

**POLITECNICO DI MILANO**  
**Polo territoriale di Como**

SCUOLA DI INGEGNERIA DELL'INFORMAZIONE  
Corso di studi in Ingegneria Informatica



## **Sistema di localizzazione tramite landmark artificiali**

Tutor universitario:  
**Prof. Andrea Bonarini**

Elaborato finale di:  
**Re Calegari Gloria**  
matr: 715624

**Anno Accademico 2010-2011**

# Sommario

Il presente lavoro si colloca nell'ambito della robotica e della visione artificiale, disciplina che si occupa di analizzare ed estrarre informazioni dall'ambiente circostante, tramite l'utilizzo di telecamere e sensori, al fine di riprodurre la vista umana. In particolare ci si è concentrati sul riconoscimento di particolari oggetti portatori di informazioni utili per il calcolo della posizione del robot.

L'obiettivo che si pone questa tesi è quello di realizzare un sistema di localizzazione assoluta basato sull'utilizzo di landmark passivi, che consenta al robot di conoscere in ogni momento la sua posizione. Il problema della localizzazione e quello dell'autonomia di un robot sono strettamente correlati: un robot, che non è in grado di conoscere la sua posizione nell'ambiente esterno, non è in grado di decidere cosa fare.

Il progetto nasce come un'estensione e modifica di un modulo già esistente e funzionante sul robot LURCH. Tale sistema si basa sull'utilizzo delle librerie ARToolKitPlus, in grado di fornire, tramite l'ausilio di marker, informazioni riguardanti il posizionamento e l'orientazione del robot.

Il modulo viene sviluppato seguendo le convenzioni di MRT, framework che garantisce una massima flessibilità e adattabilità del modulo a sistemi più complessi.

I test di verifica, effettuati in un ambiente controllato, hanno confermato i risultati attesi: il nuovo modulo ben si inserisce nei sistemi esistenti, garantendo corrispondenza con gli obiettivi proposti.

# Ringraziamenti

Ringrazio il prof. Andrea Bonarini per avermi dato la possibilità di partecipare ad un progetto così concreto e stimolante; un ringraziamento particolare a Simone Ceriani per avermi supportato e aiutato pazientemente.

Ringrazio la mia famiglia per essermi stata vicina e per avermi supportato in tutti questi anni di studio. Ringrazio Beppe, tutti i miei amici e i compagni di corso con cui ho condiviso questo percorso.

# Indice

<b>1</b>	<b>Introduzione</b>	<b>5</b>
<b>2</b>	<b>Stato dell'arte</b>	<b>8</b>
2.1	Localizzazione . . . . .	9
2.1.1	La localizzazione indoor . . . . .	10
2.1.2	Sistemi di localizzazione relativa . . . . .	11
2.1.3	Sistemi di localizzazione assoluta . . . . .	11
2.2	Localizzazione con landmark artificiali . . . . .	13
2.2.1	Fiducial Marker . . . . .	13
2.2.2	ARToolKitPlus . . . . .	14
2.3	Progetti esistenti . . . . .	15
<b>3</b>	<b>Test preliminari</b>	<b>16</b>
3.1	Ipotesi e considerazioni . . . . .	16
3.2	Ambiente di test . . . . .	18
3.3	Descrizione delle prove . . . . .	18
3.3.1	Attrezzature utilizzate . . . . .	18
3.3.2	Dimensioni del Marker . . . . .	19
3.3.3	Analisi degli errori . . . . .	19
3.4	Analisi dei risultati . . . . .	20
3.4.1	Marker piccolo . . . . .	20
3.4.2	Marker grande . . . . .	25
3.5	Considerazioni generali . . . . .	29
<b>4</b>	<b>Progettazione del sistema</b>	<b>30</b>
4.1	Descrizione del progetto . . . . .	30
4.2	Architettura MRT . . . . .	32
4.3	Principio di funzionamento . . . . .	33
4.3.1	Inizializzazione della camera e del sistema di visione . . . . .	35
4.3.2	Acquisizione frame e estrazione dell'identificativo dei marker . . . . .	35
4.3.3	Trasformazione di coordinate . . . . .	35
4.3.4	Invio dei dati calcolati al sistema . . . . .	38

4.4	Composizione del sistema . . . . .	39
4.5	Librerie utilizzate . . . . .	40
4.6	Interfacciamento del modulo . . . . .	41
4.6.1	Input-output e relazione con altri moduli del sistema . . . . .	41
<b>5</b>	<b>Implementazione</b>	<b>43</b>
5.1	La classe MarkerDetector . . . . .	43
5.2	I metodi . . . . .	44
5.2.1	inicializza() . . . . .	44
5.2.2	captureFrame() . . . . .	45
5.2.3	getPosition() . . . . .	46
5.2.4	sendMessage() . . . . .	47
5.2.5	close() . . . . .	47
<b>6</b>	<b>Integrazione su LURCH</b>	<b>48</b>
6.1	Cos'è LURCH . . . . .	48
6.2	Struttura generale del sistema . . . . .	49
6.3	Moduli interfacciati . . . . .	51
6.4	Integrazione . . . . .	52
6.5	Test effettuati . . . . .	54
6.5.1	Ambiente di test . . . . .	54
6.5.2	Pianificazione delle prove . . . . .	54
<b>7</b>	<b>Conclusioni e sviluppi futuri</b>	<b>71</b>
7.1	Conclusioni . . . . .	71
7.2	Sviluppi futuri . . . . .	72
<b>A</b>	<b>Appendice</b>	<b>76</b>
A.1	Dati risultanti dai test preliminari . . . . .	76
A.1.1	Posizione marker ricavata da ARToolKitPlus . . . . .	76
A.1.2	Tabelle del posizionamento e della rotazione della telecamera nel sistema di riferimento mondo . . . . .	90

# Capitolo 1

## Introduzione

Il presente lavoro si colloca nell'ambito della robotica e della visione artificiale, disciplina che si occupa di analizzare ed estrarre informazioni dall'ambiente circostante, tramite l'utilizzo di telecamere e sensori, al fine di riprodurre la vista umana. In particolare ci si è concentrati sul riconoscimento di particolari oggetti portatori di informazioni utili per il calcolo della posizione del robot.

L'obiettivo che si pone questa tesi è quello di realizzare un sistema di localizzazione assoluta basato sull'utilizzo di landmark passivi, che consenta al robot di conoscere in ogni momento la sua posizione. Il problema della localizzazione e quello dell'autonomia di un robot sono strettamente correlati: un robot, che non è in grado di conoscere la sua posizione nell'ambiente esterno, non è in grado di decidere cosa fare.

Il progetto nasce come un'estensione e modifica di un modulo già esistente e funzionante sul robot LURCH. Tale sistema si basa sull'utilizzo delle librerie ARToolKitPlus, in grado di fornire, tramite l'ausilio di marker, informazioni riguardanti il posizionamento e l'orientazione del robot.

Il nuovo progetto mira a fornire un sistema di localizzazione utilizzabile in più tipologie di ambienti. In special modo, la necessità di implementare tale modulo nasce dalla volontà di equipaggiare con un sistema di localizzazione i robot che si muoveranno in ambienti indoor molto grandi, come le fiere. Siccome il sistema esistente non è in grado di localizzarsi in tali ambienti, in quanto i landmark, essendo posti sul soffitto, sono molto distanti dalla telecamera e quindi non è possibile riprenderli in una maniera sufficientemente nitida da poter essere riconosciuti e analizzati, il presente progetto prevede di spostare i marker dal soffitto alle pareti, posizionando i marker non più in orizzontale, ma in verticale.

Lo sviluppo della tesi è stato articolato come segue:

- **Analisi dello stato dell'arte.** Innanzitutto è stata condotta una

analisi e un approfondimento dei concetti di localizzazione e visione artificiale nell'ambito della robotica, con particolare attenzione alle metodologie di localizzazione indoor esistenti (localizzazione assoluta e relativa). Particolare attenzione è posta sulla localizzazione con marker visuali e sui progetti esistenti.

- **Test preliminari.** Si è poi passati a verificare l'effettiva possibilità di implementazione del sistema, mediante test statici di riconoscimento dell'immagine in funzione della distanza e dell'angolo tra la telecamera e il marker. Un altro aspetto importante della fase di analisi di fattibilità è quello di identificare un ambiente di test che ben riproduca le caratteristiche dell'ambiente finale di utilizzo del sistema (luoghi ampi con illuminazione artificiale come ad esempio fiere o esposizioni). Obiettivo di questi test preliminari è, quindi, quello di acquisire dati, dai quali stimare l'errore di calcolo di posizione e orientamento.
- **Progettazione del sistema.** Nel progettare il sistema si è tenuto conto dei seguenti elementi:
  1. della piattaforma hw e sw in cui il sistema deve essere integrato. In particolare il progetto ha dovuto tener conto delle regole di interfacciamento e comunicazione imposte dal framework Device Communities Development Toolkit (DCDT) e dal Modular Robotic Toolkit (MRT), già sviluppate e utilizzate nei progetti dell'ARLab del Politecnico di Milano.
  2. delle interfacce delle librerie OpenCV per l'inizializzazione, la gestione e la acquisizione delle immagini della webcam.
  3. delle funzionalità offerte dalle librerie ARToolKitPlus per il riconoscimento dei landmark e l'estrazione dei dati di posizione e orientamento.
  4. delle trasformazioni di coordinate necessarie per definire la posizione del robot in un sistema assoluto sulla base delle immagini dei landmark acquisite dal sistema di visione.
- **Implementazione.** Il progetto è stato, poi, implementato in codice C++.
- **Integrazione su LURCH.** La scelta è stata quella di integrare il modulo *VisionExpert* sul robot LURCH, Let Unleashed Robots Crawl the House, carrozzina autonoma sviluppata presso l'AIRLab. I test effettuati hanno avuto lo scopo di verificare i seguenti punti:
  1. la precisione dei dati di posizione ricavati dal modulo *VisionExpert*
  2. l'integrazione del modulo *VisionExpert* all'interno del sistema LURCH esistente.

- **Conclusioni e sviluppi futuri.** Vengono tratte le conclusioni sul lavoro svolto e vengono esposte alcune possibili applicazioni del modulo di visione.



## Capitolo 2

# Stato dell'arte

Con l'evoluzione della tecnologia e della scienza robotica, i robot stanno via via raggiungendo livelli di autonomia e mobilità sempre maggiori. In base al grado di queste caratteristiche i robot possono essere classificati come **non autonomi** (manovrati tramite intervento umano), **semiautonomi** (possono sia essere manovrati dall'esterno sia essere in grado di prendere decisioni autonome) e **completamente autonomi** (robot completamente indipendenti). Con il termine mobilità si intende infatti la capacità di un robot di potersi muovere liberamente nel mondo, mentre con autonomia si intende la capacità di un robot di eseguire i propri compiti senza l'intervento umano.

Affinché si possano raggiungere elevati livelli di autonomia e mobilità occorre fornire al robot degli strumenti che gli consentano di analizzare l'ambiente in cui opera al fine di localizzarsi e pianificare i suoi percorsi nella maniera più autonoma possibile. I robot vengono quindi equipaggiati con sensori sempre più sofisticati e moderni e le nuove tecniche di visione artificiale stanno trovando via via sempre più spazio.

**Visione artificiale** Le tecniche di visione artificiale sono indispensabili per accrescere l'autonomia del robot in quanto sono in grado di riprodurre i processi e le rappresentazioni che stanno alla base della percezione visiva. Il robot diventa quindi capace di interpretare ciò che 'vede'. I sensori visivi acquisiscono una grande quantità di dati, rendendo disponibili al sistema una grandissima quantità di informazioni riguardanti l'ambiente circostante. Tuttavia bisogna essere in grado di gestire tutta questa mole di informazioni, in quanto può rischiare di gravare inesorabilmente sulle prestazioni generali dei sistemi stessi. La visione artificiale si basa su due processi fondamentali:

- Acquisizione dell'immagine, che trattata in forma di matrice di valori relativi all'intensità luminosa, costituisce un input al sistema

- Elaborazione ed estrazione di informazioni elementari dall'immagine, che porta alla formazione del raw sketch, ovvero uno schema grezzo su cui compiere le analisi successive.

**Navigazione** Come detto in precedenza, al fine di garantire un elevato grado di mobilità a un robot è necessario che esso possieda un adeguato sistema di navigazione. Per comprendere meglio cosa si intende per sistema di navigazione, possiamo riassumere questo concetto con in tre semplici domande:

1. **Dove sono?** Il robot deve conoscere la sua posizione per poter prendere decisioni corrette riguardo il movimento futuro. Il compito è svolto dal modulo di localizzazione
2. **Dove devo andare?** Per potersi muovere nell'ambiente e rispettare eventuali vincoli il robot ha la necessità di conoscere la posizione da raggiungere (goal).
3. **Come ci arrivo?** Una volta che il robot conosce la propria posizione e sa qual è il proprio goal, deve decidere anche in che modo raggiungerlo. Questo compito consiste nella pianificazione del percorso da intraprendere.

Gli obiettivi di questo lavoro sono quelli di implementare un sistema di localizzazione in grado di determinare quindi in ogni momento la posizione corrente del robot mediante l'utilizzo di sistemi di visione. Di seguito verrà fatta una panoramica riguardo ai sistemi di localizzazione ad oggi esistenti.

## 2.1 Localizzazione

Come visto nel paragrafo precedente il sistema di localizzazione ha il compito di rispondere alla domanda 'Dove sono?'. Un sistema di localizzazione è quindi un meccanismo che rende disponibile la posizione assoluta di un ente fisico (e.g. un robot, una persona, etc.) in modo accurato e in tempo reale. La posizione viene determinata sulla base delle informazioni sensoriali ricevute. Il problema della localizzazione è alla base dell'autonomia di un robot: se un robot non conosce la propria posizione rispetto all'ambiente esterno, non è in grado di decidere cosa fare. L'esempio forse più noto di sistema di localizzazione è il GPS, sistema di localizzazione disponibile su scala mondiale. Tale sistema, che trova numerosi ambiti applicativi, presenta però la grande limitazione di non poter essere utilizzato in ambienti indoor (all'interno di edifici).

