}$$

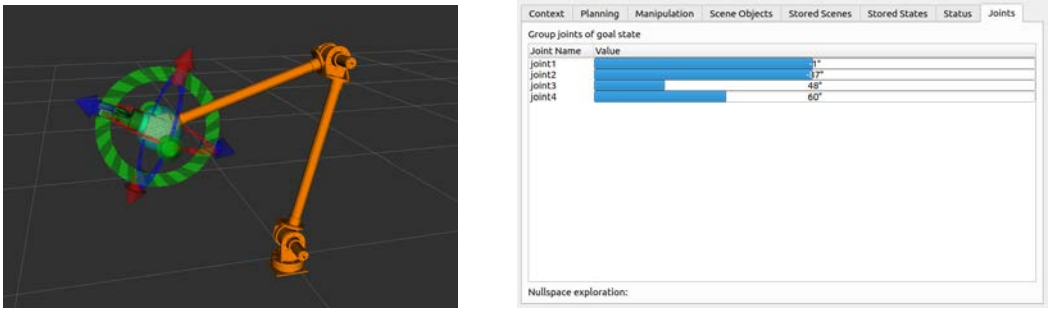
Dove $P_{braccio}$ e P_{camera} sono le coordinate del sasso rispetto alla base del manipolatore e alla camera mentre t ed R sono le trasformazioni, ottenute mediante la calibrazione occhio-mano, che legano i sistemi di riferimento

5.3 Raccolta e posa

Come ultimo passaggio per il completamento della task è richiesto di afferrare il sasso e posizionarlo in un determinato punto, presumibilmente un centro di raccolta campioni. Per rendere l'algoritmo il più generico possibile è stato scelto di dividere questa operazione in due traiettorie:

- Raccolta
- Posa

Per la progettazione delle traiettorie è stato scelto di usare *MoveIt*: un software in grado di eseguire la cinematica inversa del manipolatore (Sezione 3.4) e di progettare i profili di posizione, velocità e accelerazione che i giunti devono rispettare affinché l'*end effector* raggiunga la posizione obiettivo. Tramite MoveIt (e la GUI di Rviz) è possibile muovere il braccio sia controllando la posizione finale dell'*end effector* sia muovendo i singoli giunti.



(a) Controllo posizione finale EE **(b)** Controllo spostamento relativo tra i giunti

Figura 5.17: Controllo dei giunti

Nell'ottica di rendere il rover autonomo in queste operazioni è stato sviluppato, dai componenti del team, un pacchetto denominato *pose*. Grazie ad esso, l'operatore può inserire le coordinate obiettivo dell'*end effector* all'interno di un file *launch*. Il software è in grado di pianificare la traiettoria e di comunicarla ai motori che la eseguiranno. In futuro si metterà in comunicazione diretta il sistema di identificazione oggetti (Sezione 5.1.4) con il sistema braccio tramite un topic dedicato in modo da escludere completamente la presenza dell'operatore.

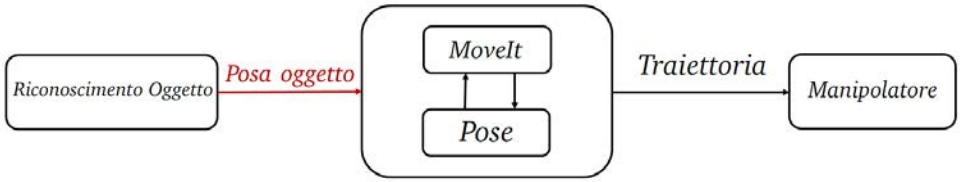


Figura 5.18: Progetto lavoro futuro

In fase di implementazione la presenza di una GUI rimane fondamentale in quanto restituisce all'operatore un feedback visivo sulla traiettoria che verrà eseguita scongiurando così possibili collisioni con l'ambiente di lavoro (strumenti e/o apparecchiature).

5.3.1 Raccolta

La traiettoria per la raccolta è stata suddivisa in tre ulteriori fasi in modo da ridurre la corrente istantanea richiesta al sistema di alimentazione. Durante la fase di avvicinamento l'*end effector* viene posizionato in una situazione intermedia tra la posizione di riposo e la posizione obiettivo.

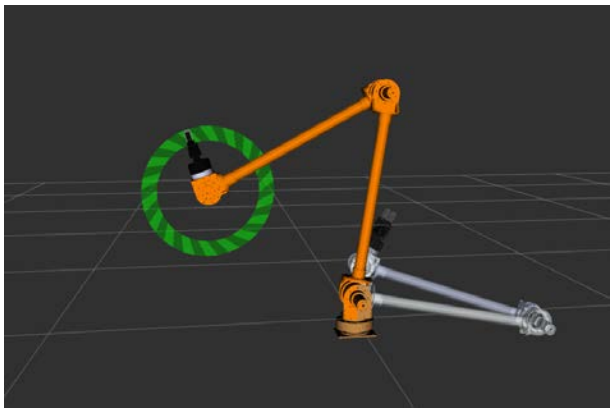


Figura 5.19: Fase di avvicinamento

Successivamente viene azionato il giunto quattro, relativo all'*end effector*, con il compito di posizionare il *gripper* perpendicolare al piano su cui si trova il sasso.

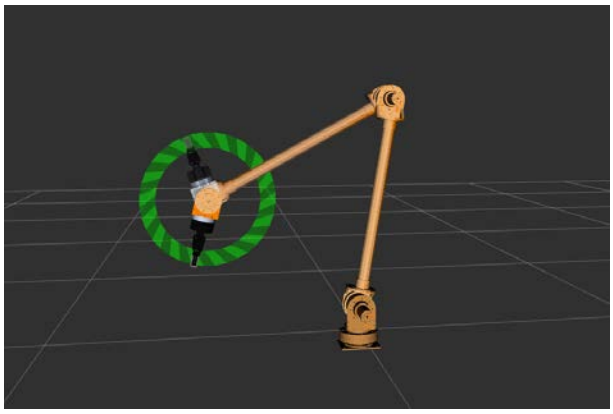


Figura 5.20: Allineamento End Effector

L'Ultimo step consiste nel raggiungimento delle coordinate del campione di roccia per effettuare la raccolta.

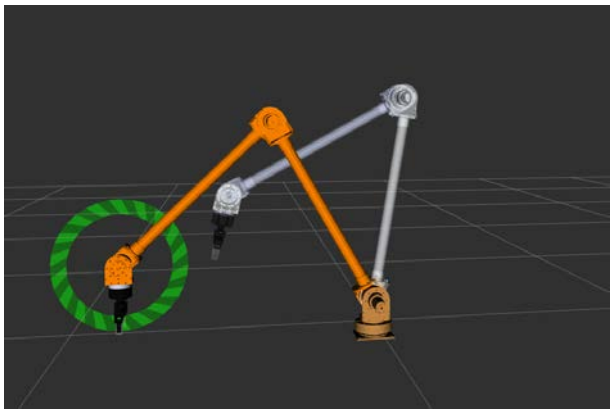


Figura 5.21: Raggiungimento obiettivo

5.3.2 Gripper

Il *gripper* (Sez. 4.2.1) viene comandato attraverso l'interfaccia grafica *rqt* (Sezione 4.3.2) che permette di modificare in tempo reale i parametri dell'End Effector.

