

UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria
Corso di Laurea Magistrale in Ingegneria Meccanica

Mappatura 3D di un Workspace Robotico con Visione Distribuita RGB-D e Movimentazione Robotica per Applicazioni di Pick and Place

3D Mapping of a Robotic Workspace Using Distributed RGB-D Vision and Robotic Manipulation for Pick and Place Applications

Candidato:
Stefano Trillini

Relatore:
Prof. Matteo Claudio Palpacelli

Correlatore:
Dott. Gaetano Lettera

Anno Accademico 2025-2026



UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria
Corso di Laurea Magistrale in Ingegneria Meccanica

Mappatura 3D di un Workspace Robotico con Visione Distribuita RGB-D e Movimentazione Robotica per Applicazioni di Pick and Place

3D Mapping of a Robotic Workspace Using Distributed RGB-D Vision and Robotic Manipulation for Pick and Place Applications

Candidato:
Stefano Trillini

Relatore:
Prof. Matteo Claudio Palpacelli

Correlatore:
Dott. Gaetano Lettera

Anno Accademico 2025-2026

Università Politecnica delle Marche
Facoltà di Ingegneria
Corso di Laurea Magistrale in Ingegneria Meccanica
Via Brecce Bianche – 60131 Ancona (AN), Italy

Alla mia famiglia
e a tutti coloro che mi hanno dato la forza
per raggiungere i miei obiettivi
con l'augurio che possano fare altrettanto

Ringraziamenti

Desidero esprimere la mia sincera gratitudine a tutte le persone che hanno contribuito, in modi diversi, alla realizzazione di questa tesi e al mio percorso accademico.

Un ringraziamento speciale va al mio relatore, il **Prof. Matteo Claudio Palpacelli**, e al mio correlatore, **Dott. Gaetano Lettera**, per la loro disponibilità, la guida e il supporto dimostrati durante tutto lo sviluppo di questo elaborato. Ringrazio inoltre tutti i docenti del corso di laurea magistrale in Ingegneria Meccanica per le competenze trasmesse e la flessibilità dimostrata nel permettermi di conciliare gli impegni universitari con quelli sportivi.

Un grazie di cuore ai miei genitori e a mia sorella Giulia, per il loro sostegno incondizionato, la pazienza e per aver sempre creduto in me appoggiando ogni mia decisione. A Giulia auguro di raggiungere ogni suo obiettivo con serenità e con la certezza del mio costante appoggio, anche se da lontano. Vi voglio bene.

Grazie a Sofia, per essermi stata sempre accanto, anche nei momenti di difficoltà, con pazienza e comprensione.

Grazie ai miei amici e compagni di squadra, per il supporto reciproco, la condivisione delle sfide e dei successi, e per aver reso questi anni un'esperienza unica e indimenticabile.

Infine, un pensiero e una dedica speciale vanno ai miei nonni. Tanti auguri, nonna, che proprio oggi, nel giorno della mia laurea, festeggeresti il tuo compleanno.

Ancona, Ottobre 2026

Stefano Trillini

Sommario

Negli ultimi anni, nei quali la robotica industriale ha un ruolo centrale nell'automazione dei processi produttivi, la cooperazione uomo-macchina si pone al centro dello sviluppo tecnologico. In questo contesto si sono affermati i robot collaborativi (*cobot*), progettati per operare in ambienti condivisi con l'operatore, migliorando efficienza, flessibilità e sicurezza dei processi. L'integrazione dei cobot con sistemi di visione 3D RGB-D e tecniche di mappatura dinamica consente la generazione di digital twin dello spazio operativo, permettendo l'adattamento in ambienti di lavoro soggetti a continue variazioni.

In questo elaborato si descrive lo sviluppo di un digital twin di una cella robotizzata basata sul cobot UR5e, integrato con la movimentazione robotica per un'applicazione di pick and place di prodotti commerciali. Il sistema di visione è composto da due sensori RGB-D fissi e uno mobile installato sul polso del manipolatore. L'infrastruttura software, sviluppata interamente in ROS 2, coordina l'acquisizione e l'elaborazione dei dati sensoriali per la generazione di una mappa dinamica dell'intero workspace (OctoMap). Tale rappresentazione permette di pianificare ed eseguire, attraverso MoveIt, la movimentazione del braccio robotico evitando altri articoli disposti nell'ambiente di lavoro in pose non predefinite, simulando un contesto realistico. Nell'applicazione sviluppata, il robot afferra un articolo da una posizione iniziale nota e lo rilascia in un punto prestabilito, pianificando traiettorie prive di collisioni.

Le prove sperimentali condotte hanno restituito una mappa 3D coerente con la realtà, confermando la corretta rilevazione degli ostacoli. Analogamente, la pianificazione e l'esecuzione di traiettorie prive di collisioni per il completamento del pick and place hanno dimostrato la capacità del sistema di adattarsi a un ambiente variabile. L'analisi dei risultati ha tuttavia evidenziato alcune criticità, come la ridotta libertà di movimento dovuta alla configurazione iniziale vincolata e alla presa degli articoli dall'alto. Inoltre, il rumore sensoriale delle telecamere fisse, posizionate a una distanza elevata dal workspace robotico, compromette la risoluzione della ricostruzione 3D impedendo sia la rilevazione di oggetti più piccoli, sia l'impiego della mappa dinamica durante la movimentazione del robot. Il sistema realizzato può essere impiegato in una vasta gamma di contesti industriali e logistici che richiedono flessibilità e adattamento all'ambiente di lavoro dinamico. Le funzionalità del sistema potranno essere estese con sviluppi futuri, orientati all'ottimizzazione delle traiettorie mediante pianificatori avanzati e vincoli di orientamento, all'integrazione di tecniche di visual servoing, o all'utilizzo di marker dedicati per la definizione automatica delle pose di presa e deposito.

Indice

1 Introduzione **1**
- 1.1 Background e motivazioni . . . 1
- 1.2 Obiettivi di ricerca e rilevanza scientifica . . . 2

2 Fondamenti teorici **5**
- 2.1 Robotica industriale e collaborativa . . . 5
- 2.2 Cinematica del robot . . . 7
- 2.3 Pianificazione del moto e delle traiettorie . . . 10
- 2.4 Sensori di visione . . . 13
  - 2.4.1 Calibrazione sensori di visione . . . 14
- 2.5 Digital twin dei sistemi robotici . . . 17
  - 2.5.1 Mappatura 3D . . . 18

3 Setup sperimentale **21**
- 3.1 Architettura hardware . . . 21
  - 3.1.1 Robot . . . 21
  - 3.1.2 Sistema di visione . . . 28
- 3.2 Architettura Software . . . 30
  - 3.2.1 ROS 2 . . . 30
- 3.3 Configurazione di laboratorio . . . 38

4 Implementazione del digital twin del sistema robotizzato **41**
- 4.1 Configurazione del simulatore digitale . . . 42
- 4.2 Integrazione del sistema di visione . . . 45
  - 4.2.1 Calibrazione estrinseca delle telecamere . . . 46
  - 4.2.2 Fusione ed elaborazione sensoriale . . . 48
- 4.3 Generazione della mappa 3D . . . 53

5 Pianificazione del moto **55**
- 5.1 Gestione delle collisioni . . . 55
- 5.2 Cinematica inversa . . . 56
- 5.3 Pianificatore di traiettorie . . . 57
- 5.4 Pipeline di pick and place . . . 59

6 Test sperimentali e analisi dei risultati **63**

7 Conclusioni e sviluppi futuri **73**

Elenco delle figure

2.1 Rototraslazione del sistema di riferimento {**B**} rispetto ad {**A**} . . . 8
2.2 Schema convenzione DH. 9
2.3 Schematizzazione metodo di pianificazione APF. 11
2.4 Distorsioni dell'immagine causate dall'ottica dei sensori di visione. . 15
2.5 Schematizzazione delle configurazioni di calibrazione telecamera-robot: a sinistra Eye-to-hand, a destra Eye-in-hand. 16
2.6 Pattern geometrici: (a) Scacchiera, (b) marker ArUco e (c) marker ChArUco. 16
2.7 Rappresentazione schematica della suddivisione di un voxel. 19
2.8 Esempi di diverse risoluzioni dei voxel, nella Figura (a) il voxel ha lato 5cm, in (b) 2cm, mentre in (c) la dimensione varia in base alla zona. 19

3.1 Robot UR5e. 22
3.2 Control box UR5e. 23
3.3 Teach Pendant UR5e. 23
3.4 Curva di carico UR5e. 24
3.5 Gripper Robotiq Hand-E. 25
3.6 Schema cinematico UR5e. 27
3.7 Orbbec Gemini 335L. 28
3.8 Orbbec Gemini 2L. 28
3.9 Modalità di comunicazione tra i nodi in ROS 2: (a) Topic, (b) Servizi, (c) Azioni. 31
3.10 Schermata RViz. 33
3.11 Interfaccia di MoveIt in RViz, in cui sono evidenziate la configurazione iniziale (in verde), la configurazione obiettivo (in arancione) e lo stato attuale del robot. 34
3.12 Definizione della configurazione obiettivo: (a) impostazione dei valori dei singoli giunti; (b) impostazione della posa del TCP tramite l'interactive marker. 35
3.13 Schermate *MoveIt Setup Assistant*: (a) importazione modello URDF, (b) ottimizzazione controllo delle collisioni, (c) definizione pose di default del robot. 37
3.14 Configurazione di laboratorio. 38
3.15 Organizzazione dell'area di lavoro. 39

3.16 UR5e con installato il supporto per la hand-eye camera e il gripper Hand-E nella configurazione iniziale. 40

4.1 Digital twin del sistema robotizzato 42
4.2 Modello CAD del supporto della Orbbec Gemini 2L. 43
4.3 Configurazione di RViz. 44
4.4 Configurazione di RViz con integrata una point cloud. 46
4.5 Configurazione del robot per la calibrazione estrinseca delle Orbbec Gemini 335L, con installato il tool di calibrazione, composto da sfera, supporto e asta filettata. 47
4.6 Pipeline di fusione ed elaborazione point cloud 49
4.7 Mappa 3D dell'ambiente operativo del robot generata con OctoMap. 54
4.8 Mappa 3D del laboratorio generata con OctoMap. 54

5.1 Schermata *MoveIt Setup Assistant* per la configurazione del gruppo di pianificazione. 58
5.2 Pose della pipeline di pick and place. 61

6.1 OctoMap del workspace robotico generata dalla Gemini 2L durante il deposito degli articoli nell'area dedicata da parte dell'operatore. . . . 64
6.2 OctoMap dell'intero laboratorio generata dalle Gemini 335L durante il deposito degli articoli nell'area dedicata da parte dell'operatore. . 64
6.3 Pick and place senza ostacoli. 66
6.4 Pick and place con altri articoli posizionati nella zona di prelievo. . . 67
6.5 Pick and place con un articolo più alto nella traiettoria base. 68
6.6 Pick and place con un articolo più alto nella traiettoria base e altri articoli posizionati nella zona di prelievo. 69
6.7 Pick and place con uno scatolone nella traiettoria base. 70

Elenco delle tabelle

3.1 Caratteristiche UR5e. 24
3.2 Caratteristiche Hand-E. 26
3.3 Parametri DH UR5e. 27
3.4 Caratteristiche telecamere RGB-D Orbbec. 29

4.1 Dimensione dei voxel per la riduzione della densità dei punti. 51
4.2 Raggio di ricerca in funzione del sensore per la rimozione dell'ambiente statico. 52
4.3 Raggio di ricerca e soglia K in funzione del sensore per la rimozione dei punti isolati. 53

Capitolo 1

Introduzione

1.1 Background e motivazioni

Negli ultimi decenni, la robotica industriale ha avuto un ruolo centrale nell'automazione dei processi produttivi, con l'obiettivo di aumentare la produttività, la precisione e la sicurezza. Oggi i robot vengono impiegati in numerosi settori industriali (dall'automotive alla chimica, dall'alimentare all'elettronica, dalla meccanica al farmaceutico), trovando applicazione in una vasta gamma di attività ripetitive o potenzialmente pericolose, come la movimentazione di oggetti pesanti, il controllo qualità, l'assemblaggio e le lavorazioni meccaniche [1].

Tradizionalmente la sicurezza è stata garantita mantenendo una netta separazione tra operatore e macchina, attraverso barriere fisiche e sistemi di protezione che impedissero qualsiasi interazione diretta. Tuttavia, l'evoluzione del panorama industriale verso il paradigma dell'Industria 5.0, che enfatizza la cooperazione uomo–macchina come elemento chiave per migliorare l'efficienza, la sostenibilità e il benessere degli operatori, insieme alla crescente richiesta di flessibilità produttiva, ha favorito lo sviluppo e la diffusione dei robot collaborativi (*cobot*) [2]. Questi sistemi, progettati per operare fianco a fianco con l'operatore in ambienti condivisi, combinano la precisione e la ripetibilità dell'automazione robotica con l'esperienza e le capacità umane.

Tale cooperazione richiede che i robot non si limitino a eseguire movimenti predefiniti, ma siano in grado di percepire e interpretare in tempo reale l'ambiente operativo, adattandosi prontamente ai cambiamenti circostanti. A questo scopo, oltre ai sensori avanzati integrati nei robot collaborativi, la mappatura tridimensionale e la generazione di digital twin dell'ambiente di lavoro sono strumenti essenziali per una rappresentazione accurata e dinamica dello spazio operativo [3].

Le moderne tecnologie di visione RGB-D, integrate attraverso l'infrastruttura di comunicazione e gestione dei dati offerta da ROS 2, consentono infatti di realizzare un digital twin aggiornato in tempo reale, grazie al quale il manipolatore adatta velocemente le proprie traiettorie in presenza di eventuali oggetti, superfici o ostacoli. Tale capacità risulta particolarmente rilevante nelle applicazioni di manipolazione e pick and place, dove la posizione e l'orientamento degli oggetti possono variare o non essere noti a priori.

In sintesi, l'integrazione di tecniche di visione tridimensionale e di modellazione digitale dell'ambiente all'interno delle architetture robotiche collaborative rappresenta un passo fondamentale verso sistemi più flessibili e sicuri. Questa evoluzione consente di superare le limitazioni delle celle robotiche tradizionali, caratterizzate da configurazioni rigide e costose, introducendo un nuovo livello di percezione e autonomia, essenziale per la robotica collaborativa, in cui robot e operatori condividono lo spazio di lavoro e operano in ambienti dinamici e soggetti a continue variazioni.

1.2 Obiettivi di ricerca e rilevanza scientifica

Nel contesto della robotica collaborativa descritta nella sezione precedente, questo elaborato si propone di sviluppare un sistema in grado di percepire, rappresentare e interpretare l'ambiente operativo di un robot collaborativo, integrando tali informazioni nei processi di pianificazione e controllo del movimento.

L'obiettivo principale è la realizzazione di un digital twin dell'intero sistema robotizzato, ottenuto a partire dai dati di profondità acquisiti dal sistema di visione composto da sensori RGB-D. Tali dati vengono elaborati per generare una OctoMap, una mappa 3D che consente di distinguere lo spazio libero dagli ostacoli, fornendo una rappresentazione compatta e continuamente aggiornata dell'area operativa. Questo modello permette di individuare e localizzare non solo oggetti statici, ma anche elementi variabili o non noti a priori, adattando le traiettorie pianificate del manipolatore per garantire un funzionamento sicuro ed efficiente in ambienti non strutturati o condivisi con l'operatore.

In linea con questi obiettivi, è stata sviluppata una pipeline di pianificazione del movimento integrata con il digital twin, impiegata in un'operazione di pick and place in cui il cobot manipola prodotti tipici da supermercato evitando gli altri articoli disposti nell'area di lavoro in posizioni non predefinite, simulando un contesto realistico logistico e collaborativo. In particolare, l'esperimento prevede la manipolazione di un articolo con posa iniziale nota e punto di rilascio definito, pianificando traiettorie che evitano collisioni con gli altri oggetti presenti nell'area di lavoro, rilevati dal sistema di visione. Questo scenario sperimentale consente di validare le strategie di percezione, modellazione e adattamento dinamico delle traiettorie sviluppate in un ambiente variabile.

Dal punto di vista scientifico e tecnologico, i contributi più significativi del progetto possono essere riassunti come segue:

- Integrazione sinergica di percezione e pianificazione, mediante un'infrastruttura basata su *ROS 2* che coordina l'acquisizione sensoriale, la modellazione 3D e la pianificazione del moto in un'architettura modulare e flessibile (vedi Capitoli 4 e 5).

- Sviluppo di un sistema di fusione delle point cloud, ottenute da più telecamere RGB-D che inquadrano la medesima area da prospettive differenti, al fine di

generare una rappresentazione tridimensionale unificata e coerente dell'ambiente operativo (vedi Capitolo 4.2).

- Realizzazione di un digital twin operativo, ottenuto a partire dalle point cloud convertite in una OctoMap e arricchito con la presenza di elementi noti della scena, in grado di fornire una rappresentazione coerente e continuamente aggiornata dello spazio operativo, impiegata per la pianificazione delle traiettorie (vedi Capitolo 4).

- Sviluppo di una pianificazione del moto adattiva, capace di sfruttare le informazioni fornite dal digital twin per generare e aggiornare in tempo reale le traiettorie del manipolatore in funzione di ostacoli o variazioni dell'ambiente, evitando le collisioni e incrementando sicurezza ed efficienza (vedi Capitolo 5).

- Validazione sperimentale in uno scenario collaborativo realistico, volto a dimostrare la capacità del sistema di operare in ambienti non strutturati e soggetti a cambiamenti, garantendo un'interazione sicura con l'operatore (vedi Capitolo 6).

Nel loro insieme, le soluzioni proposte contribuiscono ad ampliare le capacità dei robot industriali verso una maggiore autonomia e adattabilità operativa, riducendo la necessità di conoscere in anticipo l'area di lavoro.

La tecnologia sviluppata può essere impiegata in una vasta gamma di contesti industriali e logistici che richiedono flessibilità operativa e capacità di adattamento. Un esempio è rappresentato dalle celle di manipolazione e movimentazione automatica, dove il robot deve eseguire operazioni di pick and place su piani di lavoro in cui la posizione dei componenti o degli utensili varia o non è nota a priori. In questi scenari, la pianificazione adattiva consente di aggiornare in tempo reale le traiettorie, reagendo a cambiamenti improvvisi nel workspace, così da ridurre tempi di fermo e interventi di riconfigurazione. Un ulteriore ambito di impiego riguarda i sistemi di smistamento e confezionamento, nei quali i prodotti presentano forme, dimensioni e pose differenti, e dove l'infrastruttura proposta permette di adattare il comportamento del robot alle variazioni della linea produttiva, migliorando efficienza e continuità del processo [4]. Inoltre, l'approccio può essere esteso al coordinamento di più stazioni robotiche nella stessa area di lavoro, permettendo l'aggiornamento condiviso della mappa e una gestione sicura e coordinata delle loro operazioni. La stessa architettura può essere infine integrata con algoritmi di riconoscimento degli oggetti, abilitando celle completamente automatizzate capaci di identificare, selezionare e manipolare articoli specifici in base alla loro categoria o funzione.

Nel complesso, gli sviluppi presentati rappresentano un passo verso la realizzazione di ecosistemi robotici collaborativi capaci di percepire, comprendere e reagire in modo autonomo al contesto operativo. L'integrazione tra percezione, digital twin e pianificazione dinamica si inserisce pienamente nella transizione verso l'Industria

5.0, che pone al centro dell'automazione la flessibilità, l'interazione uomo-robot e la sicurezza.

Capitolo 2

Fondamenti teorici

In questo Capitolo vengono presentati i principali fondamenti teorici che supportano il lavoro di tesi, con l'obiettivo di fornire una base concettuale utile alla comprensione dei metodi, degli strumenti e delle tecnologie utilizzate. Dopo una panoramica introduttiva sulla robotica industriale e collaborativa, verranno illustrati gli aspetti principali della cinematica dei robot, fondamentali per la descrizione e l'analisi del movimento del manipolatore, e quelli relativi alla pianificazione del moto e delle traiettorie. Seguirà una sezione dedicata al sistema di visione basato su sensori RGB-D, con particolare attenzione alle procedure di calibrazione. Infine, verrà illustrato il concetto di digital twin applicato al sistema robotico, insieme al processo di mappatura 3D dell'ambiente, realizzato attraverso l'utilizzo di OctoMap.

2.1 Robotica industriale e collaborativa

La robotica industriale rappresenta uno dei pilastri dell'automazione moderna, nata con l'obiettivo di agevolare le operazioni ripetitive, faticose o potenzialmente pericolose per l'essere umano. A partire dagli anni '60, l'introduzione dei robot industriali nelle linee di produzione ha rivoluzionato i processi produttivi, migliorando in modo significativo l'efficienza, la ripetibilità e la qualità delle lavorazioni. Secondo la definizione fornita dal *Robotic Institute of America* (RIA), un robot è un "*manipolatore multifunzionale, riprogrammabile, progettato per muovere materiali, pezzi, utensili o dispositivi specializzati attraverso movimenti programmati variabili per l'esecuzione di una varietà di compiti*". Una definizione simile è contenuta nello standard ISO 8373, che descrive il robot industriale come un "*manipolatore controllato automaticamente, riprogrammabile, multifunzionale, a tre o più assi, fisso o mobile, impiegato per applicazioni di automazione industriale*". Da entrambe le definizioni emerge che la forza della robotica risiede nella versatilità e riprogrammabilità dei sistemi, che li rende adatti a una vasta gamma di contesti produttivi e operativi [5]. Dal punto di vista funzionale, un sistema robotico industriale è composto da diversi sottosistemi interconnessi:

- Sistema meccanico: costituisce la struttura portante del robot, generalmente composta da una serie di giunti e collegamenti rigidi che definiscono la sua cinematica.

- Sistema di attuazione: include motori elettrici, servomeccanismi e trasmissioni che generano e controllano il movimento dei giunti.
- Sistema sensoriale: permette di acquisire informazioni sullo stato interno del robot e sull'ambiente circostante attraverso sensori di posizione, forza, coppia, visione o prossimità.
- Sistema di controllo: elabora i dati provenienti dai sensori e comanda gli attuatori per eseguire i compiti programmati, inoltre è dotato di un'interfaccia che consente all'operatore di monitorare lo stato del sistema e inviare comandi al robot [6].

I robot industriali tradizionali operano in ambienti segregati, separati dagli operatori umani mediante barriere fisiche o gabbie di sicurezza. Questo approccio consente di lavorare ad alta velocità e forze senza compromettere la sicurezza, ma limita fortemente la flessibilità del sistema, comportando costi elevati di installazione e riconfigurazione, oltre a un'interazione impossibile con l'uomo.

Con l'evoluzione dei paradigmi produttivi verso l'Industria 5.0, è emersa la necessità di sistemi robotici capaci di operare in modo flessibile e sicuro a fianco dell'uomo, riducendone l'affaticamento e valorizzandone le competenze. In questo contesto si inserisce la robotica collaborativa, e in particolare i robot collaborativi, noti anche come cobot (collaborative robots). Questi sistemi sono progettati per condividere lo spazio di lavoro con l'operatore senza la necessità di barriere fisiche, grazie all'integrazione di sensori di forza e coppia, limitatori di velocità e algoritmi di controllo avanzati che monitorano continuamente lo stato del sistema. L'obiettivo dei cobot non è sostituire l'operatore, ma collaborare con lui, combinando la creatività e la capacità di adattamento umane alla precisione, ripetibilità e instancabilità tipiche della macchina. Le principali caratteristiche che differenziano i cobot dai robot tradizionali sono:

- **Sicurezza intrinseca**: conformità alle normative ISO/TS 15066, che definiscono limiti di forza, pressione e velocità per garantire l'incolumità degli operatori.
- **Facilità di programmazione**: grazie a interfacce grafiche intuitive e alla possibilità di programmazione per dimostrazione diretta, che consente di guidare manualmente il robot in configurazioni target che raggiungerà in autonomia.
- **Flessibilità d'integrazione**: possono essere facilmente spostati o riconfigurati all'interno della linea produttiva, anche tramite basi mobili o moduli intercambiabili.
- **Ridotti costi e ingombri**: l'assenza di barriere fisiche e la semplicità di installazione li rendono economici e adatti anche a contesti produttivi di piccole e medie dimensioni.[7]

Grazie a queste caratteristiche, i cobot trovano applicazione in numerosi settori, tra cui logistica, elettronica, alimentare, biomedicale e automotive, svolgendo compiti come l'assemblaggio leggero, le lavorazioni meccaniche, la manipolazione, il confezionamento e il controllo qualità [1].

In sintesi, mentre la robotica industriale tradizionale rappresenta un paradigma di automazione ad alte prestazioni ma rigido e isolato, la robotica collaborativa introduce un approccio orientato alla cooperazione sicura e interattiva uomo-macchina, caratterizzato da flessibilità e adattabilità operativa. La comprensione delle basi strutturali e funzionali dei sistemi robotici, insieme alla descrizione della loro cinematica (approfondita nel Capitolo successivo), nonché ai principi di percezione, controllo e pianificazione del movimento, costituisce un presupposto essenziale per l'analisi del comportamento dei robot collaborativi.

2.2 Cinematica del robot

La cinematica è una disciplina fondamentale nell'ambito della robotica, in quanto consente di analizzare e descrivere il movimento dei robot, costituendo la base per le attività di controllo, programmazione e simulazione. Essa studia il moto di un sistema di corpi senza considerare le forze e le coppie che lo generano, concentrandosi su posizione, velocità e accelerazione.

I robot vengono considerati come catene cinematiche aperte di corpi rigidi, chiamati link , connessi da giunti prismatici o rotoidali. Le equazioni della cinematica descrivono la relazione tra la posizione, la velocità e l'accelerazione di ciascun giunto, variabili che definiscono lo spazio dei giunti, e la posizione, la velocità e l'accelerazione del terminale (o end-effector) rispetto a un sistema di riferimento fisso, detto spazio di lavoro cartesiano.