### 2.1.1 La localizzazione indoor

Molti sistemi di posizionamento indoor sono stati sviluppati utilizzando trasmettitori e ricevitori di luce infrarossa, campi magnetici o altre tecniche di comunicazione senza fili. Due esempi di sistemi di localizzazione indoor sono Ubisense e StarGazer. Ubisense è un sistema preciso e affidabile basato su segnali UWB, ma al crescere degli spazi da coprire risulta essere necessario replicare varie volte il sistema, con conseguente aumento del costo. Inoltre è necessario cablare i sensori sia per le necessità di alimentazione sia per le comunicazioni che intercorrono tra di essi e la calibrazione di tutto il sistema non è banale. Anche StarGazer, richiedendo di installare nell'ambiente dei marker riflettenti, comporta un costo proporzionale alla dimensione della zona da coprire.

Un approccio completamente differente, che riduce notevolmente i costi del sistema, è basato, come accennato in precedenza, sull'uso di telecamere e tecniche di visione artificiale. Le tecniche di analisi delle immagini per la localizzazione si possono dividere in tre categorie. La prima utilizza le immagini registrate in un ambiente come base di confronto con le immagini acquisite in tempo reale, la seconda si basa sulla rilevazione di feature 3D da confrontare con un modello e la terza usa dei fiducial marker o tag posti nell'ambiente. L'uso dei primi due approcci ha degli indiscutibili vantaggi estetici, in quanto non richiede di modificare in alcun modo l'ambiente. Un esempio di robot che effettua localizzazione grazie al rilevamento automatico di caratteristiche salienti dell'ambiente è il robot Minerva. La rilevazione di feature 3D è complicata dalla necessità di costruire (manualmente o con tecniche di SLAM) il modello di confronto. L'uso di fiducial marker offre più garanzie di precisione, ma impone la modifica dell'ambiente. Per minimizzare l'impatto visivo di questi sistemi sono stati studiati speciali sistemi per produrre marker nascosti o di aspetto decorativo per l'ambiente.

A seconda dei differenti scenari in cui il robot dovrà operare, si presentano differenti aspetti e istanze del problema di localizzazione:

- **Position Tracking** Il robot conosce la propria posizione iniziale e la localizzazione si occupa di mantenere traccia della posizione durante gli spostamenti del robot nell'ambiente. Le tecniche che risolvono questo problema sono dette tecniche locali.
- **Wake-Up** Il robot non ha idea della propria posizione iniziale. Compito della localizzazione è aiutare il robot a localizzarsi attraverso l'analisi dell'ambiente circostante. Il problema è più complesso del precedente e il robot potrebbe avere indicazioni e stime discordanti sulla propria posizione.
- **Kidnapped Robot** Il robot è perfettamente localizzato, ma viene improvvisamente spostato in una nuova posizione senza esserne ag-

giornato. In realtà il Wake-Up Problem è un sotto-caso del Kidnapped Robot Problem, in cui il robot è avvertito di essere stato spostato e di doversi rilocalizzare.

Esistono due approcci differenti per risolvere il problema della localizzazione: utilizzo di sistemi di localizzazione relativa (la posizione viene stimata in base alle posizioni precedenti del robot in moto) e sistemi di localizzazione assoluta (il calcolo della posizione avviene basandosi su misure correnti).

### 2.1.2 Sistemi di localizzazione relativa

I sistemi di localizzazione relativa (detti anche di dead reckoning) sono i più utilizzati e diffusi. Tali sistemi basano le stime della posizione sulla base di misure relative del moto del robot. Questi metodi di misurazione sono soggetti a errori, in quanto la misura corrente viene calcolata basandosi sulla misura precedente che può essere affetta da errore. In questo modo quindi l'errore si trascina di passaggio in passaggio, rendendo la stima molto rischiosa e imprecisa. L'errore aumenta con il tempo durante la fase di movimento del robot. Al fine di migliorare la stima della posizione, spesso si trovano accoppiati a sistemi di localizzazione assoluta. Le posizioni relative del robot possono essere acquisite generalmente tramite odometria e navigazione inerziale

**L'odometria** L'odometria lavora integrando nel tempo le informazioni acquisite sul movimento. Attraverso l'impiego di sensori (encoder) sulle ruote del robot, si è in grado di ricostruire con l'odometria la distanza percorsa e la direzione di navigazione. Il vantaggio fondamentale dell'odometria è la semplicità, in quanto garantisce misure rapide e poco costose in termini computazionali. Lo svantaggio dell'odometria è costituito dagli errori sistematici: l'attrito, lo slittamento delle ruote sul terreno, l'eventuale imprecisione degli encoder. Piccoli errori sull'orientazione del robot si ripercuotono come ampi errori sulla posizione.

**La navigazione inerziale** Nella navigazione inerziale vengono utilizzati giroscopi e accelerometri per misurare velocità di rotazione e accelerazione. Come nel caso dell'odometria, la navigazione inerziale è molto sensibile alle variazioni del terreno, e dovendo integrare i dati una o più volte nel tempo è possibile che gli errori crescano con il tempo.

### 2.1.3 Sistemi di localizzazione assoluta

I sistemi di localizzazione assoluta calcolano la posizione assoluta del robot senza basarsi su misure precedenti. La posizione del robot è infatti calcolata sulla misura corrente, e non viene stimata integrando misure passate come

avviene nei sistemi di localizzazione relativa. Il pregio di questi sistemi è naturalmente l'assenza di errori sistematici che crescono in base al tempo durante il movimento del robot.

I sistemi di localizzazione assoluta si dividono in metodi basati su landmark e metodi basati su mappe topologiche.

**Metodi basati su mappe** I metodi di Model Matching e di confronto di mappe calcolano la posizione del robot in un modo diverso dai metodi precedenti. I sensori del robot cercano features geometriche distintive dell'ambiente esterno e le confrontano con informazioni raccolte a priori.

Esistono due approcci diversi per rappresentare e confrontare mappe:

- **Mappe geometriche:** contengono la descrizione dell'ambiente in un sistema di coordinate globale
- **Mappe topologiche:** l'ambiente è rappresentato tramite un'interconnessione di nodi (i luoghi dell'ambiente) e archi (le azioni da intraprendere per spostarsi da un nodo all'altro).

I metodi basati su mappe sono in generale molto precisi, tuttavia richiedono la disponibilità di una grande quantità di informazioni provenienti dai dati sensoriali e necessitano di una capacità computazionale alquanto elevata.

**Metodi basati su landmark** I landmark sono features (letteralmente - caratteristiche) dell'ambiente che il robot è in grado di rilevare e riconoscere. La navigazione tramite landmark è una delle tecniche messe a disposizione per la localizzazione. A partire dai dati provenienti da sensori e telecamere il sistema è in grado di analizzare e ricavare dati riguardanti la sua localizzazione.

In base al modo in cui interagiscono con il robot, i landmark si distinguono tra passivi e attivi

**Landmark attivi** I landmark attivi, per lo più noti come boe (beacons), sono landmark che inviano al robot informazioni riguardanti la loro posizione. Generalmente si tratta di apparecchiature radio. Normalmente per calcolare la posizione assoluta del robot attraverso l'utilizzo di landmark attivi si applica il sistema della triangolazione. Questo metodo utilizza le distanze e gli angoli relativi a tre o più landmark 'visibili' dal robot. Attraverso l'analisi di tali dati il robot è in grado di calcolare la propria posizione e orientazione nel sistema di riferimento assoluto.

**Landmark passivi** La metodologia che utilizza i landmark passivi si basa sull'idea di porre dei markers nell'ambiente in cui si muoverà il robot. I

markers sono dei disegni contenenti informazione, ovvero una sorta di codice a barre. L'immagine è portatrice di un codice identificativo univoco. Una volta riconosciuto uno di questi disegni, il sistema di localizzazione è in grado di estrarre le informazioni relative alla posizione del marker nel sistema di riferimento. Sulla base di queste informazioni, applicando tecnologie di visione artificiale, il robot è in grado di determinare la posizione corrente.

Il progetto, oggetto di questa tesi, riguarda lo sviluppo di un sistema di localizzazione assoluta basato su marker passivi, in quanto si è ritenuto il più rispondente ai requisiti: un sistema di localizzazione per ambienti indoor di grandi dimensioni e dal costo limitato.

## **2.2 Localizzazione con landmark artificiali**

### **2.2.1 Fiducial Marker**

I sistemi basati su fiducial marker sono utilizzati in svariati ambiti, dalla robotica alle applicazioni di realtà aumentata. Un marker è solitamente costituito da una superficie planare di forma nota con un contenuto informativo. In questo lavoro, sulla base delle esperienze precedenti, è stato preso in considerazione il sistema ARToolKitPlus.

I marker di questi sistemi hanno forma quadrata con un vistoso bordo nero e possono essere stampati con una comune stampante su un foglio di carta sufficientemente rigido, a un costo irrisorio. Il contenuto informativo è costituito da un'immagine che viene confrontata con un modello precedentemente memorizzato e che varia da sistema a sistema. Il processo di identificazione di marker in un'immagine segue in genere i seguenti passi:

1. Identificazione dei possibili marker presenti nell'immagine.
2. Riproiezione dell'immagine in una vista frontale.
3. Confronto dell'immagine del marker con un modello e restituzione di un codice che identifica univocamente il contenuto informativo del marker o un codice di errore qualora il contenuto informativo non corrisponda a nessun modello noto.

L'output di un sistema di identificazione di marker è una lista di codici univoci di identificazione dei marker riconosciuti in un'immagine e la loro posizione e orientamento espressa relativamente al sistema di riferimento solidale con la telecamera. Per una corretta stima delle distanze è necessario che le dimensioni del marker siano note a priori con precisione. La presenza del prefisso AR nel nome dei sistemi citati indica che il loro sviluppo è stato principalmente legato a studi di realtà aumentata (Augmented Reality)

### 2.2.2 ARToolKitPlus

ARToolKitPlus è un'evoluzione di ARToolKit; utilizza lo stesso metodo del predecessore per riconoscere i contorni dei marker, introducendo però un sistema di autocalibrazione della soglia che lo rende più flessibile alla variabilità delle condizioni di luce.

I marker di ARToolKitPlus sono di forma quadrata con un marcato bordo nero (vedi figura 3.1).

Il contenuto informativo è costituito da un'immagine in scala di grigi. Il processo di riconoscimento dei marker in un frame è basato su una sogliaatura che permette di ricavare un'immagine binaria in bianco e nero.

Dall'immagine binaria vengono estratti i contorni e identificate le linee. I quadrilateri, ovvero le catene di quattro linee che formano figure chiuse, sono soggetti ad analisi per il riconoscimento del contenuto informativo.

I pattern che possono essere utilizzati come contenuto informativo sono memorizzati all'interno del sistema e confrontati, dopo aver effettuato il raddrizzamento dell'immagine, con l'uso della correlazione.

È possibile utilizzare sia marker con contenuto informativo generico (i.e., un'immagine da confrontare come nel caso di ARToolKit) sia con un identificativo digitale (i.e. un codice a barre bidimensionale). I marker predefiniti con contenuto digitale sono di due tipi:

- simple ID con 512 marker disponibili (vedi figura 3.1)
- BCH ID con 4096 marker.

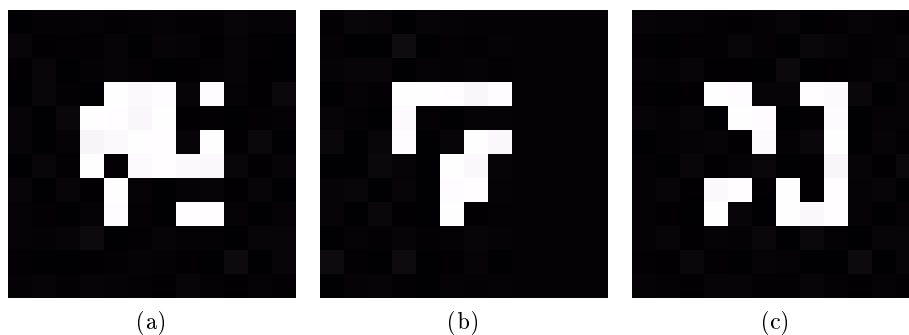


Figura 2.1: Esempi di marker simple ID

Le principali migliorie introdotte da ARToolKitPlus rispetto ad ARToolKit sono:

- Meccanismo di soglia automatica per la creazione dell'immagine in bianco e nero da analizzare
- Compensazione del fenomeno di vignettatura (zone scure negli angoli dell'immagine) che migliora il riconoscimento dei marker anche quando questi sono vicini al bordo dell'immagine

- Semplificazione del processo di calibrazione della telecamera, che si può effettuare con qualsiasi sistema standard (e.g. Camera Calibration Toolbox for Matlab)
- Introduzione di un algoritmo di stabilizzazione della stima della matrice di rototraslazione (Robust Planar Pose Tracking, RPP ) che porta dal sistema di riferimento telecamera al sistema di riferimento tag.

## 2.3 Progetti esistenti

Il lavoro effettuato si appoggia su un sistema di localizzazione già esistente, sviluppato nel AIRLab del Politecnico di Milano nell'ambito del progetto LURCH. Il sistema sviluppato utilizza le ARToolKit Plus come librerie per la gestione dei marker. Il sistema è stato progettato per utilizzare dei marker della grandezza di 15x15 cm posizionati sul soffitto. (vedere se estendere questa parte)

Tale sistema è inserito in un'architettura modulare chiamata MRT, Modular Robotic Toolkit. MRT è un'architettura software per i robot autonomi, grazie alla quale è possibile combinare insieme e personalizzare moduli sw per creare applicazioni con poco tempo e sforzo. Questo framework si basa su un'architettura modulare, dove i vari moduli interagiscono tra di loro utilizzando un linguaggio comune attraverso un ambiente orientato al passaggio dei messaggi. Questo approccio modulare garantisce benefici in termini di flessibilità e riusabilità. Tutti i progetti sviluppati nell'AIRLab utilizzano MRT come framework.

Su LURCH il membro che attualmente si occupa di calcolare la posizione della carrozzina nel sistema di riferimento assoluto è chiamato VisionExpert, con ovvio riferimento al suo compito di analizzare delle immagini tramite tecniche di visione artificiale.