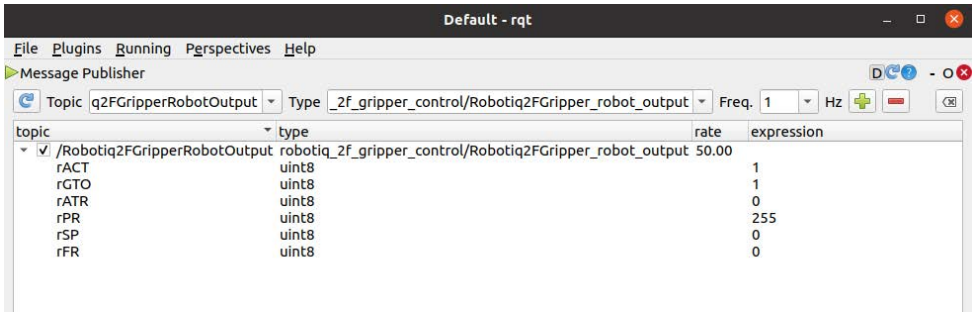
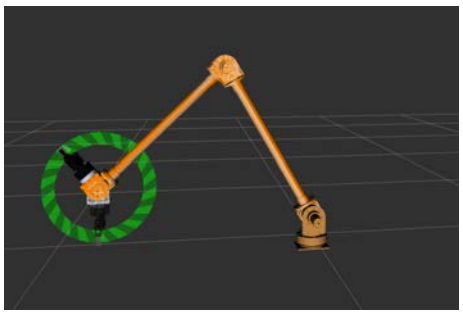


Figura 5.22: GUI rqt controllo gripper

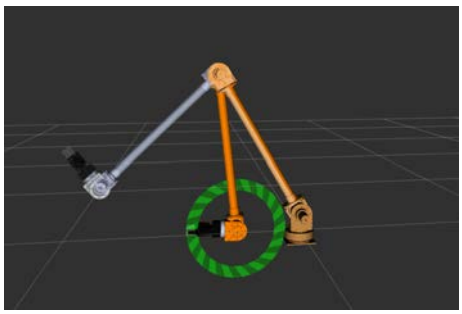
L'interfaccia grafica permette di controllarne la posizione inserendo un parametro numerico compreso tra 0 e 255 dove 0 significa totalmente aperto mentre 255 totalmente chiuso. La velocità di esecuzione del movimento è anch'essa modificabile tramite un parametro avente la stessa scala cioè 0 velocità minima e 255 velocità massima. Inoltre, integrato nel gripper è presente un sensore di forza utilizzato come feedback per un controllo in forza. Perciò quando il sensore misura un valore superiore al limite imposto dall'operatore, sempre con scala da 0 a 255, l'avanzamento viene interrotto.

5.3.3 Posa

Dopo aver afferrato il campione di roccia lo si vuole posizionare in un centro di raccolta. Data la posizione in cui si trova il braccio, il momento alla base è notevole perciò è consigliato ridurlo eseguendo una manovra ad hoc: si solleva il sasso muovendo il giunto quattro e si ritrae il link due muovendo esclusivamente il giunto tre.



(a) Rotazione giunto End Effector quattro



(b) Rotazione giunto tre

Figura 5.23: Rotazione dell'*end effector*

Successivamente si procede con la posa effettiva del campione nel punto prestabilito.

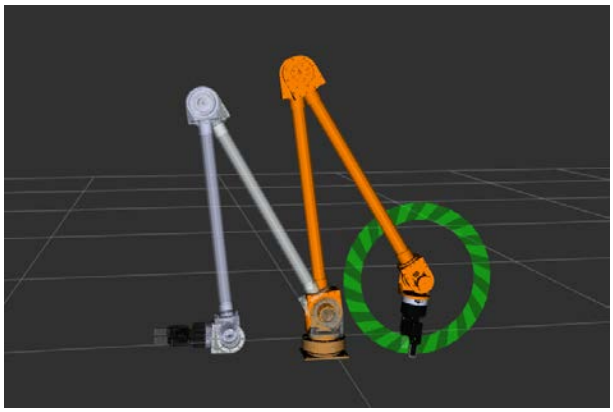


Figura 5.24: Posa campione roccioso in centro di raccolta

Capitolo 6

Risultati

In questo capitolo verranno presentati tutti i risultati ottenuti in laboratorio durante lo svolgimento del progetto di tesi. Verranno ripercorsi i passaggi fondamentali del capitolo precedente mostrando cosa è stato concretamente ottenuto.

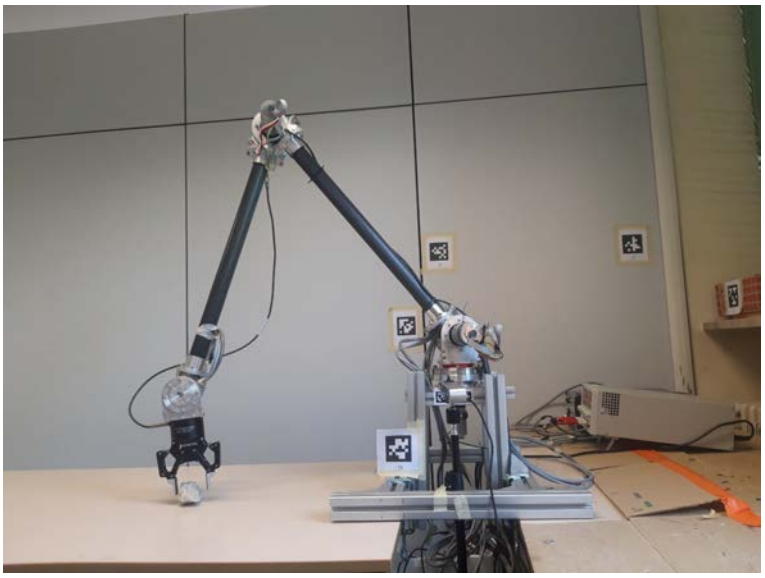


Figura 6.1: Esempio di raccolta di un campione roccioso

6.1 Risultati riconoscimento oggetto

Il riconoscimento oggetto si è basato sull'utilizzo della stereo camera Zed e, come mostrato nella Sezione 5.1.1, l'immagine stereo viene acquisita dal nodo *usb cam* e divisa dal nodo *splitter*.

(a) Immagine Sinistra

(b) Immagine Destra

Figura 6.2: Coppia immagine stereo in output dal nodo *splitter*

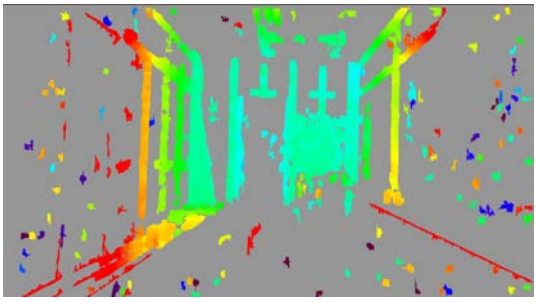
Per eseguire il processo di rettifica della coppia di immagini ed eliminare le distorsioni è stato eseguito il processo di **calibrazione stereo** il quale ha fornito i parametri intrinseci ed estrinseci della stero camera ZED (presenti in appendice). La calibrazione è stata eseguita implementando il metodo di Zhang [14] utilizzando una scacchiera avente 9×7 quadrati di dimensioni $35\text{mm} \times 35\text{mm}$ posta su un foglio A3 (Sezione 2.4). Grazie ai dati forniti dalla calibrazione è stato possibile eseguire la rettifica e l'eliminazione delle distorsioni:

(a) Immagine Sinistra rettificato e non distorta

(b) Immagine Destra rettificata e non distorta

Figura 6.3: Rettifica e correzione immagini

Inoltre, come spiegato nella Sezione 5.1.3, il package *stero image proc* fornisce la *disparity map* (e la relativa *point cloud*):