Le configurazioni cinematiche, ossia le posizioni dei link o dei giunti del robot, vengono definite attraverso due tipologie di sistemi di riferimento:

- **Sistema globale (terna mondo)**: fisso rispetto all'ambiente in cui opera il robot, solitamente ancorato alla sua base, e utilizzato per rappresentare la posizione e l'orientamento complessivi del manipolatore.
- **Sistemi locali (terne solidali ai link del robot)**: ciascun sistema di riferimento è ancorato a un punto specifico di un link e si muove solidalmente con esso, consentendo di definire le trasformazioni relative tra i diversi segmenti del robot e di effettuare i calcoli cinematici.

La variazione di uno o più giunti del manipolatore provoca il movimento di uno o più link successivi, ciò comporta che le origini dei sistemi di riferimento locali traslino reciprocamente, mentre gli assi variano orientamento [8].

Come illustrato in Figura 2.1, la posizione dell'origine O_b di un sistema di riferimento locale $\{\mathbf{B}\}$ rispetto a quello precedente $\{\mathbf{A}\}$ è descritta dal vettore di traslazione

$Q \in \mathbb{R}^3$, mentre l'orientamento relativo è rappresentato dalla matrice di rotazione $\mathbf{R}_A^B \in \mathbb{R}^{3\times3}$. Entrambi questi elementi consentono di definire la trasformazione $\mathbf{T}_A^B$ del sistema $\{\mathbf{B}\}$ rispetto a $\{\mathbf{A}\}$, i quali rappresentano due link consecutivi. La trasformazione è espressa in modo compatto, tramite il formalismo delle matrici di trasformazione omogenea (HTM, *Homogeneous Transformation Matrix*), con una matrice 4×4 che combina rotazione e traslazione:

$$\mathbf{T}_A^B = \begin{bmatrix} \mathbf{R}_A^B & Q \\ 0 & 1 \end{bmatrix}$$

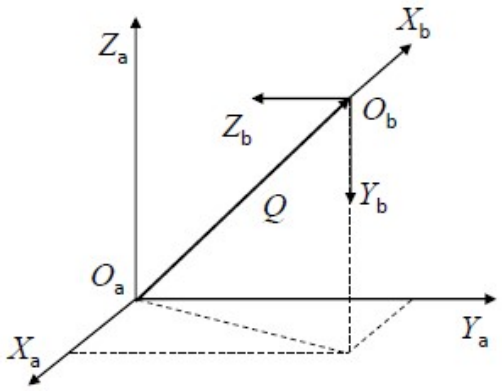


Figura 2.1: Rototraslazione del sistema di riferimento $\{\mathbf{B}\}$ rispetto ad $\{\mathbf{A}\}$

Questo formalismo consente inoltre di concatenare trasformazioni multiple mediante semplici moltiplicazioni di matrici, descrivendo così l'intera catena cinematica del robot a partire dalle relazioni tra giunto e giunto.

Al fine di definire in modo univoco i sistemi di riferimento associati ai link e di generare in maniera sistematica le corrispondenti matrici di trasformazione, viene adottata la convenzione di Denavit-Hartenberg (DH) illustrata nello schema riportato in Figura 2.2. Secondo tale convenzione, la relazione tra due giunti consecutivi di un manipolatore può essere descritta tramite quattro parametri fondamentali:

- a_i: distanza tra gli assi z_i e z_{i+1}, misurata lungo l'asse x_i;
- α_i: angolo tra gli assi z_i e z_{i+1}, misurato attorno all'asse x_i;
- d_i: distanza tra gli assi x_{i-1} e x_i, misurata lungo l'asse z_i;
- θ_i: angolo di rotazione tra gli assi x_{i-1} e x_i, misurato attorno all'asse z_i.

Questi parametri consentono di formalizzare in maniera uniforme la descrizione geometrica del robot, indipendentemente dalla sua complessità o dal numero di gradi di libertà.

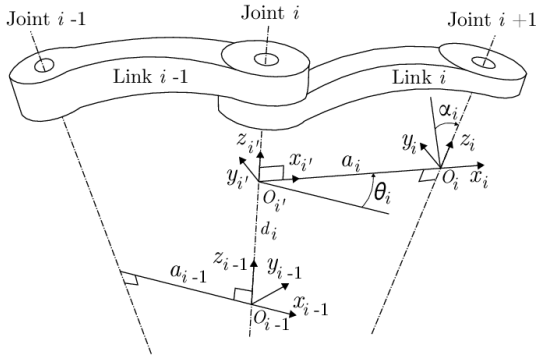


Figura 2.2: Schema convenzione DH.

La trasformazione $\mathbf{T}_i^{i+1}$ tra un sistema di riferimento i e quello successivo $i+1$ viene così descritta da una matrice omogenea costruita a partire dai parametri DH, seguendo una sequenza prestabilita di rotazioni e traslazioni definite dalla convenzione stessa:

$$\mathbf{T}_i^{i+1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

La trasformazione complessiva tra il sistema di riferimento base e quello dell'organo terminale (end-effector), in funzione dei valori dei giunti, rappresenta il problema della cinematica diretta, e si ottiene come prodotto delle matrici di trasformazione associate ai singoli giunti:

$$\mathbf{T}_0^n = \mathbf{T}_0^1 \cdot \mathbf{T}_1^2 \cdot \mathbf{T}_2^3 \cdots \mathbf{T}_{n-1}^n$$

dove ciascuna $\mathbf{T}_i^{i+1}$ è ottenuta a partire dai parametri DH corrispondenti al giunto i. Nel caso di strutture seriali, il problema è ben definito e ammette un'unica soluzione per ogni configurazione di giunti. In questo modo, il formalismo DH permette di derivare, a partire dai dati geometrici del robot, una rappresentazione compatta ed efficiente della cinematica diretta, utile nelle fasi di simulazione, pianificazione del moto e controllo del manipolatore. Tale formalismo costituisce anche la base per la risoluzione della cinematica inversa che consiste nel determinare il valore dei giunti a partire dalla posa (posizione e orientamento) desiderata dell'end-effector. A differenza della cinematica diretta, il problema inverso risulta generalmente più complesso, in quanto:

- Può ammettere più soluzioni (ridondanza).
- Può non ammettere soluzioni, quando la posa desiderata è al di fuori dello spazio di lavoro.
- Può presentare configurazioni singolari, ossia punti in cui il robot perde uno o più gradi di libertà effettivi.

La cinematica inversa può essere risolta mediante due principali approcci:

1. **Metodi analitici**: utilizzati per robot con configurazioni relativamente semplici, consentono di determinare una soluzione esplicita tramite la risoluzione diretta delle equazioni geometriche e trigonometriche che legano le variabili di giunto alla posa desiderata dell'end-effector.

2. **Metodi numerici**: adottati per robot con architetture più complesse, si basano su algoritmi iterativi che, data una funzione obiettivo, cercano le configurazioni dei giunti che minimizzano l'errore rispetto alla posa desiderata.

La cinematica inversa è essenziale nel controllo dei robot, in quanto consente di pianificare traiettorie nello spazio cartesiano specificando direttamente le pose desiderate del terminale [9].

2.3 Pianificazione del moto e delle traiettorie

La pianificazione del moto costituisce uno degli aspetti fondamentali della robotica, in quanto consente di determinare una traiettoria che permetta a un manipolatore di spostarsi da una configurazione iniziale a una finale in modo efficiente, rispettando vincoli geometrici, cinematici e dinamici. Basandosi sui modelli cinematici del robot, gli algoritmi di pianificazione calcolano percorsi continui e fisicamente realizzabili, garantendo il raggiungimento dell'obiettivo prefissato ed evitando collisioni con eventuali ostacoli presenti nell'ambiente.

Le prime tecniche elaborate per la pianificazione del moto, come i metodi graph-based, discretizzano lo spazio di lavoro e utilizzano algoritmi di ricerca su grafo (una rappresentazione di configurazioni e collegamenti) per individuare il percorso ottimale. Questo approccio garantisce una pianificazione precisa in ambienti semplici, tuttavia, con l'aumentare dei gradi di libertà dei manipolatori, diventa rapidamente inutilizzabile, a causa della crescita esponenziale della complessità computazionale.

Per superare tali limiti, sono stati sviluppati metodi più evoluti, capaci di operare in spazi continui e di gestire configurazioni complesse con maggiore efficienza. I principali sono descritti di seguito:

- **Sampling-Based Algorithms**. Questi metodi esplorano lo spazio delle configurazioni attraverso campioni generati in modo casuale o quasi casuale, con l'obiettivo di connettere le configurazioni tramite traiettorie localmente valide, costruendo roadmap (reti di connessioni tra configurazioni valide) o alberi di esplorazione. Tali approcci mantengono un'elevata efficienza anche in spazi ad alta dimensionalità, pur non garantendo soluzioni sempre ottimali. A differenza dei metodi esaustivi, essi offrono completezza probabilistica, ossia la probabilità di trovare un percorso cresce con l'aumentare del tempo di esecuzione. Gli algoritmi si differenziano principalmente per la modalità con cui campionano lo spazio e costruiscono la connettività tra i nodi. Alcune

varianti introducono strategie di campionamento guidato (o *biased sampling*) per accelerare la convergenza verso la soluzione. Ad esempio RRT-Connect estende progressivamente l'albero di esplorazione in direzione della configurazione obiettivo finché non la raggiunge o incontra un ostacolo, migliorando l'efficienza in spazi aperti. Il Potential Function-Based RRT* impiega invece campi potenziali per orientare il campionamento, combinando una componente attrattiva verso il goal e una repulsiva dagli ostacoli, ottenendo percorsi più fluidi e sicuri. Infine, approcci più recenti, come il Critical PRM, impiegano reti neurali per identificare configurazioni chiave e connetterle nello spazio di configurazione, generando percorsi preferenziali che aumentano l'efficienza del processo di pianificazione. Nonostante la buona scalabilità, tali metodi generano spesso traiettorie irregolari e poco fluide, rendendo necessarie fasi di post-ottimizzazione o strategie di semplificazione della roadmap.

- **Artificial Potential Field (APF)**. Questo metodo si basa sulla costruzione di un campo potenziale attrattivo generato dalla posizione di goal e un campo potenziale repulsivo generato dagli ostacoli, come schematizzato in Figura 2.3. Il robot quindi si muove verso l'obiettivo evitando gli ostacoli, guidato dal campo potenziale risultante dalla combinazione dei due. L'APF è utilizzato per la sua semplicità e capacità di funzionamento in tempo reale, ma presenta alcune limitazioni, come la vulnerabilità ai minimi locali che impedisce al manipolatore di raggiungere l'obiettivo, e lo rende meno adatto per ambienti dinamici.

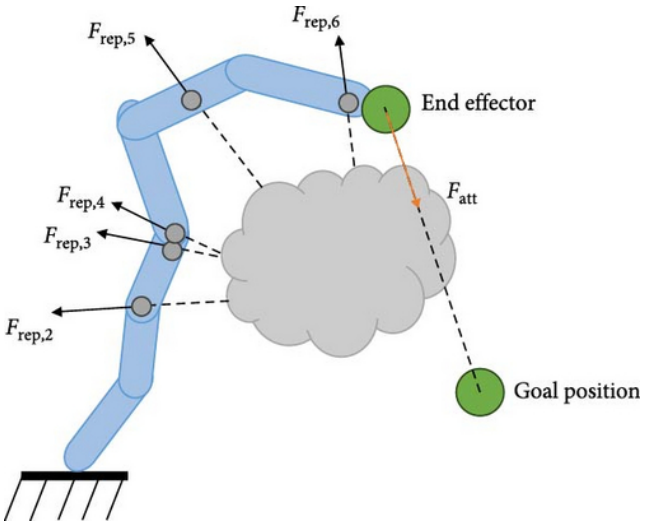


Figura 2.3: Schematizzazione metodo di pianificazione APF.

- **Optimization-Based Algorithms**. Formulano la pianificazione del moto come un problema di ottimizzazione di una traiettoria iniziale, minimizzando una funzione di costo soggetta a una serie di vincoli cinematici, dinamici e di collisione. La funzione di costo valuta la qualità complessiva della traiettoria,

mentre i vincoli garantiscono il rispetto dei limiti fisici del robot e dei requisiti specifici del compito, come il raggiungimento degli obiettivi o l'evitamento degli ostacoli. Nella funzione di costo, i diversi algoritmi introducono ulteriori criteri di ottimizzazione, ad esempio l'efficienza temporale (GOMP), la minimizzazione della lunghezza del percorso (TrajOpt e ISIMP) o la fluidità della traiettoria (CHOMP). Questi approcci consentono di generare traiettorie fluide e prive di collisioni, risultando ideali per manipolatori industriali in ambienti strutturati.

- **Deep Reinforcement Learning (DRL)**. Gli algoritmi DRL, si basano sulla teoria matematica del MDP (Markov Decision Process), un processo decisionale in cui un agente, in questo caso il robot, osserva lo stato in cui si trova, compie un'azione e riceve una ricompensa che misura quanto quell'azione lo avvicina all'obiettivo prefissato. Sulla base di queste esperienze, il robot aggiorna progressivamente la propria strategia di comportamento, detta policy, con l'obiettivo di massimizzare la somma delle ricompense future. La modalità con cui vengono scelte le azioni dipende dall'algoritmo adottato: dai metodi più semplici, come Q-learning e Deep Q-Network (DQN), fino agli approcci più avanzati e stabili, come Deep Deterministic Policy Gradient (DDPG), Twin Delayed DDPG (TD3) e Soft Actor-Critic (SAC). Questi ultimi consentono di operare in spazi d'azione continui e garantiscono maggiore efficienza e robustezza nell'apprendimento, anche in ambienti dinamici o complessi.

In sintesi, i metodi basati su campionamento sono efficienti in spazi complessi, mentre le tecniche di ottimizzazione di traiettoria privilegiano la qualità e la fluidità del movimento, risultando ideali per la rifinitura di traiettorie già eseguibili. Gli approcci di tipo DRL, infine, rappresentano una frontiera più recente e autonoma, in grado di apprendere politiche di pianificazione ottimali direttamente dall'interazione con l'ambiente [10].

Un'ampia gamma di algoritmi sampling-based, tra cui PRM, PRM*, RRT, RRT* e KPIECE, è raccolta all'interno di una delle librerie open-source più diffuse per la pianificazione del moto: OMPL (Open Motion Planning Library). OMPL è una libreria software sviluppata in C++, progettata per essere indipendente dal modello del robot e dall'ambiente di lavoro, e facilmente integrabile all'interno di framework esterni per la simulazione e il controllo del movimento, come ROS 2. In particolare, OMPL viene integrata in piattaforme avanzate come MoveIt, che gestiscono il modello cinematico del robot, la descrizione dell'ambiente, il controllo delle collisioni e l'interfaccia con i sensori, consentendo di verificare la validità del percorso pianificato e di tradurlo in traiettorie effettivamente eseguibili. Questa integrazione permette inoltre di sfruttare la flessibilità di OMPL nella selezione e combinazione dei diversi algoritmi di pianificazione, anche all'interno di una stessa applicazione, assegnando a ciascun movimento o fase operativa l'algoritmo più adatto alle esigenze specifiche, ad esempio privilegiando la rapidità di calcolo in alcune fasi o l'ottimizzazione del percorso in altre [11].

2.4 Sensori di visione

I sensori di visione rappresentano una componente fondamentale per la percezione dell'ambiente circostante in numerose applicazioni di robotica, realtà aumentata e realtà virtuale (AR e VR). Tradizionalmente, si utilizzano telecamere RGB (Red, Green, Blue), che acquisiscono immagini bidimensionali in cui ogni pixel è descritto da tre valori di intensità che rappresentano il contributo di rosso, verde e blu. La combinazione di tali valori consente di ottenere la gamma di colori percepita dall'occhio umano. Le telecamere RGB non forniscono informazioni sulla profondità, infatti due oggetti con caratteristiche visive simili, ma posti a distanze diverse dalla telecamera, possono risultare indistinguibili.

Per superare questa limitazione sono stati sviluppati i sensori RGB-D, dispositivi che, oltre ad acquisire l'immagine a colori, associano a ciascun pixel un'informazione di distanza dalla telecamera. In altre parole, ai tradizionali canali R, G e B viene aggiunto un canale D (Depth), che rappresenta la coordinata di profondità z di ciascun punto della scena rispetto al sensore, generando una mappa di profondità [12].

Le tecniche per la rilevazione della profondità si basano su principi fisici differenti, le principali tecnologie utilizzate nelle telecamere RGB-D sono le seguenti:

- **Structured light (a luce strutturata)**: proietta sull'ambiente un pattern luminoso a infrarossi noto, come griglie o punti. Il pattern, una volta deformato dalle superfici tridimensionali su cui è proiettato, viene rilevato dal sensore e conoscendo la posizione relativa tra proiettore e sensore, è possibile ricostruire la distanza dei punti tramite triangolazione ottica. I sistemi basati su questa tecnologia risultano altamente efficaci in scenari interni e a distanza ravvicinata, offrendo mappe di profondità dettagliate e ad alta risoluzione. Sono spesso impiegati nel riconoscimento dei gesti e nella scansione di piccoli oggetti, ma risultano meno adatti per ambienti esterni o di grandi dimensioni a causa dell'interferenza della luce solare.

- **Time of flight (ToF) (tempo di volo)**: emette impulsi di luce infrarossa e misura il tempo impiegato dalla luce per riflettersi dagli oggetti e ritornare al sensore, utilizzando tale ritardo per calcolare con precisione la distanza da ciascun punto all'interno del campo visivo. Esistono due tipologie principali: ToF diretto (dToF), che misura il tempo di volo di singoli impulsi luminosi, e ToF indiretto (iToF), che emette un fascio luminoso modulato in intensità (ad esempio un'onda sinusoidale) e deduce la profondità misurando lo sfasamento tra il segnale emesso e quello ricevuto. Questi sistemi forniscono dati di profondità molto accurati con frame rate elevati, il che li rende ideali per ambienti dinamici, d'altra parte, possono presentare difficoltà con superfici riflettenti e forte luce ambientale.

- **Stereo vision (sistemi di visione stereo)**: si ispira al principio della visione binoculare umana, utilizzando due o più telecamere RGB poste a una distanza nota tra loro. Ogni telecamera cattura una proiezione bidimensionale della stessa scena da un punto di vista leggermente diverso e, per ciascun punto visibile, è possibile calcolare la disparità, ossia lo spostamento apparente della sua posizione nelle diverse immagini acquisite. La profondità viene poi ricavata tramite triangolazione geometrica. I sistemi basati su questa tecnologia sono versatili e possono funzionare in diverse condizioni di illuminazione, risultando adatti anche per applicazioni in ambienti esterni. Richiedono tuttavia algoritmi sofisticati per gestire occlusioni e superfici prive di texture o troppo uniformi, oltre a presentare in genere elevate esigenze computazionali. Questa tecnologia è impiegata anche nelle telecamere Orbbec Gemini 2L e Orbbec Gemini 335L utilizzate nel presente studio. In particolare, la Gemini 2L adotta la tecnologia Active Stereo IR, una variante della stereo vision tradizionale in cui, oltre alle due telecamere stereo, è integrato anche un proiettore infrarosso (IR) che emette un pattern luminoso invisibile. Tale pattern crea texture artificiali sulla scena, rendendo più semplice e robusta la ricerca di corrispondenze tra i due canali stereo, ampliando il funzionamento anche in ambienti uniformi e scarsamente illuminati, poiché il pattern infrarosso è rilevabile anche al buio [13].

Ogni tecnologia di acquisizione della profondità presenta vantaggi e limitazioni specifiche, che ne guidano la scelta in funzione del contesto applicativo. I sensori RGB-D rappresentano un elemento chiave per la percezione tridimensionale, permettendo ai robot di comprendere e interagire accuratamente con l'ambiente circostante, a condizione che la relazione geometrica tra sensore e scena sia determinata con precisione attraverso un'adeguata calibrazione.

2.4.1 Calibrazione sensori di visione

La calibrazione dei sensori di visione rappresenta un processo fondamentale per garantire una rappresentazione accurata e affidabile dello spazio reale. Essa consente di determinare la relazione tra le coordinate dei punti rilevati dal sensore e quelle del mondo reale, correggendo le distorsioni ottiche introdotte dal sistema di acquisizione e definendo in modo preciso la posizione e l'orientamento della telecamera rispetto a un riferimento esterno. Una calibrazione accurata è quindi essenziale per tutte le applicazioni di robotica, come la ricostruzione tridimensionale dell'ambiente, in cui la coerenza spaziale dei dati influisce direttamente sulle prestazioni del sistema.

Il processo di calibrazione può essere suddiviso in due fasi principali: la calibrazione intrinseca, che riguarda i parametri interni della telecamera, ovvero le caratteristiche ottiche e geometriche del sistema di acquisizione, e la calibrazione estrinseca, che definisce la relazione spaziale tra la telecamera e l'ambiente esterno.

La calibrazione intrinseca consente di stimare la lunghezza focale, il punto principale (cioè la proiezione del centro ottico sul piano dell'immagine) e i coefficienti di distorsione radiale e tangenziale dell'obiettivo. Questi parametri sono indispensabili per correggere le distorsioni introdotte dall'ottica illustrate in Figura 2.4 [14].



Figura 2.4: Distorsioni dell'immagine causate dall'ottica dei sensori di visione.

La calibrazione estrinseca, invece, permette di determinare con precisione la posizione (rappresentata dal vettore di traslazione $\mathbf{t} \in \mathbb{R}^{3\times1}$) e l'orientamento (descritto dalla matrice di rotazione $\mathbf{R} \in \mathbb{R}^{3\times3}$) del sensore rispetto a un sistema di riferimento esterno. Nel contesto di un sistema robotizzato, tale procedura assume un ruolo cruciale poiché individua la posizione e l'orientamento del sensore rispetto al robot, rendendo possibile la fusione sensoriale e la costruzione di mappe coerenti, in cui è necessario integrare le informazioni provenienti da più telecamere con il modello cinematico del manipolatore. I parametri estrinseci ($\mathbf{t}$ e $\mathbf{R}$) vengono organizzati nella matrice di rototraslazione omogenea $\mathbf{T}_{camera\to robot}$:

$$\mathbf{T}_{camera\to robot} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}$$

la quale consente di trasformare le coordinate dei punti $\mathbf{q}_i$, espresse nel sistema di riferimento della telecamera, nelle corrispondenti coordinate $\mathbf{p}_i$, espresse nel sistema di riferimento del robot:

$$\mathbf{p}_i = \mathbf{T}_{camera\to robot}\ \mathbf{q}_i$$

Esistono due configurazioni principali di calibrazione telecamera-robot, che variano in base alla posizione del sensore rispetto al manipolatore, come mostrato schematicamente in Figura 2.5:

- **Eye-in-hand**: la telecamera è montata sull'end-effector del robot. In questo caso, la calibrazione consente di determinare la posizione e l'orientamento relativi tra la telecamera e il terminale del manipolatore. La procedura prevede l'acquisizione di una serie di immagini di un oggetto di geometria nota, mantenuto statico, mentre il robot si sposta in diverse configurazioni.

- **Eye-to-hand**: la telecamera è fissa rispetto all'ambiente e osserva il robot durante il movimento. La calibrazione serve quindi a determinare la posizione

e l'orientamento della telecamera rispetto al sistema di riferimento del robot. In questo caso, un oggetto di forma nota, come una sfera o un marker, viene fissato all'end-effector e ripreso da varie posizioni [15].

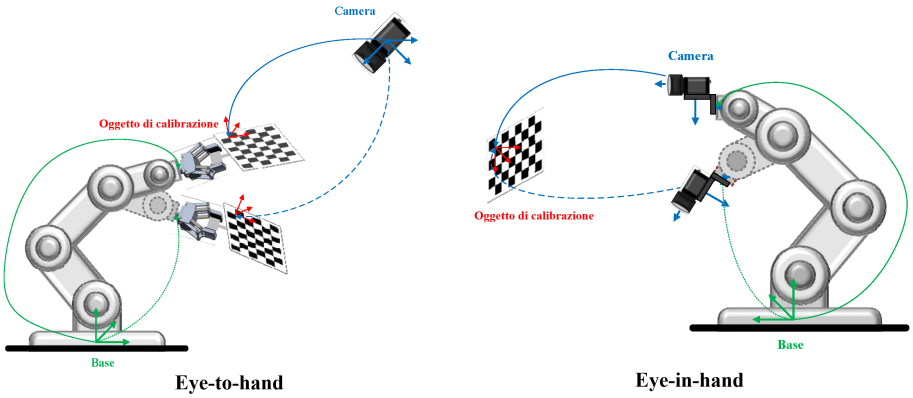


Figura 2.5: Schematizzazione delle configurazioni di calibrazione telecamera-robot: a sinistra Eye-to-hand, a destra Eye-in-hand.

Tradizionalmente, la calibrazione, sia intrinseca che estrinseca, viene eseguita utilizzando pattern geometrici noti, come scacchiere o marker ArUco/ChArUco (Figura 2.6), chiaramente visibili dalla telecamera [16].



(a) (b) (c)

Figura 2.6: Pattern geometrici: (a) Scacchiera, (b) marker ArUco e (c) marker ChArUco.

Il metodo più diffuso è quello di *Zhang* [17], che consente di stimare i parametri di calibrazione a partire da un insieme di immagini di una scacchiera di riferimento acquisite da diverse angolazioni. In ciascuna immagine vengono individuati i vertici della scacchiera (corner detection) e utilizzati per calcolare la relazione geometrica tra le coordinate note del pattern nel mondo reale e le loro proiezioni nell'immagine. Dall'insieme di tali relazioni si ottiene un sistema di equazioni che, una volta risolto, restituisce i parametri intrinseci ed estrinseci della telecamera. Tale approccio presenta alcune limitazioni, poiché richiede l'intervento manuale per il corretto posizionamento del pattern e la completa visibilità del riferimento in ciascuna immagine. Per superare queste limitazioni, sono stati sviluppati metodi alternativi basati su

oggetti di forma semplice, come sfere o marker tridimensionali, che permettono una calibrazione più robusta e facilmente automatizzabile. Tra questi, il metodo di *Liu et al.* [18] utilizza una telecamera RGB-D spostata in configurazioni predefinite e un insieme di sfere statiche come riferimento, eseguendo una stima iniziale della trasformazione tra i sistemi di riferimento, seguita da un affinamento tramite ottimizzazione non lineare.