Tenuti in considerazione questi aspetti, l'obiettivo del lavoro sarà quello di creare un nuovo modulo 'VisionExpert', compatibile con l'architettura MRT, in grado di fornire un sistema alternativo di localizzazione. Partendo dal progetto di localizzazione esistente si intende sviluppare un modulo in grado di autolocalizzarsi utilizzando dei marker posti in posizione verticale (nel progetto esistente i marker sono posti orizzontalmente sul soffitto). L'idea di questa modifica nasce dalla necessità di avere un sistema di localizzazione alternativo per quei luoghi in cui l'utilizzo del soffitto non è possibile (fiere espositive-capannoni con soffitti molto alti, ambienti aperti).



## Capitolo 3

# Test preliminari

Prima di iniziare la progettazione effettiva del sistema di localizzazione si è ritenuto indispensabile effettuare una serie di test, basati sull'uso delle librerie ARToolKitPlus, per capire se la soluzione dei marker in verticale sia in grado di dare gli risultati comparabili, in termini di precisione e affidabilità, della soluzione attualmente in uso, che prevede l'utilizzo di marker posti sul soffitto in posizione orizzontale.

Sono, quindi, stati condotti una serie di test con lo scopo di valutare sperimentalmente gli errori in funzione della distanza, dell'angolo di rotazione del marker e delle dimensioni del marker. Il risultato consiste in una mappa di riconoscimento del marker e una mappa degli errori di posizione e orientazione rilevati.

### 3.1 Ipotesi e considerazioni

Per poter utilizzare ARToolKitPlus è stato innanzitutto necessario calibrare la telecamera. Questa operazione è stata effettuata utilizzando Camera Calibration Toolbox for Matlab che permette di stimare sia la matrice fondamentale della telecamera sia i parametri di correzione della distorsione radiale dell'ottica.

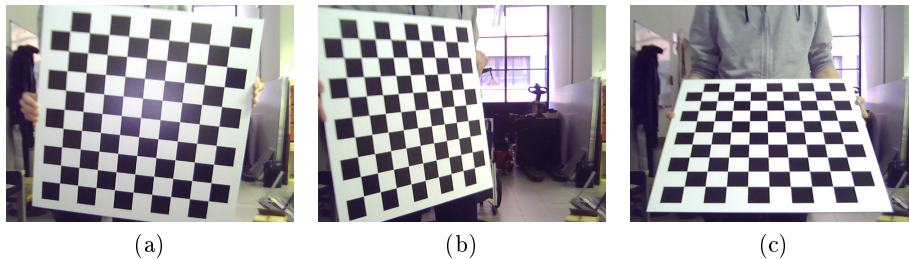


Figura 3.1: Scacchiera utilizzata per la calibrazione della camera

Durante lo svolgimento delle prove sono state tenute in considerazione le seguenti ipotesi.

Si è calcolato sperimentalmente che l'angolo di vista della telecamera è pari a  $\pm 20^\circ$  rispetto all'asse ottico, pertanto le prove sono state effettuate entro questi limiti.

Sulla base di prove preliminari effettuate si è riscontrata una simmetria attorno all'asse ottico della telecamera e di conseguenza i test sono stati eseguiti solo su metà campo visivo.

L'angolo massimo attorno all'asse y in cui il marker viene riconosciuto è pari a  $45^\circ$ - $50^\circ$  circa. Pertanto sulla base di questi dati si è scelto di effettuare i test ruotando il marker attorno all'asse y di  $0^\circ$ ,  $\pm 15^\circ$  e  $\pm 30^\circ$  (spostando il marker lungo l'asse x si aggiunge un'ulteriore componente di angolo).

Nell'effettuare le prove gli assi di riferimento della telecamera e del marker considerati sono quelli riportati nell'immagine seguente (vedi figura 3.2):

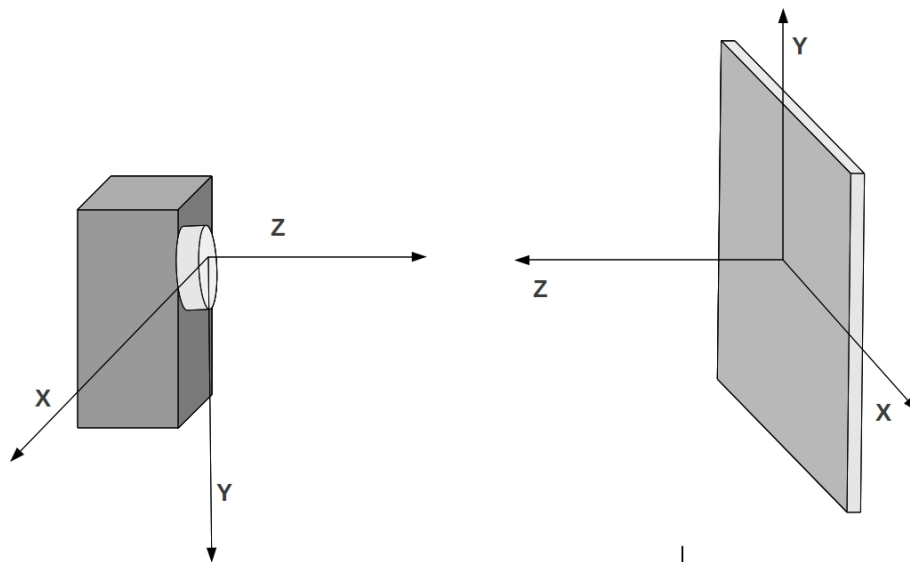


Figura 3.2: Sistemi di riferimento marker e webcam

I test sono stati effettuati prendendo per ogni posizione/orientazione del marker 5 immagini in successione con telecamera in modalità filmato al fine di riprodurre le condizioni reali di funzionamento.

Nei calcoli è stato poi utilizzato il valor medio di queste cinque misurazioni in quanto la varianza misurata tra questi campioni è risultata essere molto bassa (ordini di grandezza tra  $10^{-9}$  e  $10^{-4}$ ).

Al fine di semplificare il calcolo degli errori in termini di posizionamento e angolo dell'asse ottico della telecamera si è ipotizzato di far coincidere il sistema di riferimento mondo con quello della telecamera, ponendo il sistema 'mondo' ruotato di  $180^\circ$  attorno all'asse x come riportato nel disegno. (vedi

figura 3.3)

Nelle prove effettuate il marker è stato posto all'incirca alla stessa altezza della telecamera. Si è deciso di utilizzare i marker con disegno più semplice, in modo tale da semplificare il loro riconoscimento.

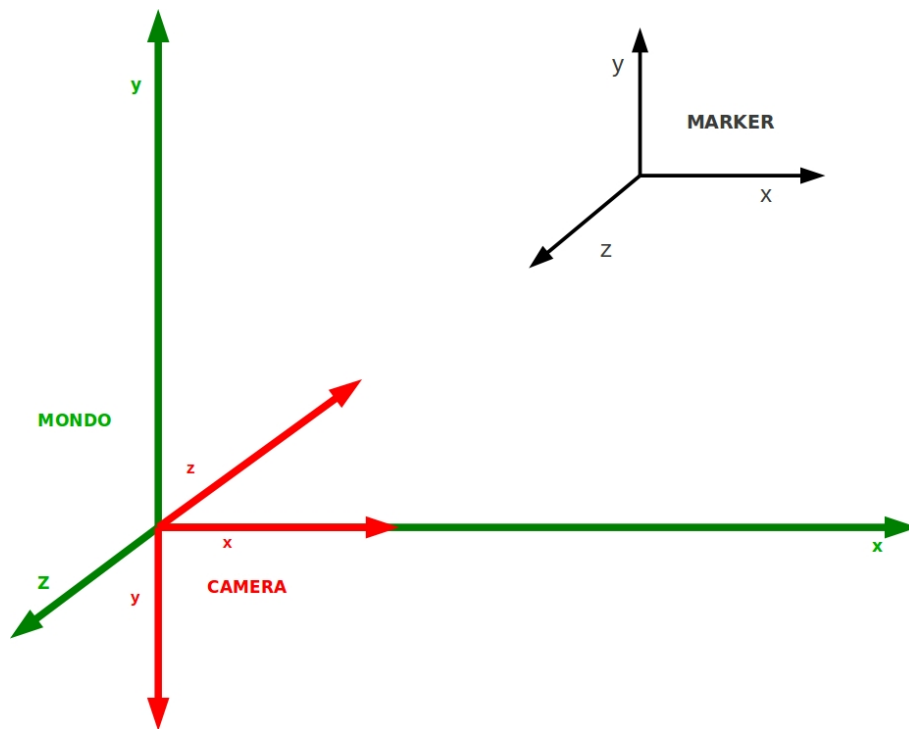


Figura 3.3: Sistema di riferimento di marker, webcam e mondo

## 3.2 Ambiente di test

Le prove sono state effettuate in una palestra illuminata con luci al neon. Un'intera parete della palestra è composta da vetrate, dalle quali entra abbondante luce. Le prove sono state effettuate tra le ore 11.00 e 16.00, avendo quindi un ambiente illuminato sia da luce artificiale al neon sia dalla luce naturale entrante dalle finestre. (vedi figura 3.4)

## 3.3 Descrizione delle prove

### 3.3.1 Attrezzature utilizzate

- Webcam TECHSOLO TCA-4900
- cavalletto con possibilità di rotazione sui tre assi (x,y e z), dotato di indicazione dei gradi di rotazione
- Software library ArtToolKit Plus per il calcolo della posizione e orientamento relativo dei marker
- Matlab per l'analisi dei dati acquisiti



Figura 3.4: Ambiente in cui sono stati effettuati i test

### 3.3.2 Dimensioni del Marker

Sono state effettuate prove con marker di due dimensioni differenti:

- Marker piccolo: 15cm x15 cm
- Marker grande: 25cm x25 cm

Entrambi i marker utilizzati sono stati plastificati come rivestimento anti-usura.

### 3.3.3 Analisi degli errori

Una volta ricavata la matrice  $T_C^W$  è possibile analizzare gli errori tra la posizione teorica della telecamera e la posizione calcolata tramite la visione del marker.

Per analizzare gli errori di posizionamento ci si è concentrati sull'analisi della quarta colonna della matrice  $T_C^W$ . Siccome è stato scelto di posizionare il sistema di riferimento mondo coincidente con quello della telecamera (vedi figura 3.3) il valore atteso delle componenti  $t_x$  e  $t_z$  è 0. Il valore letto sulla quarta colonna coincide quindi anche con l'errore di posizionamento stesso. Si può considerare costante il valore della traslazione lungo l'asse y e quindi tralasciare la sua analisi.

Per analizzare gli errori di rotazione, invece, si è posta l'attenzione sulla direzione dell'asse ottico della telecamera, ovvero l'asse z. Siccome il sistema mondo è stato posizionato coincidente con quello della telecamera ma ruotato di  $180^\circ$  attorno all'asse x (vedi figura 3.3) ci si aspetta di trovare l'asse ottico della camera con la stessa direzione dell'asse z mondo e verso opposto. Per analizzare ciò si deve quindi analizzare la terza colonna della matrice  $T_C^W$ , rappresentante le componenti del versore z della telecamera. Ricavando l'arcotangente della componente x sulla componente z del versore si ricava l'angolo di cui il sistema di riferimento camera è ruotato rispetto all'asse z del mondo.(vedi figura 3.5)

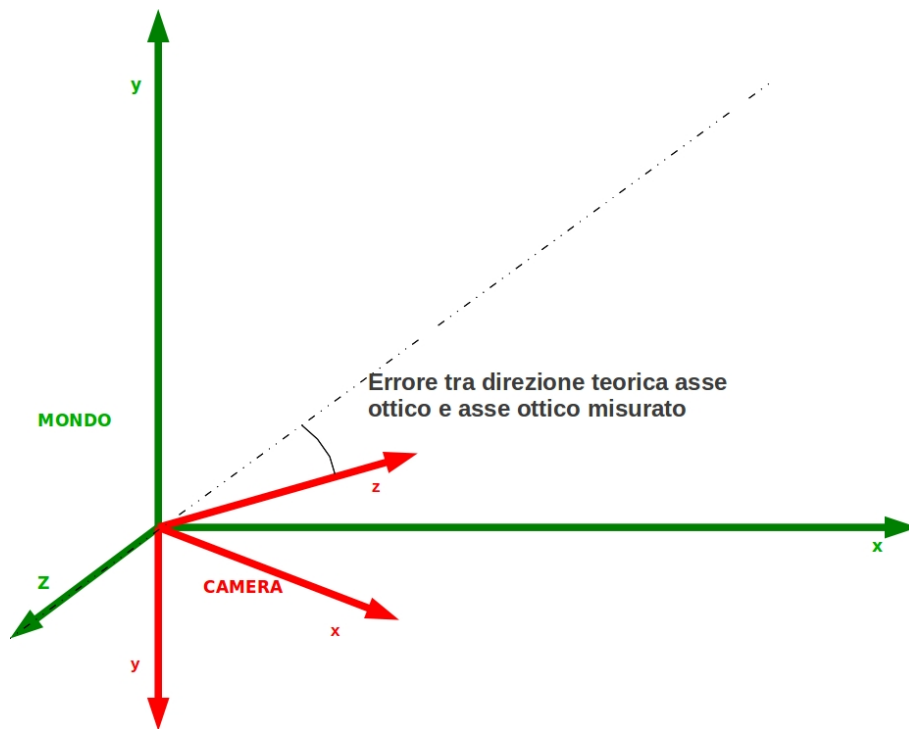


Figura 3.5: Errore della direzione dell'asse ottico della camera

## 3.4 Analisi dei risultati

### 3.4.1 Marker piccolo

**Analisi dati** Le misurazioni fatte con inclinazione di  $0^\circ$  attorno all'asse  $x$  sono state piuttosto deludenti in quanto la percentuale di riconoscimento nelle varie posizioni è stata veramente bassa. Il riconoscimento arriva infatti a 3m di distanza lungo l'asse  $z$  e 1m lungo l'asse  $x$ , solo in alcune particolari condizioni. Queste condizioni sono il marker inclinato di  $15^\circ$ -  $30^\circ$  a destra attorno all'asse  $y$ . In questo caso l'errore di posizione delle coordinate  $x$  e  $z$  risulta essere notevole (circa 2 m di errore).

La causa di questo risultato deludente è da imputare ai riflessi che la luce esterna causa sul marker plastificato.