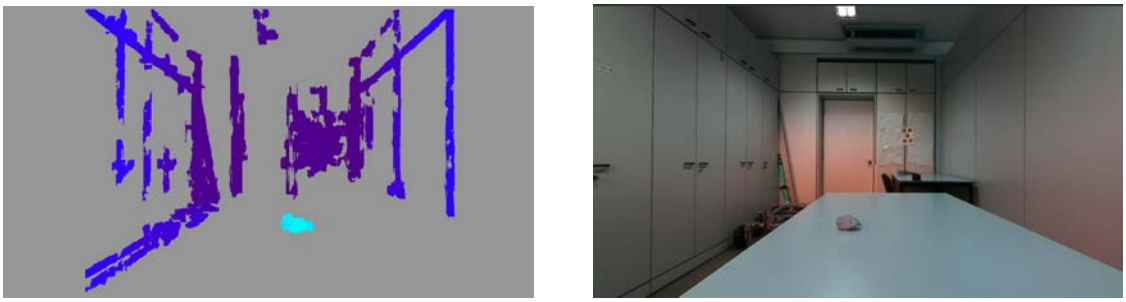
(a) *Disparity map*

(b) Immagine Destra rettificata e non distorta

Figura 6.4: *Disparity map* ottenuta da una coppia di immagini rettificate non distorte

Effettuando la riconfigurazione dinamica dei parametri è stato possibile migliorarne il risultato (Sezione 5.1.3). Data la posizione ravvicinata del sasso è stato scelto di aumentare il parametro *disparity range* e per eliminare le regioni troppo piccole, causate probabilmente dal rumore presente nell'immagine, si è agito sul *speckle size*. Il valore dei parametri modificati utilizzati:

- *disparity range*: 190
- *speckle size*: 1000



(a) *Disparity map* con parametri riconfigurati

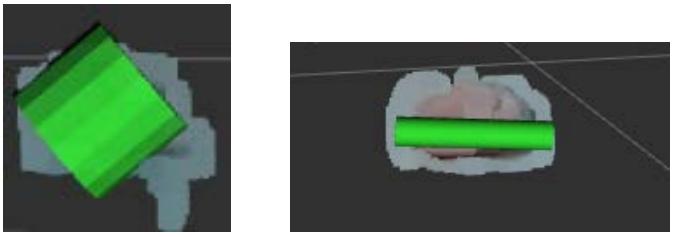
(b) Immagine Destra rettificata e non distorta

Figura 6.5: *Disparity map* ottenuta con parametri riconfigurati

Come illustrato nella Sezione 5.1.4, per la ricerca dell'oggetto viene applicato un filtro alla *point cloud* escludendo tutti i punti esterni dalla regione di interesse. Nel caso specifico è stata considerata rilevante unicamente la zona in prossimità della stereo-camera e raggiungibile dal braccio. Perciò sono stati applicati dei filtri lungo ogni asse (rispetto al sistema camera di Fig. 5.7):

- Lungo asse Z da 0m a 1.7m
- Lungo asse X da -0.3m a 1m
- Lungo asse Y da -0.5m a 0.5m

L'algoritmo di riconoscimento oggetto è stato in grado di identificare sempre il sasso paragonandolo a un cilindro anche se le dimensioni di quest'ultimo non rispecchiavano sempre quelle del campione di roccia. In base alle condizioni di luminosità il cilindro poteva essere delle dimensioni corrette o inferiori. Un' altra fonte di errore è stato il tavolo che riflettendo l'immagine del sasso induceva il software in inganno.



(a) Riconoscimento oggetto accurato

(b) Riconoscimento oggetto impreciso

Figura 6.6: Riconoscimento oggetto in diversi casi di accuratezza

Le coordinate ottenute del cilindro tramite l'algoritmo variano a seconda della posa del sasso. Ne viene riportato un esempio per confrontare la misura con quella effettuata da un misuratore laser.
L'algoritmo fornisce la distanza dalla camera al baricentro dell'oggetto mentre la misurazione laser fornisce la distanza dalla camera al punto di proiezione del baricentro sulla superficie. Quindi è stato necessario elaborare algebricamente i dati per metterli a confronto:

$$Z_{superficie} = Z_{baricentro} - r \tag{6.1}$$

Dove r è la dimensione del raggio del cilindro. Le coordinate X e Y non subiscono variazioni.

Tabella 6.1: Risultati riconoscimento oggetto

	Riconoscimento oggetto [m]	Misuratore laser [m]
Z	0.48 $\pm$ 0.02	0.525 $\pm$ 0.005
X	0.05 $\pm$ 0.02	0.045 $\pm$ 0.005
Y	0.20 $\pm$ 0.02	0.200 $\pm$ 0.005

Grazie alla Tabella 6.1 è possibile osservare che le misure effettuate lungo gli assi X e Y sono perfettamente compatibili mentre le misure lungo l'asse Z presentano un leggero scostamento.
L'errore relativo all'algoritmo di riconoscimento oggetto è stato ottenuto mediante Eq. (5.1) considerando i seguenti parametri:

- $Risoluzione = 2208 \times 1242$ px
- $a = 0.01805$
- $b = 0.1746$
- $Z = 1$ m

6.2 Calibrazione occhio mano

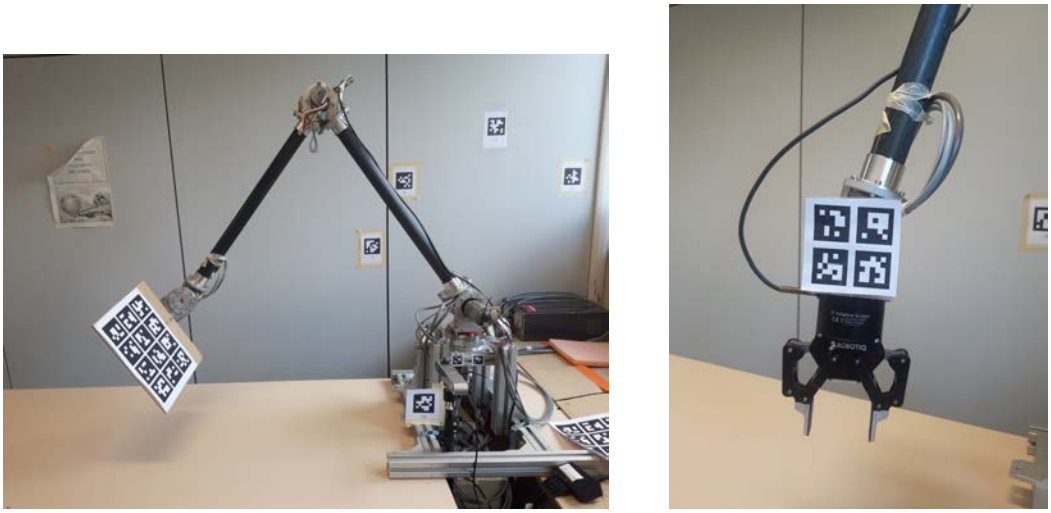
Per determinare la matrice di rototraslazione che lega il sistema di visione con il sistema base del braccio robotico si è usato il processo di calibrazione occhio mano. Data la morfologia attuale dell'hardware del rover si è scelto di utilizzare la calibrazione *eye on base*. Sono state utilizzate due tipologie di *tag* per confrontarne i risultati:

Tabella 6.2: Caratteristiche tag utilizzati

	N° QR code	Lato QR code [mm]	Distanza [mm]
Primo *tag*	12	58 $\pm$ 1	6$\pm$ 1
Secondo *tag*	4	35$\pm$ 1	6$\pm$ 1

Il primo *tag* è stato utilizzato per testare l'algoritmo e durante la calibrazione sostituiva il *gripper* date le sue dimensioni ingombranti. Nell'ottica di eseguire la calibrazione ogni qual volta fosse necessario, senza

dover sostituire l'*end effector*, si è deciso di effettuare una seconda calibrazione con un *tag* notevolmente più piccolo in modo che possa rimanere montato sul robot.