Una calibrazione accurata, ottenuta combinando i parametri intrinseci ed estrinseci, assicura la coerenza geometrica tra percezione e realtà fisica, migliorando la ricostruzione e l'interpretazione dell'ambiente. In questa prospettiva, l'evoluzione delle tecniche di calibrazione, dai pattern geometrici tradizionali ai metodi automatizzati basati su forme semplici o ottimizzazione numerica, consente di ottenere sistemi di visione più affidabili, anche in scenari dinamici con telecamere mobili.

2.5 Digital twin dei sistemi robotici

Un digital twin è una replica digitale di un sistema fisico reale che non si limita a riprodurne la geometria o il comportamento statico, ma integra informazioni dinamiche, parametri operativi e dati provenienti dal sistema reale. Il modello virtuale e il sistema reale interagiscono attraverso una connessione bidirezionale, che consente di monitorare, simulare e ottimizzare i processi con elevata accuratezza [19].

Nel contesto robotico, il digital twin riveste un ruolo fondamentale, poiché consente la rappresentazione, la gestione e il controllo dell'ambiente operativo. La stazione robotizzata viene modellata digitalmente nella sua completezza, includendo il manipolatore, le attrezzature, gli ostacoli e le aree di lavoro libere e accessibili. Questa rappresentazione consente di analizzare e pianificare le operazioni in condizioni controllate, prevedendo potenziali collisioni o inefficienze nella disposizione spaziale. A differenza di una semplice rappresentazione digitale o di una simulazione tradizionale, il digital twin è un modello interattivo che, oltre a ricevere dati da robot, sensori e telecamere, è in grado di influenzare direttamente il comportamento del sistema reale attraverso la pianificazione del moto e l'invio di comandi di controllo. Nel contesto di ROS 2, ad esempio, il digital twin sfrutta le point cloud e le mappe di occupazione 3D per il rilevamento degli ostacoli e la pianificazione di traiettorie sicure, successivamente convertite in comandi di movimento inviati al robot, chiudendo così il ciclo di interazione tra mondo virtuale e fisico. Allo stesso tempo, nell'ambiente di simulazione (ad esempio in RViz) è possibile visualizzare e modificare parametri di controllo quali la velocità dei giunti, la forza di presa del gripper o la configurazione finale del braccio.

Il digital twin è costituito da un insieme di componenti statiche e dinamiche che, interagendo tra loro, ne determinano la struttura complessiva e il funzionamento. Tra le componenti statiche rientra il modello del robot, descritto mediante file URDF (Unified Robot Description Format), che ne specifica la geometria, la struttura cinematica e le proprietà fisiche. All'interno dello stesso modello possono essere integrati

anche gli ostacoli permanenti e le attrezzature della stazione robotizzata, rappresentati da modelli CAD importati come mesh nel file URDF, al fine di riprodurre con fedeltà la forma e la disposizione degli oggetti. Pur appartenendo alla parte strutturale del modello, il robot e i componenti aggiuntivi collegati assumono un comportamento dinamico, poiché la loro posizione e orientamento vengono aggiornati in funzione del movimento del sistema reale. La componente dinamica del digital twin è invece costituita dalla mappatura tridimensionale aggiornata continuamente, ottenuta attraverso l'elaborazione dei dati provenienti dai sensori di visione. Tali informazioni vengono convertite in mappe di occupazione 3D, come le OctoMap (vedi Capitolo successivo), che consentono di rilevare la presenza di ostacoli variabili o oggetti in movimento. Gli elementi così individuati vengono interpretati come oggetti di collisione all'interno del modello digitale, permettendo una pianificazione del moto sicura, adattiva e coerente con lo stato dell'ambiente.

Nel paragrafo seguente verrà approfondito il processo di mappatura 3D, illustrando le metodologie e le tecnologie impiegate per la costruzione e l'aggiornamento continuo della rappresentazione della stazione robotizzata.

2.5.1 Mappatura 3D

I sensori RGB-D generano una rappresentazione tridimensionale dell'ambiente sotto forma di point cloud, un insieme di punti a cui sono associate le coordinate spaziali (x, y, z). Grazie ai parametri ottenuti dalla calibrazione estrinseca della telecamera, la mappa di profondità viene convertita in una point cloud riferita al sistema di coordinate del sensore, ottenendo una mappa 3D accurata e densa della scena. Tuttavia, le point cloud grezze possono contenere rumore, punti errati (outlier) o dati ridondanti. Per questo motivo, prima dell'utilizzo nelle fasi successive, vengono generalmente sottoposte a operazioni di filtraggio ed elaborazione, come la rimozione dei punti isolati, la riduzione della densità o l'eliminazione dei punti appartenenti a regioni specifiche.

Dopo questa fase, le point cloud elaborate vengono impiegate per costruire mappe di occupazione tridimensionali tramite OctoMap [20], una libreria open-source che implementa un framework di mappatura probabilistica 3D basato su strutture dati gerarchiche (octree), in grado di rappresentare l'ambiente in modo efficiente sia in termini di memoria sia di aggiornamento dati. L'octree è una struttura dati ad albero che suddivide ricorsivamente lo spazio tridimensionale in celle cubiche, dette voxel, ciascuna delle quali rappresenta una porzione dello spazio che può essere ulteriormente suddivisa in otto sotto-regioni (da cui il termine octree). La Figura 2.7 mostra una rappresentazione schematica di tale suddivisione, che può proseguire fino a raggiungere il livello di dettaglio desiderato o imposto dalla densità dei dati acquisiti. In questo modo, le aree della mappa più complesse e ricche di dettagli vengono descritte mediante voxel più piccoli e numerosi, mentre le zone uniformi o prive di particolari sono rappresentate in modo più grossolano, garantendo un uso

bilanciato delle risorse computazionali e di memoria. La Figura 2.8 mostra l'effetto della variazione della dimensione dei voxel sulla rappresentazione della mappa.

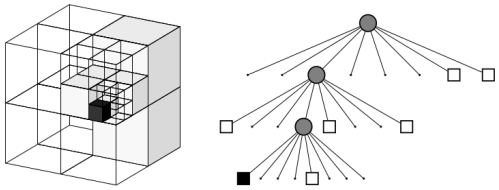
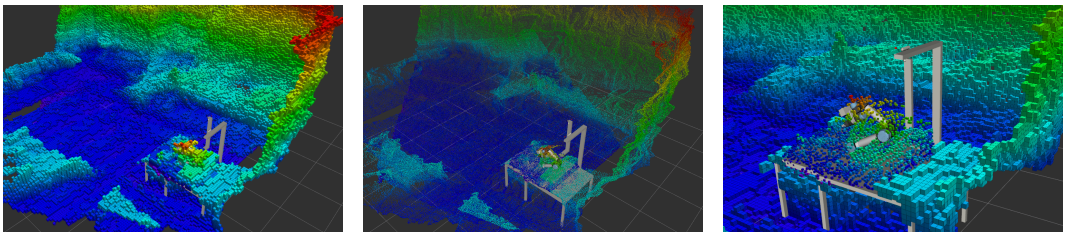


Figura 2.7: Rappresentazione schematica della suddivisione di un voxel.



(a) (b) (c)

Figura 2.8: Esempi di diverse risoluzioni dei voxel, nella Figura (a) il voxel ha lato 5cm, in (b) 2cm, mentre in (c) la dimensione varia in base alla zona.

Alla fase di discretizzazione spaziale, OctoMap affianca un modello probabilistico che assegna a ciascun voxel uno stato: occupato, libero o sconosciuto, sulla base delle point cloud acquisite dalle telecamere RGB-D. Come si può osservare nelle mappe riportate in Figura 2.8, solo le regioni contrassegnate come occupate vengono visualizzate, mentre quelle libere o sconosciute non sono rappresentate. La stima dello stato di ciascun voxel avviene tramite una procedura di ray-casting, che traccia un raggio dall'origine del sensore verso ogni punto misurato. Lungo ciascun raggio, le celle attraversate vengono marcate come libere, poiché il fascio le percorre senza incontrare ostacoli, mentre quella corrispondente al punto terminale è marcata come occupata. Le regioni non attraversate da alcun raggio rimangono implicitamente nello stato iniziale, ovvero sconosciuto, finché non vengono coinvolte da nuove misurazioni. Inoltre, OctoMap consente di arricchire la rappresentazione spaziale memorizzando informazioni aggiuntive nei voxel, come caratteristiche cromatiche o geometriche. In questo modo, è possibile generare mappe tridimensionali a colori, in cui ciascun elemento assume una tonalità in funzione di una grandezza geometrica, ad esempio la coordinata z rispetto al frame di riferimento del sensore, come mostrato in Figura 2.8. Questa estensione consente lo sviluppo di numerose applicazioni avanzate, come la classificazione semantica dell'ambiente, la localizzazione basata sull'aspetto e la costruzione di digital twin realistici.

Nel caso specifico di questo progetto, OctoMap è impiegato generare una rappresentazione 3D accurata dell'ambiente operativo del robot collaborativo, che funge da base per la pianificazione del moto e la modellazione del digital twin. La mappa

volumetrica viene integrata nel framework MoveIt (si veda il Capitolo 3.2.1), che interpreta i voxel occupati come oggetti di collisione, consentendo al pianificatore di calcolare traiettorie sicure in grado di evitare gli ostacoli.

In conclusione, la mappatura tridimensionale dell'ambiente rappresenta un elemento fondamentale nei contesti di robotica collaborativa, poiché consente al robot di percepire e interagire in modo sicuro con lo spazio circostante. In tale ambito, OctoMap è una soluzione estremamente efficace, grazie alla sua capacità di combinare accuratezza geometrica ed efficienza computazionale.

Capitolo 3

Setup sperimentale

Il presente Capitolo descrive l'architettura del sistema sperimentale sviluppato per la validazione del digital twin e della pianificazione del moto del robot collaborativo. Sono illustrate nel dettaglio la componente hardware e software, insieme alla configurazione di laboratorio, che descrive l'organizzazione fisica della stazione robotizzata e l'integrazione dei diversi moduli necessari per l'esecuzione dei test sperimentali.

Tutte le attività di sviluppo, integrazione e sperimentazione sono state condotte presso il Laboratorio di Meccatronica e Robotica (MIR) del Dipartimento di Ingegneria Industriale e Scienze Matematiche (DIISM) dell'Università Politecnica delle Marche, che ha fornito le risorse hardware e software necessarie.

3.1 Architettura hardware

Questa sezione presenta l'architettura hardware del sistema robotizzato sviluppato illustrando tutti i dispositivi che lo compongono. In particolare, il robot collaborativo UR5e, il gripper Hand-E della Robotiq, il sistema di visione basato su telecamere RGB-D Orbbec, nonché i supporti meccanici e le unità di connessione necessari all'integrazione del sistema.

3.1.1 Robot

Il robot collaborativo UR5e, mostrato in Figura 3.1, è uno dei modelli di punta della gamma e-Series sviluppata da Universal Robots [21]. Si tratta di un manipolatore antropomorfo a sei gradi di libertà, caratterizzato da altrettanti giunti rotoidali che gli conferiscono un'elevata flessibilità e la capacità di raggiungere pose complesse nello spazio tridimensionale. Tali caratteristiche rendono il robot adatto a un'ampia varietà di applicazioni industriali e di ricerca, come assemblaggio, movimentazione di componenti di piccole e medie dimensioni, operazioni di controllo qualità e collaudo. Progettati secondo il principio Power and Force Limited (PFL) tipico dei robot collaborativi, l'UR5e è dotato di un design ergonomico privo di spigoli vivi e di sensori di forza e coppia integrati, che consentono un'interazione sicura con l'operatore attraverso il controllo della forza e della velocità di movimento. L'impiego del robot in applicazioni collaborative, in cui l'attuatore finale o il pezzo in lavorazione possono entrare in contatto con l'operatore, è possibile solo a condizione che venga effettuata

una valutazione del rischio specifica dell'applicazione, poiché la sola certificazione del robot, conforme agli standard EN ISO 13849-1 (Cat. 3, PL d) e ISO 10218-1 rilasciata da TÜV Nord, non è sufficiente a garantirne la sicurezza complessiva [22].

L'UR5e è dotato di un driver ufficiale fornito da Universal Robots, *UNIVERSAL ROBOTS ROS 2 Driver* [23], che permette il controllo e il monitoraggio del cobot attraverso gli strumenti di pianificazione, simulazione e visualizzazione dell'ecosistema ROS 2, offrendo un'interfaccia continuamente aggiornata per lo sviluppo di applicazioni robotiche avanzate.

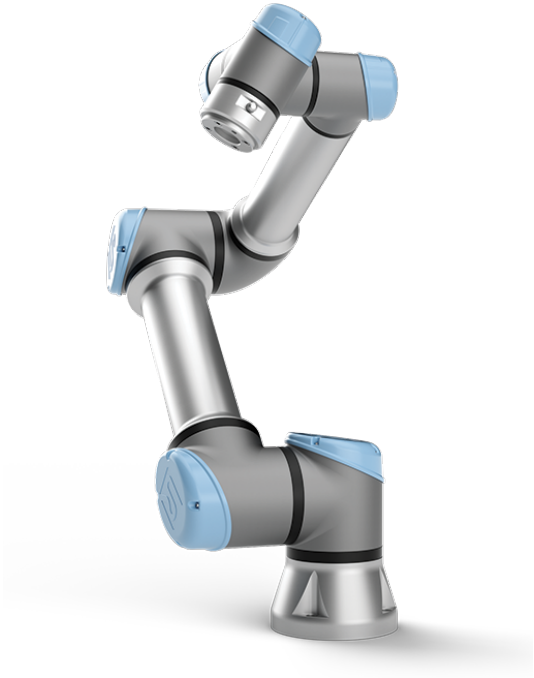


Figura 3.1: Robot UR5e.

Il cuore operativo dell'UR5e è il Control Box e-Series (Figura 3.2), che gestisce l'alimentazione, elabora i dati acquisiti dai sensori del robot e coordina l'esecuzione dei comandi provenienti dalle diverse interfacce operative. La comunicazione con i sistemi esterni è resa possibile dalle connessioni Ethernet e dalle porte I/O digitali e analogiche integrate nel Control Box.

Grazie all'integrazione con ROS 2, l'unità di controllo elabora i comandi di movimento generati all'interno del framework, trasmessi tramite topic o azioni, traducendoli in istruzioni compatibili con l'architettura interna del robot e restituendo in tempo reale lo stato dei giunti e dei sensori.

Un'ulteriore interfaccia di controllo è rappresentata dal Teach Pendant (Figura 3.3), una console collegata via cavo al Control Box, che costituisce lo strumento principale per la programmazione e il monitoraggio del robot, integrando anche il pulsante di arresto di emergenza. Grazie al software *PolyScope* integrato, l'operatore può interagire con il sistema in modo intuitivo visualizzando in tempo reale le grandezze

interne come posizioni, velocità, accelerazioni e stato dei motori. Inoltre, può configurare i principali parametri di sicurezza, tra cui limiti di velocità e accelerazione, aree di lavoro consentite e barriere virtuali. Il Teach Pendant consente anche il controllo diretto degli end-effector, ad esempio gripper o utensili, e supporta la programmazione per dimostrazione diretta, che permette di guidare manualmente il braccio robotico registrando traiettorie complesse senza la necessità di scrivere codice, con feedback immediato sullo stato operativo del robot.



Figura 3.2: Control box UR5e.



Figura 3.3: Teach Pendant UR5e.

Le principali caratteristiche tecniche dell'UR5e sono riassunte nella Tabella 3.1:

Caratteristica	**Valore**
Carico utile	5 kg
Sbraccio	850 mm
Peso	20.7 kg
Ingombro a terra	ϕ 149 mm
Classificazione IP	IP 54
Rumorosità	$<$ 65 dB
Temperatura di esercizio	0 $-$50 °C
Range di movimento dei giunti	$\pm$ 360°
Velocità massima dei giunti	180 °/s
Velocità massima dell'end-effector	circa 1 m/s
Frequenza di aggiornamento del sistema	500 Hz
Accuratezza sensore di forza/coppia	4 N
Ripetibilità di posa	$\pm$ 0.03 mm

Tabella 3.1: Caratteristiche UR5e.

La Figura 3.4 mostra invece la curva di carico dell'UR5e in funzione della distanza dell'end-effector dalla base del robot, utile per valutare come la capacità di carico vari al variare della posizione operativa.

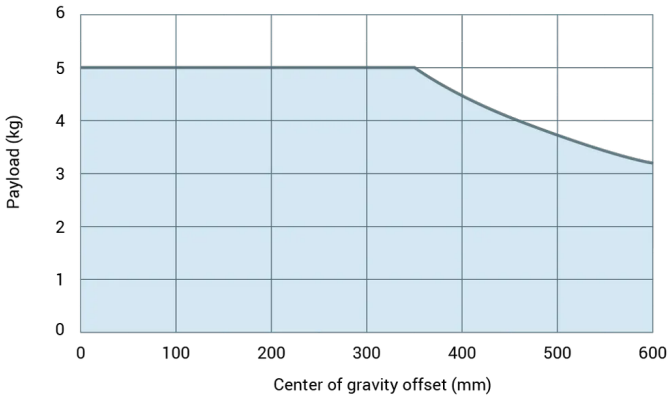


Figura 3.4: Curva di carico UR5e.

End-effector

Per questa applicazione, all'UR5e è stato installato come end-effector il gripper Hand-E (Figura 3.5), sviluppato da Robotiq, un dispositivo versatile progettato specificatamente per applicazioni di robotica collaborativa [24]. Si tratta di un sistema di presa meccanico a due dita parallele, con una corsa di 50 mm e un'elevata precisione di posizionamento, caratteristiche che lo rendono adatto a operazioni di assemblaggio e manipolazione di oggetti di piccole e medie dimensioni. Inoltre, i due

polpastrelli sono intercambiabili, permettendo di selezionare la forma più idonea in funzione della geometria degli oggetti da afferrare.



Figura 3.5: Gripper Robotiq Hand-E.

L'Hand-E, progettato secondo una logica Plug&Play, garantisce un'installazione rapida sulla flangia del robot e una messa in funzione immediata, risultando adatto a contesti collaborativi che richiedono elevata semplicità e adattabilità operativa.

Nel caso specifico, tuttavia, il terminale non è montato direttamente sulla flangia del manipolatore: tra i due elementi è stato interposto un supporto, realizzato tramite stampa 3D e già disponibile presso il laboratorio MIR, in quanto utilizzato e collaudato in attività sperimentali precedenti. Il supporto integra il sistema di visione, fornendo l'alloggio per una telecamera Orbbec Gemini 2L, impiegata come hand-eye camera, garantendo al contempo il corretto fissaggio del gripper senza compromettere la funzionalità complessiva. Il gripper risulta pienamente integrato nell'ecosistema Universal Robots ed è programmabile direttamente tramite la Teach Pendant, attraverso cui è possibile configurare parametri operativi quali posizione, velocità e forza di presa.

Le principali caratteristiche tecniche dell'Hand-E sono riportate nella Tabella 3.2:

Caratteristica	**Valore**
Corsa	50 mm
Carico utile	7 kg
Peso	1 kg
Garanzia	5000000 cicli
Forza di presa	da 20 a 185 N
Velocità di chiusura	da 20 a 150 mm/s
Ripetibilità	0.025 mm
Grado di protezione	IP 67
Temperatura di esercizio	da -10 a 50 °C

Tabella 3.2: Caratteristiche Hand-E.

Cinematica UR5e

Per l'analisi cinematica dell'UR5e è stata adottata la convenzione di Denavit-Hartenberg, descritta nel Capitolo 2.2. In Figura 3.6 è riportata la rappresentazione schematica del manipolatore, con il posizionamento dei sistemi di riferimento associati a ciascun link, mentre la Tabella 3.3 riporta i valori dei corrispondenti parametri di Denavit-Hartenberg. Sia la Figura 3.6, sia i valori dei parametri sono tratti dalla documentazione ufficiale fornita da Universal Robots [25].

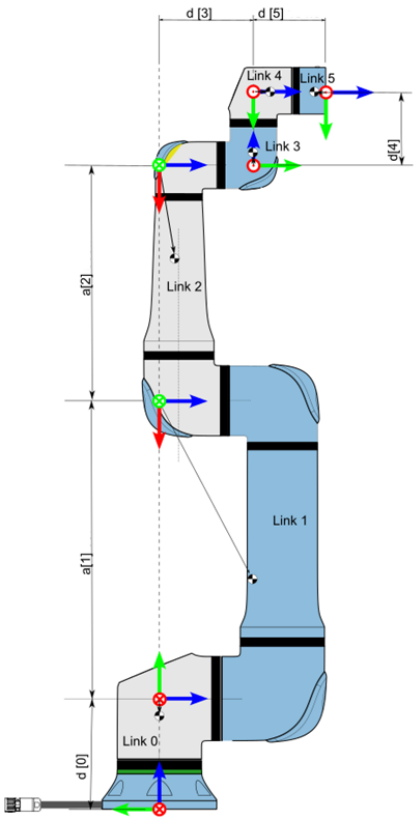


Figura 3.6: Schema cinematico UR5e.

	a [m]	d [m]	alpha [rad]
Joint 1	0	0.1625	$\pi/2$
Joint 2	-0.425	0	0
Joint 3	-0.3922	0	0
Joint 4	0	0.1333	$\pi/2$
Joint 5	0	0.0997	$-\pi/2$
Joint 6	0	0.0996	0

Tabella 3.3: Parametri DH UR5e.

3.1.2 Sistema di visione

Il sistema di visione è composto da tre telecamere RGB-D: due Orbbec Gemini 335L e una Orbbec Gemini 2L, mostrate rispettivamente in Figura 3.7 e Figura 3.8, [26] [27].



Figura 3.7: Orbbec Gemini 335L.



Figura 3.8: Orbbec Gemini 2L.

L'impiego di sensori RGB-D, come approfondito nel Capitolo 2.4, consente di acquisire simultaneamente informazioni cromatiche (RGB) e di profondità (Depth), fornendo una rappresentazione tridimensionale accurata della scena, fondamentale per le operazioni successive di mappatura e manipolazione degli oggetti.

Le telecamere Orbbec della serie Gemini si basano su piattaforme hardware e software molto simili tra loro, progettate per garantire prestazioni affidabili in un'ampia varietà di applicazioni di visione artificiale e robotica.

Il cuore di entrambe è rappresentato dal processore specializzato ASIC MX6800, sviluppato da Orbbec per l'elaborazione in tempo reale dei dati di profondità e per la gestione interna dell'allineamento spaziale tra l'immagine RGB e la mappa di profondità. Questo approccio riduce significativamente la latenza e limita la dipendenza dalla potenza di calcolo del computer host, rendendo il sistema complessivamente più leggero ed efficiente. A completare l'hardware, i sensori integrano una IMU interna, composta da accelerometro e giroscopio, utile per il tracciamento del movimento in applicazioni di robotica mobile, e un sistema di acquisizione stereo combinato (attivo e passivo), che li rende efficienti sia in ambienti interni che esterni, anche in condizioni di illuminazione molto variabili: dal buio completo alla piena luce solare. Per la comunicazione con ambienti di sviluppo come ROS 2, Orbbec fornisce un Software Development Kit (SDK [28]) che garantisce un accesso completo e personalizzabile ai

dati e alle funzioni delle telecamere. L'integrazione con ROS 2 consente di gestire i flussi di dati provenienti dai sensori, applicare filtri di post-elaborazione e combinare le informazioni con quelle di altre sorgenti, come ulteriori sensori, strumenti di visualizzazione o robot. L'SDK include inoltre strumenti per la configurazione e la calibrazione del dispositivo, permettendo di regolare parametri come risoluzione, esposizione e sincronizzazione temporale (*timestamp*).

Le principali differenze tra i due modelli risultano evidenti dalla Tabella 3.4, in cui sono riportate e confrontate le rispettive caratteristiche tecniche. In sintesi, la Gemini 335L si distingue per un range di profondità più ampio e per una frequenza di acquisizione superiore delle immagini RGB, mentre la Gemini 2L è ottimizzata per applicazioni a corto raggio, offrendo una maggiore precisione nelle misure ravvicinate grazie al sensore di prossimità integrato, il quale elimina i punti ciechi all'interno dell'intero intervallo operativo (0 - 10 m).

Caratteristica	**Gemini 335L**	**Gemini 2L**
Tecnologia di acquisizione profondità	Stereo	Active stereo IR
Lunghezza d'onda supportata	850 nm	850 nm
Intervallo di profondità	0.17 - 20 m (range ottimale: 0.5 - 6 m)	0.2 - 10 m
Risoluzione profondità / FPS	fino a 1280x800 @ 30fps	fino a 1280x800 @30fps
FOV profondità	H90° V65°	H91° V66°
Risoluzione RGB / FPS	fino a 1280x800 @60fps	fino a 1280x800 @30fps
FOV RGB	H94° V68°	H94° V68°
Processore	Orbbec ASIC	Orbbec ASIC
Connessione dati / Alimentazione	USB 3.0 Type-C	USB 3.0 / USB 2.0
Condizioni operative	−10°/50°; Indoor/Outdoor; 5% - 90% RH (no condensa)	0°/40°; Indoor/Semi-Outdoor; 8% - 90% RH (no condensa)
Dimensioni (WxHxD)	124 mm x 29 mm x 27 mm	124 mm x 29 mm x 26 mm
Peso	133 g	152 g

Tabella 3.4: Caratteristiche telecamere RGB-D Orbbec.

In base a tali caratteristiche, le due tipologie di sensori sono state collocate in posizioni differenti: le Gemini 335L sono installate in posizione sopraelevata, con

l'obiettivo di acquisire una visione d'insieme dell'ambiente e monitorare l'intero spazio operativo del robot, mentre la Gemini 2L, montata sul polso del manipolatore, è impiegata come *hand-eye camera*, dedicata alla percezione della scena posta di fronte all'end-effector e solidale al movimento del braccio.