Di seguito alcune immagini di marker non riconosciuti per questo motivo. (vedi figura 3.6)

Al fine di eliminare il riflesso le misure sono state ripetute con un angolo di inclinazione del marker di  $15^\circ$  attorno all'asse  $x$  (in senso antiorario). In tali condizioni il riconoscimento è garantito fino ai 3m di distanza lungo l'asse  $z$  tra l'80% e il 100%. Ci sono però due eccezioni:

1. il marker posto a 1m e 2m di distanza lungo l'asse  $z$  con rotazione di  $30^\circ$  a destra attorno all'asse  $y$  (vedi figura 3.7a e 3.7b)

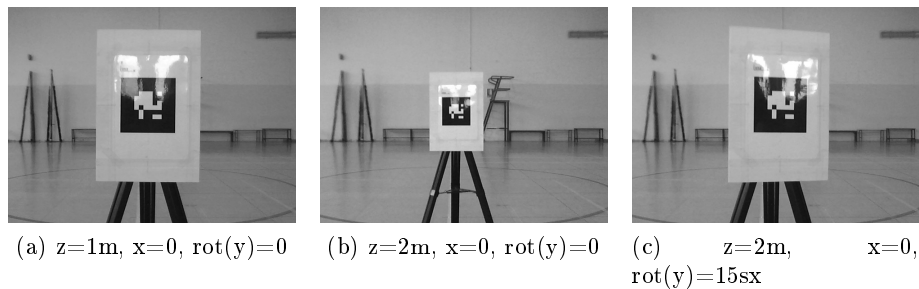


Figura 3.6: Marker non riconosciuti

2. il marker posto a 3 m di distanza lungo z e 1m lungo x e con rotazione  $30^\circ$  a sinistra attorno all'asse y (vedi figura 3.7c)

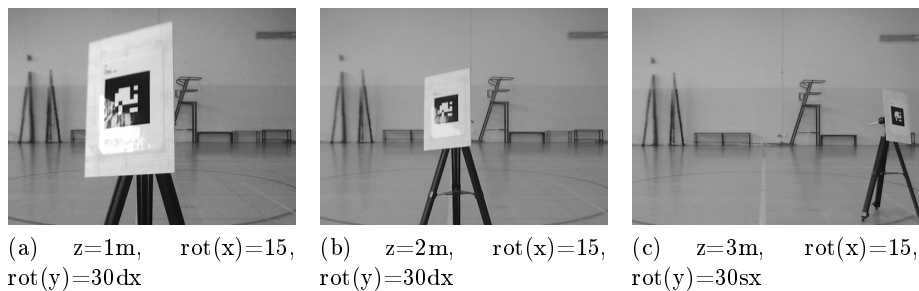


Figura 3.7: Marker non riconosciuti

Analizzando le immagini sopra riportate si nota che il mancato riconoscimento del punto 1) precedente è dovuto nuovamente a un problema di riflessi luminosi sul marker. Le motivazioni legate al mancato riconoscimento di cui al punto 2) sono da imputare al fatto che l'angolo massimo attorno all'asse y in cui i marker vengono riconosciuti è pari a  $45-50^\circ$ . In questo caso oltre all'inclinazione di  $30^\circ$  del marker c'è da aggiungere un'ulteriore rotazione dovuta allo spostamento del marker lungo l'asse x. Essendo spostato di 3m lungo z e 1m lungo x l'angolo di rotazione aggiuntivo è pari  $\arctan(1/3)=18,5^\circ$ . Pertanto in questa situazione la telecamera vede il marker ruotato di  $49^\circ$  circa, angolazione per la quale il marker inizia a non essere più riconosciuto.

A 4 m di distanza lungo l'asse z i il marker viene riconosciuto solo in particolari condizioni. L'errore di posizionamento associato a questa distanza è di molto superiore rispetto a quello dei marker posti fino a 3m di distanza.

**Considerazioni** Dalle prove effettuate si può ritenere che il comportamento del marker inclinato di  $0^\circ$  attorno all'asse x è assimilabile a quello inclinato di  $15^\circ$  a patto che venga garantita l'assenza di riflessi luminosi sul marker.

L'analisi degli errori viene quindi eseguita sulle prove effettuate con inclinazione di  $15^\circ$  attorno all'asse x, generalizzando i risultati ottenuti anche alla situazione di marker non inclinato attorno all'asse x.

Dalle prove effettuate si può concludere che se i marker vengono posizionati in modo tale da garantire l'assenza di riflessi luminosi il riconoscimento del marker viene garantito fino a 3 m di distanza lungo l'asse z e 1m lungo l'asse x con risultati sufficientemente precisi (tenendo conto che il posizionamento effettuato durante i test risente di errori entro i 5cm).

Di seguito gli errori medi di posizionamento lungo z (0-3m) e x (0-1m all'interno dell'angolo di vista di  $20^\circ$ ) e gli errori di rotazione (errore dell'angolo rispetto all'asse ottico teorico).

	Errore medio	Varianza	Dev. standard
Asse x	-0,09 m	0,030 $m^2$	0,17m
Asse Z	-0,07m	0,13 $m^2$	0,11m
Angolo	-0,718°	9,44°	3,1°

Per quanto riguarda il riconoscimento del marker posizionato a 4m di distanza lungo l'asse z si nota un netto peggioramento dei risultati, come evidenziato nella tabella seguente. Anche il calcolo dell'angolo perde di precisione, con una deviazione standard molto elevata. Questo valore così alto di varianza ci porta a considerare non attendibili le posizioni calcolate a 4 m di distanza. A causa dell'elevata distanza tra telecamera e marker è possibile che i contorni dell'immagine non siano riconosciuti correttamente e quindi si ha un errato calcolo dei valori di rotazione e posizionamento del marker.

	Errore medio	Varianza	Dev. standard
Asse x	0,91 m	3,35 $m^2$	1,83 m
Asse Z	-1,02 m	0,24 $m^2$	0,48 m
Angolo	13,2°	644,41°	25,4°

Di seguito sono riportati i grafici della percentuale di riconoscimento (vedi figura 3.8) e degli errori di angolo (vedi figura 3.11) e posizionamento (vedi figura 3.9,3.10) calcolati con un marker inclinato di  $15^\circ$  attorno all'asse x e  $0^\circ$  attorno all'asse y.



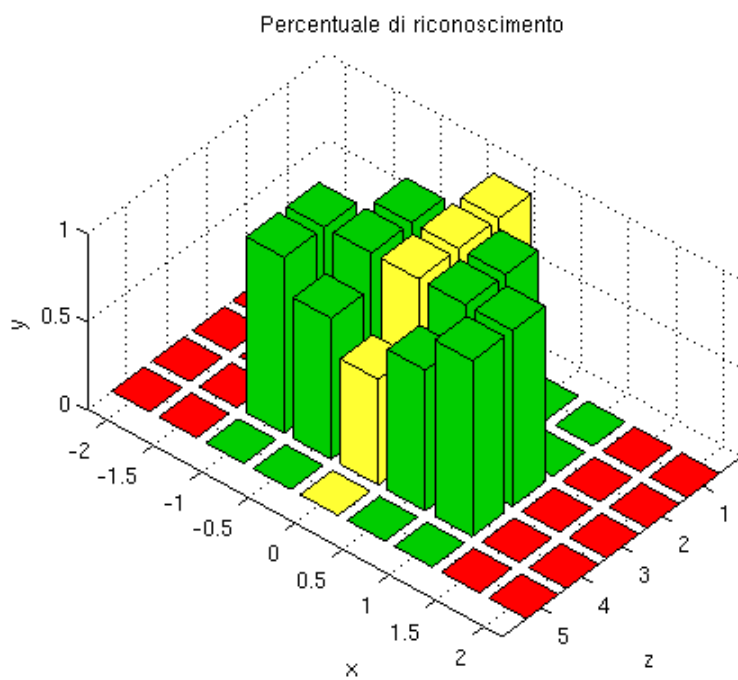


Figura 3.8: Percentuale di riconoscimento

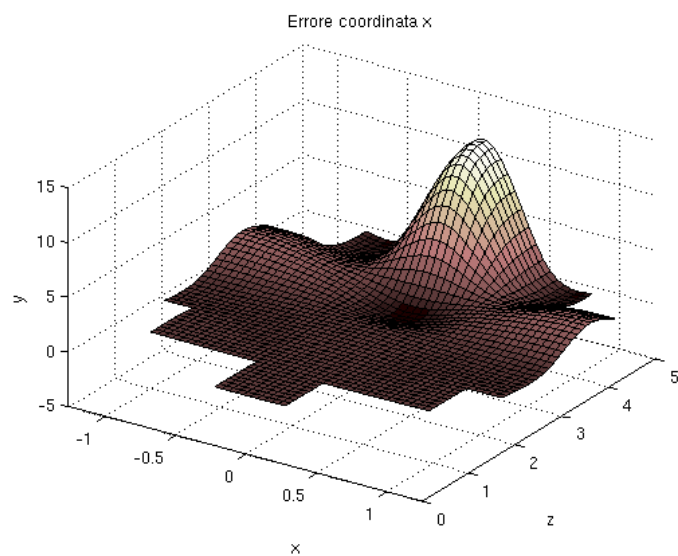


Figura 3.9: Errore coordinata x

NOTA: nei punti in cui il marker non viene riconosciuto è stato impostato come errore il valore costante 10

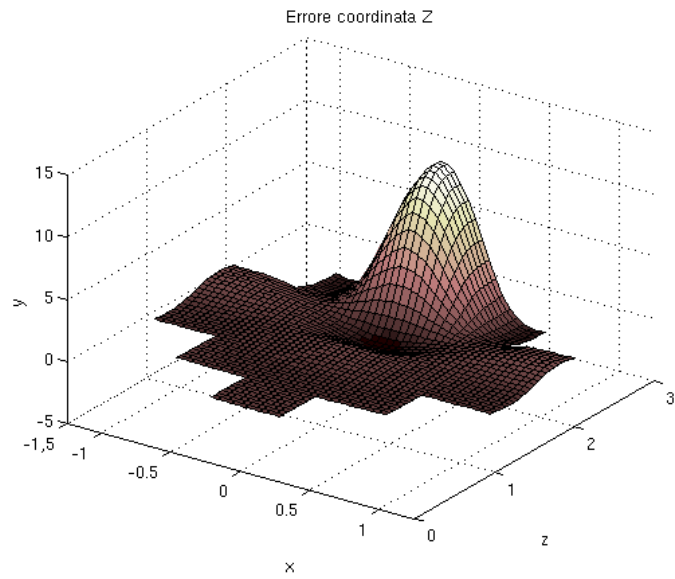


Figura 3.10: Errore coordinata z

NOTA:nei punti in cui il marker non viene riconosciuto è stato impostato come errore il valore costante 10

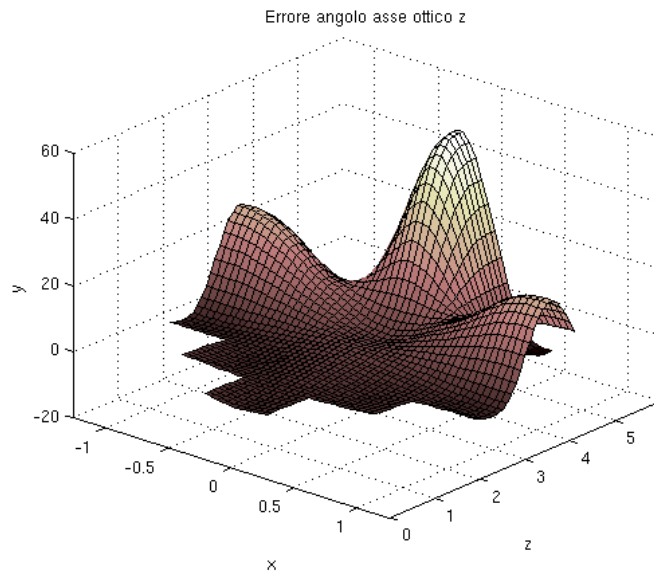


Figura 3.11: Errore angolo asse ottico

NOTA:nei punti in cui il marker non viene riconosciuto è stato impostato come errore il valore costante 40

### 3.4.2 Marker grande

**Analisi dati** Dati i problemi dovuti ai riflessi di luce riscontrati nelle prove effettuate con il marker piccolo, si è deciso di incominciare ad effettuare le prove con il marker grande inclinandolo di  $15^\circ$  attorno all'asse x.

Tali prove hanno evidenziato una ottima percentuale di riconoscimento del marker (tra 80% e 100%) fino alla distanza di 5m lungo l'asse z, salvo rare eccezioni, quali:

1. marker inclinato di  $30^\circ$  a destra attorno all'asse y e posto alla distanza di 1m (immagine 4) e 2m (immagine 5) lungo l'asse z
2. marker inclinato di  $30^\circ$  a sinistra attorno all'asse y e posto alla distanza di 5m lungo z e di 1m lungo x (immagine 6)

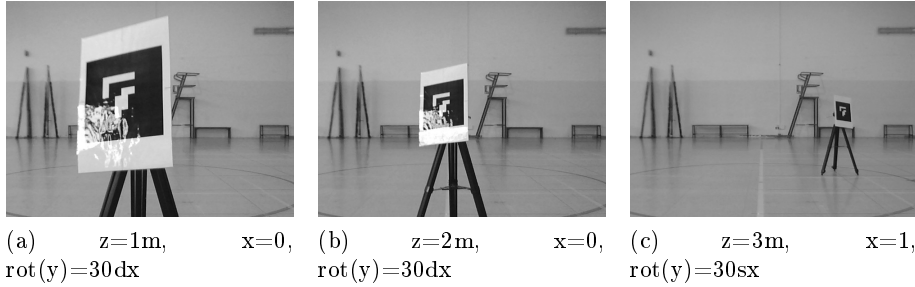


Figura 3.12: Marker non riconosciuti

Per il punto 1 si può dedurre che la causa del non riconoscimento è attribuibile al riflesso presente sul marker. Per il punto 2 invece la causa del mancato riconoscimento potrebbe essere l'elevata angolazione del marker ( $30^\circ + \arctan(1/5) = 12^\circ$ ) aggiunta alla lontananza in cui si trova il marker.

A 6 m di distanza lungo l'asse z il marker viene riconosciuto solo in particolari condizioni. L'errore di posizionamento associato a questa distanza è di molto superiore rispetto a quello dei marker posti fino a 5 m di distanza. A 7 m nessun marker viene più riconosciuto, ad eccezione del marker inclinato  $0^\circ$  attorno all'asse x e sull'asse ottico della telecamera.