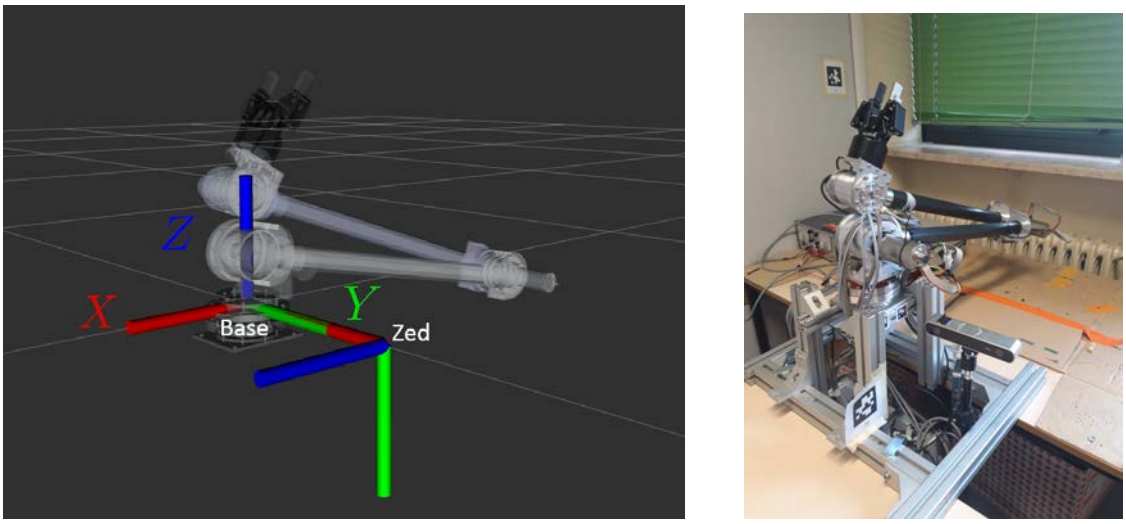


(a) Tag sul pannello

(b) Tag sul giunto

Figura 6.7: Diverse pose del tag

In totale sono state eseguite 18 calibrazioni: 10 con il primo *tag* e 8 con il secondo. Per ogni test sono stati raccolti 20 *samples* superando così il numero minimo raccomandato: tra le 10 e le 12. Ogni calibrazione ha fornito la matrice di rototraslazione che lega il sistema di visione alla base del braccio.



(a) Sistema occhio mano e relativi sistemi di riferimento. Asse Z colore blu; asse X colore rosso; asse Y colore verde

(b) Foto del sistema reale

Figura 6.8: Confronto tra sistema reale e sistema simulato

Tabella 6.3: Risultati calibrazione occhio mano con il primo tag

Test	Traslazione X [m]	Traslazione Y [m]	Traslazione Z [m]	Roll [rad]	Pitch [rad]	Yaw[rad]
1°	-0.01	0.42	0.06	-1.61	0.16	-1.52
2°	-0.03	0.40	0.07	-1.57	0.17	-1.52
3°	-0.01	0.39	0.07	-1.57	0.15	-1.53
4°	-0.02	0.41	0.07	-1.62	0.16	-1.52
5°	-0.02	0.35	0.04	-1.53	0.09	-1.49
6°	-0.01	0.35	0.05	-1.51	0.09	-1.50
7°	-0.04	0.34	0.05	-1.47	0.10	-1.49
8°	-0.01	0.31	0.04	-1.37	0.07	-1.51
9°	0.02	0.40	0.07	-1.64	0.16	-1.54
10°	0.00	0.44	0.04	-1.64	0.11	-1.49

Tabella 6.4: Risultati calibrazione occhio mano con il secondo tag

Test	Traslazione X [m]	Traslazione Y [m]	Traslazione Z [m]	Roll [rad]	Pitch [rad]	Yaw[rad]
1°	-0.06	0.40	0.05	-1.51	0.06	-1.52
2°	-0.06	0.36	0.05	-1.50	0.02	-1.52
3°	-0.01	0.60	0.13	-1.76	0.23	-1.64
4°	-0.01	0.52	0.10	-1.75	0.16	-1.57
5°	-0.01	0.50	0.07	-1.75	0.13	-1.56
6°	0	0.49	0.10	-1.71	0.14	-1.57
7°	0	0.43	0.10	-1.68	0.12	-1.57
8°	-0.01	0.50	0.10	-1.65	0.15	-1.56

Dato il numero di test ($N_1 = 10$ e $N_2 = 8$), è stata usata la distribuzione *t di Student* (Fig. 5.16) con un livello di confidenza pari al 99.7%. Per il calcolo dell'intervallo di confidenza si è usata l'Eq. (5.8):

Tabella 6.5: Intervalli di confidenza prima prova (tag1)

	Media ($\overline{x_i}$)	Deviazione (s)	Limite inferiore	Limite superiore
Traslazione X [m]	-0.02	0.02	-0.04	0.00
Traslazione Y [m]	0.38	0.04	0.33	0.43
Traslazione Z [m]	0.05	0.01	0.03	0.07
Roll [rad]	-1.55	0.05	-1.50	-1.61
Pitch [rad]	0.12	0.04	0.08	0.17
Yaw [rad]	-1.51	0.02	-1.53	-1.48

Tabella 6.6: Intervalli di confidenza seconda prova (tag2)

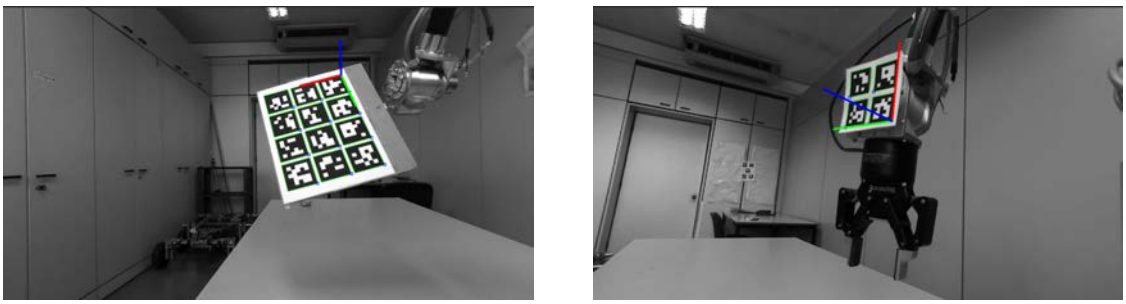
	Media ($\overline{x_i}$)	Deviazione (s)	Limite inferiore	Limite superiore
Traslazione X [m]	-0.02	0.03	-0.05	0.01
Traslazione Y [m]	0.48	0.07	0.38	0.57
Traslazione Z [m]	0.09	0.03	0.05	0.12
Roll [rad]	-1.66	0.06	-1.75	-1.57
Pitch [rad]	0.13	0.06	0.04	0.21
Yaw [rad]	-1.56	0.04	-1.56	-1.51

Come strumento di controllo dei risultati è stato utilizzato un misuratore laser che ha fornito i seguenti dati di traslazione tra sistema di riferimento camera e sistema base:

Tabella 6.7: Parametri fisici

	Misura laser [m]
Traslazione X	0.000 ± 0.005
Traslazione Y	0.031 ± 0.005
Traslazione Z	0.001 ± 0.005

La sicurezza che il *tag* rimanesse all'interno del campo di vista della camera durante l'operazione di acquisizione dati è stata garantita da Rviz, mostrando l'immagine con sovrapposto il *tag* (se rilevato)