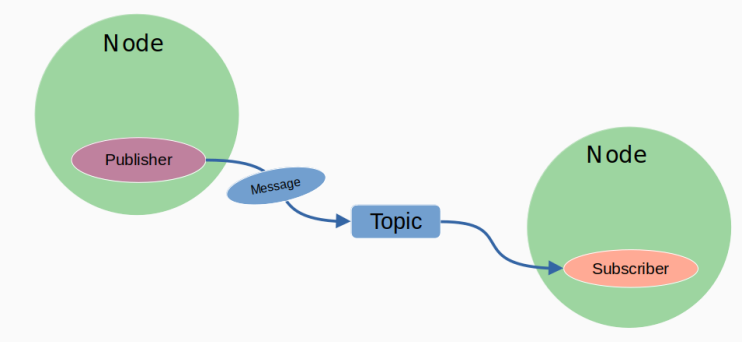
3.2 Architettura Software

3.2.1 ROS 2

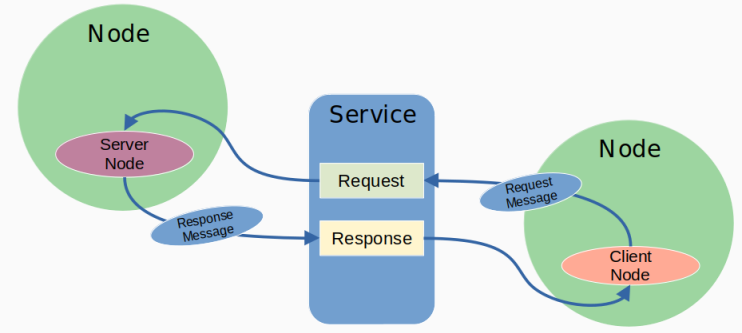
L'intera architettura software sviluppata è basata su ROS 2 (Robot Operating System 2), nella versione Humble, che costituisce la piattaforma principale per il controllo del robot, per l'acquisizione e l'elaborazione delle immagini, e per la comunicazione tra i diversi software e dispositivi coinvolti [29]. ROS 2 rappresenta l'evoluzione del framework open-source ROS, nato per facilitare lo sviluppo di applicazioni robotiche attraverso un insieme di strumenti e librerie. Rispetto alla prima versione, introduce miglioramenti sostanziali, tra cui l'integrazione con linguaggi moderni (C++ e Python), la compatibilità multipiattaforma (Linux, macOS, Windows e altri sistemi), una maggiore sicurezza e la capacità di operare in ambienti real-time. La natura open-source e il supporto di una vasta community ne hanno favorito la diffusione sia in ambito accademico che industriale, rendendolo oggi uno degli strumenti più utilizzati per la progettazione di sistemi robotici complessi.

Grazie alla sua architettura modulare, ROS 2 consente di suddividere un'applicazione in più nodi, unità fondamentali di elaborazione, ciascuna delle quali svolge un compito specifico, come l'acquisizione dei dati da un sensore, l'elaborazione delle informazioni o il controllo di un attuatore. Lo scambio di informazioni tra i nodi avviene principalmente in tre modalità schematizzate in Figura 3.9:

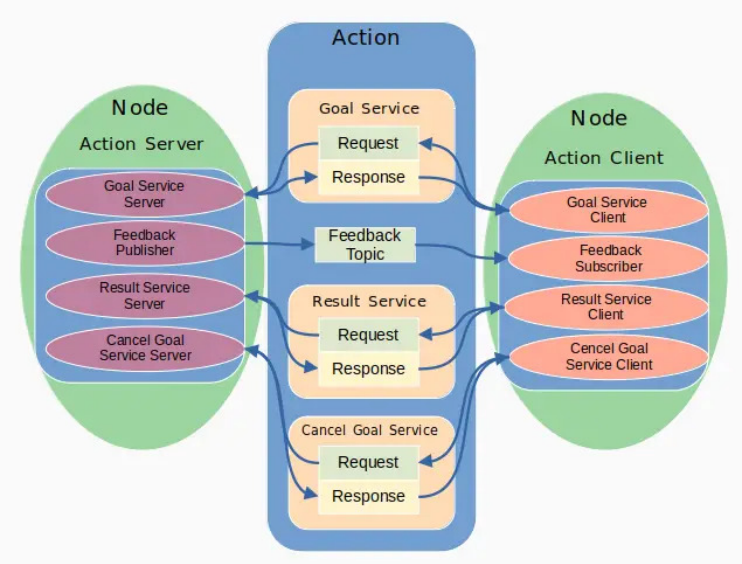
- **Topic**: consentono una comunicazione continua e asincrona secondo il modello *publisher-subscriber*: un nodo (*publisher*) invia messaggi su un canale nominato e uno o più nodi iscritti a quel canale (*subscriber*) li ricevono in tempo reale.
- **Servizi**: implementano una comunicazione *request-response*: un nodo client invia una richiesta a un nodo server, che elabora la richiesta e restituisce una risposta, utile per operazioni sincrone e rapide.
- **Azioni**: estendono il modello dei servizi per gestire operazioni di lunga durata, permettendo di ricevere aggiornamenti sullo stato dell'esecuzione e di interrompere il processo se necessario.



(a)



(b)



(c)

Figura 3.9: Modalità di comunicazione tra i nodi in ROS 2: (a) Topic, (b) Servizi, (c) Azioni.

Questa architettura rende il framework particolarmente adatto allo sviluppo di progetti complessi che richiedono l'integrazione di numerosi moduli software e hardware. Oltre alla funzionalità di comunicazione, ROS 2 mette a disposizione una serie di strumenti dedicati alla visualizzazione, al monitoraggio e al debug del sistema, tra

cui RViz, utilizzato in questa attività di ricerca per la rappresentazione e l'analisi dei dati in tempo reale, come descritto nel dettaglio nella sezione successiva.

Nell'architettura software sviluppata, ROS 2 gestisce la comunicazione e il coordinamento tra il cobot UR5e, le telecamere RGB-D, l'ambiente virtuale di simulazione e i vari nodi impiegati per l'elaborazione dei dati e la pianificazione dei movimenti. Grazie ai driver ufficiali Universal Robots, ROS 2 mette in comunicazione il Control Box del robot con l'ambiente software, pubblicando in tempo reale le informazioni sullo stato del manipolatore e inviando i comandi di movimento sia da codice che tramite l'interfaccia grafica. Parallelamente, gestisce l'acquisizione e l'elaborazione delle point cloud, integrandoli con i modelli del sistema per costruire un digital twin interattivo in RViz dell'intero sistema robotizzato. Su questa base si integra MoveIt, uno dei principali framework open-source per la pianificazione e l'esecuzione di movimenti robotici integrati in ROS 2 (approfondito nella sezione dedicata), che sfrutta le informazioni del digital twin per generare traiettorie sicure e ottimizzate.

In conclusione, ROS 2 non si limita a fornire un mezzo di comunicazione tra i vari moduli, ma costituisce l'elemento unificante dell'intero sistema, integrando componenti hardware e software in un'unica piattaforma flessibile, consentendo lo sviluppo di applicazioni robotiche complesse in modo modulare ed estensibile.

RViz

Integrato nell'ecosistema ROS 2, RViz rappresenta lo strumento principale per la visualizzazione tridimensionale e l'interazione con il sistema robotico. Consente di monitorare in tempo reale i dati pubblicati nei topic, offrendo una visione completa dello stato del robot, dei sensori e dell'ambiente circostante. Attraverso la sua interfaccia grafica, RViz fornisce un ambiente interattivo in cui è possibile integrare, analizzare e controllare le informazioni provenienti dai vari nodi ROS 2, oltre a inviare comandi e messaggi ai componenti del sistema.

Una schermata tipica di RViz è mostrata in Figura 3.10, in cui è visualizzata una point cloud acquisita dalla Orbbec Gemini 335L.

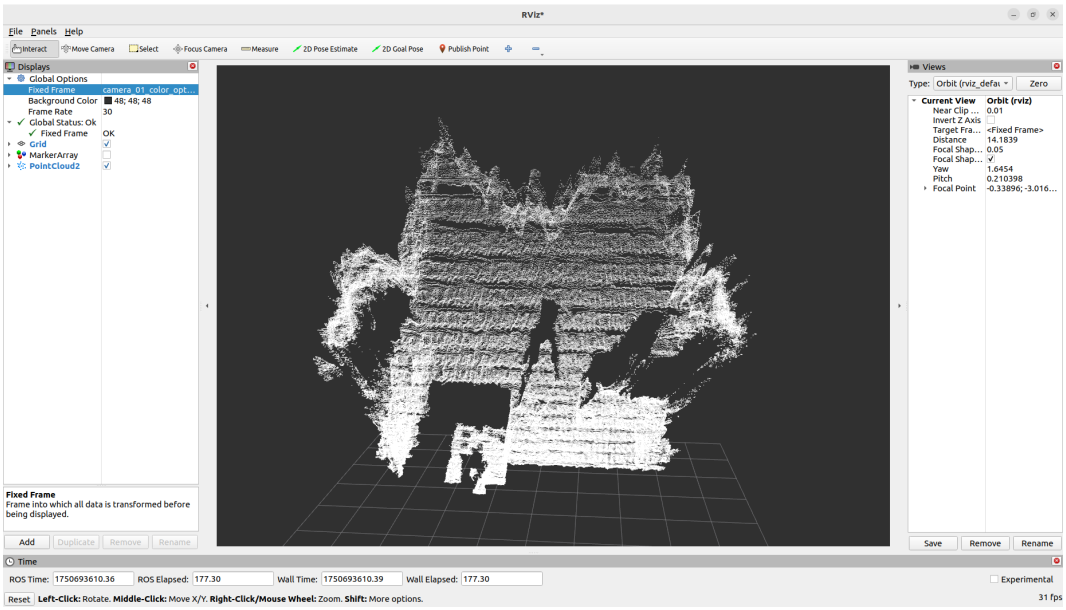


Figura 3.10: Schermata RViz.

Nell'applicazione sviluppata, RViz è stato utilizzato per visualizzare le immagini RGB, le point cloud, le mappe 3D generate con OctoMap e il modello del robot, descritto mediante file dedicati. L'integrazione di questi elementi in un unico ambiente consente di ottenere una rappresentazione completa e continuamente aggiornata dello stato del manipolatore e dell'ambiente operativo, evidenziando le aree di collisione. Inoltre, la visualizzazione dei frame di riferimento permette di verificare la corretta gestione delle trasformazioni tra i vari sistemi di coordinate. Grazie all'integrazione con MoveIt, l'ambiente interattivo di RViz consente di definire in modo intuitivo le pose dell'end-effector e le configurazioni del manipolatore, gestendo la pianificazione, la simulazione e l'esecuzione delle traiettorie direttamente dall'interfaccia grafica.

MoveIt

Il controllo del manipolatore e la pianificazione delle traiettorie sono gestiti da MoveIt, una piattaforma integrata in ROS 2 che combina algoritmi di pianificazione, moduli per la gestione delle collisioni e strumenti di simulazione [30]. Grazie alla sua struttura flessibile e modulare, MoveIt può essere adattato a differenti contesti applicativi della robotica manipolativa. Include inoltre un plugin dedicato in RViz, denominato *Motion Planning*, attraverso il quale è possibile configurare le pose obiettivo, simulare e avviare l'esecuzione delle traiettorie pianificate sul robot reale, direttamente dall'interfaccia grafica.

Un esempio dell'interfaccia grafica di MoveIt integrata in RViz è mostrato in Figura 3.11, dove sono evidenziate la configurazione iniziale (in verde), la configurazione obiettivo (in arancione) e lo stato attuale del robot.

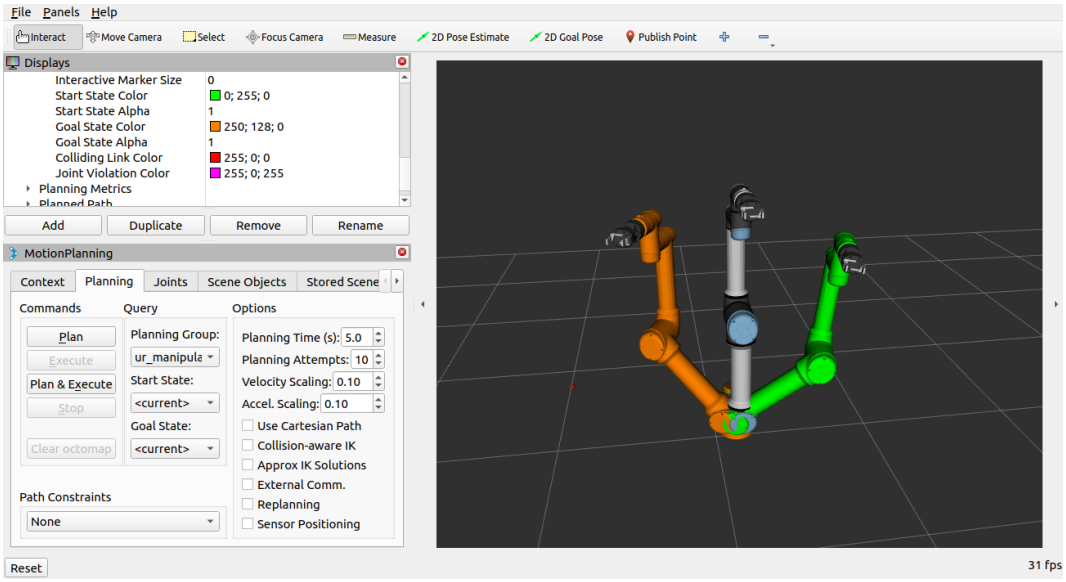
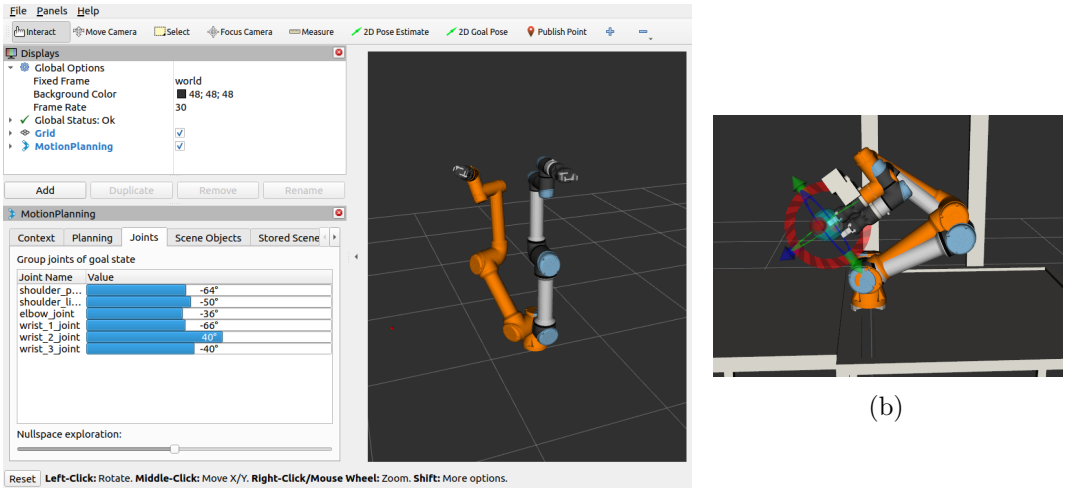


Figura 3.11: Interfaccia di MoveIt in RViz, in cui sono evidenziate la configurazione iniziale (in verde), la configurazione obiettivo (in arancione) e lo stato attuale del robot.

Alla base del funzionamento si trovano i moduli di cinematica inversa (IK), che determinano i valori dei giunti necessari a raggiungere una posizione e un orientamento desiderati del TCP (Tool Center Point), ovvero il punto più estremo dell'end effector, posizionato al centro e definito da un sistema di riferimento con asse z uscente dal terminale. MoveIt integra di default il solutore di cinematica inversa *Kinematics and Dynamics Library (KDL)*, ma consente anche l'utilizzo di solutori alternativi, come *TRAC-IK*, che offrono soluzioni ottimizzate in base a specifici criteri di prestazione. In questa applicazione, l'algoritmo *TRAC-IK* calcola configurazioni finali del manipolatore il più possibile prossime a quelle iniziali, migliorando la fluidità dei movimenti. Su questa base si innesta il processo di pianificazione delle traiettorie, che genera percorsi cinematicamente validi e privi di collisioni, nel rispetto dei vincoli geometrici e dinamici del robot. Gli obiettivi di movimento possono essere definiti come pose cartesiane del TCP oppure come configurazioni di giunti, in base alle esigenze operative. Ad esempio, come mostrato in Figura 3.12, in RViz è possibile specificare manualmente i valori dei singoli giunti, o impostare la posa del TCP tramite l'interactive marker con l'aggiornamento automatico della configurazioni di giunti.



(a)

(b)

Figura 3.12: Definizione della configurazione obiettivo: (a) impostazione dei valori dei singoli giunti; (b) impostazione della posa del TCP tramite l'interactive marker.

MoveIt integra, come libreria di pianificazione predefinita la Open Motion Planning Library (OMPL), approfondita nel Capitolo 2.3, offrendo la possibilità di scegliere tra diverse strategie di pianificazione in funzione della complessità e della velocità computazionale richiesta. Come per i solutori di cinematica inversa è possibile integrare planner alternativi, come *STOMP* o *CHOMP*, per adattarsi a scenari particolari o soddisfare specifici requisiti prestazionali.

La pianificazione è strettamente collegata alla gestione delle collisioni, componente fondamentale nella robotica collaborativa per garantire la sicurezza degli operatori e l'integrità dell'ambiente circostante. A tal fine, MoveIt mantiene un modello costantemente aggiornato dello spazio operativo, che integra sia di ostacoli statici, derivati da modelli CAD caricati nel modello di descrizione del robot (URDF), sia elementi dinamici, rilevati tramite sensori o mappe tridimensionali, consentendo ai planner di generare traiettorie ottimizzate e prive di collisioni.

Il coordinamento tra i vari moduli di MoveIt è gestito dal nodo ROS 2 `move_group`, che rappresenta l'interfaccia principale per la pianificazione e l'esecuzione dei movimenti sul robot reale. L'interazione con tale nodo può avvenire attraverso tre approcci principali, in funzione del linguaggio o dell'ambiente di lavoro adottato:

- in C++, tramite il pacchetto `move_group_interface`;
- in Python, mediante il pacchetto `moveit_commander`;
- tramite interfaccia grafica, utilizzando *RViz* e il modulo *Motion Planning*.

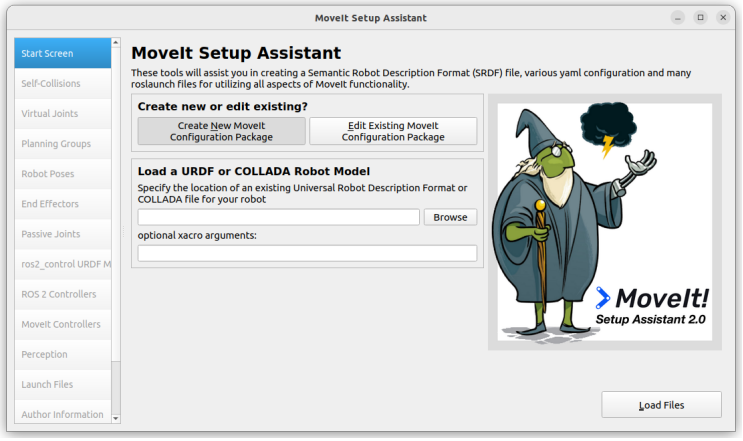
Il nodo `move_group` opera sulla base delle informazioni fornite tramite il *ROS Parameter Server*, il servizio di ROS 2 dedicato alla gestione e alla condivisione dei

parametri tra i nodi. I dati principali, che compongono il pacchetto di configurazione di MoveIt e risultano necessari al corretto funzionamento del robot, comprendono:

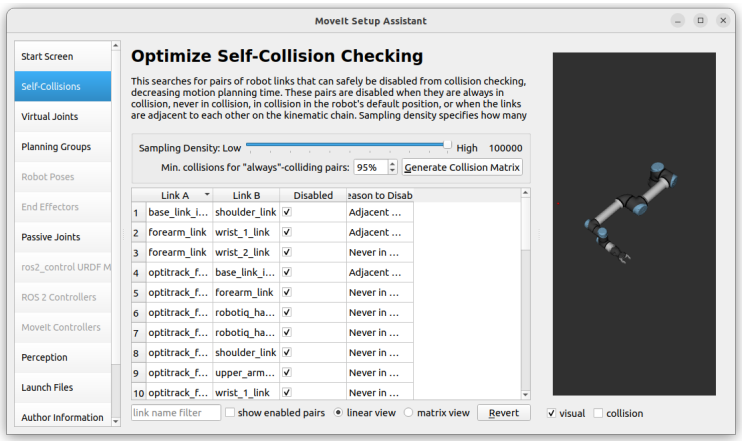
- il modello URDF (*Unified Robot Description Format*), disponibile nel parametro *robot_description*, che descrive la struttura e la cinematica del robot;
- il modello SRDF (*Semantic Robot Description Format*), associato al parametro *robot_description_semantic*, che definisce gli aspetti logico-funzionali del manipolatore, come i gruppi di giunti, le configurazioni predefinite e i vincoli di collisione;
- ulteriori configurazioni specifiche di MoveIt, comprendenti limiti dei giunti, parametri di pianificazione e impostazioni cinematiche e di percezione.

Sia il file SRDF sia i quelli contenti le suddette configurazioni (generalmente file YAML) vengono generalmente generati attraverso il *MoveIt Setup Assistant*, uno strumento grafico interattivo che guida l'utente nella definizione dei parametri necessari all'integrazione del robot all'interno dell'ambiente MoveIt.

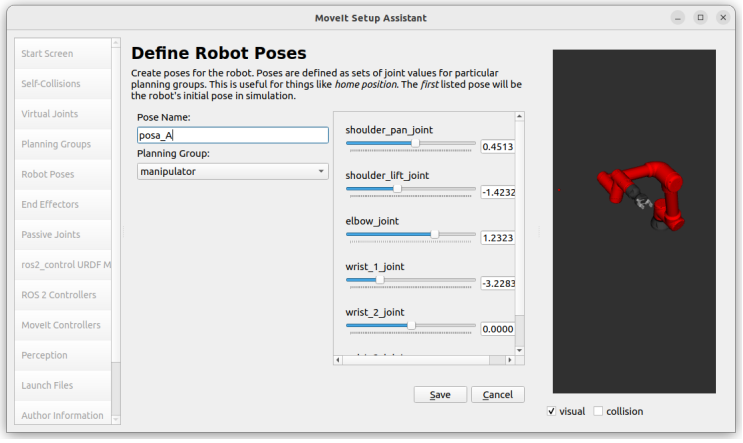
La Figura 3.13 mostra alcune schermate del *MoveIt Setup Assistant*. Nella prima viene importato il modello URDF, nella seconda si esclude dal controllo di collisione le coppie di link sempre in contatto o che non possono fisicamente interferire, così da ottimizzare e velocizzare la pianificazione, infine, nella terza sono definite le pose di default del robot.



(a)



(b)



(c)

Figura 3.13: Schermate *MoveIt Setup Assistant*: (a) importazione modello URDF, (b) ottimizzazione controllo delle collisioni, (c) definizione pose di default del robot.

3.3 Configurazione di laboratorio

Il sistema robotizzato sviluppato presso il laboratorio MIR riproduce una stazione di lavoro collaborativa, nella quale differenti componenti hardware e software cooperano per consentire la percezione e la mappatura dell'ambiente, la pianificazione e l'esecuzione di movimenti robotici sicuri ed efficienti.

La Figura 3.14 mostra la disposizione complessiva della postazione di lavoro, costituita dal cobot UR5e della *Universal Robots* (3.1.1), equipaggiato con il gripper adattivo Hand-E della *Robotiq* (3.1.1), che consente la manipolazione di oggetti di diversa forma e dimensione. La percezione dell'ambiente è affidata a tre telecamere RGB-D *Orbbec* (3.1.2), disposte in modo da garantire una copertura visiva completa e la ricostruzione tridimensionale dell'area operativa. Completa la configurazione il computer centrale, a cui sono connessi tutti i dispositivi, con installato ROS 2 per la gestione e l'elaborazione dei dati.



Figura 3.14: Configurazione di laboratorio.

L'UR5e è stato installato al centro di un tavolo alto 75 cm, largo 1 m e lungo 1.7 m, configurazione che garantisce un'area di lavoro ampia e accessibile su tutti i lati. Il robot è stato rigidamente ancorato al tavolo mediante bullonatura per assicurare stabilità anche durante movimenti rapidi o in presenza di carichi applicati al polso, evitando vibrazioni che potrebbero compromettere la precisione del movimento e la ripetibilità delle traiettorie. La disposizione del tavolo, progettata per riprodurre

un tipico scenario operativo come quello di un supermercato, prevede una zona di prelievo nella parte anteriore e una zona di deposito posteriore, posizionata su un piano rialzato di 21 cm che simula uno scaffale. Tale organizzazione, schematizzata in Figura 3.15, consente di riprodurre un ciclo di manipolazione completo di tipo pick and place.

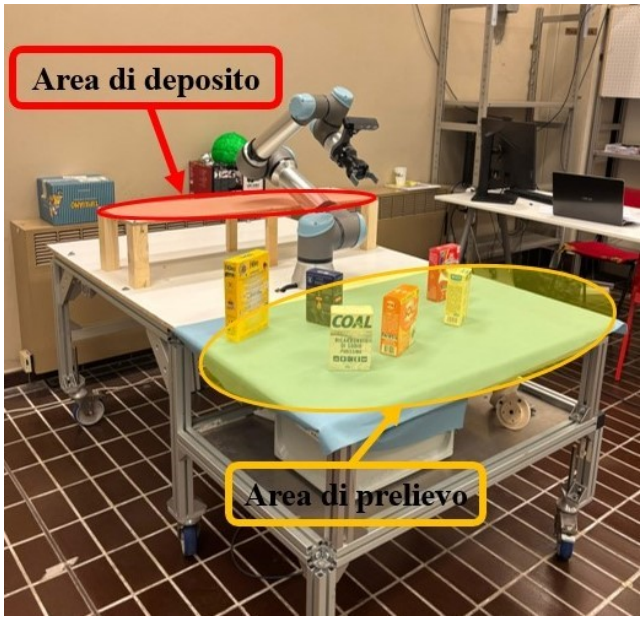


Figura 3.15: Organizzazione dell'area di lavoro.