Come già osservato per il marker piccolo, anche il marker grande inclinato di  $15^\circ$  e quello inclinato di  $0^\circ$  attorno all'asse x presentano un comportamento analogo, fatto salvo che venga garantita l'assenza di luce riflessa sul marker. Nei test effettuati con il marker non inclinato attorno all'asse x ci si è quindi soffermati a testare la zona critica, ovvero la fascia dei 6 m di distanza lungo z. Tali test hanno evidenziato un miglioramento nella percentuale di riconoscimento. Tale miglioramento non è però sufficiente per considerare la fascia dei 6m come fascia di buona visione della telecamera.

**Considerazioni** Con l'utilizzo di un marker più grande si è aumentata la zona di buon riconoscimento dei marker, passata dai 3 m con il marker piccolo (15 cm x 15 cm) ai 5 m del marker grosso (25 cm x 25 cm).

Dalle prove effettuate si può concludere che se i marker vengono posizionati in modo tale da garantire l'assenza di riflessi luminosi il riconoscimento del marker viene garantito fino a 5 m di distanza lungo l'asse z e 2 m lungo l'asse x con risultati sufficientemente precisi (tenendo conto che il posizionamento effettuato durante i test risente di errori entro i 5 cm).

Di seguito gli errori medi di posizionamento lungo z (0-5m) e x (0-2m all'interno dell'angolo di vista di 20°) e gli errori di rotazione (errore dell'angolo rispetto all'asse ottico teorico).

	Errore medio	Varianza	Dev. standard
Asse x	0,024m	0,0644 $m^2$	0,25m
Asse Z	0,2m	0,021 $m^2$	0,14m
Angolo	1,26°	10,58°	3,25°

Per quanto riguarda il riconoscimento del marker posizionato a 6m di distanza lungo l'asse z si nota un netto peggioramento dei risultati, soprattutto per quanto riguarda l'asse x e l'angolo, come mostrato in tabella.

	Errore medio	Varianza	Dev. standard
Asse x	0,67 m	1,30 $m^2$	1,14 m
Asse Z	-0,11m	0,23 $m^2$	0,48 m
Angolo	7,56°	100°	10°

Di seguito sono riportati i grafici della percentuale di riconoscimento e degli errori di angolo e posizionamento calcolati con un marker inclinato di 15° attorno all'asse x e 0° attorno all'asse y.

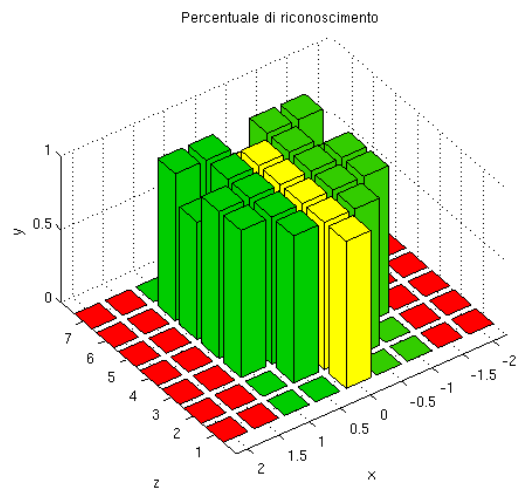


Figura 3.13: Percentuale di riconoscimento

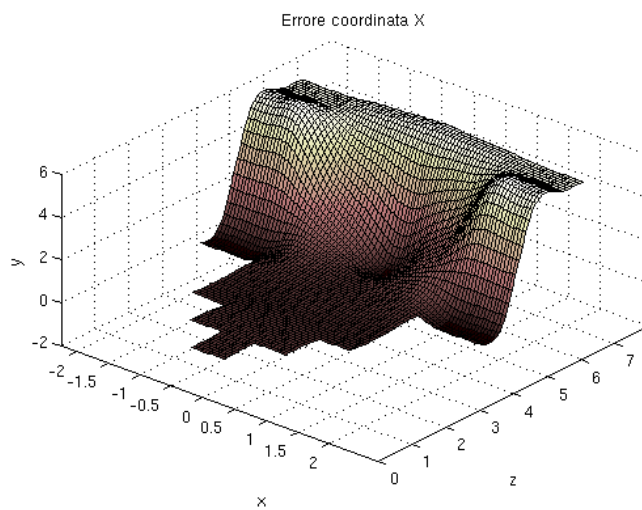


Figura 3.14: Errore coordinata x

NOTA: nei punti in cui il marker non viene riconosciuto è stato impostato come errore il valore costante 5

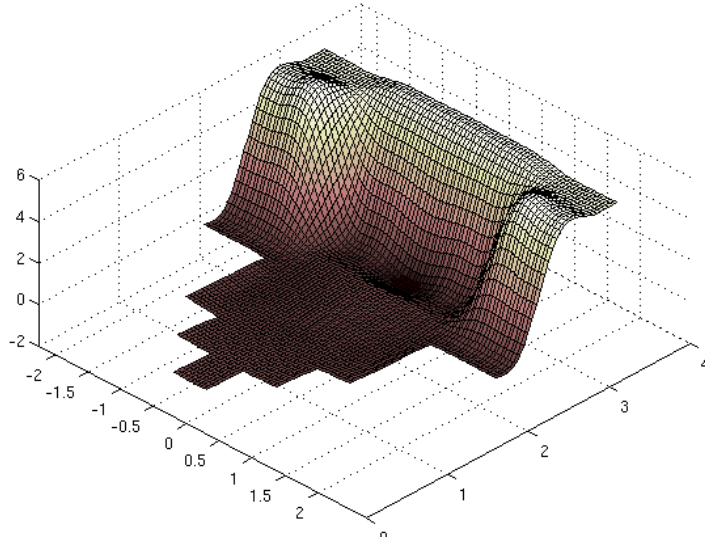


Figura 3.15: Errore coordinata z

NOTA:nei punti in cui il marker non viene riconosciuto è stato impostato come errore il valore costante 5

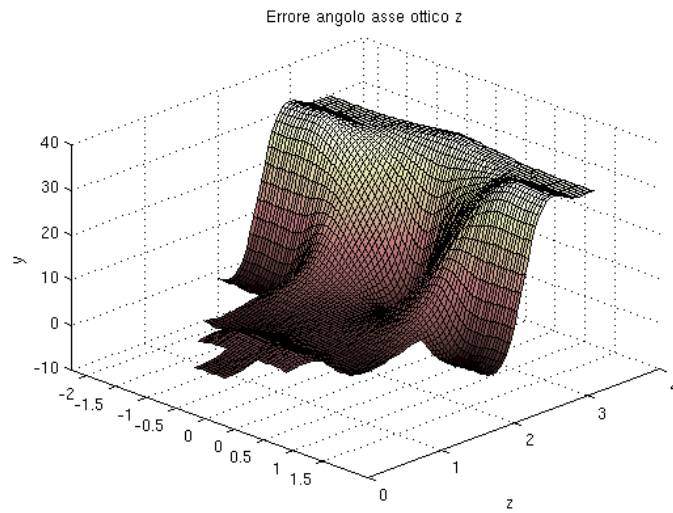


Figura 3.16: Errore angolo asse ottico

NOTA:nei punti in cui il marker non viene riconosciuto è stato impostato come errore il valore costante 30

### 3.5 Considerazioni generali

**Luminosità** Oltre ai test sopra descritti (effettuati in una palestra illuminata con luce artificiale al neon), sono stati compiuti test in un luogo aperto sfruttando l'illuminazione naturale. A fronte di analoghi strumenti di misura ed analisi e analogo metodo di posizionamento dei marker sono stati ottenuti risultati totalmente differenti. Le percentuali di riconoscimento dei marker sottoposti a luce naturale in ambiente aperto sono state nettamente inferiori a quelle dei marker illuminati con luce al neon e ambiente chiuso. Da ciò si può capire come il tipo di illuminazione sia un fattore determinante nel riconoscimento dei marker. È pertanto opportuno testare il sistema in ogni ambiente in cui si desidera utilizzarlo.

**Posizionamento e tipo di marker** È opportuno fare un'osservazione riguardo al problema nel quale ci siamo imbattuti ripetutamente durante le prove: i riflessi di luce sul marker. Il fatto di avere un marker plastificato non ha sicuramente aiutato in questo senso. Nelle nostre prove abbiamo provato ad ovviare a questo inconveniente ruotando il marker di  $15^\circ$  attorno all'asse x in modo da non trovarci con il marker e il piano camera paralleli (condizione di massimo riflesso). Ciò ha migliorato la situazione ma il problema non è stato risolto totalmente, in quanto in alcune posizioni il riflesso era ancora presente. Probabilmente è opportuno pensare di utilizzare qualche forma differente di rivestimento del marker, ad esempio un rivestimento opaco. Durante la fase di installazione dei marker è opportuno controllare, nel limite del possibile, l'entità dei riflessi che potrebbero interessare lo stesso, in modo da minimizzare l'interferenza con essi.

## Capitolo 4

# Progettazione del sistema

Una volta testata l'effettiva fattibilità del progetto si è potuti passare alla fase progettuale e implementativa.

Il lavoro effettuato si appoggia su un sistema di localizzazione già esistente, sviluppato presso l'IRLab del Politecnico di Milano nell'ambito del progetto LURCH.

Prerequisito importante da tenere in considerazione è il contesto in cui il progetto nasce. Il modulo che si andrà a sviluppare dovrà essere integrabile all'interno di sistemi basati su architettura MRT, un'architettura modulare che consente una rapida e efficiente aggregazione di più moduli in un'unica applicazione.

Le fasi di progetto, dovendo tenere conto di questo aspetto, sono state così strutturate: una prima fase di implementazione del modulo e una seconda di integrazione dello stesso nel sistema esistente. Verrà prestata maggiore attenzione sugli aspetti dell'integrazione nel capitolo successivo, in cui sarà esposta la reale integrazione effettuata su LURCH.

Nel capitolo corrente verrà, invece, spiegato la struttura e il principio di funzionamento del modulo, le librerie e i tool di supporto utilizzati, l'interfacciamento del modulo con il sistema esterno (input e output) e il sistema di gestione di interazione e comunicazione tra moduli.

### 4.1 Descrizione del progetto

Il progetto sviluppato si appoggia su un sistema di localizzazione già esistente, sviluppato nel IRLab del Politecnico di Milano nell'ambito del progetto LURCH. Il sistema sviluppato utilizza le ARToolKit Plus come librerie per la gestione dei marker ed è stato progettato per utilizzare dei marker della grandezza di 15x15 cm posizionati sul soffitto.

Su LURCH il membro che attualmente si occupa di calcolare la posizione della carrozzina nel sistema di riferimento assoluto è chiamato *VisionExpert*,



con ovvio riferimento al suo compito di analizzare delle immagini tramite tecniche di visione artificiale. L'architettura generale del sistema LURCH è visibile in figura 6.2.

Tenuti in considerazione questi aspetti, l'obiettivo del lavoro sarà quello di creare un nuovo modulo 'VisionExpert', compatibile con l'architettura MRT, in grado di fornire un sistema alternativo di localizzazione.

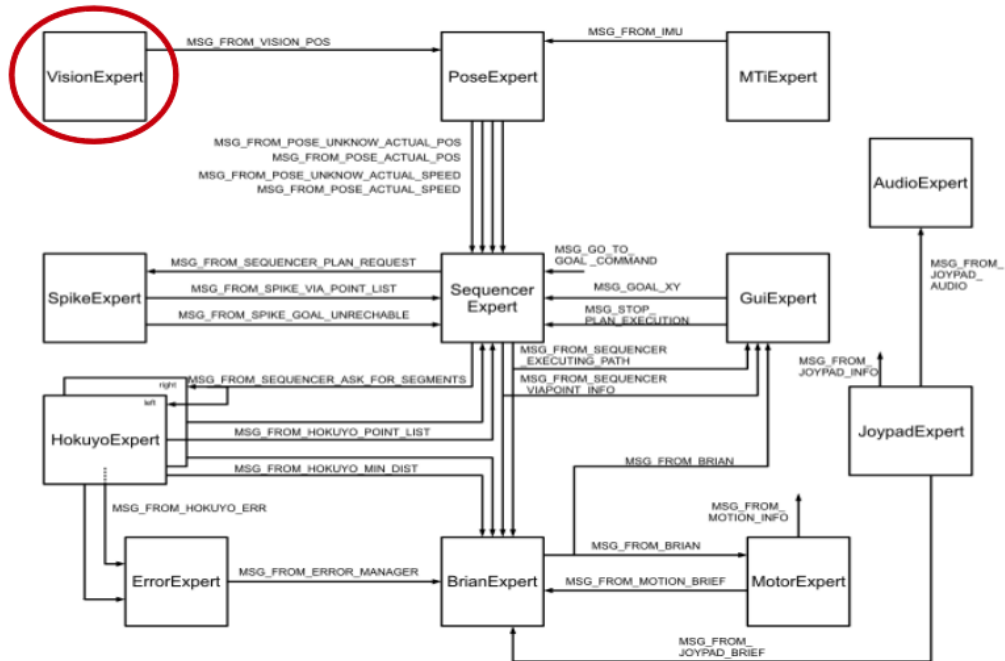


Figura 4.1: Architettura generale del sistema Lurch

Partendo dal progetto di localizzazione esistente si intende sviluppare un modulo in grado di autolocalizzarsi utilizzando dei marker posti in posizione verticale (nel progetto esistente i marker sono posti orizzontalmente sul soffitto). L'idea di questa modifica nasce dalla necessità di avere un sistema di localizzazione alternativo per quei luoghi in cui l'utilizzo del soffitto non è possibile (fiere espositive-capannoni con soffitti molto alti, ambienti aperti).

Vengono sfruttate le potenzialità dei marker, quali portatori di informazioni di posizione e orientazione. Come mostrato in figura 4.2, la telecamera, installata sul robot, inquadra, durante il suo spostamento, i marker posizionati nell'ambiente estraendo e calcolando in real-time dati utili per la sua localizzazione.