(a) Rilevamento tag1 **(b)** Rilevamento tag2

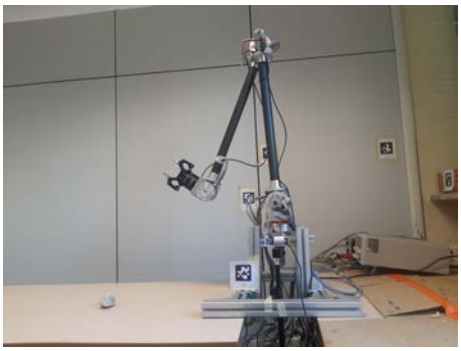
Figura 6.9: Rilevamento tag

6.3 Presa e posa del campione roccioso

Tramite l'Eq. (5.9) è possibile calcolare la posa del campione rispetto al sistema di riferimento base del manipolatore. Successivamente si procede con il posizionamento dell'*end effector* in prossimità del sasso. In accordo con la fase di implementazione vengono eseguite tre manovre:

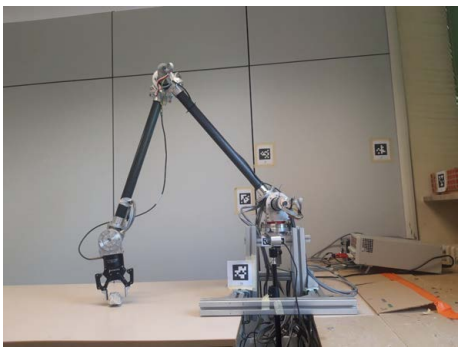
- Avvicinamento
- Orientamento dell'End Effector
- Posizionamento esatto

Tramite *MoveIt* e il package *Pose* si sono ottenuti i seguenti risultati:



(a) Avvicinamento

(b) Orientamento dell'End Effector



(c) Avvicinamento al campione

(d) Ingrandimento dell'avvicinamento al campione

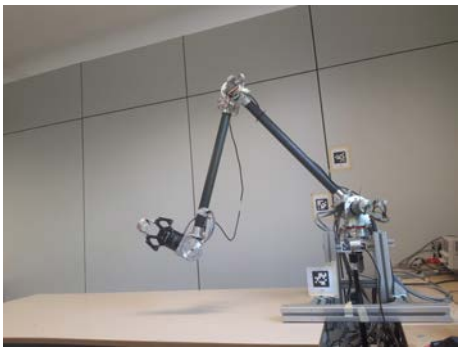
Figura 6.10: Avvicinamento al campione roccioso

Si procede con la chiusura della pinza per afferrare il sasso. Utilizzando la GUI di rqt (Fig. 5.22) è stato possibile controllare l'*end effector* mediante i seguenti parametri:

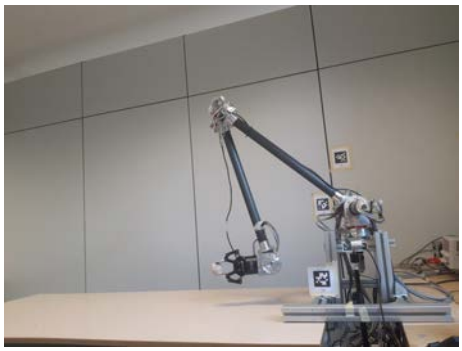
- rPR: (posizione richiesta al *gripper*) 255 cioè totalmente chiuso
- rSP: velocità di esecuzione 10 velocità moderata
- rFR: 255 soglia massima per il controllo di forza così da assicurarsi una presa forte e decisa sul sasso

Figura 6.11: Presa del campione roccioso

Infine si procede con la posa del campione di roccia nel punto prestabilito. Per minimizzare la coppia alla base sono state eseguite due rotazioni distinte dei giunti quattro e tre (Sezione 5.3.3).



(a) Rotazione giunto 4



(b) Rotazione giunto 3

Figura 6.12: Rotazione dei giunti

L'ultima traiettoria eseguita porta il campione di roccia nel centro di raccolta

Figura 6.13: Posa del campione in un ipotetico centro di raccolta

Capitolo 7

Conclusioni

In quest'ultimo capitolo vengono analizzati i risultati ottenuti evidenziandone i punti di forza e di debolezza affiancandoli a possibili soluzioni. La presentazione si sviluppa in tre paragrafi che valutano rispettivamente:

- Riconoscimento oggetto
- Calibrazione occhio mano
- Presa e posa del campione roccioso

Complessivamente ogni sezione ha portato risultati soddisfacenti, raggiungendo così l'obiettivo predisposto cioè il riconoscimento del campione roccioso tramite stereo camera e relativa raccolta mediante braccio robotico. I limiti dell'algoritmo implementato, riscontrati durante la realizzazione del progetto di tesi, sono stati utili per individuare possibili sviluppi futuri.

Concludendo, il lavoro in team mi ha offerto la possibilità di confrontare le mie idee con quelle degli altri membri del progetto stimolando la maturazione di soluzioni ai problemi incontrati. Inoltre, il progetto Morpheus mi ha permesso di applicare le conoscenze, acquisite durante il mio percorso di studi, in maniera concreta realizzando così questo lavoro di tesi.

7.1 Valutazioni riconoscimento oggetto

Il sistema di **visione** (Fig. 5.3) ha permesso l'acquisizione delle immagini stereo aventi un frame rate e la risoluzione elevata (2K=2208x1242pixel). Risultato non banale in quanto la camera ZED nasce per lavorare con hardware specifico che generalmente non è presente nei PC dei membri del team (Sezione 4.1). Per questo è stato necessario sviluppare un sistema ad hoc adottando una serie di workaround (nodo splitter e synch Fig. 5.3) che avrebbero potuto ridurre la qualità delle informazioni ottenute. Ora, si è in grado di utilizzare la camera ZED su qualsiasi dispositivo laptop dotato del sistema operativo ROS.

Il processo di **calibrazione** ha permesso di ottenere i valori dei parametri intrinseci ed estrinseci della stereo camera, grazie ai quali è stato possibile rimuovere le distorsioni ed effettuare la rettifica delle

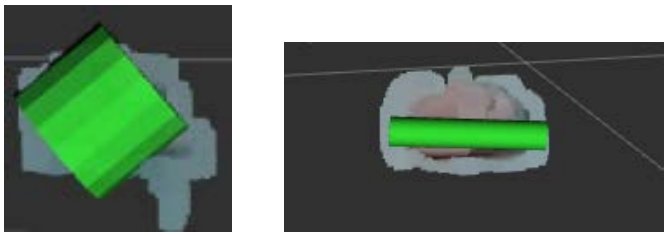
immagini stereo. L'utilizzo di ROS e dei relativi package ha permesso di eseguire una procedura semplice, efficace e accurata sebbene le dimensioni ridotte della scacchiera utilizzata (foglio A3).

(a) immagine distorta **(b)** immagine non distorta

Figura 7.1: Risultati della calibrazione

L'**elaborazione della point cloud** ha riscontrato problemi dovuti alle condizioni al contorno: la triangolazione dei punti avveniva in maniere sempre più accurata con l'aumentare della luminosità della scena , viceversa le zone in ombra difficilmente erano presenti nella disparity map. Inoltre, la presenza di oggetti nelle immediate vicinanze del camera veniva ignorata: per distanze inferiori ai trenta centimetri il campione roccioso non rientrava mai nella disparity map. La riconfigurazione dinamica dei parametri relativi al matching features ha permesso di mitigare gli errori (Sezione 5.1.3), ma la necessità continua di aggiornarli dovuta alla variazione di luminosità presente in laboratorio durante l'arco della giornata ha esplicitato la necessità, in futuro, di creare un dataset contenente i parametri in funzione della scena osservata. Il rover, in maniera autonoma o con l'aiuto dell'operatore, potrà così decidere quale set di dati utilizzare in base all'ambiente circostante. Per migliorare ulteriormente il risultato si potrà elaborare un altro dataset contenente i parametri della camera: ISO, tempo di esposizione e apertura del diaframma (Sezione 2.1.3) sempre in funzione della scena osservata.