Il gripper Hand-E è montato sulla flangia del robot tramite un supporto dedicato, progettato per ospitare anche la telecamera *Orbbec Gemini 2L*, come mostrato in Figura 3.16. Tale supporto, leggero e compatto, assicura sia il corretto ancoraggio dell'end-effector sia il preciso orientamento del sensore RGB-D verso l'area di lavoro, garantendo un'acquisizione dei dati stabile e accurata. Il gripper è connesso al robot mediante un cavo con connettore inserito nella parte inferiore della flangia, che fornisce sia l'alimentazione sia la comunicazione diretta con il sistema di controllo.

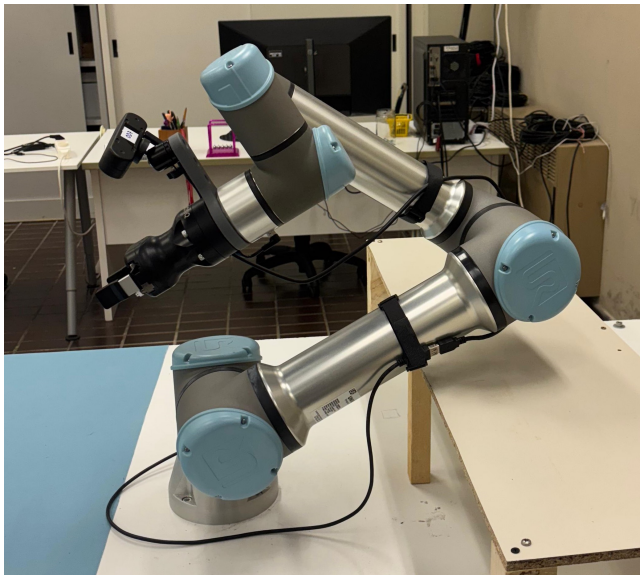


Figura 3.16: UR5e con installato il supporto per la hand-eye camera e il gripper Hand-E nella configurazione iniziale.

Il cobot è collegato tramite cablaggio diretto al proprio Control Box, che comunica con il computer centrale in modalità wireless, soluzione che offre una maggiore flessibilità nella disposizione della postazione e riduce l'ingombro dei cavi. Al Control Box è connesso anche il Teach Pendant, utilizzato per impostare i principali parametri di configurazione del robot, come il payload, calcolato considerando il peso complessivo dell'Hand-E, della telecamera Orbbec Gemini 2L e del relativo supporto. È infine definita la configurazione iniziale del manipolatore, in modo che la telecamera montata al polso (hand-eye camera) inquadri l'intera area di lavoro posta di fronte al robot, garantendo il corretto riconoscimento degli oggetti presenti sul tavolo.

Il sistema di visione è completato da due telecamere Orbbec Gemini 335L, installate nella parte superiore del laboratorio a 2.4 metri di altezza per garantire una visione complessiva dell'ambiente operativo e monitorare l'intera stazione robotizzata.

La disposizione dei sensori RGB-D, mostrata in Figura 3.14, è concepita per esaltare le loro caratteristiche, descritte nel Capitolo 3.1.2, infatti le Gemini 335L, più adatte alla visione a lungo raggio, sono collocate lontane dall'area di manipolazione, mentre la Gemini 2L, progettata per acquisizioni ravvicinate, è montata sul polso del robot. Questa configurazione ibrida, che combina telecamere fisse e una dinamica, consente di ottenere una percezione completa e coerente dell'intero ambiente, riducendo le zone d'ombra e migliorando la precisione nelle operazioni di presa.

Tutte le telecamere sono collegate al computer tramite cavi USB-C, prestando particolare attenzione al cablaggio della hand-eye camera, il cui cavo segue l'intera struttura del robot come visibile in Figura 3.16, così da evitare interferenze con i suoi movimenti.

Capitolo 4

Implementazione del digital twin del sistema robotizzato

L'implementazione del digital twin del sistema robotizzato consiste nella creazione di una rappresentazione virtuale fedele e costantemente aggiornata, ottenuta combinando elementi statici con informazioni dinamiche provenienti dai sensori. Il risultato è un modello digitale completo e interattivo, capace di riflettere in tempo reale lo stato del sistema fisico.

La prima fase riguarda la configurazione del simulatore digitale, in cui vengono importati il modello del robot e le mesh statiche delle attrezzature presenti nell'area operativa. Successivamente viene integrato il sistema di visione, la cui pipeline di calibrazione, elaborazione e fusione dei dati consente di rappresentare con precisione gli elementi dinamici che circondano il robot. A partire dalle point cloud generate da tale processo, il sistema genera mappe 3D tramite OctoMap, continuamente aggiornate, che permettono al digital twin di rilevare ostacoli variabili e supportare la pianificazione di traiettorie sicure.

La Figura 4.1 mostra il risultato finale, mentre le sezioni successive descrivono nel dettaglio le fasi della sua realizzazione.

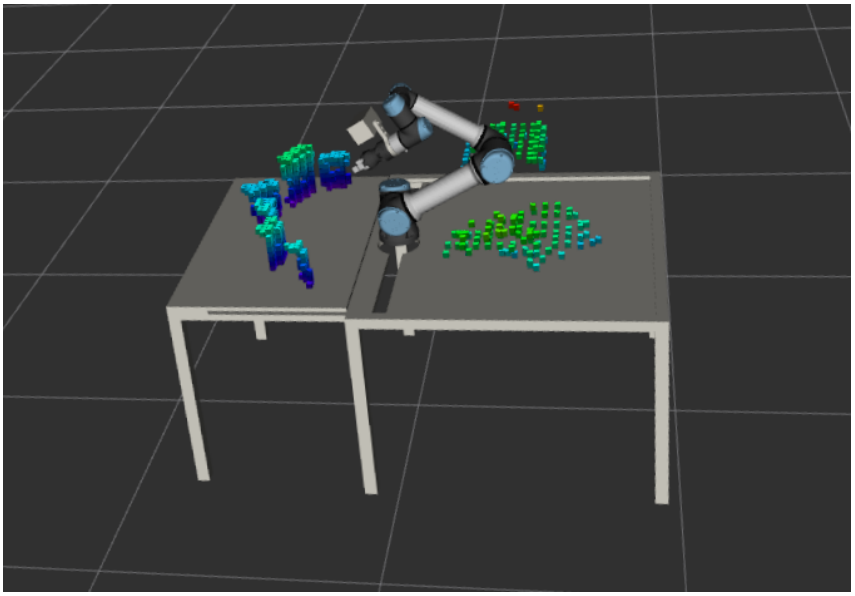


Figura 4.1: Digital twin del sistema robotizzato

4.1 Configurazione del simulatore digitale

La configurazione del simulatore digitale rappresenta la fase iniziale nella costruzione del digital twin del sistema robotizzato. In questa fase vengono importate in RViz le componenti statiche, ovvero quegli elementi che possono variare in posizione e orientamento all'interno della scena virtuale, ma non di forma.

Il primo elemento importato è il modello del robot UR5e, pubblicando sul topic *robot_description* il relativo file URDF. Si tratta di un documento XML, incluso nei driver ufficiali *universal_robots_driver*, che descrive la struttura meccanica e cinematica del manipolatore, specificando i link (corpi rigidi) e i giunti (connessioni cinematiche) che lo compongono. Per ciascun componente sono inoltre importate mesh statiche, derivate dai rispettivi modelli CAD (in formato *stl* o *dae*), che definiscono le geometrie di visualizzazione e di collisione. In questo modo, il modello virtuale riproduce fedelmente sia la struttura funzionale sia l'aspetto del robot reale, assumendone la stessa posa grazie all'aggiornamento continuo dello stato dei giunti pubblicati nel topic *joint_states*. I link che compongono l'UR5e sono:

- *base_link*
- *shoulder_link*
- *upper_arm_link*
- *forearm_link*
- *wrist_1_link*
- *wrist_2_link*

- *wrist_3_link*
- *flange*
- *tool0*

L'URDF originale è stato modificato introducendo un frame di riferimento globale denominato *world*, ottenuto traslando il frame *base_link* lungo l'asse z negativo di 75 cm, pari all'altezza del tavolo, in modo che risulti in corrispondenza del pavimento del laboratorio MIR e mantenga lo stesso orientamento del *base_link*. In RViz il frame *world* è selezionato come *Fixed Frame*, garantendo che la visualizzazione e la gestione delle trasformazioni avvengano rispetto a un riferimento statico, ancorato all'ambiente e indipendente dal manipolatore.

Nel file URDF sono stati integrati anche il gripper installato sul cobot e il supporto per la hand-eye camera su cui è fissato. Quest'ultimo è definito come un nuovo link rigidamente fissato alla flangia del robot, al quale è associata la mesh statica derivata dal modello CAD utilizzato per la stampa 3D del supporto (Figura 4.2).

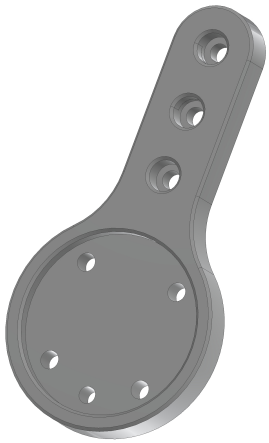


Figura 4.2: Modello CAD del supporto della Orbbec Gemini 2L.

La struttura cinematica e la geometria dell'Hand-E sono state importate dal repository pubblico GitHub *robotiq_hande_description* [31]. La catena di link e giunti che descrive il gripper è aggiunta al manipolatore mediante collegamento fisso tra il *robotiq_hande_coupler*, che rappresenta il primo link dell'Hand-E, e il supporto, mantenendo il corretto posizionamento e orientamento relativi osservati nel sistema reale. A ciascun link è associata una geometria di visualizzazione, coincidente con quella di collisione, descritta dai modelli CAD, forniti nel repository del gripper, importati come mesh statiche. Il gripper è composto dai seguenti link:

- *robotiq_hande_coupler*
- *robotiq_hande_link*

- *robotiq_hande_left_finger*
- *robotiq_hande_right_finger*
- *robotiq_hande_end*

Con l'installazione del supporto e dell'end-effector, è stato inoltre necessario aggiornare la posizione del TCP (Tool Center Point), originariamente posizionato in corrispondenza della flangia del robot con asse z uscente, applicando un offset lungo z per portarlo nel punto terminale del gripper. Al supporto è stato fissato un ulteriore link rappresentativo del sensore Orbbec Gemini 2L, la cui geometria è stata semplificata mediante un box che ne approssima l'ingombro, sufficiente ai fini della visualizzazione e del controllo delle collisioni. Infine, nell'URDF caricato in RViz è stato incluso anche il tavolo su cui è installato il robot, definito come un link a cui è associata una mesh statica derivata dal relativo modello CAD precedentemente realizzato per altre applicazioni di laboratorio.

In Figura 4.3 è mostrato il risultato della configurazione in RViz, ottenuta importando il file URDF dell'UR5e modificato e arricchito con le attrezzature accessorie presenti nella stazione robotizzata.

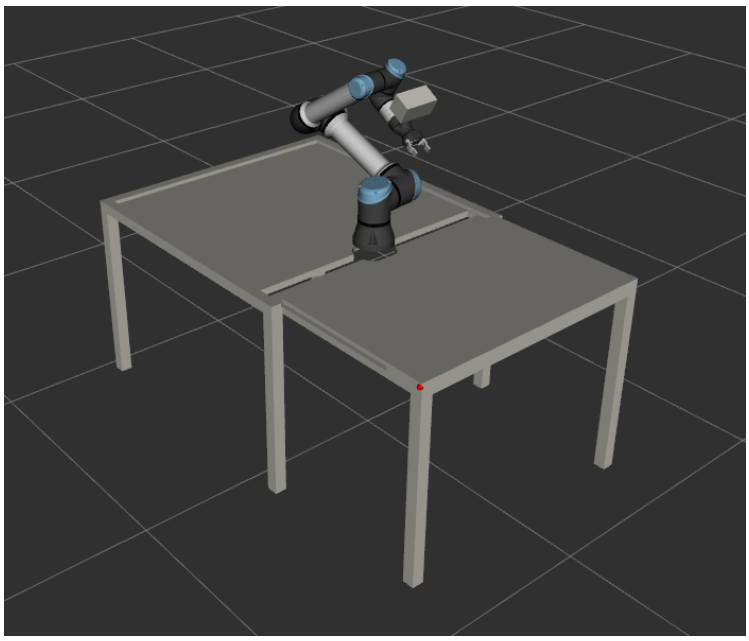


Figura 4.3: Configurazione di RViz.

4.2 Integrazione del sistema di visione

Dopo aver configurato il simulatore digitale, viene integrato il sistema di visione, necessario per la mappatura degli oggetti dinamici presenti nella scena così da consentire la pianificazione di traiettorie sicure e prive di collisioni. Il sistema di visione, descritto al Capitolo 3.1.2, deve rilevare sia gli articoli commerciali presenti nell'area di lavoro, sia l'eventuale intrusione di un operatore nella zona di movimentazione del robot.

Le point cloud acquisite dai tre sensori RGB-D installati nella stazione robotizzata vengono pubblicate nei rispettivi topic dai driver ufficiali Orbbec , insieme al frame di riferimento e al timestamp (tempo di acquisizione). Per integrare correttamente i dati di profondità nel simulatore digitale è necessaria la calibrazione estrinseca dei sensori, descritta nella sezione successiva, che consente di determinare con precisione la posizione e l'orientamento di ciascuna camera rispetto al frame di riferimento *world*. Va precisato che la calibrazione fornisce inizialmente la matrice di rototraslazione tra il frame del sensore e quello *base_link*. Tuttavia, poiché la trasformazione tra *base_link* e *world* è già definita nell'URDF, ROS 2 compone le due trasformazioni per ottenere quella diretta rispetto al sistema di riferimento globale. Le trasformazioni tra i frame dei sensori e il frame *base_link* vengono pubblicate tramite il pacchetto `tf2_ros`, che le rende costantemente accessibili agli altri nodi ROS 2 per la corretta gestione e visualizzazione delle point cloud. In questo modo, ogni punto acquisito viene visualizzato in RViz nella posizione corrispondente a quella effettiva nel mondo reale.

La calibrazione estrinseca risulta inoltre fondamentale per le successive operazioni di fusione ed elaborazione sensoriale, descritte nel Capitolo 4.2.2, poiché consente di esprimere tutte le point cloud in un medesimo sistema di riferimento. Ciò permette di combinarle in una rappresentazione unica e coerente dell'ambiente, in modo da evidenziare solo gli ostacoli presenti nello spazio operativo.

Completate le operazioni di calibrazione ed elaborazione, la point cloud risultante, così come quelle intermedie, può essere visualizzata in RViz, insieme al modello del robot e alle componenti accessorie introdotte nella configurazione del simulatore digitale, come mostrato in Figura 4.4. In alternativa, la point cloud può essere utilizzata per generare la mappa 3D, il cui procedimento è illustrato nel Capitolo 4.3, che completa l'implementazione del digital twin del sistema robotizzato.

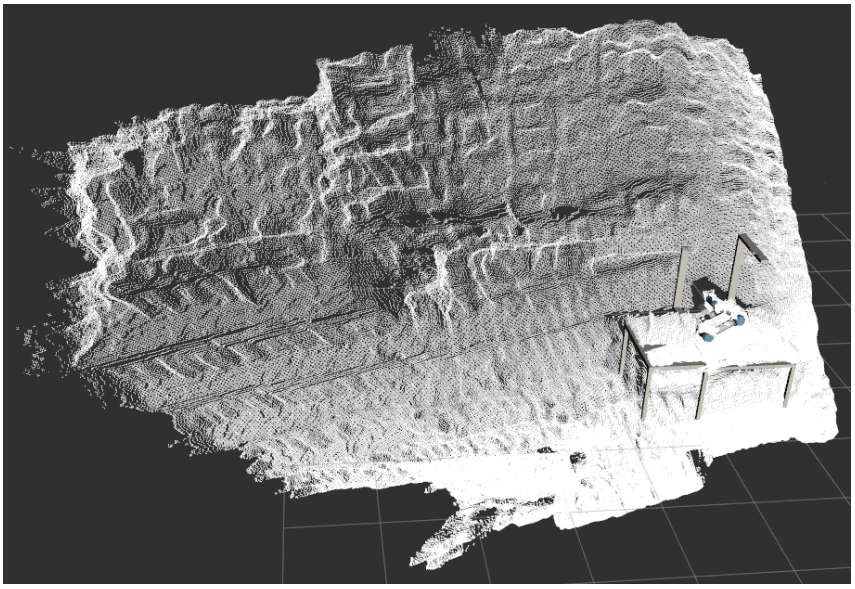


Figura 4.4: Configurazione di RViz con integrata una point cloud.

4.2.1 Calibrazione estrinseca delle telecamere

La calibrazione estrinseca delle telecamere RGB-D consente di determinare la posizione e l'orientamento di ciascun sensore rispetto al sistema di riferimento del robot, permettendo l'integrazione coerente dei dati di profondità nel simulatore digitale.

La Orbbec Gemini 2L montata sul polso del manipolatore è già stata calibrata in precedenti applicazioni sperimentali, pertanto si dispone già della matrice di trasformazione tra il sistema di riferimento della hand-eye camera (*gemini2L_1_link*) e il TCP, che ha la seguente forma:

$$\mathbf{T}_{gemini2L_1_link \to TCP} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}$$

Sfruttando la cinematica diretta del robot, ROS 2 determina automaticamente la matrice di trasformazione tra il sensore e il frame *base_link* ($\mathbf{T}_{gemini2L_1_link \to base_link}$), la quale varia in funzione della configurazione del manipolatore, poiché la telecamera è solidale alla flangia del robot.

La calibrazione delle due Orbbec Gemini 335L si basa sul lavoro di tesi di Pepi Mattia [15], che definisce una procedura automatizzata per la calibrazione estrinseca, validata sulle stesse telecamere e nella medesima configurazione adottata in questa applicazione. La procedura, sviluppata interamente in ROS 2, prevede, tramite un algoritmo di sphere-tracking, il rilevamento di un target sferico solidale al polso dell'UR5e. Il quale assume diverse configurazioni mediante un algoritmo di movimentazione automatica del robot, secondo un approccio eye-to-hand. Il target utilizzato

è una sfera di diametro 20.8 cm, fissata alla flangia del robot tramite un supporto stampato in 3D e un'asta filettata, come mostrato in Figura 4.5.

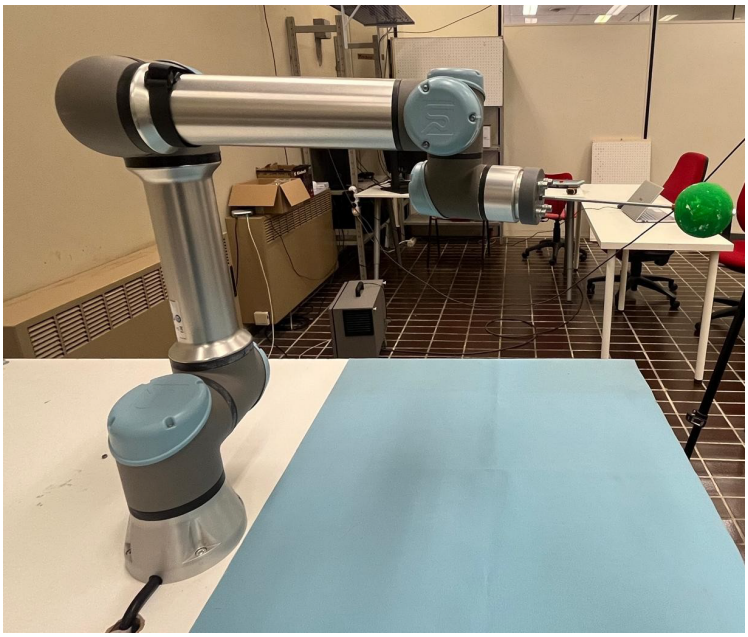


Figura 4.5: Configurazione del robot per la calibrazione estrinseca delle Orbbec Gemini 335L, con installato il tool di calibrazione, composto da sfera, supporto e asta filettata.

Dalle immagini acquisite dalla telecamera RGB-D fissa, l'algoritmo di *sphere-tracking* estrae la posizione del centro della sfera nel sistema di riferimento della camera. Poiché la distanza tra il TCP, coincidente in questo caso con il centro della sfera, e la flangia del robot è nota, per ciascuna configurazione si determina la posizione corrispondente della sfera nel sistema di riferimento del robot. Si ottengono così coppie di punti espresse nei due frame, necessarie alla stima della trasformazione rigida $\mathbf{T}_{camera \to robot}$ tramite l'algoritmo di *Umeyama* che ammette una soluzione in forma chiusa. Da tale procedura si ricavano la matrice di rotazione $\mathbf{R} \in SO(3)$ e il vettore di traslazione $\mathbf{t} \in \mathbb{R}^3$ che compongono la matrice di trasformazione nella forma omogenea:

$$\mathbf{T}_{camera \to robot} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}$$

Questa matrice consente di convertire le coordinate di un punto $\mathbf{q}_i$ espresse nel sistema di riferimento della telecamera (*camera_01_link* o *camera_02_link*) nel corrispondente punto $\mathbf{p}_i$ nel sistema di riferimento del robot (*base_link*):

$$\mathbf{p}_i = \mathbf{T}_{camera \to robot} \begin{bmatrix} \mathbf{q}_i \\ 1 \end{bmatrix}$$

Stimata la matrice di trasformazione, la sua accuratezza viene verificata tramite due approcci complementari: una validazione numerica, basata sull'analisi dell'errore residuo tra i punti trasformati, e una validazione visiva tramite RViz, che consente di verificare qualitativamente l'allineamento tra i frame.

4.2.2 Fusione ed elaborazione sensoriale

Le point cloud acquisite dalle tre telecamere RGB-D, precedentemente calibrate e integrate nel sistema, vengono combinate per ottenere una rappresentazione unica e completa della scena, riducendo le occlusioni. Tale point cloud costituisce la base per la generazione della mappa 3D dell'ambiente e per le successive fasi di pianificazione del moto e validazione delle traiettorie in MoveIt. A questo scopo è stata sviluppata una pipeline di elaborazione delle point cloud, integrata in ROS 2, organizzata come una sequenza di trasformazioni applicate progressivamente, ciascuna realizzata da un nodo dedicato. Ogni nodo pubblica la point cloud risultante in un topic intermedio, al quale si sottoscrive il nodo responsabile della fase successiva, così da comporre una catena di elaborazione. I nodi della pipeline vengono avviati tramite un unico file *launch*, all'interno del quale sono inoltre configurati i parametri principali e specificati i topic di ingresso e uscita. Questa organizzazione centralizzata rende l'intero sistema modulare, facilmente estendibile e semplice da modificare.

Il processo parte dai dati grezzi acquisiti dai sensori e produce una point cloud che rappresenta esclusivamente gli ostacoli rilevati all'interno del workspace robotico con precisione e senza rumore o elementi non utili alla pianificazione delle traiettorie. Le operazioni di filtraggio, trasformazione e sincronizzazione sono realizzate tramite librerie dedicate di ROS 2, in particolare la Point Cloud Library (PCL), che offre gli strumenti necessari per processare e combinare in modo efficiente e modulare i dati tridimensionali [32].

La pipeline è composta dai seguenti passaggi principali:

- Sincronizzazione temporale;
- Trasformazione in uno stesso frame;
- Fusione preliminare delle Orbbec Gemini 335L;
- Riduzione della densità dei punti;
- Isolamento del workspace robotico;
- Rimozione dell'ambiente statico;
- Rimozione dei punti appartenenti al robot;
- Rimozione dei punti isolati;
- Integrazione della Gemini 2L con le Gemini 335L;

I passaggi mostrati in Figura 4.6 vengono descritti nel dettaglio nei paragrafi seguenti, includendo i parametri utilizzati e le scelte implementative adottate in ciascun nodo.