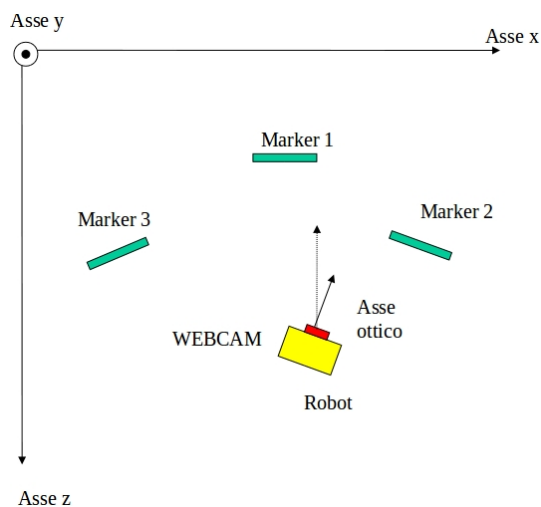


Figura 4.2: Schema del sistema di localizzazione assoluta basata su Landmark

## 4.2 Architettura MRT

MRT, Modular Robotic Toolkit, è un'architettura sw utilizzata per robot autonomi, dove diversi moduli possono essere facilmente combinati insieme e personalizzati per creare applicazioni robotiche con il minimo sforzo e tempo. I moduli interagiscono tra loro utilizzando un linguaggio comune e un ambiente predisposto allo scambio di messaggi. ciò garantisce benefici in termini di flessibilità e riusabilità.

I moduli sono semplici unità funzionali in cui il sistema è suddiviso. In tal modo risulta facile parallelizzare e suddividere il lavoro e inoltre si ha la possibilità di riusare i moduli in differenti applicazioni a seconda delle necessità. I moduli possono girare su macchine differenti e l'integrazione tra i moduli viene svolta dal DTCT, come spiegato in precedenza.

In un'applicazione robotica esistono, tipicamente, tre tipologie di moduli: moduli *sensing*, moduli *reasoning* e moduli *acting*. I moduli *sensing* sono quelli che interagiscono direttamente con i sensori (sonar, giroscopi, laser). Il loro scopo è quello di acquisire dati grezzi ed elaborarli per mandarli ai moduli *reasoning*, i quali rappresentano il *core* del sw. Il modulo *VisionExpert* ne è un esempio: dall'immagine grezza acquisita dalla telecamera vengono estratti i dati necessari per il calcolo della posizione.

I moduli *acting*, invece, sono invece i responsabili dei movimenti degli attuatori; essi seguono le indicazioni fornite dai moduli *reasoning* che sono responsabili dei processi decisionali che determinano il comportamento del robot. In tal modo, questi ultimi non necessitano di conoscere nulla riguardo ai sensori o attuatori montati sul robot.

### **4.3 Principio di funzionamento**

Il sistema di localizzazione assoluta, quando viene utilizzato in un ambiente opportunamente attrezzato con markers (ARtoolKitplus), è in grado di svolgere le attività di seguito descritte e schematizzate nella figura 4.3

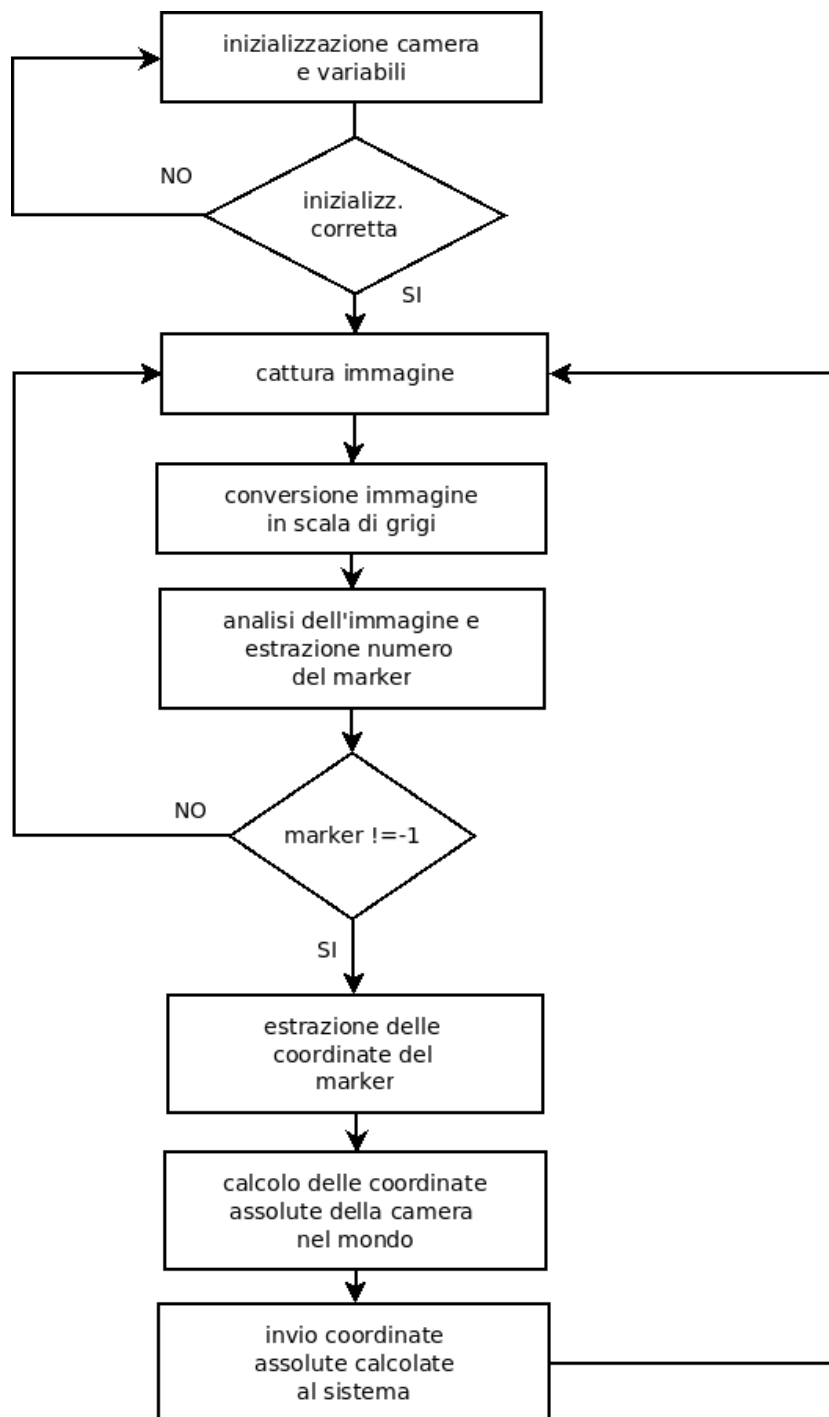


Figura 4.3: Algoritmo per l'analisi dei marker: principio base di funzionamento del sistema

### 4.3.1 Inizializzazione della camera e del sistema di visione

Il passaggio iniziale dell'algoritmo richiede una fase di inizializzazione della webcam. Una volta inizializzata, la camera è in grado di catturare periodicamente un frame fintanto che la camera non viene rimossa. Se l'inizializzazione della camera non va a buon fine il processo si blocca e occorre riprovare a inizializzare la camera. Nella fase iniziale, oltre all'inizializzazione della camera, vengono anche caricati in memoria le posizioni e le orientazioni dei marker, rispetto al sistema mondo, posti nell'ambiente in cui il robot si muoverà.

### 4.3.2 Acquisizione frame e estrazione dell'identificativo dei marker

Ogni frame che viene catturato dalla webcam viene convertito in scala di grigi prima di essere processato dalle funzioni dell' ARToolKit Plus. La prima analisi, che viene effettuata, riguarda la ricerca di marker nell'immagine. Se viene trovato un marker, viene restituito in output dalle funzioni dell' ARToolKit il suo codice identificativo. Se nessun marker viene riconosciuto il valore di ritorno è -1 e l'algoritmo continua acquisendo un nuovo frame.

### 4.3.3 Trasformazione di coordinate

Nel caso in cui il codice identificativo sia diverso da -1, l'algoritmo estrae dal marker la matrice  $T_M^C$ , matrice di rototraslazione che descrive la posizione e l'orientamento del marker nel sistema di riferimento solidale con la camera. Attraverso operazioni di trasformazione di coordinate (la matrice  $T_M^C$  viene invertita  $T_C^M$  e moltiplicata per la matrice di rototraslazione  $T_W^M$  che descrive la posizione del marker nel mondo) si ottiene la matrice che indica la posizione e orientazione della camera nel sistema di riferimento mondo  $T_W^C$ . Occorre effettuare queste trasformazioni per rendere omogenei i dati posseduti riguardanti il posizionamento dei marker e le informazioni acquisite dall'ambiente, in quanto la telecamera e i marker utilizzano sistemi di riferimento differenti.

I sistemi di riferimento coinvolti sono raffigurati nella figura 4.4.

Nei paragrafi seguenti viene spiegato nel dettaglio come avvengono queste trasformazioni di coordinate.

## Posizione assoluta con camera mobile

Le immagini acquisite tramite webcam, analizzate tramite il software ARToolKit Plus, forniscono informazioni riguardo il riconoscimento o non riconoscimento

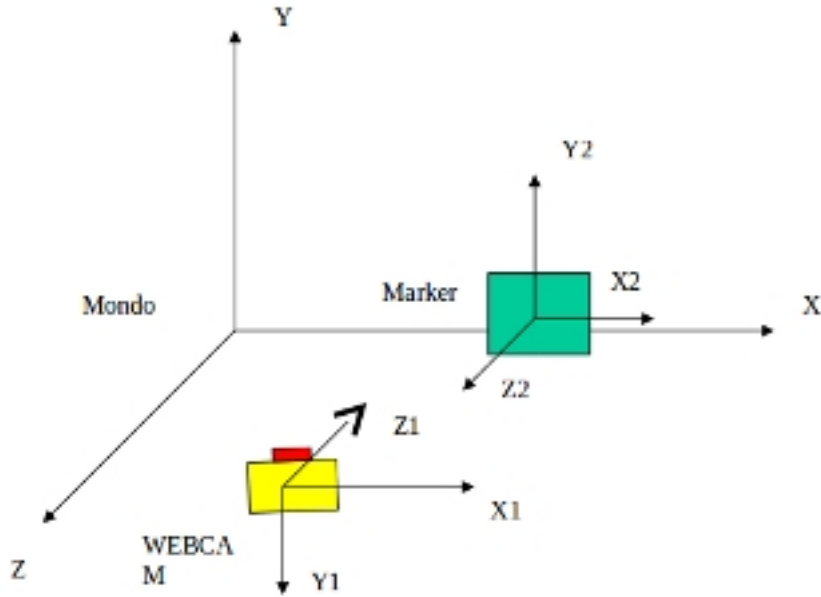


Figura 4.4: Sistemi di riferimento mondo, marker e webcam

to del marker. In caso affermativo ArtToolKit Plus restituisce il codice identificativo del marker e una matrice di roto-traslazione che indica la posizione e la rotazione del marker nel sistema di riferimento telecamera. La matrice si presenta nella seguente forma:

$$\mathcal{T}_M^C = \begin{pmatrix} u_1 & v_1 & w_1 & t_x \\ u_2 & v_2 & w_2 & t_y \\ u_3 & v_3 & w_3 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

La quarta colonna  $(t_x, t_y, t_z)$  indica le componenti di traslazione lungo gli assi x, y e z, mentre le prime tre colonne  $(u_1 \dots w_3)$  costituiscono la componente di rotazione degli assi. I valori degli angoli  $\alpha$  (rotazione attorno all'asse z),  $\beta$  (rotazione attorno all'asse y) e  $\gamma$  (rotazione attorno all'asse x) si ricavano dal seguente sistema di equazioni:

$$\begin{cases} u_1 = \cos\alpha\cos\beta \\ u_2 = \sin\alpha\cos\beta \\ u_3 = -\sin\beta \\ v_1 = \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \\ v_2 = \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma \\ v_3 = \cos\beta\sin\gamma \\ w_1 = \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ w_2 = \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ w_3 = \cos\beta\cos\gamma \end{cases}$$

Tale matrice deve essere successivamente invertita al fine di ottenere la posizione e la rotazione della telecamera nel sistema di riferimento del marker. ( $T_C^M$ )

Essendo noto il sistema di riferimento del Marker rispetto a quello 'mondo' (dati caricati nella fase di inizializzazione del sistema), per ottenere la posizione e la rotazione della telecamera nel sistema di riferimento mondo occorre moltiplicare la matrice inversa precedentemente calcolata ( $T_C^M$ ) per la matrice che indica la posizione e la rotazione del marker nel sistema di riferimento mondo ( $T_M^W$ ).

L'equazione seguita è la seguente:

$$T_C^W = T_M^W T_C^M$$

Di seguito è riportato uno schema a blocchi che riassume le operazioni effettuate per ottenere le coordinate della camera nel sistema mondo.

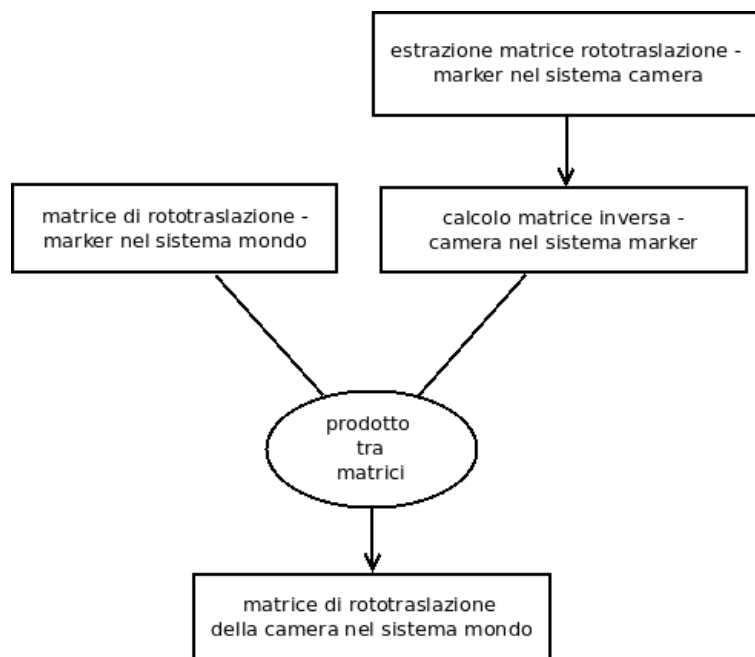


Figura 4.5: Operazioni per ottenere le coordinate della camera nel sistema mondo

#### 4.3.4 Invio dei dati calcolati al sistema

Una volta ricavati i dati di posizionamento della camera nel mondo, questi vengono inviati tramite messaggi agli altri moduli del sistema, tramite il middleware DCDT.