A cascata, anche l'algoritmo di **ricerca oggetto** risente delle condizioni al contorno a cui è sottoposto il sistema ma, nonostante ciò, ha fornito risultati eccellenti riuscendo sempre a identificare il campione roccioso e a restituirne la posa del baricentro (Fig. 6.6) . L'utilizzo della zona di interesse non è estendibile, in generale, a tutte le applicazioni dei rover, ma in questo progetto di tesi è stato sfruttato per diminuire notevolmente il costo computazionale e quindi il tempo di elaborazione, passando da decine di secondi a pochi secondi, e la richiesta di memoria RAM, tuttavia alta (alcuni gigabyte). In aggiunta si è osservato che l'ambiente strutturato del laboratorio risulta più svantaggioso rispetto ad un ambiente caotico: un tavolo lucido presenta dei riflessi che possono portare a falsi matching e sono difficilmente eliminabili via software, questo problema non si pone su un terreno granulare.



(a) Riconoscimento oggetto accurato

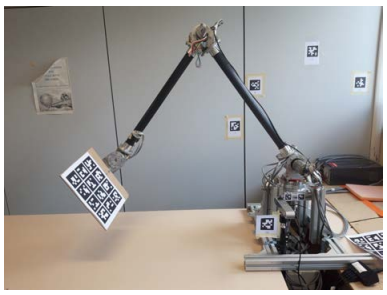
(b) Riconoscimento oggetto impreciso

Figura 7.2: Confronto tra un riconoscimento impreciso dell'oggetto e uno preciso

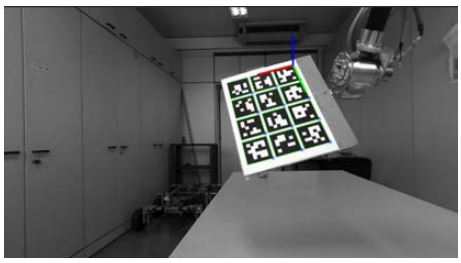
Si tiene a precisare che la difficoltà nel determinare, in maniera corretta, le dimensioni del cilindro associato al sasso (Fig. 7.2b) non risiede nell'algoritmo di riconoscimento oggetto, ma è dovuta alla non eccellente point cloud fornitagli in input. L'algortimo utilizzato per la ricerca oggetti, inoltre, presenta un livello di intelligenza artificiale primordiale, lo stato dell'arte mostra tecniche molto avanzate che impiegano reti neurali garantendo risultati affidabili e precisi, ovviamente a discapito di potenza di calcolo. L'utilizzo di questo tipo di tecnologia permetterebbe al rover di riconoscere un oggetto specifico in mezzo a tanti discriminandolo non solo per la forma, ma anche per il colore.

7.2 Valutazioni calibrazione occhio mano

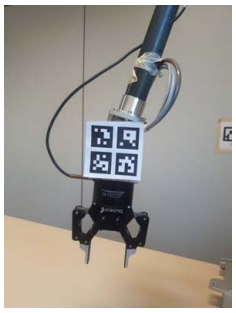
Tramite l'esecuzione della calibrazione occhio mano in modalità *eye in hand* è stato possibile calcolare la matrice di rototraslazione che lega il sistema di visione con il sistema braccio robotico. Sono state eseguite due tipologie di test, ognuna delle quali sfruttava un tag diverso (Fig. 7.3).



(a) Calibrazione occhio mano tramite tag1



(b) Riconoscimento tag1



(c) Calibrazione occhio mano tramite tag2

(d) Riconoscimento tag2

Figura 7.3: Calibrazione occhio mano e riconoscimento *tag*

Per ogni *tag* sono state eseguite 10 prove di calibrazione (Tabella 6.4) ognuna delle quali contava al proprio interno 20 *sample* cioè 20 coppie di posizioni relative tra i sistemi camera e *tag* e i sistemi base ed *end effector*. Infine è stata eseguita la misura di traslazione tra i sistemi di riferimento del rover tramite un misuratore laser per poter effettuare un confronto tra l'algoritmo e realtà.

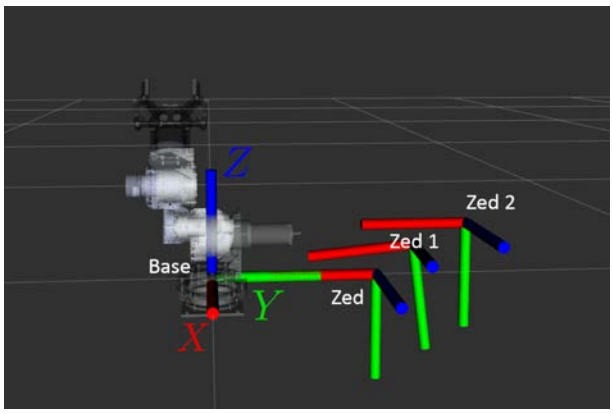


Figura 7.4: Rappresentazione della media dei risultati ottenuti tramite l'utilizzo dei 2 tag e dalla misurazione laser

Purtroppo, come mostrato qualitativamente dalla Fig. 7.4, e quantitativamente dalla Tabella 6.4, i test effettuati non hanno fornito un risultato direttamente utilizzabile: il discostamento tra la misura calcolata e la misura reale è troppo elevato. Però, studiando i valori ottenuti durante le varie prove si è notato come i risultati siano all'interno di un intervallo di confidenza sufficientemente stretto (Tabella 6.6) per poter affermare che l'errore sia sistematico e costante. Ciò implica che, attraverso un'elaborazione numerica in *post processing*, sia possibile correggere l'errore della calibrazione occhio mano. L'inesattezza dei risultati deriva unicamente dai dati relativi all'asse Y infatti, in generale, i risultati della calibrazione sono buoni: le rotazioni Roll, Pitch e Yaw sono ragionevolmente prossime alla realtà come le traslazioni lungo l'asse X e Z (Fig. 7.4), l'errore maggiore ($\sim$ 10 cm) si presenta sulla traslazione lungo l'asse Y. La situazione appena descritta può essere frutto di diverse problematiche:

- La cinematica del braccio robotico non consente la rotazione dell'End Effector rispetto all'asse di X
- Il giunto alla base del braccio, che permette la rotazione attorno all'asse Z, presenta un gioco di diversi gradi
- Gli assi del sistema di riferimento camera e base sono allineati

Per ogni problematica è stata individuata una possibile soluzione che in futuro potrà essere adoperata:

- Introduzione di un nuovo grado di libertà che permetta all'End Effector di eseguire la manovra di rollio
- Ripristino dei componenti meccanici che si occupano della trasmissione del moto
- Scelta di una posizione relativa tra *camera* e *braccio* diversa

Risulta interessante confrontare i risultati ottenuti utilizzando due scacchiere diverse di tag. La scacchiera di dimensioni maggiori è stata utilizzata anche per testare i software di calibrazione occhio mano

oltre che per poter unire le informazione trovate dal sistema di ricerca oggetto con le informazioni richieste dal sistema di raccolta. Le sue dimensioni eccessive (12 QR code distribuiti su un foglio A4) non la rendono un target idoneo per applicazioni ripetute. Infatti, è necessario smontare l'End Effector e sostituirlo con il pannello contenente i tag ogniqualvolta si voglia eseguire la calibrazioni. Per questo motivo si è scelto di testare anche un target di dimensioni ridotte in modo che possa essere essemblato insieme al braccio in maniera permanente. I risultati ottenuti dalle calibrazioni col secondo tag hanno rispecchiato le aspettative: le dimensioni minori rendono il sistema maggiormente integrabile ma le informazioni acquisite perdono di accuratezza. Oltre alla dimensioni anche il numero dei QR code è causa della diminuzione della precisione in quanto ogni code rappresenta un piano per cui deve passare (con una certa tolleranza) il piano del sistema di riferimento target, perciò maggiore è il numero di code maggiore è il numero di vincoli e migliore sarà l'identificazione.