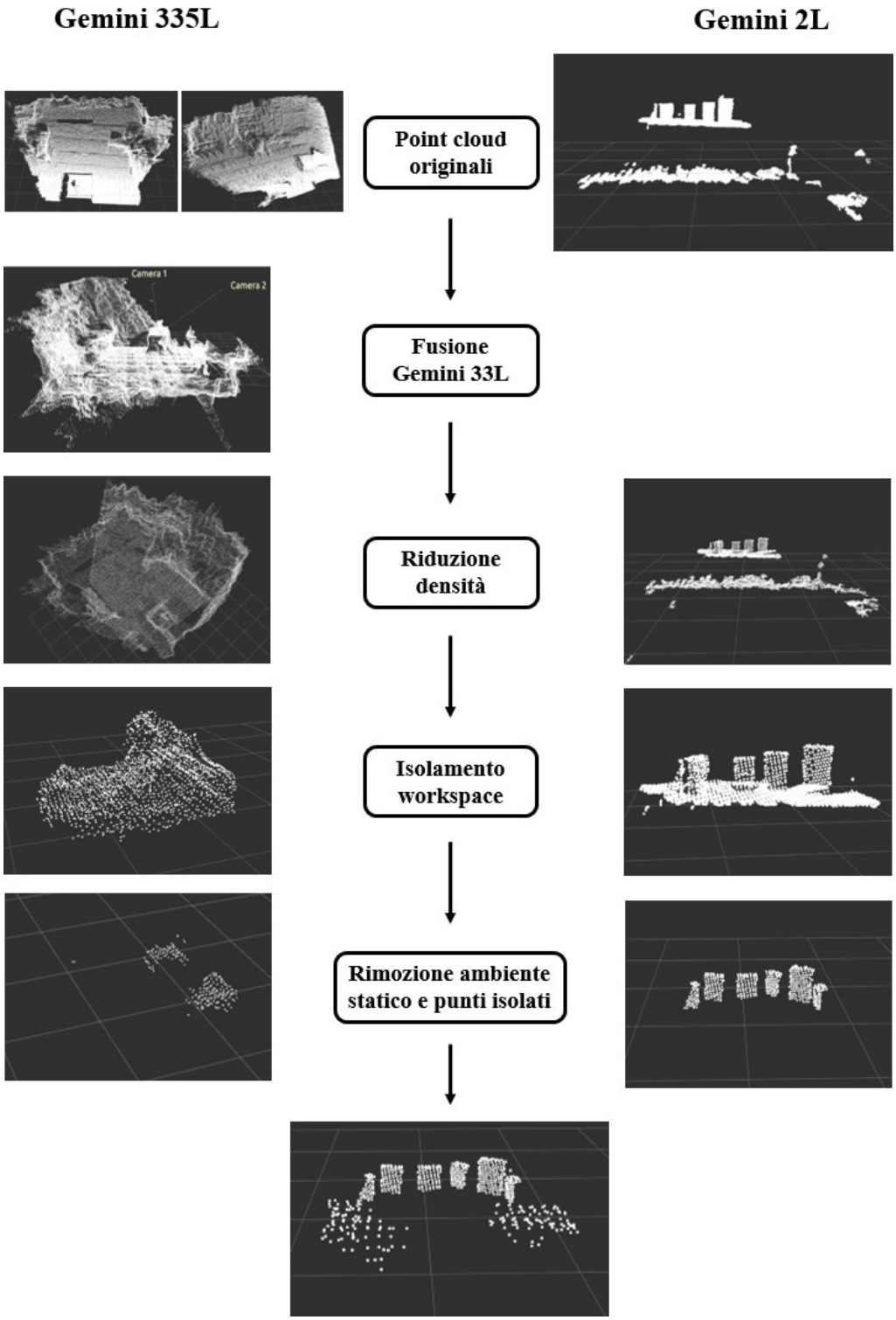


Figura 4.6: Pipeline di fusione ed elaborazione point cloud

Sincronizzazione temporale

La prima operazione consiste nella sincronizzazione temporale delle point cloud acquisite dalle tre telecamere RGB-D, necessaria per garantire che i dati si riferiscano allo stesso istante di acquisizione. Sebbene ciascun sensore pubblichi continuamente le point cloud sui rispettivi topic (*camera_01/depth/points*, *camera_02/depth/points* per le Gemini 335L e *gemini2L_1/depth/points* per la Gemini 2L), le acquisizioni non risultano perfettamente simultanee, a causa di normali differenze nei tempi di esposizione e dei ritardi di comunicazione. La sincronizzazione è ottenuta tramite un nodo dedicato che utilizza il pacchetto `message_filters` di ROS 2, il quale allinea temporalmente più flussi di messaggi sulla base dei rispettivi timestamp tramite il criterio *ApproximateTime*. In questo modo vengono considerati solo i messaggi con i tempi di acquisizione sufficientemente vicini tra loro.

Trasformazione in uno stesso frame

Dopo la sincronizzazione, le point cloud vengono trasformate nel frame di riferimento *base_link* utilizzando le matrici ottenute dalla calibrazione estrinseca. Questo rende le coordinate dei punti delle diverse point cloud direttamente confrontabili, permettendone la corretta fusione in una unica. La trasformazione delle coordinate dei punti è gestita tramite il pacchetto `tf2` di ROS 2, che rende disponibili le trasformazioni tra i vari frame, sia statiche sia dinamiche. La libreria PCL applica tali trasformazioni a ciascun punto, assicurando che ogni point cloud sia correttamente posizionata all'interno dello stesso sistema di riferimento.

Fusione preliminare delle due Orbbec Gemini 335L

In questa fase le point cloud acquisite dalle due Orbbec Gemini 335L vengono combinate in una unica utilizzando la funzione *concatenate* della libreria PCL, che unisce i punti della seconda in quelli della prima. La fusione è possibile perché le point cloud sono già sincronizzate e riferite allo stesso sistema di riferimento.

Riduzione della densità dei punti

Le point cloud acquisite dai sensori RGB-D presentano una densità di punti spesso superiore al necessario per descrivere l'ambiente, appesantendo notevolmente il carico computazionale delle elaborazioni successive. Per ottimizzare le prestazioni viene applicato un filtro di *downsampling* (riduzione della densità) basato sul metodo *Voxel Grid* della libreria PCL, sia alle point cloud delle Gemini 335L sia a quella della Gemini 2L. Il volume acquisito viene suddiviso in una griglia tridimensionale di piccoli cubi regolari detti voxel. Tutti i punti contenuti in ciascun voxel vengono sostituiti da un unico punto, la cui posizione è calcolata come la media delle coordinate dei punti originari presenti in quel volume. La Tabella 4.1 riporta la dimensione dei voxel in funzione del sensore da cui proviene la point cloud.

Sensore	Lunghezza lato del voxel
Gemini 335L	5 cm
Gemini 2L	2 cm

Tabella 4.1: Dimensione dei voxel per la riduzione della densità dei punti.

Isolamento del workspace robotico

In questa fase vengono isolati solo i punti che ricadono nell'area di lavoro del robot, ovvero la porzione di spazio effettivamente rilevante per la pianificazione di traiettorie sicure, riducendo ulteriormente il carico computazionale delle elaborazioni successive. Il volume di interesse è definito dall'intersezione tra una sfera di raggio 80 cm (0.8 m) centrata nel punto di ancoraggio del robot (*base_link*) e lo spazio posto al di sopra del piano del tavolo, generando complessivamente una semisfera. Vengono quindi mantenuti solo i punti le cui coordinate, riferite al *base_link*, soddisfano:

$$\sqrt{x^2 + y^2 + z^2} \leq 0.8 \ , \quad z > 0.$$

Oltre a questo primo filtraggio applicato alle point cloud delle Gemini 335L e della Gemini 2L, da quella statica sono stati rimossi i punti che ricadono nel campo visivo di quella dinamica. L'obiettivo è evitare ridondanze e sfruttare la maggiore accuratezza garantita dalla telecamera montata nel polso del robot nelle regioni da essa osservate. Per identificare tali punti viene utilizzata una ricerca radiale basata su KD-tree. Il KD-tree (k-dimensional tree) è una struttura dati gerarchica che organizza i punti suddividendo ricorsivamente lo spazio lungo gli assi cartesiani. Questa rappresentazione permette di escludere rapidamente intere regioni che non possono contenere punti vicini, rendendo la ricerca molto più efficiente rispetto a un controllo punto per punto. In questo caso, il KD-tree contiene i punti acquisiti dalla Gemini 2L e per ogni punto delle Gemini 335L la funzione *radiusSearch* verifica se esistono punti della 2L entro un raggio di 15 cm. Se non vengono trovati vicini, il punto delle 335L viene mantenuto, in caso contrario viene scartato. In questo modo, all'interno del campo visivo della Gemini 2L vengono conservati esclusivamente i punti da essa acquisiti, mentre, le 335L contribuiscono solo nelle zone non coperte dal sensore mobile. Questo assicura una rappresentazione completa dell'ambiente senza duplicazioni né sovrapposizioni inutili. Infine, le porzioni di point cloud esterne al workspace non vengono eliminate, ma pubblicate nel topic *pcl_outside_ws_335L*, così da poter essere riutilizzate per una visione più ampia della scena o per future estensioni del sistema.

Rimozione dell'ambiente statico

La rimozione del background statico consente di rappresentare nella point cloud soltanto gli elementi dinamici, cioè quelli che il sistema di visione deve effettivamente

mappare. All'avvio del nodo viene acquisita una point cloud di riferimento, chiamata *background*, che rappresenta lo stato iniziale e immobile della scena. Nei frame successivi ciascuna nuova point cloud viene confrontata con il riferimento. I punti che coincidono, o che risultano sufficientemente vicini a quelli del background, vengono eliminati; quelli che non trovano un punto corrispondente vengono invece conservati e classificati come *foreground*, ovvero la parte dinamica. Anche in questo caso l'individuazione dei punti vicini è basata su un KD-tree, utilizzando per ogni point cloud un raggio di ricerca specifico (riportato in Tabella 4.2). Se il KD-tree restituisce almeno un vicino entro tale raggio, il punto viene scartato, se invece non viene trovato alcun vicino, il punto è considerato nuovo e viene aggiunto alla *foreground*. Il risultato di questa elaborazione è una point cloud dinamica che rappresenta esclusivamente gli oggetti o le porzioni della scena che differiscono dall'ambiente statico iniziale.

Sensore	**Raggio di ricerca**
Gemini 335L	10 cm
Gemini 2L	2 cm

Tabella 4.2: Raggio di ricerca in funzione del sensore per la rimozione dell'ambiente statico.

Rimozione dei punti appartenenti al robot

Quando la mappa 3D viene aggiornata mentre il robot è in movimento, oppure quando si trova in posizione diversa rispetto a quella utilizzata per acquisire il background, le Gemini 335L rilevano il manipolatore, che non è rimosso dai filtri precedenti e verrebbe quindi interpretato come ostacolo. In queste condizioni, sarebbe impossibile pianificare qualunque movimento, poiché MoveIt segnalerebbe continuamente collisioni tra il robot e se stesso. Per evitare questo problema è necessario eliminare dalla point cloud tutti i punti che rappresentano la geometria del manipolatore. A tale scopo viene utilizzato un nodo che combina la descrizione geometrica del robot ricavata dall'URDF con la posizione dei link ottenuta dal pacchetto `tf2`, aggiornata in tempo reale dal *robot_state_publisher* sulla base dei dati pubblicati nel topic *joint_states*. Questo consente di posizionare correttamente i link nello spazio e individuare con precisione i punti che ricadono all'interno dei loro volumi. Dopo aver caricato le geometrie, per ogni punto della point cloud viene costruita una sfera dal raggio di 5 cm. La funzione *collide* della libreria FCL (Flexible Collision Library) verifica se tale sfera interseca la geometria del robot nella configurazione corrente. Se l'intersezione viene rilevata, il punto corrispondente viene rimosso dalla point cloud.

Rimozione dei punti isolati

A causa del rumore dei sensori, le point cloud contengono punti isolati che non rappresentano alcun elemento reale della scena. Questi punti fanno apparire come

occupate zone che in realtà sono libere, compromettendo la corretta pianificazione del movimento. Per eliminarli è stato sviluppato un nodo che li identifica tramite una ricerca dei vicini basata su un KD-tree, che consente di individuare in modo efficiente i punti spazialmente prossimi nella point cloud. Per ciascun punto vengono cercati i vicini entro un raggio di ricerca, diverso a seconda della point cloud, utilizzando la funzione *radiusSearch*. Se il numero di vicini trovati è inferiore alla soglia K, il punto viene considerato isolato e quindi rimosso mediante la funzione *ExtractIndices* della libreria PCL, producendo una point cloud più pulita e priva di rumore sparso. Al contrario, se il punto ha un numero sufficiente di vicini, viene mantenuto nella point cloud finale. I valori del raggio di ricerca e della soglia K utilizzati in entrambe le point cloud per classificare un punto isolato, sono riportati in Tabella 4.3.

Sensore	**Raggio di ricerca**	**K**
Gemini 335L	10 cm	2
Gemini 2L	2 cm	2

Tabella 4.3: Raggio di ricerca e soglia K in funzione del sensore per la rimozione dei punti isolati.

Integrazione della Gemini 2L con le Gemini 335L

Le point cloud ottenute dai dati acquisiti dalle Gemini 335L e dalla Gemini 2L, dopo essere state filtrate ed elaborate nei nodi precedenti, vengono fuse in una unica così da ottenere una rappresentazione completa del workspace del robot. La 2L offre maggiore accuratezza nella zona davanti all'end-effector, mentre le 335L garantiscono una copertura più ampia dell'ambiente circostante. La fusione, resa possibile dal fatto che le point cloud sono sincronizzate ed espresse nello stesso frame di riferimento, viene effettuata tramite la funzione *concatenate* della libreria PCL. La point cloud risultante viene pubblicata nel topic *pcl_final*, e costituisce l'input del nodo incaricato di generare la mappa 3D tramite OctoMap, successivamente utilizzata per la pianificazione e la validazione del movimento.

4.3 Generazione della mappa 3D

La point cloud prodotta dalla pipeline di fusione ed elaborazione sensoriale descritta nella sezione precedente fornisce una rappresentazione accurata degli oggetti presenti nell'area di lavoro del robot. Per permettere a MoveIt e al planner di moto di riconoscere questi elementi come ostacoli, la point cloud deve essere trasformata in una mappa 3D.

In ROS 2, il metodo più efficace per generare questa mappa è l'utilizzo di OctoMap, come descritto nel Capitolo 2.5.1. Il nodo incaricato della conversione da point cloud a OctoMap si sottoscrive al topic *pcl_final*, da cui riceve i dati di profondità. La generazione della mappa avviene tramite la libreria *octomap*, utilizzando la classe

`octomap::OcTree`, che crea una struttura ad albero spaziale con una risoluzione del voxel pari a 2 cm. Ogni punto della point cloud viene inserito nella mappa chiamando la funzione *updateNode*, mentre *updateInnerOccupancy* aggiorna la struttura interna dell'albero per mantenere la coerenza dell'occupazione. L'octree risultante viene poi trasformato in un messaggio ROS 2 tramite *binaryMapToMsg* e pubblicato sul topic *octomap*, così da poter essere visualizzato in RViz come mappa 3D degli ostacoli (Figura 4.7). I voxel marcati come occupati vengono interpretati dal pianificatore come zone di collisione da evitare, consentendo così la generazione di traiettorie sicure.

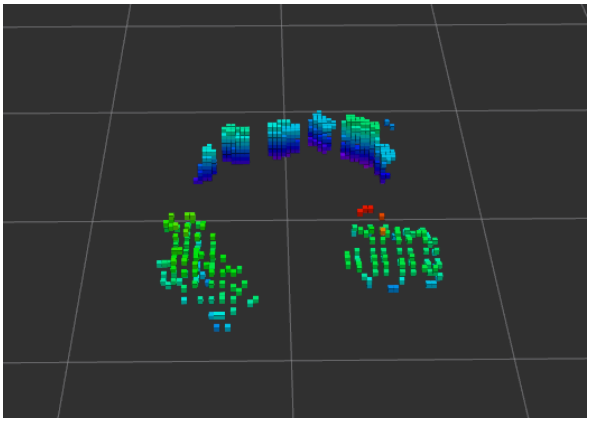


Figura 4.7: Mappa 3D dell'ambiente operativo del robot generata con OctoMap.

Oltre alla mappa del workspace, è possibile generare anche una rappresentazione dell'area esterna allo spazio operativo del robot sottoscrivendosi al topic *pcl_outside_ws_335L*. L'OctoMap dell'intero laboratorio, mostrata in Figura 4.8, può risultare utile per scenari futuri in cui si vogliano integrare ulteriori postazioni robotiche e ottenere una rappresentazione completa dell'intero laboratorio.

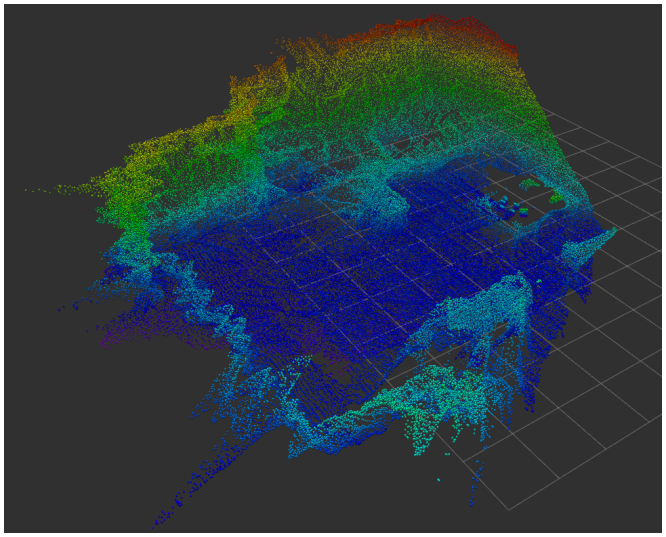


Figura 4.8: Mappa 3D del laboratorio generata con OctoMap.

Capitolo 5

Pianificazione del moto

La pianificazione del moto consente al robot di generare movimenti sicuri e coerenti con l'ambiente reale, integrando in un unico nodo ROS 2 tutte le componenti necessarie alla movimentazione: gestione delle collisioni, risoluzione della cinematica inversa, pianificazione delle traiettorie ed esecuzione della pipeline di pick and place. Il sistema si basa sulla mappa 3D prodotta dal modulo di percezione, che consente a MoveIt di valutare la presenza di ostacoli statici e dinamici durante la generazione delle traiettorie. Le pose da raggiungere nelle diverse fasi del pick and place sono definite in modo da afferrare dall'alto gli articoli da supermercato posti nella zona di prelievo e depositarli su un piano rialzato nella parte posteriore dell'area operativa.

I paragrafi seguenti approfondiscono queste componenti, descrivendo come ciascuna contribuisca a garantire movimenti privi di collisioni.

5.1 Gestione delle collisioni

La pianificazione del moto deve garantire in ogni istante traiettorie prive di collisioni, valutando sia gli ostacoli statici presenti nella scena sia gli elementi dinamici rilevati dal sistema di visione. L'obiettivo è ottenere movimenti per un'applicazione di pick and place sicuri, sfruttando la mappa 3D ottenuta nel Capitolo precedente.

Durante la pianificazione, il sistema deve prevenire diverse tipologie di interferenze:

- Collisioni del robot con se stesso (self-collision): contatti tra i link del manipolatore.
- Collisioni robot–ambiente statico: urti con tavolo, attrezzature o altri elementi permanenti dell'area operativa.
- Collisioni robot–oggetti da movimentare: contatti indesiderati con oggetti non ancora afferrati e mappati con il sistema di visione.
- Collisioni oggetto afferrato–ambiente: l'oggetto trasportato dal robot non deve urtare altri elementi della scena, comprese altre parti del robot.

Le self-collision e le collisioni del robot con l'ambiente statico sono gestite automaticamente da MoveIt grazie alle geometrie di collisione definite nell'URDF importato

in RViz, che includono sia la struttura del manipolatore sia le mesh CAD statiche importate nella fase di configurazione del simulatore digitale (Capitolo 4.1).

La configurazione include anche il file SRDF, nel quale sono definite alcune coppie di link per cui le collisioni vengono disabilitate. Queste esclusioni riguardano quelle tra link adiacenti, che risultano sempre in contatto e quindi non devono essere considerate durante la pianificazione. Vengono inoltre escluse le collisioni che non possono verificarsi per ragioni geometriche o per vincoli cinematici del robot, così da ridurre il carico computazionale, rendendo la pianificazione più rapida ed efficiente senza compromettere la sicurezza del movimento.

Gli ostacoli dinamici, invece, derivano dalla mappa di occupazione generata tramite OctoMap, continuamente aggiornata a partire dalle point cloud prodotte dal sistema di visione. I voxel marcati come occupati vengono gestiti come oggetti di collisione che vengono aggiunti a quelli già presenti nella scena e aggiornati in tempo reale. In questo modo, eventuali variazioni nell'area di lavoro, ad esempio l'aggiunta di un oggetto da parte di un operatore tra una movimentazione e l'altra, vengono immediatamente riconosciute, e la nuova traiettoria pianificata terrà conto della presenza del nuovo ostacolo.

L'ambiente di lavoro viene gestito tramite la classe `moveit::planning_interface::PlanningSceneInterface`, che permette al nodo dedicato alla movimentazione di aggiungere, rimuovere e aggiornare gli oggetti di collisione presenti nella scena. Attraverso questa interfaccia, MoveIt acquisisce automaticamente gli ostacoli statici definiti nell'URDF e integra quelli dinamici derivati dalla mappa 3D generata tramite OctoMap. La stessa interfaccia permette inoltre la gestione degli oggetti afferrati: quando il robot preleva un pezzo, questo viene aggiunto alla scena come *attached object*, con dimensioni e posizione coerenti con l'oggetto reale. In questo modo vengono permesse le collisioni tra l'oggetto e i polpastrelli del gripper, pur mantenendo attive tutte le altre, consentendo a MoveIt di pianificare correttamente il movimento durante la fase di trasporto.

La pianificazione del robot, approfondita più avanti, è eseguita tramite la classe `MoveGroupInterface`, responsabile dell'impostazione dei target, del calcolo delle traiettorie e dell'esecuzione dei movimenti. Essa utilizza la scena costantemente aggiornata dalla *PlanningSceneInterface*, così da garantire percorsi privi di collisioni.

5.2 Cinematica inversa

La cinematica inversa, come descritto nel dettaglio al Capitolo 2.2, ha l'obiettivo di determinare i valori dei giunti corrispondenti a una posa del TCP. Poiché per una stessa posa esistono più configurazioni articolari, selezionare una soluzione il più possibile vicina e coerente a quella di partenza contribuisce a generare traiettorie più brevi, fluide ed efficienti. Per questo motivo, anziché utilizzare il solutore di default di MoveIt (KDL), viene scaricata la libreria *TRAC-IK*, che permette di ottenere

configurazioni ottimizzate rispetto alla distanza dalla configurazione iniziale. Nel nodo viene creato e configurato il solutore *TRAC-IK*, inizializzato con:

- i nomi dei link di partenza e arrivo della catena cinematica, dove *chain_start* coincide con *base_link* e *chain_end* con *tool0* (ovvero il TCP);
- il modello URDF del robot, utilizzato per estrarre la geometria e i limiti articolari;
- il timeout di 1 secondo e la tolleranza numerica di convergenza pari a 1^{e-5};
- la metrica di ottimizzazione impostata su *Distance*, che seleziona la soluzione più vicina allo stato iniziale.

Per ogni movimentazione, il nodo si sottoscrive al topic *JointState* per acquisire in tempo reale la configurazione articolare corrente. Questi valori vengono utilizzati come configurazione iniziale da fornire al solutore, guidando *TRAC-IK* verso soluzioni il più possibile vicine allo stato attuale del robot. In parallelo, sono definiti la posizione e l'orientamento target del TCP, generata dalla pipeline di pick and place definita nel Capitolo 5.4. Il metodo *CartToJnt* calcola quindi la configurazione articolare corrispondente alla posa richiesta. Una volta trovata una soluzione, questa viene verificata mediante il controllo delle collisioni. Qualora la configurazione risultasse non valida, il solutore viene richiamato per ricercarne un'altra. Questo scenario si presenta raramente, poiché le configurazioni di partenza tendono a essere lontane dagli ostacoli, garantendo movimenti fluidi e sicuri. Se la soluzione supera il controllo delle collisioni, i valori articolari ottenuti vengono infine trasferiti al *MoveGroupInterface*, che li utilizza come obiettivo per la successiva pianificazione della traiettoria.

5.3 Pianificatore di traiettorie

Definita e aggiornata dinamicamente la scena di collisione nella *PlanningSceneInterface*, MoveIt pianifica le traiettorie tenendo conto di tutti gli ostacoli presenti. La pianificazione ha il compito di trovare un percorso sicuro e cinematicamente valido che porti il robot da una configurazione iniziale a una finale, rispettando vincoli, limiti e geometrie di collisione. Si tratta del passaggio che traduce un obiettivo di movimento in una sequenza continua ed eseguibile di stati articolari.

La pianificazione è gestita operativamente dal nodo dedicato alla movimentazione tramite la classe *MoveGroupInterface*, che fornisce le API per configurare e controllare il comportamento del robot. La classe viene inizializzata specificando il gruppo di giunti definito nell'SRDF *manipulator*, corrispondente alla catena che va dal link *base_link* a *flange*, così da definire esplicitamente quali giunti partecipano al movimento. La schermata del *MoveIt Setup Assistant* utilizzata per configurare questo gruppo di giunti è mostrata in Figura 5.1.

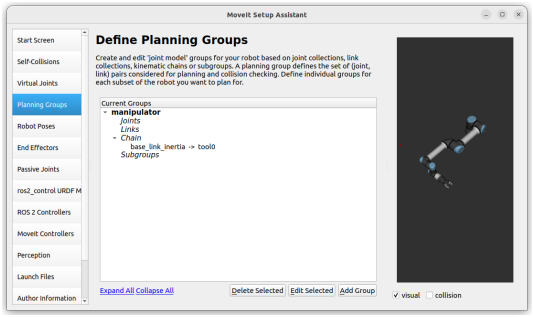


Figura 5.1: Schermata *MoveIt Setup Assistant* per la configurazione del gruppo di pianificazione.

L'obiettivo del movimento viene definito assegnando direttamente i valori dei giunti tramite la funzione *setJointValueTarget*, utilizzando quelli calcolati dalla risoluzione della cinematica inversa (illustrata nella sezione precedente). La scelta della configurazione finale è un aspetto cruciale: soluzioni di IK lontane dallo stato iniziale comportano traiettorie più lunghe, movimenti ampi e tempi di esecuzione più elevati, mentre fornire configurazioni il più possibile vicine a quella attuale contribuisce invece a ottenere percorsi più brevi, fluidi ed efficienti.

La generazione del percorso è affidata alla libreria OMPL, configurata tramite i file YAML prodotti dal *MoveIt Setup Assistant* (come descritto nel Capitolo 3.2.1). All'interno dello stesso *MoveGroupInterface* viene impostato l'algoritmo *RRTstarkConfigDefault* tramite la funzione *setPlannerId*. Questo planner appartiene alla famiglia degli RRT*, metodi sampling-based che esplorano lo spazio delle configurazioni in modo progressivo, raffinando la soluzione nel tempo disponibile (si veda il Capitolo 2.3 per la spiegazione dettagliata). Il tempo massimo di ricerca è impostato a 1 secondo attraverso *setPlanningTime*, così da garantire una pianificazione rapida e compatibile con i tempi tipici di un'applicazione di pick and place. Quando necessario, è possibile introdurre vincoli cinematici specifici tramite la funzione *setPathConstraints*, ad esempio vincoli di orientamento per mantenere il TCP e quindi l'oggetto trasportato in una posizione desiderata lungo tutto il movimento. In questa applicazione tali vincoli non vengono utilizzati, poiché renderebbero la pianificazione significativamente più complessa e aumenterebbero la probabilità che il planner non trovi una soluzione valida, soprattutto in presenza di numerosi ostacoli e regioni di collisione nella scena.