#### Middleware DCDT

DCDT, Device Communities Development Toolkit, è un framework usato per integrare tutti i moduli in MRT. Questo middleware è stato implementato per semplificare lo sviluppo di applicazioni in cui più attività vengono eseguite contemporaneamente e necessitano di scambiarsi messaggi tra loro. DCDT è un framework basato sul meccanismo publish-subscribe in grado di supportare diversi mezzi fisici di comunicazione in modo trasparente e facile. Si tratta di un sistema orientato alla gestione degli agenti. Un agente è un'entità astratta rappresentata nel codice da classi di tipo Expert. Per svolgere l'attività a cui sono destinati, gli agenti hanno bisogno di eseguire i loro compiti in totale autonomia e di poter scambiare messaggi con gli altri agenti operanti nel medesimo ambiente.

DCDT rende disponibile un ambiente multi-agente senza gravare l'utente degli aspetti legati alla programmazione in multithreading e della comunicazione tra i diversi thread.

DCDT è costituita da un oggetto principale, chiamato Agorà, che ospita e gestisce i diversi moduli software, chiamati Membri. I membri sono programmi/thread concorrenti che possono essere eseguiti periodicamente o alla notifica di un evento. Ogni Membro dell'Agorà può scambiare messaggi con altri Membri della stessa Agorà o con Membri di altre Agorà su macchine differenti. La comunicazione tra agenti è realizzata mediante pacchetti TCP/IP.

Il ciclo di vita di ogni membro è costituito da:

1. Inizializzazione del membro
2. Ripetizione periodica della funzione specifica del membro
3. Chiusura del membro

La creazione di un membro avviene semplicemente specificando le azioni che ognuna delle tre fasi identificate deve svolgere. Alla creazione dell'Agorà ogni singolo membro esegue la routine di inizializzazione e successivamente ripete la routine che specifica il compito di cui si occupa. DCDT permette di specificare la periodicità di ogni membro. Ad esempio, si può ipotizzare che un membro che si occupa di acquisire e analizzare immagini da una telecamera sia ripetuto 15 volte al secondo, quindi con un periodo di 66ms. Qualora



la singola esecuzione supera il tempo limite di 66msec, l'esecuzione successiva è immediata. Ogni membro ha la facoltà di chiedere la terminazione dell'Agorà in cui opera, provocando l'esecuzione della routine di chiusura di tutti i membri.

I messaggi sono tipizzati, ovvero sono distinguibili in base ad un codice che li caratterizza. Il meccanismo che gestisce la ricezione dei messaggi è basato su un sistema ad iscrizioni, ovvero ogni membro specifica quali messaggi vuole ricevere. Ad esempio, considerando A, B e C membri distinti, X e Y tipi di messaggi disponibili, A iscritto alla ricezione di X e B iscritto alla ricezione di Y, è possibile fare in modo che C produca messaggi sia di tipo X che Y (e.g., una lettura sensoriale codificata in due differenti modi) e che A e B li ricevano. Si noti che nessun membro è a conoscenza della presenza degli altri, ma solo delle informazioni scambiate. E dunque possibile creare un sistema flessibile. Ad esempio, la sostituzione del membro C con un nuovo membro D che produce gli stessi tipi di messaggi è del tutto trasparente al sistema. L'uso di DCDT permette di creare facilmente un'applicazione multiagente e semplifica la gestione dello scambio di informazioni tra i moduli software da sviluppare.

**Messaggi** Lo scambio dei messaggi è rappresentabile con un modello di un ipotetico ufficio postale che ha il compito di ricevere, smistare e consegnare i messaggi mediante una classe chiamata Dispatcher che svolge il lavoro di postino.

DCDT ha una politica pubblica/sottoscrivi che nasconde la distribuzione fisica dei messaggi. In questo modello, tutti gli agenti hanno la possibilità di pubblicare messaggi inviandoli all'ufficio postale.

Ad ogni messaggio è assegnato un tipo e un agente ha la possibilità di sottoscrivere ad una lista di interesse, specificando il tipo di messaggi che intende ricevere. In questo modo solo i diretti interessati possono ottenere le informazioni precedentemente richieste e non vengono, quindi, prodotte copie inutili dello stesso messaggio, aumentando l'efficienza e l'affidabilità del sistema.

## 4.4 Composizione del sistema

Per rispondere al requisito precedente si è ipotizzato di progettare un sistema che faccia uso di componenti hardware e software disponibili sul mercato. L'architettura del sistema utilizzata al fine di rispondere ai requisiti è la seguente (vedi figura 4.6):

I componenti hw utilizzati sono un personal computer e una telecamera per l'acquisizione di immagini. La camera è una normale webcam USB (TECHSOLO TCA-4900), disponibile in commercio (vedi figura 4.7). Il sistema operativo utilizzato è Ubuntu 11.04. Le librerie di supporto utiliz-

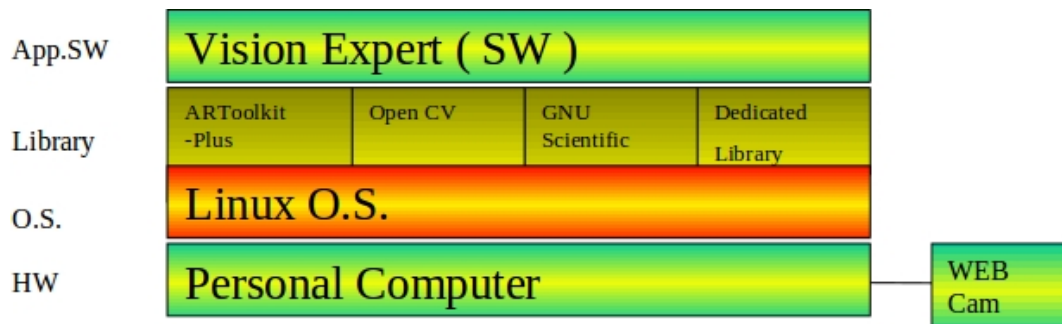


Figura 4.6: Architettura hw e sw del sistema

zate sono le OpenCV per la gestione della webcam e le ARToolKit Plus per la gestione dei marker (tali librerie verranno spiegate nel dettaglio nella sezione successiva). Il modulo software che si andrà a implementare diventerà, nell'ottica dell'architettura MRT, l'esperto che tratterà i problemi della visione del robot (VisionExpert).

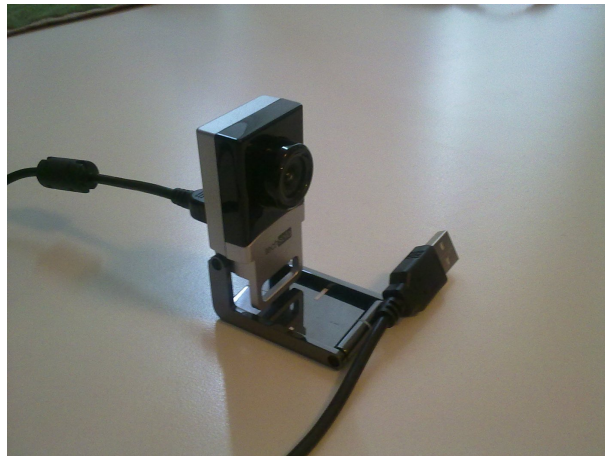


Figura 4.7: Webcam utilizzata per l'acquisizione delle immagini

## 4.5 Librerie utilizzate

**OpenCV** OpenCV è una libreria open source di computer vision. La libreria è scritta in C e C++ ed è possibile sviluppare applicazioni sotto qualsiasi sistema operativo, aspetto per noi fondamentale essendo il sistema funzionante totalmente in Linux. La libreria è nata da un'iniziativa di ricerca di Intel per migliore applicazione fortemente stressanti per la CPU. Con questo obiettivo, vennero avviati diversi progetti, dal tracciamento di laser alla vi-

sualizzazione di immagini 3D. Durante una visita in un'università, uno degli sviluppatori decise di applicare la stessa politica di passaggio libero del codice tra persone per migliorare delle librerie già esistenti di computer vision, evitando di dover riscrivere il codice ex novo. Iniziò così un progetto finalizzato all'ottenimento di un'infrastruttura universale ed efficiente per la computer vision.

Proprio per la natura iniziale del progetto di Intel, OpenCV è stata pensata con un occhio di riguardo alla complessità computazionale e un utilizzo in applicazioni real-time. La libreria è ottimizzata per C e può avvalersi dei vantaggi di processori multi-core. Essa contiene oltre 500 funzioni che si occupano di varie aree dell'analisi d'immagini, tra cui la calibrazione delle videocamera, la visione stereoscopica, l'estrazione di informazioni utili.

**ARToolKit Plus** ARToolKitPlus è una libreria di visione artificiale, scritta in linguaggio C. Il software è in grado di riconoscere e calcolare in tempo reale la posizione e l'orientamento di specifici oggetti all'interno dei frame ricevuti, grazie all'utilizzo di appositi 'marker', ovvero di immagini con caratteristiche particolari che il software è 'addestrato' a riconoscere. ciò consente lo sviluppo di molte applicazioni basate sulla Realtà Aumentata (Augmented Reality). È un'evoluzione delle librerie ARToolKit. Alcuni dei vantaggi nell'utilizzare le ARToolKit Plus piuttosto che le ARToolKit base consistono nell'avere un numero maggiore di marker a disposizione (4096) evitando quindi di dover costruire per proprio conto i marker desiderati, nell'avere un sistema di autoregolazione della soglia tra bianco e nero a seconda delle diverse condizioni di luce, nell'avere più tipologie di formati di immagine supportati (RGB565, Gray images) e nel consentire una pifacile calibrazione della camera.

## 4.6 Interfacciamento del modulo

Vengono analizzati ora gli aspetti riguardanti l'integrazione del modulo all'interno del sistema generale.

### 4.6.1 Input-output e relazione con altri moduli del sistema

Come detto il modulo sviluppato prende il nome di VisionExpert, con ovvio riferimento al suo compito di analizzare delle immagini tramite tecniche di visione artificiale. Gli input e gli output del modulo sono quelli schematizzati in figura 4.8.

L'input ricevuto da questo modulo è costituito essenzialmente dai fotogrammi acquisiti periodicamente dalla telecamera. Una volta acquisite

le immagini viene calcolata, quando possibile, la posizione della carrozzina rispetto al sistema di riferimento assoluto.

L'output di VisionExpert è costituito da un messaggio di tipo MSG FROM VISION POS, contenente le seguenti informazioni:

- *marker id* indica il numero del marker individuato nell'immagine.
- *time stamp sec* e *time stamp usec* rappresentano l'istante di tempo rispetto all'avvio del membro VisionExpert in cui si è rilevata l'immagine. Il timestamp è codificato su due diversi valori, il primo rappresenta il conteggio dei secondi, il secondo i microsecondi.
- *X*, *Y* e *A* rappresentano la posizione e l'orientamento della carrozzina rispetto al sistema di riferimento assoluto utilizzato. La posizione è espressa in millimetri e l'orientamento in radianti.

I messaggi inviati da VisionExpert sono ricevuti da PoseExpert.

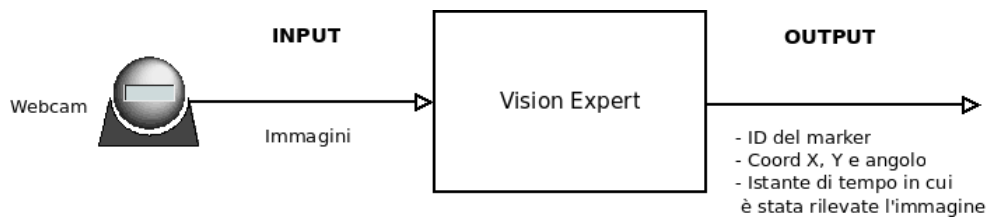


Figura 4.8: input e output del modulo

## Capitolo 5

# Implementazione

### 5.1 La classe MarkerDetector

L'algoritmo è organizzato all'interno di una classe, `MarkerDetector`, in modo da rendere l'implementazione trasparente al programma (nel nostro caso il `VisionExpert`) che la utilizza. L'interfaccia pubblica della classe è la seguente:

---

```
1  public :
2
3  MarkerDetector () ;
4  ~MarkerDetector () ;
5
6  void  inizializza ( CvCapture*block1_capture ,
7                    GestioneCoordinateAssolute*cordAss , IplImage*pRGBImg ,
8                    IplImage*pGrayImg , ARToolkitWrapper*art , Inizializzazione2
9                    *ini , Matrix3DRotoTraslation*tmp , Matrix3DRotoTraslation*
10                   resultPose ) ;
11
12
13  int  captureFrame ( CvCapture*block1_capture , IplImage*pRGBImg ,
14                   IplImage*pGrayImg , unsigned char*img ) ;
15
16  int  getPosition ( unsigned char*img , ARToolkitWrapper*art ,
17                   Matrix3DRotoTraslation*tmp , Matrix3DRotoTraslation*
18                   resultPose , GestioneCoordinateAssolute*cordAss , int& mark ,
19                   double& rotAsseZ ) ;
20
21  void  sendMessage ( int  mark , Matrix3DRotoTraslation*resultPose ,
22                   double& rotAsseZ , struct  timeval  tdiff ) ;
23
24  void  close () ;
```

---

Oltre a costruttore e distruttore, la classe contiene cinque metodi utili per le operazioni di inizializzazione, l'acquisizione e analisi delle immagini, l'invio dei dati calcolati al resto del sistema e il rilascio delle risorse. Questi metodi saranno dettagliatamente descritti nei seguenti paragrafi.

## 5.2 I metodi

### 5.2.1 inizializza()

---

```
1 void MarkerDetector::inizializza(CvCapture *block1_capture ,
  GestioneCoordinateAssolute *cordAss, IplImage *pRGBImg,
  IplImage *pGrayImg, ARToolkitWrapper *art, Inizializzazione2 *
  ini, Matrix3DRotoTraslation * tmp, Matrix3DRotoTraslation *
  resultPose)
```

---

Compito di questo metodo è quello di istanziare e inizializzare gli oggetti utili all'acquisizione e analisi dell'immagine.

In particolare vengono inizializzati due oggetti di tipo `IplImage` che andranno a contenere l'immagine a colori acquisita direttamente dalla webcam e l'immagine convertita in scala di grigi.