Concludendo, grazie ai seguenti test è stato verificato che la calibrazione occhio mano, applicata al rover Morpheus, fornisce dei buoni risultati, ma che necessitano di elaborazione. In futuro, l'algoritmo di calibrazione occhio mano non sarà limitato al sistema di raccolta ma risolverà una problematica importante del rover: l'assenza di encoder assoluti nei motori del manipolatore. Dopo ogni accensione, il *braccio* eseguirà una traiettoria predefinita e tramite la calibrazione occhio mano sarà possibile identificare l'orientamento del sistema *base* rispetto al sistema *camera* (fisso al telaio del rover), permettendo così la trasformazione delle informazioni fornite dagli encoder da relative ad assolute.

7.3 Valutazioni pick and place

Comunicando al package *Pose* la posizione del campione roccioso rispetto alla base è stato possibile effettuare la raccolta seguendo il procedimento riportato nella Sezione 6.3. I risultati sono stati soddisfacenti perché si è stati in grado di afferrare il campione roccioso e di posizionarlo nel centro di raccolta.

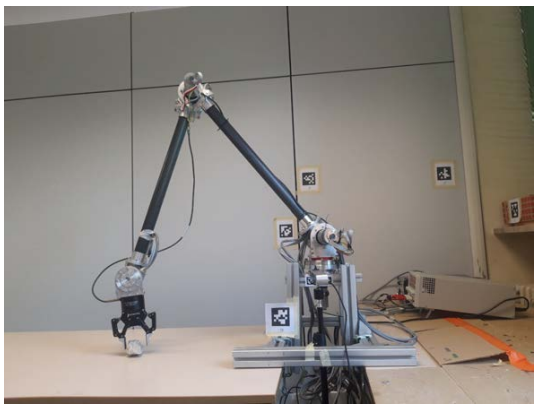


Figura 7.5: Raccolta campione roccioso tramite braccio robotico

Tuttavia, la presenza dei giochi meccanici all'interno dei giunti del manipolatore portavano spesso l'End Effector a non essere perfettamente allineato con l'obiettivo e, in rari casi, hanno determinato l'insuccesso della prova. Per ovviare al problema si sta studiando una soluzione che si avvalga della stero camera ZED e dei tag posti sull'End Effector necessari per la calibrazione occhio mano. Data la conoscenza

della posizione del campione roccioso e del tag rispetto al sistema di riferimento camera, è possibile identificare la distanza relativa tra oggetto e End Effector. Pianificando una traiettoria che minimizzi la distanza appena citata si è in grado di compensare sia gli errori dovuti al gioco dei giunti sia quelli relativi al riconoscimento dell'oggetto.

Le valutazioni relative al **gripper** sono eccellenti. Il suo utilizzo mediante GUI è stato semplice ed efficace. La presenza del controllo in forza ha reso superflua la conoscenza delle dimensioni reali del campione roccioso in quanto la chiusura della pinza veniva arrestata in seguito al riconoscimento oggetto, rendendo più semplice la raccolta dei campioni di roccia.

Bibliografia

[1] D. Paganini, "MORPHEUS: ideazione e progettazione del rover di Ateneo Relatore:" tesi di laurea mag., Dip. Ing. Industriale, Università di Padova, 2017.

[2] E. Morellato, "Algoritmi e software di controllo per la trazione del rover "Morpheus"," tesi di laurea mag., Dip. Ing. Industriale, Università di Padova, 2016.

[3] M. Reschiglian, "Sviluppo e implementazione di una interfaccia grafica in realtà aumentata per il controllo semplificato di sistemi robotici da remoto o in ambiente ostile," tesi di laurea mag., Dip. Ing. Industriale, Università di Padova, 2018.

[4] G. Franchini, "Progettazione di un braccio robotico per applicazioni spaziali integrato sul rover di ateneo Morpheus," tesi di laurea mag., Dip. Ing. Industriale, Università di Padova, 2021.

[5] *Esempio di androide*. indirizzo: https://www.ecodibergamo.it/stories/la-salute/robot-umanoide-contro-lautismo_1034831_11/.

[6] *Esempio di robot elettrodomestico*. indirizzo: https://www.ikohs.com/it/.

[7] *Esempio di veicolo autonomo*. indirizzo: https://www.mbda-systems.com/press-releases/mbda-milrem-robotics-develop-anti-tank-unmanned-ground-vehicle/.

[8] *ExoMars Rosalind Franklin, Rover ESA*. indirizzo: https://www.esa.int/ESA_Multimedia/Videos/2020/03/ExoMars_Rosalind_Franklin_rover_3601.

[9] J. Weng, P. Cohen e M. Herniou, "Camera Calibration with Distortion Models and Accuracy Evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, n. 10, 1992.

[10] P. Zanuttigh, C. D. Mutto, L. Minto, G. Marin, F. Dominio e G. M. Cortelazzo, *Time-of-flight and structured light depth cameras: Technology and applications*. Springer International Publishing Switzerland, 2016, pp. 1–355, ISBN: 9783319309736. DOI: 10.1007/978-3-319-30973-6.

[11] D. Scaramuzza e F. Fraundorfer, "Visual Odometry Part II: Matching, Robustness, Optimization, and Applications," rapp. tecn. 4, 2011, pp. 80–92. DOI: 10.1109/mra.2011.943233.

[12] S. Se, D. Lowe e J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *International Journal of Robotics Research*, vol. 21, n. 8, pp. 735–758, 2002, ISSN: 02783649. DOI: 10.1177/027836402761412467.

[13] H. Bay, T. Tuytelaars e L. Van Gool, "SURF: Speeded up robust features," 2006.

[14] Z. Zhang, "A Flexible New Technique for Camera Calibration," rapp. tecn., 1998.

[15] *Lente sottile*. indirizzo: https://it.wikipedia.org/wiki/Lente_sottile.

[16] *Diaframma ottico*. indirizzo: https://it.wikipedia.org/wiki/Diaframma_(ottica).

[17] *Profondità di campo*. indirizzo: https://it.wikipedia.org/wiki/Profondit%C3%A0_di_campo.

[18] *Posa scacchiere calibrazione*. indirizzo: http://mesh.brown.edu/3DP-2016/hw2/hw2.html.

[19] *Distorsione a cuscino e a barile*. indirizzo: https://ceeroprodsite.wordpress.com/2017/02/01/alle-origini-del-fisheye/.

[20] *ESA foto, Katwijk Beach Planetary Rover Dataset*. indirizzo: https://robotics.estec.esa.int/datasets/katwijk-beach-11-2015/.

[21] *La dinamica in robotica*. indirizzo: https://www.meccanismocomplesso.org/robot-dynamics-la-dinamica-in-robotica/.

[22] B. Siciliano, L. Sciavicco, L. Villani e G. Oriolo, *Robotics: Modeling, Planning, and Control*. London: Springer, 2009, ISBN: 9781846286414. DOI: 10.1109/MRA.2009.934833.

[23] StereoLabs, "ZED Camera and SDK Overview," rapp. tecn., 2018, p. 2.

[24] Maxon Motor, "Planetary Gearhead GP52," rapp. tecn. 223093, pp. 1–7.