Una volta impostati tutti i parametri, la funzione *plan* calcola una traiettoria priva di collisioni che collega la configurazione iniziale con quella target. Se la pianificazione ha successo, il percorso viene prima visualizzato in RViz, permettendo di valutarne rapidamente la correttezza e la coerenza del movimento, e successivamente la traiettoria viene eseguita tramite la funzione *execute*. La procedura viene ripetuta per ciascuna movimentazione prevista, aggiornando di volta in volta la configurazione di giunti da raggiungere in base all'obiettivo corrente.

5.4 Pipeline di pick and place

Dopo aver configurato il modello del robot, la gestione delle collisioni, il risolutore di cinematica inversa e il pianificatore di traiettorie, il passo successivo consiste nel definire le pose che il TCP deve raggiungere durante le diverse fasi operative della pipeline di pick and place. Per ciascuna di queste pose, il nodo sviluppato determina la configurazione articolare target tramite la cinematica inversa, pianifica una traiettoria valida priva di collisioni, e invia al robot il comando per l'esecuzione del movimento.

La configurazione iniziale del robot coincide con quella illustrata nel Capitolo 3.3, scelta affinché la telecamera montata al polso (hand-eye camera) inquadri l'intera area di prelievo posta di fronte al robot. Questa scelta garantisce il corretto funzionamento dell'applicazione di riconoscimento e localizzazione degli oggetti, già sviluppata e validata in laboratorio, che fornisce la posa dell'oggetto insieme alle sue dimensioni [15]. Sulla base di tali informazioni viene definita la terna di presa, ovvero la posa che il TCP deve assumere per afferrare l'oggetto dall'alto. La terna è posizionata 2.5 cm al di sotto della faccia superiore dell'articolo e orientata con l'asse z verso il basso, mentre gli assi x e y sono allineati rispettivamente al lato corto e al lato lungo dell'oggetto.

La posa di deposito viene invece ottenuta tramite programmazione per dimostrazione diretta (vedi Capitolo 3.1.1): il gripper, con l'oggetto già afferrato, viene guidato manualmente nella posizione desiderata sopra il piano rialzato posto nella zona di deposito, e le coordinate del TCP vengono lette nell'interfaccia del Teach Pendant. In questo modo si individua una posizione sicura, che consente di rilasciare l'oggetto correttamente senza urtare il piano di deposito e senza rischiare cadute che renderebbero instabile il posizionamento. Tutte le pose utilizzate nella pipeline di movimentazione, dalla configurazione iniziale alla presa e al deposito, sono espresse rispetto al *base_link*, garantendo coerenza nella definizione dell'intera sequenza operativa del pick and place.

Nelle applicazioni di pick and place non si effettua un passaggio diretto dalla posa iniziale alla posa di presa, né da quella di presa a quella di rilascio. È invece necessario introdurre pose intermedie che permettono di avvicinarsi e allontanarsi dall'oggetto in modo lineare, controllato e compatibile con i vincoli di sicurezza e di collisione. Per questo motivo, l'intero ciclo di movimentazione è strutturato secondo la seguente pipeline:

- configurazione iniziale
- pre-pick
- pick $\rightarrow$ chiusura gripper
- post-pick
- pre-place

- place → apertura gripper
- post-place

L'intera sequenza delle pose della pipeline di pick and place è illustrata in Figura 5.2.

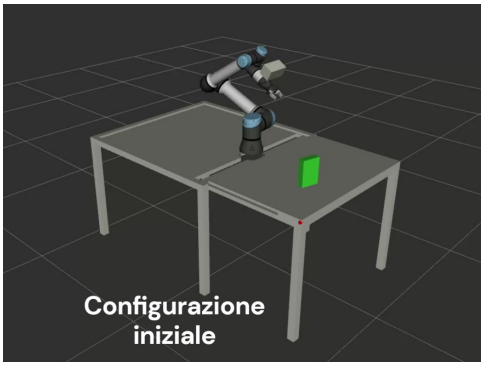
Le pose intermedie pre-pick, post-pick, pre-place e post-place mantengono lo stesso orientamento delle rispettive pose di pick e di place, differenziandosi unicamente per una traslazione di 10 cm lungo l'asse z positivo del *base_link*. Per assicurare movimenti rigorosamente verticali nelle fasi più sensibili della manipolazione, le transizioni tra le pose principali (pick e place) e le rispettive pose intermedie vengono eseguite tramite spostamenti cartesiani, evitando la pianificazione di una traiettoria tra configurazioni articolari. Questa scelta consente di:

- evitare contatti indesiderati tra i polpastrelli del gripper e l'articolo durante l'avvicinamento alla posa di presa (transizioni pre-pick → pick e place → post-place);
- prevenire sfioramenti tra l'oggetto afferrato e il piano di lavoro durante l'allontanamento dal punto di prelievo o l'avvicinamento al punto di deposito (transizioni pick → post-pick e pre-place → place).

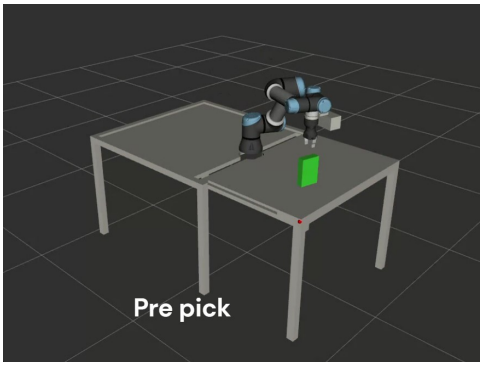
La generazione dello spostamento cartesiano è comunque affidata a MoveIt, che verifica in tempo reale l'assenza di collisioni e il rispetto dei vincoli cinematici.

Prima della fase di afferraggio è necessario disattivare la collisione tra i polpastrelli del gripper e l'oggetto, in caso contrario, il pianificatore rileverebbe costantemente la collisione, invalidando ogni movimento. Dopo la fase di rilascio, tale collisione deve essere nuovamente riattivata per garantire traiettorie sicure nelle successive movimentazioni.

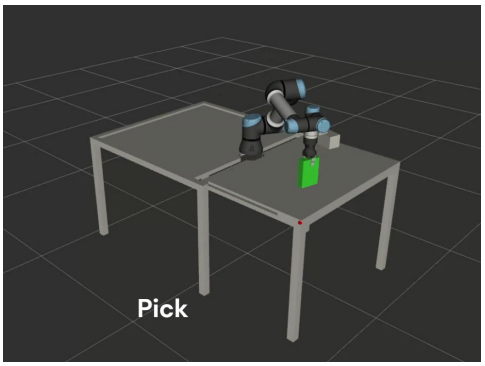
Raggiunta la posa di pick, il gripper viene chiuso per afferrare l'oggetto. Il comando di chiusura è impartito dall'operatore tramite l'interfaccia del Teach Pendant, ciò è possibile poiché, una volta raggiunta la posa di pick, il nodo attende 3 secondi prima di procedere con la movimentazione successiva. Allo stesso modo, una volta raggiunta la posa di place, i polpastrelli vengono riaperti per rilasciare l'oggetto.



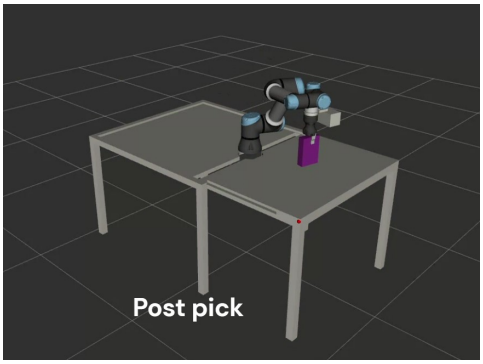
(a) Configurazione iniziale



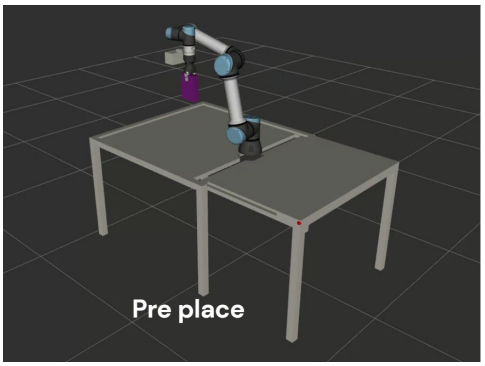
(b) Pre-pick



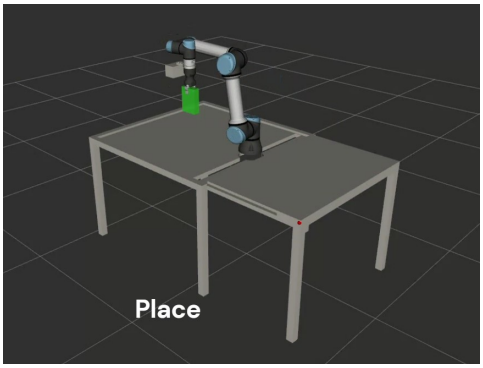
(c) Pick



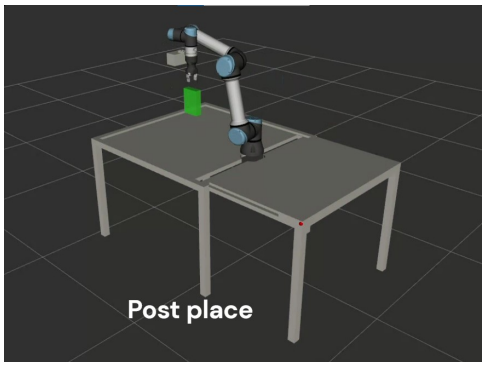
(d) Post-pick



(e) Pre-place



(f) Place



(g) Post-place

Figura 5.2: Pose della pipeline di pick and place.

Capitolo 6

Test sperimentali e analisi dei risultati

In questo Capitolo vengono presentati i test sperimentali condotti per validare il funzionamento complessivo del sistema, comprendente il modulo di visione, la generazione della mappa tridimensionale mediante OctoMap, la pianificazione del movimento e l'esecuzione delle traiettorie necessarie a completare il ciclo di pick and place. L'obiettivo principale è verificare la capacità del sistema di riconoscere correttamente gli ostacoli presenti nell'ambiente, aggiornare la rappresentazione della scena e adattare il percorso di pick and place in funzione delle condizioni operative, garantendo sempre movimenti privi di collisioni e compatibili con i vincoli cinematici del manipolatore.

La prima fase di test è stata dedicata alla validazione del sistema di visione e della ricostruzione 3D ottenuta tramite OctoMap. Per analizzarne il comportamento in un contesto realistico, è stata effettuata una prova preliminare in cui un operatore attraversa la stanza e posiziona diversi articoli nella zona di prelievo davanti al robot. Durante questa attività è stato possibile osservare l'aggiornamento in tempo reale della mappa tridimensionale, sia all'interno dell'area di lavoro del robot sia dell'intero laboratorio.

Nella porzione di mappa relativa al workspace del robot, generata principalmente grazie ai dati acquisiti dalla hand-eye camera, gli oggetti posizionati sul tavolo compaiono progressivamente con una ricostruzione coerente sia dal punto di vista del posizionamento sia delle dimensioni (Figura 6.1). Questa mappa, integrata con i dati delle Gemini 335L relativi alla porzione di ambiente utile alla pianificazione del moto, rappresenta la scena utilizzata da MoveIt per il calcolo delle traiettorie. Parallelamente, la mappa globale ottenuta dall'ampio campo visivo delle Gemini 335L mostra l'intera stanza, all'interno della quale è visibile anche il movimento dell'operatore (Figura 6.2). Ciò conferma la capacità del sistema di rilevare oggetti voluminosi e dinamici anche al di fuori dell'area di lavoro.

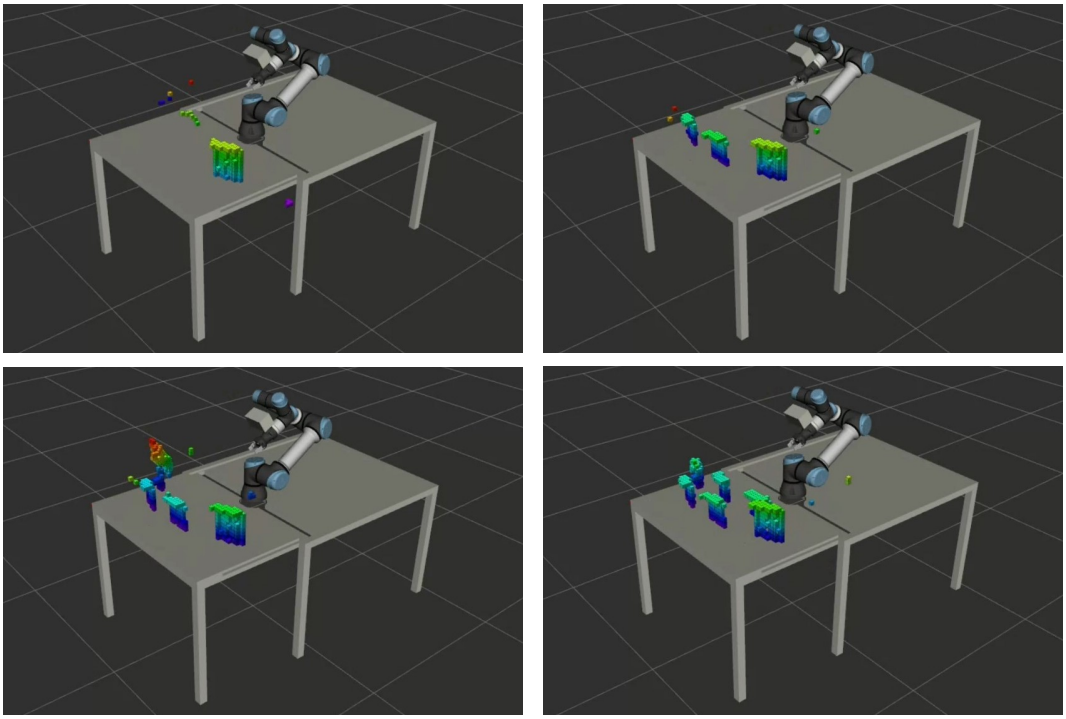


Figura 6.1: OctoMap del workspace robotico generata dalla Gemini 2L durante il deposito degli articoli nell'area dedicata da parte dell'operatore.

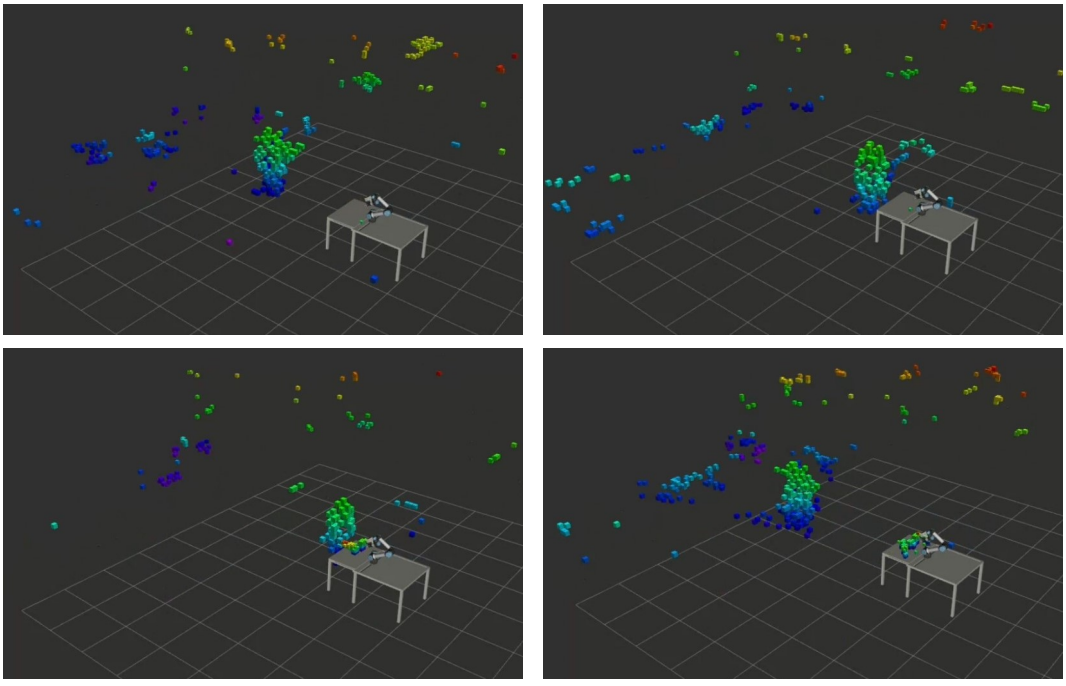


Figura 6.2: OctoMap dell'intero laboratorio generata dalle Gemini 335L durante il deposito degli articoli nell'area dedicata da parte dell'operatore.

La mappa è stata inoltre osservata durante il movimento del robot, così da valutarne il comportamento non solo nella fase di posizionamento degli oggetti ma anche durante la manipolazione. Da questi test sono emerse alcune limitazioni riconducibili alle caratteristiche dei sensori o alla configurazione di laboratorio. Le Gemini 335L, pur garantendo un ampio campo visivo, operano a distanze elevate rispetto al robot e ciò introduce un rumore significativo: i punti acquisiti risultano meno precisi e tendono a generare aree irregolari intorno alle superfici reali. Di conseguenza, durante il movimento del robot, nella mappa compare un insieme di punti incoerenti che non rappresentano la sua effettiva geometria e che, proprio per questo motivo, non vengono rimossi dal filtro dedicato all'eliminazione dei punti appartenenti al robot. In una OctoMap dinamica queste zone verrebbero interpretate come ostacoli, causando il fallimento sistematico delle pianificazioni. Per evitare questo problema, l'aggiornamento dinamico della mappa è stato disattivato durante l'esecuzione del movimento. La mappa viene quindi generata, successivamente arricchita con eventuali ostacoli rilevati per testare la pianificazione di traiettorie sicure e infine congelata interrompendo l'aggiornamento. Quindi, se durante il movimento venissero inseriti nuovi oggetti nella scena, questi non verrebbero rilevati. È importante evidenziare che il filtro di rimozione dei punti appartenenti al robot è stato comunque progettato e integrato nella pipeline, in vista di future configurazioni dotate di sensori meno rumorosi o posizionati a distanze inferiori, nelle quali una OctoMap dinamica potrebbe risultare realmente efficace.

In questa fase è stata inoltre valutata la frequenza di aggiornamento della mappa che si mantiene elevata, circa 20 fps, nonostante i dati di profondità, acquisiti dai sensori a 30 fps, vengano processati più volte richiedendo un carico computazionale significativo.

Sono stati infine definiti i parametri dei filtri utilizzati nella pipeline di fusione ed elaborazione sensoriale (Capitolo 4.2.2), con l'obiettivo di bilanciare due esigenze contrapposte: da un lato ridurre il più possibile il rumore generato dai sensori, evitando falsi ostacoli; dall'altro mantenere una rappresentazione sufficientemente dettagliata, che permetta di visualizzare gli articoli più piccoli che possono interferire con le traiettorie di pick and place.

Sulla base di queste valutazioni è stato possibile caratterizzare le capacità di rilevamento dei sensori: la hand-eye camera riesce a identificare oggetti con dimensioni minime di circa 4 cm, mentre le Gemini 335L non rilevano in modo affidabile oggetti più piccoli di circa 10 cm a causa della distanza operativa. Di conseguenza, gli articoli di dimensioni ridotte vengono identificati esclusivamente dalla camera sul polso, mentre gli ostacoli più voluminosi e posizionati al di fuori del campo visivo della Gemini 2L vengono rilevati solo dalle telecamere installate in alto nel laboratorio.

Con il modulo di visione validato, sono stati eseguiti test completi di pick and place, nei quali tutti i componenti del sistema, dalla percezione alla pianificazione ed esecuzione delle traiettorie, operano in modo coordinato. L'obiettivo è verificare che l'intera applicazione sia in grado di generare e realizzare movimentazioni sicure e

complete in presenza di diverse configurazioni di oggetti nell'area di lavoro, valutando la capacità del sistema di adattarsi ai cambiamenti della scena senza compromettere la corretta esecuzione del task.

La prima prova ha previsto un semplice pick and place con un unico articolo e senza ostacoli aggiuntivi, eccetto il piano rialzato di deposito, mappato dal sistema di visione. Il robot ha seguito la traiettoria più breve, definendo un caso base di riferimento per le configurazioni successive (Figura 6.3).

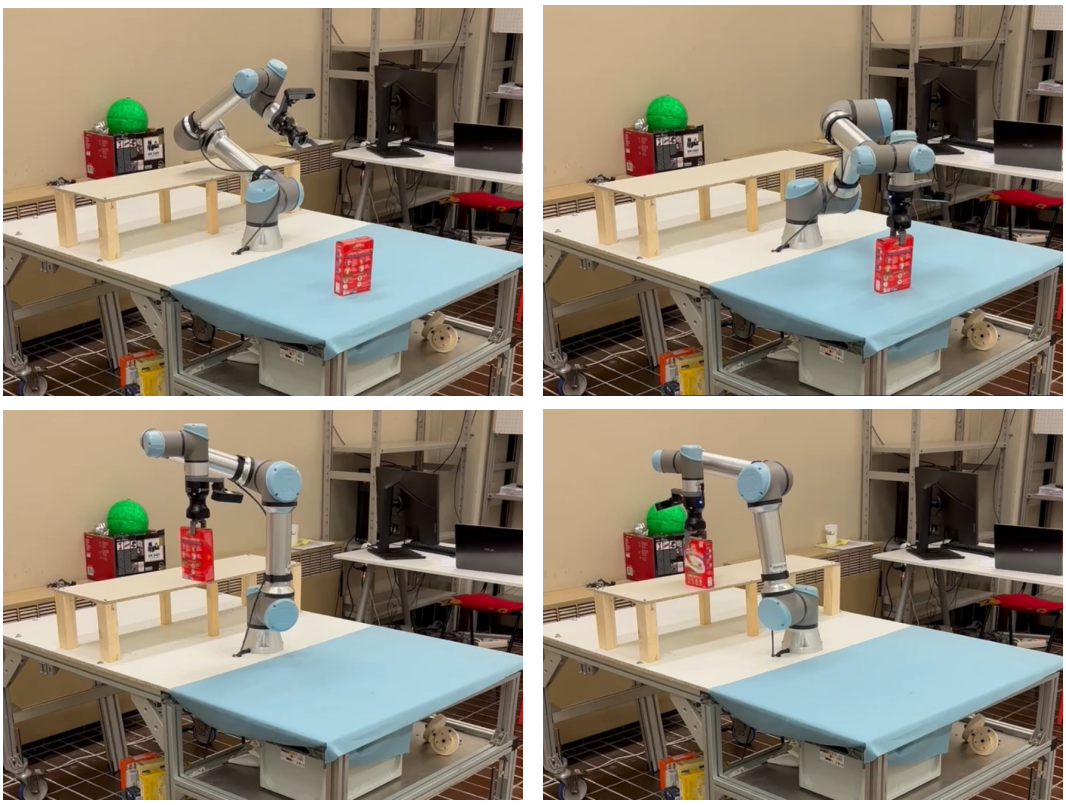


Figura 6.3: Pick and place senza ostacoli.

Nel test seguente sono stati collocati alcuni articoli attorno all'oggetto da prelevare, ma senza interferire con la traiettoria naturale. Il pianificatore li ha rilevati correttamente nella mappa, tuttavia non ha ritenuto necessario deviare il percorso, riproducendo esattamente lo stesso movimento osservato nel caso base (Figura 6.4).

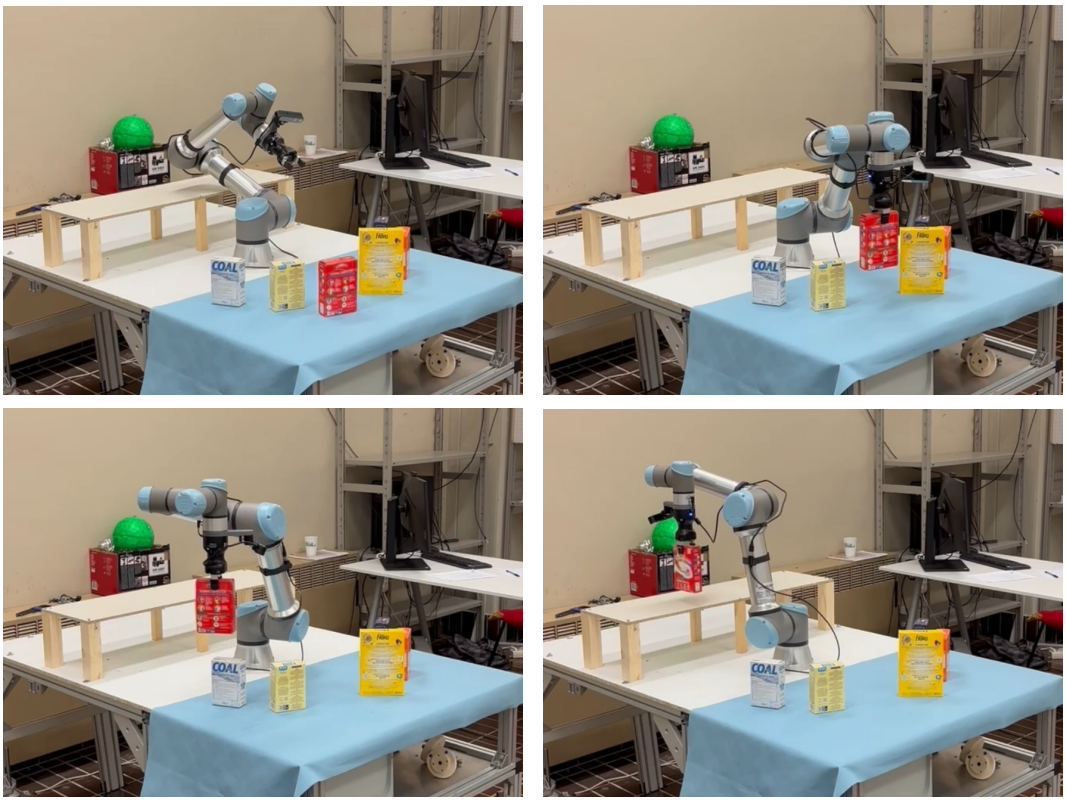


Figura 6.4: Pick and place con altri articoli posizionati nella zona di prelievo.

Una configurazione più complessa ha previsto il posizionamento di un singolo articolo, leggermente più alto, lungo la traiettoria diretta verso la zona di deposito. In questo scenario il robot ha generato automaticamente un percorso alternativo, aggirando l'ostacolo e completando correttamente il task, senza allungare di molto la traiettoria (Figura 6.5).

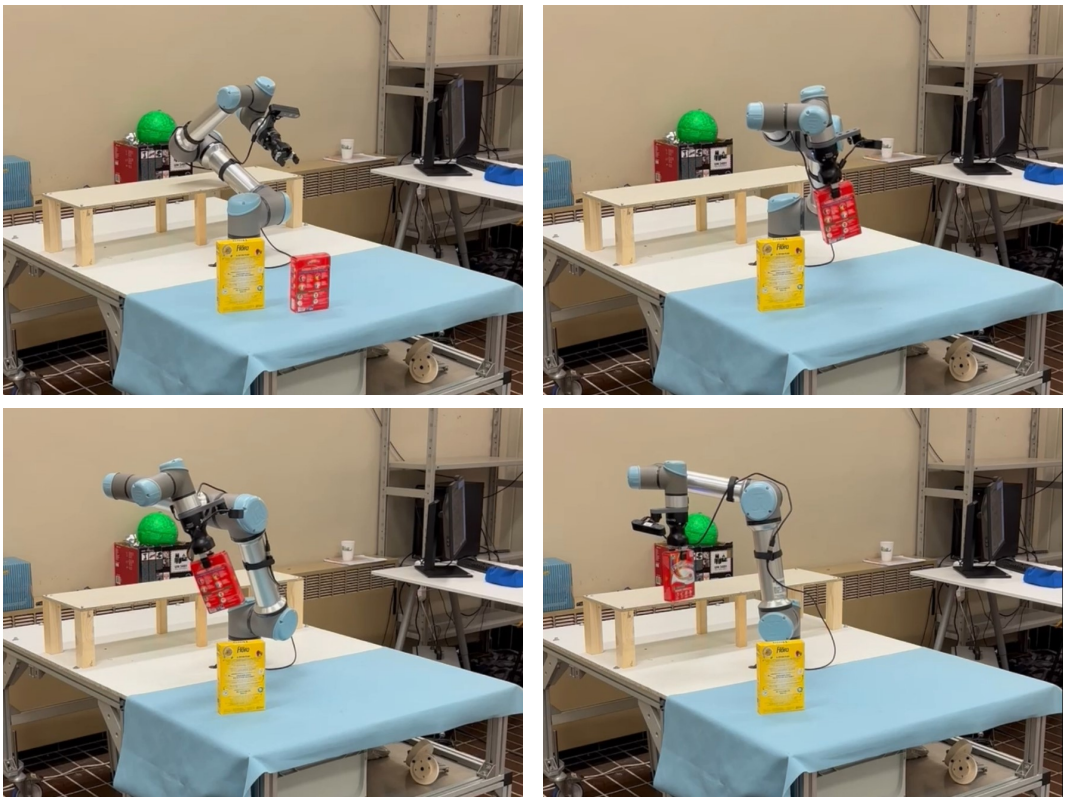


Figura 6.5: Pick and place con un articolo più alto nella traiettoria base.

Aumentando ulteriormente la complessità, oltre allo stesso ostacolo è stata inserita nella zona di prelievo una serie di articoli con dimensioni simili a quello trasportato. In questo scenario la deviazione minima non risultava praticabile, poiché la scatola in movimento avrebbe rischiato di collidere con gli oggetti disposti nell'area. Per completare la movimentazione è stato quindi necessario forzare una ripianificazione. La soluzione adottata consiste nel vincolare il valore del primo giunto nella configurazione di place, imponendogli un orientamento opposto rispetto a quello che verrebbe scelto nella traiettoria base. In questo modo il robot raggiunge la stessa posa finale passando dal lato opposto, generando una traiettoria più ampia ma priva di collisioni e consentendo al pianificatore di esplorare percorsi alternativi che altrimenti non verrebbero considerati (Figura 6.6).

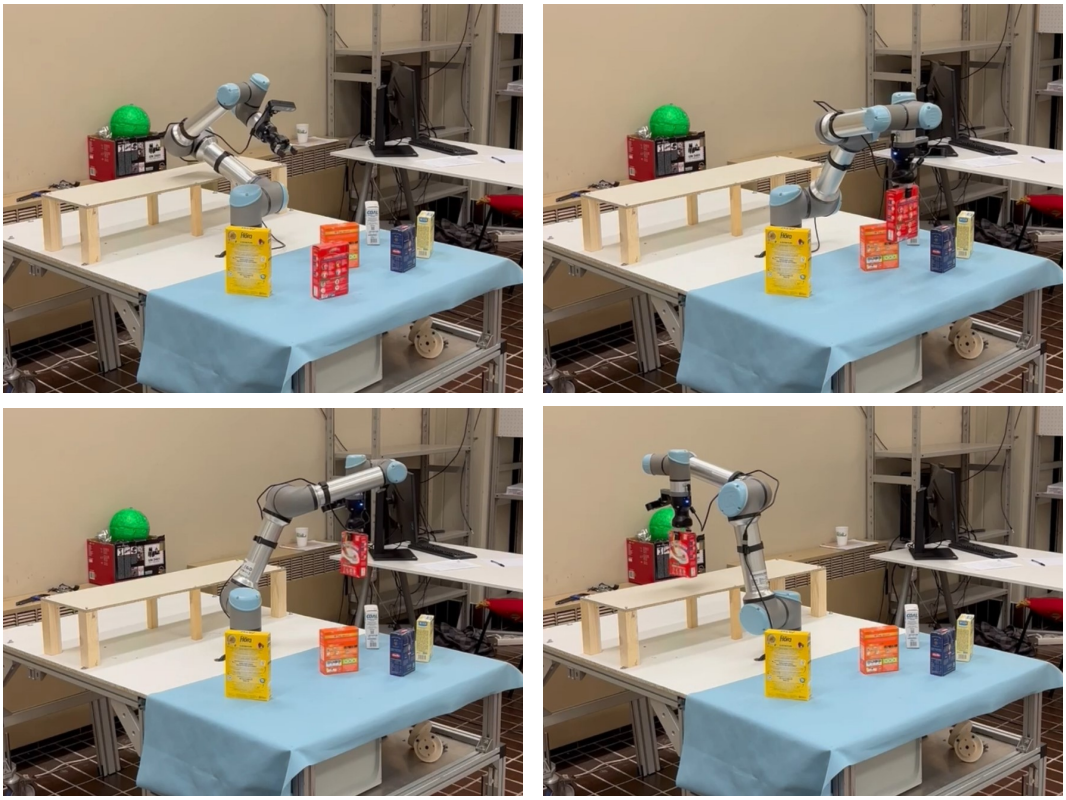


Figura 6.6: Pick and place con un articolo più alto nella traiettoria base e altri articoli posizionati nella zona di prelievo.

Nell'ultimo test è stato inserito uno scatolone di dimensioni significativamente maggiori rispetto agli articoli movimentati, posizionato in modo da interferire con la traiettoria base. Anche in questo caso il pianificatore non è stato in grado di individuare un percorso valido senza ricorrere a una ripianificazione della configurazione di place. L'orientamento modificato del primo giunto ha dunque permesso al robot di generare una traiettoria più lunga, passando dal lato opposto dell'ostacolo e completando correttamente il task. Questo test è stato introdotto per evidenziare l'importanza delle Gemini 335L, che risultano essere le uniche in grado di rilevare lo scatolone aggiunto, poiché posizionato al di fuori del campo visivo della hand-eye camera (Figura 6.7).

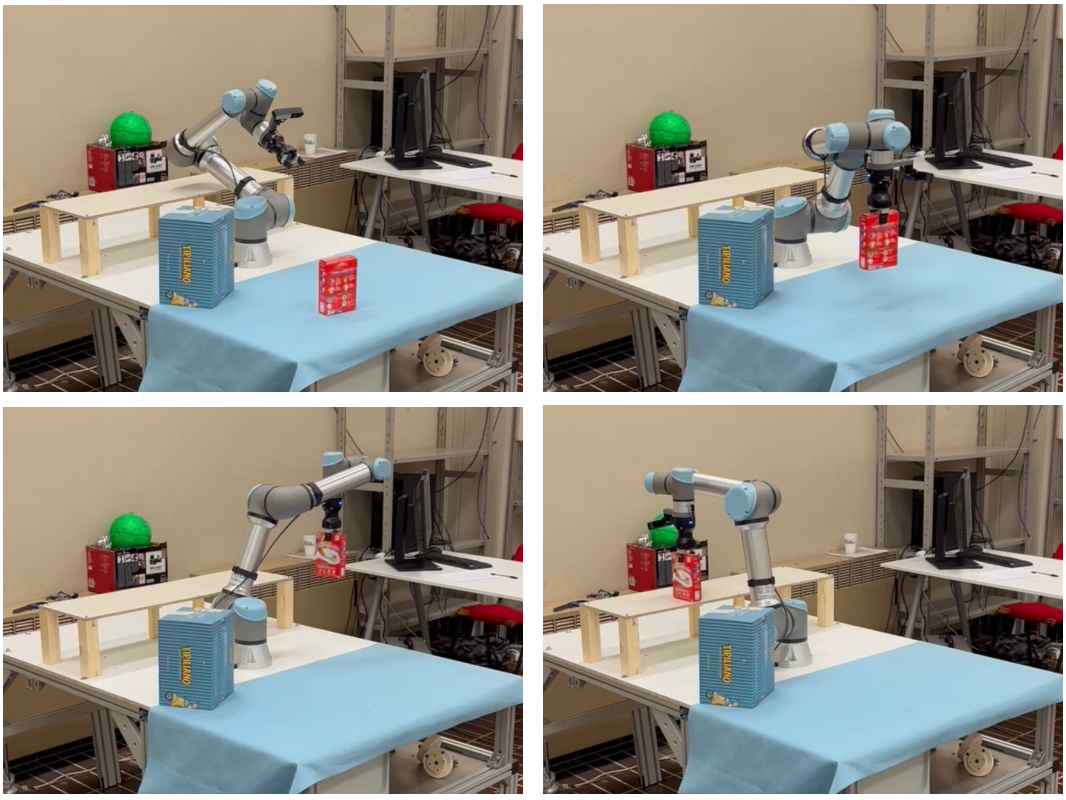


Figura 6.7: Pick and place con uno scatolone nella traiettoria base.

Complessivamente, i test effettuati permettono non solo di verificare il corretto funzionamento del sistema, ma anche di evidenziare alcuni aspetti che influenzano in modo significativo sia la pianificazione del movimento sia l'affidabilità della percezione.

Un primo elemento emerso dalle prove riguarda l'influenza della configurazione iniziale del robot sulla capacità di generare traiettorie brevi e sicure in presenza di ostacoli numerosi o ingombranti. Per garantire alla hand-eye camera una visione completa della zona di prelievo, il robot deve infatti partire da configurazioni prossime ai limiti di alcuni giunti. Questa condizione, insieme alla presa dell'articolo dall'alto,

riduce la libertà di movimento inducendo il planner a preferire percorsi più ampi anche quando l'ostacolo potrebbe essere superato con deviazioni minime.

I test hanno inoltre permesso di valutare la ripetibilità del sistema. Quando la scena rimane invariata e la mappa non viene rigenerata, la pianificazione risulta estremamente stabile: il robot riproduce sempre lo stesso percorso. Se invece il sistema di visione viene riattivato, le piccole variazioni nella ricostruzione tridimensionale, dovute al rumore sensoriale, modificano leggermente la forma e la posizione percepita degli oggetti e generano traiettorie diverse rispetto alla pianificazione precedente. Ne emerge quindi un comportamento molto affidabile in condizioni statiche, ma più sensibile quando la scena viene aggiornata frequentemente.

Un ulteriore aspetto riguarda l'interazione tra il robot e un operatore presente nella scena. Quando la mappa è attiva e viene avviata la pianificazione, qualsiasi parte del corpo che intersechi il volume attraversato dalla traiettoria viene interpretata come un ostacolo, causando il fallimento della pianificazione stessa. Al contrario, se l'operatore rimane all'esterno delle zone interessate dal pick and place, il movimento viene eseguito senza difficoltà. Poiché durante l'esecuzione la mappa viene congelata, eventuali interferenze dell'operatore con l'area di lavoro non vengono rilevate dal sistema, e il robot prosegue lungo la traiettoria pianificata senza deviazioni, con il conseguente rischio di collisione.

Dal punto di vista percettivo, i test evidenziano che la hand-eye camera è essenziale per ricostruire gli oggetti più piccoli, ma il suo campo visivo limitato ne riduce l'efficacia quando il gripper si orienta verticalmente e si avvicina al tavolo, restringendo ulteriormente l'area osservata. Di conseguenza la percezione fornita della camera sul polso deve essere integrata con una ricostruzione globale ottenuta tramite sensori posizionati in modo più favorevole rispetto alle Gemini 335L presenti in laboratorio.

Nel complesso, i test mostrano come il comportamento del sistema dipenda in modo significativo da due fattori principali: le limitazioni cinematiche, imposte dalla configurazione iniziale e dalla presa dall'alto degli oggetti, e il rumore introdotto dai sensori più distanti. Entrambi influenzano la qualità della mappa tridimensionale e, di conseguenza, le soluzioni generate dal pianificatore di movimento. Questi aspetti rappresentano gli elementi tecnici più rilevanti emersi dalla sperimentazione e consentono di interpretare in modo completo le prestazioni osservate nelle diverse configurazioni testate.

Capitolo 7

Conclusioni e sviluppi futuri

Il lavoro aveva come obiettivo la realizzazione di una pipeline completa capace di percepire l'ambiente tramite sensori RGB-D e generare un digital twin del sistema robotizzato basato su una mappa 3D sviluppata con OctoMap. Infine, utilizzare questa rappresentazione per pianificare e adattare le traiettorie di un robot collaborativo in un contesto di pick and place non strutturato. In questo quadro, l'intento principale era integrare percezione, mappatura tridimensionale, pianificazione del movimento ed esecuzione robotica in un'unica applicazione completa, validata attraverso una serie di test sperimentali.

Il sistema di visione ha prodotto una mappa 3D coerente con la scena, utilizzata da MoveIt per calcolare traiettorie sicure in presenza di ostacoli che variano di volta in volta, e il robot ha portato a termine le operazioni di pick and place previste, dimostrando la capacità del sistema di adattarsi. L'analisi sperimentale ha però evidenziato alcune limitazioni che influenzano le prestazioni complessive del sistema. Da un lato, la configurazione iniziale del manipolatore, necessaria per garantire alla hand-eye camera una visione adeguata della zona di prelievo, porta alcuni giunti ad operare vicino ai propri limiti, riducendo la libertà di movimento. A questa condizione si aggiunge il vincolo imposto dalla presa degli articoli dall'alto, che richiede un orientamento specifico del polso e costringe il robot a operare in una zona prossima al limite della propria area operativa, limitando ulteriormente le configurazioni disponibili. Dall'altro lato, il rumore introdotto dai sensori fissi, posizionati a una distanza maggiore rispetto a quella effettivamente necessaria per inquadrare l'area operativa del robot, compromette la qualità della ricostruzione 3D. Questa condizione rende difficile la rilevazione accurata degli oggetti più piccoli e impedisce l'utilizzo di una mappa dinamica durante l'esecuzione del movimento. Tali fattori limitano la possibilità di generare traiettorie più compatte, ottimizzate o adattabili in tempo reale.

Un possibile miglioramento riguarda quindi il posizionamento delle telecamere fisse, avvicinandole al robot pur mantenendo la visione completa dell'area operativa, privilegiando la qualità locale rispetto alla copertura globale del laboratorio. Questa soluzione ridurrebbe il rumore e renderebbe più precisa la mappatura 3D dell'ambiente, favorendo la rilevazione degli ostacoli più piccoli e permettendo l'utilizzo di una mappa dinamica durante la movimentazione del robot. Parallelamente, l'impiego di

pianificatori più avanzati rispetto a quelli disponibili nella libreria OMPL potrebbe produrre traiettorie ottimizzate, più efficienti, e adatte a contesti operativi reali. Inoltre, lo studio di strategie di presa alternative, ad esempio afferrando l'articolo lateralmente, aumenterebbe lo spazio di manovra del robot nel superamento degli ostacoli. L'adozione di queste soluzioni faciliterebbe anche l'introduzione di vincoli di orientamento durante il trasporto, utili a mantenere l'oggetto verticale in ogni momento.

Ulteriori sviluppi futuri possono invece ampliare le funzionalità del sistema e renderlo più versatile nelle sue applicazioni, ad esempio attraverso:

- l'impiego della libreria **ViSP** (Visual Servoing Platform), una piattaforma open-source in C++ progettata specificamente per il *visual servoing* e l'*object tracking*. ViSP mette a disposizione un ampio set di algoritmi in grado di identificare, localizzare e seguire un oggetto all'interno di sequenze di immagini, anche se l'oggetto o la telecamera sono in movimento. Il *visual servoing* consente di controllare il movimento del robot utilizzando informazioni visive come feedback, sfruttando i dati forniti dall'*object tracking* per aggiornare in tempo reale la posa da raggiungere. Nel contesto di questo progetto, tale libreria potrebbe essere impiegata per identificare e seguire l'oggetto da manipolare, regolando la posa di presa del robot tramite feedback visivo e migliorando precisione e adattabilità della fase di afferraggio.

- l'utilizzo di **QR code o marker dedicati** consentirebbe di impostare automaticamente sia la posa di presa sia quella di deposito, rendendole specifiche per ogni tipologia di prodotto. Il sistema di visione, una volta riconosciuto il marker, potrebbe recuperare le informazioni associate all'oggetto e generare automaticamente pose ottimali. La mappa 3D dell'ambiente continuerebbe comunque a essere utilizzata per rilevare gli ostacoli ed evitare collisioni durante l'esecuzione del movimento.

In conclusione, il sistema sviluppato rappresenta una base solida e completamente funzionante per operazioni di pick and place automatiche e flessibili, in grado di adattare la movimentazione alla ricostruzione 3D dell'ambiente invece di seguire traiettorie rigide e preimpostate. Allo stesso tempo, le criticità emerse evidenziano la necessità di migliorare la qualità della percezione e di adottare tecniche di pianificazione e controllo più avanzate, così da aumentare la robustezza, la precisione e l'autonomia del sistema.

Bibliografia

[1] Universal Robots, "Applicazioni." https://www.universal-robots.com/it/applicazioni/, 2025. consultato il 4 dicembre 2025.

[2] K. A. Demir, G. Döven, and B. Sezen, "Industry 5.0 and human-robot coworking," *Procedia Computer Science*, vol. 158, pp. 688–695, 2019. 3rd WORLD CONFERENCE ON TECHNOLOGY, INNOVATION AND ENTREPRENEURSHIP"INDUSTRY 4.0 FOCUSED INNOVATION, TECHNOLOGY, ENTREPRENEURSHIP AND MANUFACTURE" June 21-23, 2019.

[3] A. Malik, G. Shao, and J. Tarakhovsky, "Digital twins for robot systems in manufacturing," in *Digital Twins, Simulation, and Metaverse*, 529 14th St NW, Washington, DC: Springer, 2024.

[4] Universal Robots, "Automazione robotica." https://www.universal-robots.com/it/lp/automazione-robotica/, 2025. consultato il 4 dicembre 2025.

[5] Automazione News, "La robotica industriale." https://www.automazionenews.it/la-robotica-industriale/, 2018. consultato il 4 dicembre 2025.

[6] N. Conterno, "Sviluppo di una linea di confezionamento con robot collaborativi," tesi di laurea magistrale, Università degli Studi di Padova, Padova, Italia, 2021. pp. 9–14.

[7] Universal Robots, "Perché i cobot?." https://www.universal-robots.com/it/approfondimenti/perche-i-cobot/, 2025. consultato il 4 dicembre 2025.

[8] A. Falappa, "Mimica robotica in tempo reale: Un algoritmo per la riproduzione dei movimenti del braccio tramite visione artificiale," tesi di laurea magistrale, Università Politecnica delle Marche, 2023. pp. 9–18.

[9] P. Rocco, "Cinematica del robot." Slide disponibili online, 2016. consultato il 9 agosto 2025.

[10] J. Liu, H. J. Yap, and A. S. M. Khairuddin, "Review on motion planning of robotic manipulator in dynamic environments," *Journal of Sensors*, vol. 2024, no. 1, p. 5969512, 2024.

[11] E. Tosello, "Motion planning per un robot a molti gradi di libertà," Master's thesis, Università degli studi di Padova, Apr 2013. Tesi di laurea magistrale, pp. 11–26.

[12] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb, "State of the art on 3d reconstruction with rgb-d cameras," *Computer Graphics Forum*, vol. 37, pp. 625–652, 05 2018.

[13] MRDVS, "How depth sensing cameras empower smarter robotics." `https://mrdvs.com/how-depth-sensing-cameras-empower-smarter-robotics/`, Sep 2025. Blog post, consultato il 5 dicembre 2025.

[14] A. Vina, "Una guida alla calibrazione della telecamera per la computer vision nel 2025." `https://www.ultralytics.com/it/blog/a-guide-to-camera-calibration-for-computer-vision-in-2025`, Feb 2025. Blog post, consultato il 5 dicembre 2025.

[15] M. Pepi, "Calibrazione automatica di telecamera rgb-d mediante framework ros 2," Master's thesis, Università Politecnica delle Marche, Facoltà di Ingegneria, 2025. Relatore: Prof. Matteo Claudio Palpacelli; Correlatori: Prof. Gaetano Lettera, Prof. Federico Neri.

[16] W. Lin, P. Liang, G. Luo, Z. Zhao, and C. Zhang, "Research of online hand–eye calibration method based on charuco board," *Sensors*, vol. 22, no. 10, 2022.

[17] Z. Zhang, "A flexible new technique for camera calibration," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, pp. 1330 – 1334, 12 2000.

[18] L. Hongyan, D. Qu, F. Xu, F. Zou, J. Song, and K. Jia, "Approach for accurate calibration of rgb-d cameras using spheres," *Optics Express*, vol. 28, pp. 19058–19073, 06 2020.

[19] A. Mazumder, M. Sahed, Z. Tasneem, P. Das, F. Badal, M. Ali, M. Ahamed, S. Abhi, S. Sarker, S. Das, M. Hasan, M. Islam, and M. Islam, "Towards next generation digital twin in robotics: Trends, scopes, challenges, and future," *Heliyon*, vol. 9, no. 2, p. e13359, 2023.

[20] A. Hornung, K. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, 04 2013.

[21] U. Robots, "Ur5e — cobot leggero e versatile." `https://www.universal-robots.com/it/prodotti/ur5e/`, 2025. Pagina ufficiale prodotto, consultata il 5 dicembre 2025.

[22] U. Robots, "Faq – domande frequenti." `https://www.universal-robots.com/it/approfondimenti/faq/`. Pagina ufficiale FAQ, consultata il 5 dicembre 2025.

[23] U. Robots, "Universal robots ros2 driver supporting cb3 and e-series." `https://github.com/UniversalRobots/Universal_Robots_ROS2_Driver`, 2025. Repository GitHub, versione 4.4.0 (aggiornato 14 Nov 2025), consultato il 5 dicembre 2025.

[24] Robotiq, "Hand-e adaptive gripper." `https://robotiq.com/products/hand-e-adaptive-robot-gripper`. Pagina ufficiale prodotto, consultata 5 dicembre 2025.

[25] U. Robots, "Dh parameters for calculations of kinematics and dynamics." `https://www.universal-robots.com/articles/ur/application-installation/dh-parameters-for-calculations-of-kinematics-and-dynamics`. Sito ufficiale, consultato il 5 dicembre 2025.

[26] Orbbec, "Gemini 335L – stereo vision 3d camera." `https://www.orbbec.com/products/stereo-vision-camera/gemini-335l/`. Pagina ufficiale prodotto, consultata il 5 dicembre 2025.

[27] Orbbec, "Orbbec gemini 2L – stereo-vision 3d camera." `https://www.orbbec.com/products/stereo-vision-camera/gemini-2l/`. Pagina ufficiale del prodotto, consultata il 5 dicembre 2025.

[28] Orbbec, "Orbbec sdk v1&v2 pre-compiled repo." `https://github.com/orbbec/OrbbecSDK`. Repository GitHub, versione attiva (2025), consultato il 5 dicembre 2025.

[29] O. Robotics, "Ros 2 documentation — humble hawksbill." `https://docs.ros.org/en/humble/index.html`. Documentazione ufficiale ROS 2, versione Humble, consultata il 5 dicembre 2025.

[30] M. Developers, "Moveit — robotic manipulation framework." `https://moveit.picknik.ai/main/index.html`. Sito ufficiale MoveIt, documentazione e risorse, consultato il 5 dicembre 2025.

[31] M. Aleksandrowicz, K. Klimek, and J. Płachno, "Robotiq hand-e ros 2 description package." `https://doi.org/10.5281/zenodo.15047935`. Versione 0.1.1, rilasciato il 18 marzo 2025.

[32] A. Campomaggiore, M. Costanzo, G. Lettera, and C. Natale, "A fuzzy inference approach to control robot speed in human-robot shared workspaces," in *Proceedings of the 16th International Conference on Informatics in Control, Automation and Robotics - Volume 2: ICINCO,*, pp. 78–87, INSTICC, SciTePress, 2019.