[25] Maxon Motor, "Planetary Gearhead GP 32," rapp. tecn. 166943, 2010, pp. 3–5.

[26] Maxon Motor, "Planetary Gearhead GP42," rapp. tecn. 203129, pp. 1–7.

[27] Robotiq, "2f-85 Adaptive Gripper," rapp. tecn., 2020. indirizzo: https://blog.robotiq.com/hubfs/Product-sheets/Adaptive%20Grippers/Product-sheet-Adaptive-Grippers-EN.pdf?%7B%5C_%7Dga=2.103676870.773845636.1566226093-89642537.1540500648.

[28] *ROS*. indirizzo: https://www.ros.org/.

[29] *Ubuntu*. indirizzo: https://www.ubuntu-it.org/.

[30] *Rappresentazione middleware*. indirizzo: https://www.rfid-soluzioni.com/cosa-puo-fare-per-te-un-middleware/.

[31] *rqt*. indirizzo: http://wiki.ros.org/rqt.

[32] *Rviz*. indirizzo: http://wiki.ros.org/rviz/UserGuide.

[33] K. Bussmann, A. Dietrich e C. Ott, "Whole-Body Impedance Control for a Planetary Rover with Robotic Arm: Theory, Control Design, and Experimental Validation," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 910–917, 2018, ISSN: 10504729. DOI: 10.1109/ICRA.2018.8460533.

[34] F. Furrer, M. Wermelinger, H. Yoshida, F. Gramazio, M. Kohler, R. Siegwart e M. Hutter, "Autonomous robotic stone stacking with online next best object target pose planning," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2350–2356, 2017, ISSN: 10504729. DOI: 10.1109/ICRA.2017.7989272.

[35] I. Fassi e G. Legnani, "Hand to sensor calibration: A geometrical interpretation of the matrix Equation AX=XB," *Journal of Robotic Systems*, vol. 22, n. 9, pp. 497–506, 2005, ISSN: 07412223. DOI: 10.1002/rob.20082.

[36] R. Y. Tsai e R. K. Lenz, "A New Technique for Fully Autonomous and Efficient 3D Robotics Hand/Eye Calibration," *IEEE Transactions on Robotics and Automation*, vol. 5, n. 3, 1989.

[37] A. Trebi-Ollennu, W. Kim, K. Ali, O. Khan, C. Sorice, P. Bailey, J. Umland, R. Bonitz, C. Ciarleglio, J. Knight, N. Haddad, K. Klein, S. Nowak, D. Klein, N. Onufer, K. Glazebrook, B. Kobeissi, E. Baez, F. Sarkissian, M. Badalian, H. Abarca, R. G. Deen, J. Yen, S. Myint, J. Maki, A. Pourangi, J. Grinblat, B. Bone, N. Warner, J. Singer, J. Ervin e J. Lin, "InSight Mars Lander Robotics Instrument Deployment System," *Space Science Reviews*, vol. 214, n. 5, 2018, ISSN: 15729672. DOI: 10.1007/s11214-018-0520-7. indirizzo: http://dx.doi.org/10.1007/s11214-018-0520-7.

[38] T. E. Lee, J. Tremblay, T. To, J. Cheng, T. Mosier, O. Kroemer, D. Fox e S. Birchfield, "Camera-to-Robot Pose Estimation from a Single Image," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 9426–9432, 2020, ISSN: 10504729. DOI: 10.1109/ICRA40945.2020.9196596. arXiv: 1911.09231.

[39] L. E. Ortiz, E. V. Cabrera e L. M. Gonçalves, "Depth data error modeling of the ZED 3D vision sensor from stereolabs," *Electronic Letters on Computer Vision and Image Analysis*, vol. 17, n. 1, pp. 1–15, 2018, ISSN: 15775097. DOI: 10.5565/rev/elcvia.1084.

[40] *Package usb cam*. indirizzo: http://wiki.ros.org/usb_cam.

[41] *Package camera calibration*. indirizzo: http://wiki.ros.org/camera_calibration.

[42] *Package stereo image proc*. indirizzo: http://wiki.ros.org/stereo_image_proc.

[43] *MoveIT*. indirizzo: https://moveit.ros.org/.

[44] *Calibrazione eye on base*. indirizzo: https://github.com/IFL-CAMP/easy_handeye.

[45] *Calibrazione occhio mano immagini*. indirizzo: https://www.youtube.com/watch?v=xQ79ysnrzUk.

[46] *Plugin per Rviz di MoveIt*. indirizzo: https://ros-planning.github.io/moveit_tutorials/doc/hand_eye_calibration/hand_eye_calibration_tutorial.html.

[47] K. Daniilidis, "Hand-Eye Calibration Using Dual Quaternions," *The International Journal of Robotics Research*, vol. 18, n. 3, pp. 286–298, 1999. DOI: 10.1177/02783649922066213.

Appendice A

Risultati della calibrazione

Di seguito vengono riportati i risultati numerici ottenuti dal processo di calibrazione presentato nella Sezione 2.3.

Per la camera **sinistra** sono stati ottenuti i seguenti risultati.

Matrice di calibrazione K:

$$\Big[K\Big] = \begin{bmatrix} 701.599 & 0.000 & 686.599 \\ 0.000 & 703.951 & 370.730 \\ 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{A.1}$$

Coefficienti di distorsione D:

$$\Big[D\Big] = \begin{bmatrix} -0.164 & 0.015 & -0.001 & -0.001 & 0.000 \end{bmatrix} \tag{A.2}$$

Matrice per la rettifica delle immagini Rec:

$$\Big[Rec\Big] = \begin{bmatrix} 0.999 & -0.003 & 0.013 \\ 0.003 & 0.999 & 0.011 \\ -0.013 & -0.011 & 0.999 \end{bmatrix} \tag{A.3}$$

Matrice di proiezione P:

$$\Big[P\Big] = \begin{bmatrix} 691.613 & 0.000 & 694.377 & 0.000 \\ 0.000 & 691.613 & 371.749 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \end{bmatrix} \tag{A.4}$$

Per la camera **destra** sono stati ottenuti i seguenti risultati.

Matrice di calibrazione K:

$$\Big[K\Big] = \begin{bmatrix} 701.727 & 0.000 & 667.117 \\ 0.000 & 703.011 & 374.288 \\ 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{A.5}$$

Coefficienti di distorsione D:

$$\Big[D\Big] = \Big[-0.183 \quad 0.032 \quad -0.003 \quad -0.001 \quad 0.000\Big] \tag{A.6}$$

Matrice per la rettifica delle immagini Rec:

$$\Big[Rec\Big] = \begin{bmatrix} 0.999 & -0.005 & 0.008 \\ 0.005 & 0.999 & -0.011 \\ -0.008 & 0.011 & 0.999 \end{bmatrix} \tag{A.7}$$

Matrice di proiezione P:

$$\Big[P\Big] = \begin{bmatrix} 691.613 & 0.000 & 694.377 & -81.383 \\ 0.000 & 691.613 & 371.749 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \end{bmatrix} \tag{A.8}$$

La **matrice di rotazione** $[R]$ che lega i sistemi di riferimento CCD della camera di destra e sinistra:

$$\Big[R\Big] = \begin{bmatrix} 0.999 & 0.002 & 0.005 \\ -0.002 & 0.999 & 0.022 \\ -0.005 & -0.022 & 0.999 \end{bmatrix} \tag{A.9}$$

Il **vettore traslazione** $[t]$ che lega i sistemi di riferimento CCD della camera di destra e sinistra:

$$\Big[t\Big] = \begin{bmatrix} -0.117 \\ 0.001 \\ -0.001 \end{bmatrix} \tag{A.10}$$