Sono poi istanziate due oggetti di tipo `Matrix3DRotoTraslation`, classe che contiene i metodi per la creazione e la gestione delle matrici di roto-traslazione.

Tramite l'oggetto `cordAss` di tipo `GestioneCoordinateAssolute()` è possibile gestire i marker posti nell'ambiente e calcolare la posizione in termini di coordinate assolute.

Occorre poi fornire al sistema le informazioni riguardanti la dimensione dei marker e la calibrazione della telecamera effettuata. Queste vengono passate come parametri al costruttore della classe `ARToolKitWrapper()`.

---

```
1 this->pRGBImg = 0;
2 this->pGrayImg = 0;
3
4 this->tmp= new Matrix3DRotoTraslation();
5 this->resultPose= new Matrix3DRotoTraslation();
6
7 this->cordAss=new GestioneCoordinateAssolute();
8
9 this->art=new ARToolkitWrapper("../webcamCal.cal", dim);
```

---

Nella fase di inizializzazione è opportuno inoltre caricare nel sistema le informazioni riguardanti i marker posti nell'ambiente in cui il robot si muoverà. Per far ciò si è pensato di scrivere due file di testo: `numberOfTag.txt` contenente il numero dei marker posti nell'ambiente in cui il robot si muoverà e (`tagRelation.txt`) contenente l'ID di ogni marker presente, la sua posizione e orientazione rispetto al sistema di riferimento 'mondo'. È compito della classe `Inizializzazione2()` analizzare questi file e inserire questi dati in variabili visibili al sistema (`tagNum`, `tagX`, `tagY`, `tagZ`, `tagAlpha`, `tagBeta`, `tagGamma`, `numTag`).

Tramite il metodo `addTag(tagNum [u], tagX[u], tagY[u], tagZ[u], tagAlpha[u], tagBeta[u], tagGamma[u])` i marker vengono resi disponibili al sistema per il calcolo delle posizioni in coordinate assolute.

---

```

1 this->ini=new Inizializzazione2 ();
2 int numTag=this->ini->initNumMarkerDaFile ();
3
4 this->ini->initCordMarkerDaFile (tagNum, tagX, tagY, tagZ, tagAlpha,
    tagBeta, tagGamma, numTag);
5
6 for (int u=0; u<numTag; u++){
7     this->cordAss->addTag (tagNum [u], tagX [u], tagY [u], tagZ [u],
        tagAlpha [u], tagBeta [u], tagGamma [u]);
8 }

```

---

L'inizializzazione della webcam avviene utilizzando i metodi delle librerie OpenCV, di seguito riportati:

---

```

1 this->block1_capture = NULL;
2 this->block1_capture = cvCaptureFromCAM (0);

```

---

### 5.2.2 captureFrame()

---

```

1 int MarkerDetector::captureFrame (CvCapture * block1_capture,
    IplImage * pRGBImg, IplImage * pGrayImg, unsigned char *img)

```

---

Il metodo si occupa della acquisizione dell'immagine da webcam e della sua trasformazione in scala di grigi.

Come detto in precedenza, per la gestione della camera si è scelto di utilizzare la libreria OpenCv. Le istruzioni per catturare il frame sono le seguenti:

---

```

1 cvGrabFrame (this->block1_capture);
2 this->pRGBImg = cvRetrieveFrame (this->block1_capture);

```

---

Sempre utilizzando questa libreria viene effettuata la conversione dell'immagine in scala di grigi.

---

```

1 this->pGrayImg = cvCreateImage ( cvSize (this->pRGBImg->width,
    this->pRGBImg->height), this->pRGBImg->depth, 1 );
2 cvCvtColor (this->pRGBImg, this->pGrayImg, CV_RGB2GRAY);
3
4 this->img=reinterpret_cast <unsigned char*> (this->pGrayImg->
    imageData);

```

---

L'ultima riga di codice è necessaria per effettuare un cast sull'immagine in quanto la funzione `ARToolKit` che analizza l'immagine necessita di un tipo `unsigned char*`.

### 5.2.3 getPosition()

---

```
1 int MarkerDetector::getPosition(unsigned char *img,
    ARToolkitWrapper *art, Matrix3DRotoTraslation * tmp,
    Matrix3DRotoTraslation * resultPose,
    GestioneCoordinateAssolute *cordAss, int &mark, double &
    rotAsseZ)
```

---

Questo metodo racchiude l'elaborazione e l'analisi dell'immagine vera e propria. È da questo metodo che, dando in input l'immagine acquisita e opportunamente convertita, viene calcolata la posizione assoluta della camera nel sistema di riferimento assoluto.

L'immagine viene passata al metodo `processImage(img)` dell'oggetto di tipo `ARToolKitWrapper` che restituisce il numero del marker riconosciuto (-1 se nessun marker è stato trovato).

---

```
1 int mark=this->art->processImage(img);
```

---

Viene calcolata l'inversa (matrice di rototraslazione della camera nel sistema marker) della matrice restituita dall'analisi effettuata dall' `ARToolKit`. Viene inoltre cercata una corrispondenza tra il marker trovato nell'immagine e i marker caricati da file nella fase di inizializzazione. Se c'è corrispondenza (`this->cordAss->getAbsolute(mark, tmp)==0`) il sistema è in grado di calcolare la posizione assoluta della camera tramite l'istruzione alla riga 3.

---

```
1 tmp = invMatrix3D(this->art->getMatrix3D(), NULL);
2 if (this->cordAss->getAbsolute(mark, tmp)==0){
3 this->resultPose=this->cordAss->getResultPose();
4 ...
5 }
```

---

Il risultato ottenuto dalla riga 3 è la matrice `resultPose` di tipo `Matrix3DRotoTraslation`, una matrice di rototraslazione che indica la posizione assoluta della camera nel sistema di riferimento mondo.

Per il calcolo dell'orientazione della camera viene utilizzata tale matrice; vengono estratti i valori della terza colonna, rappresentanti le componenti del versore z della telecamera (righe 1 e 2). Grazie a tali valori è possibile calcolare il valore dell'arcotangente, rappresentante la rotazione della camera.

---

```
1 double vx=resultPose->getValue(0,2);
2 double vz=resultPose->getValue(2,2);
3 double tangente=(vx/vz);
4 if (vz>0){
5     double arcotangenteRad=atan(tangente);
6     this->rotAsseZ=(360*arcotangenteRad)/(2*PIGRECO);
7 }
```

---



```

8     }else {
9     double arcotangenteRad=atan(tangente);
10    this->rotAsseZ=((360*arcotangenteRad)/(2*PIGRECO))+180;
11
12    }
13 }

```

---

### 5.2.4 sendMessage()

```

1 void MarkerDetector::sendMessage(int mark, Matrix3DRotoTraslation
   * resultPose, double &rotAsseZ, struct timeval tdiff)

```

---

Le funzioni fondamentali per mandare messaggi presenti in questo metodo sono: `NewMessage(tipomessaggio,stringa)` (tipomessaggio serve ad identificare il mittente e il tipo del messaggio, mentre stringa serve solo per debug) e `mCurrentMessage <<D-OPEN-DATUM...<D-CLOSE-DATUM;` in cui vengono spediti i dati veri e propri.

```

1
2 mCurrentMessage <<fixed<<D_OPEN_DATUM <<VISION_X      << " " <<
   x <<D_CLOSE_DATUM;
3 mCurrentMessage <<D_OPEN_DATUM <<VISION_Y      << " " << y <<
   D_CLOSE_DATUM;
4 mCurrentMessage <<D_OPEN_DATUM <<VISION_Z      << " " << z <<
   D_CLOSE_DATUM;
5
6 mCurrentMessage <<D_OPEN_DATUM <<VISION_ANGLE    << " " <<
   rotAsseZ <<D_CLOSE_DATUM;

```

---

Come si può notare dalle righe di codice riportate qui sopra i dati inviati al sistema sono la posizione (x,y,z) e la rotazione (rotAsseZ) calcolate.

### 5.2.5 close()

```

1 void MarkerDetector::close()

```

---

In questo metodo vengono rilasciate le risorse utilizzate come la camera e le immagini. Viene utilizzato i metodi delle OpenCV `cvReleaseCapture` e `cvReleaseImage`.

```

1 cvReleaseImage(&pRGBImg);
2 cvReleaseImage(&pGrayImg);
3 cvReleaseCapture(&block1_capture);

```

---

## Capitolo 6

# Integrazione su LURCH

È stato scelto di integrare il modulo *VisionExpert* sul robot LURCH, Let Unleashed Robots Crawl the House, carrozzina autonoma sviluppata presso l'AILab. Sono stati successivamente svolti dei test per verificare la precisione dei dati di posizione ricavati dal modulo *VisionExpert* e l'integrazione del modulo all'interno del sistema LURCH esistente. Nel capitolo seguente vengono spiegate le diverse tipologie di test effettuati e vengono analizzati i risultati ottenuti.

### 6.1 Cos'è LURCH

LURCH è l'acronimo di Let Unleashed Robots Crawl the House e, oltre a richiamare alla mente il celebre personaggio della famiglia Addams, è la carrozzina autonoma sviluppata presso l'AILab del Politecnico di Milano.

LURCH è una versione estesa delle normali carrozzine elettriche che si trovano in commercio (Rabbit by OttoBock) equipaggiata, in aggiunta, con 2 computers (PCBricks) alimentati dalle batterie della carrozzina, un monitor touchscreen, un sistema per il calcolo dell'odometria, una telecamera FireI400 e una webcam USB, sensori laser.

Obiettivi principali del progetto LURCH sono lo sviluppo di un sistema di navigazione autonoma attraverso un sistema di autolocalizzazione, di path planning e di collision and obstacle avoidance; Si punta inoltre ad aggiungere nuove e differenti interfacce per guidare e comandare la carrozzina (come Joypad wireless, comandi vocali, controllo del movimento dei muscoli facciali e brian computer interface -BCI-), sensori e funzionalità tipiche della robotica.



Figura 6.1: La carrozzina LURCH

## 6.2 Struttura generale del sistema

Di seguito è riportata lo schema della struttura complessiva del sistema. Dall'immagine si possono notare le interazioni tra i diversi esperti attraverso lo scambio di messaggi e la presenza di un 'Esperto' per ogni compito/attività che il robot svolge.

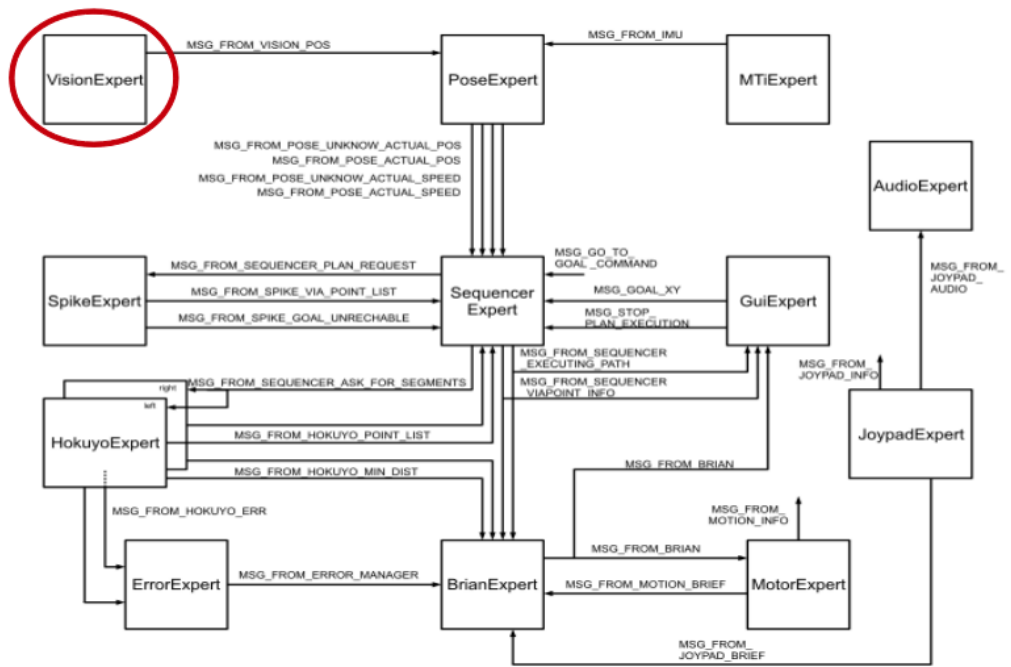


Figura 6.2: Architettura generale del sistema

### 6.3 Moduli interfacciati

Focalizziamo ora la nostra attenzione sul modulo *VisionExpert*, modulo che abbiamo re-implementato in questo progetto.

Il modulo, come rappresentato nella figura 6.3, non riceve nessun dato in input da parte di altri moduli del sistema, ma riceve solo le immagini da analizzare dalla webcam dedicata.

In output, invece, il nuovo modulo deve inviare al *PoseExpert* le stesse informazioni inviate dal modulo già esistente. Quando la posizione della carrozzina è disponibile, il membro *VisionExpert* invia un messaggio di tipo MSG FROM VISION POS a *PoseExpert*.

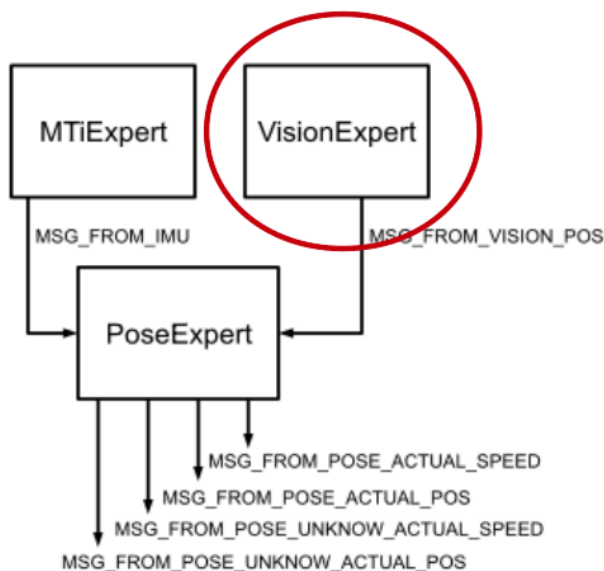


Figura 6.3: Moduli interfacciati

*PoseExpert* è un modulo che invia periodicamente due tipi di messaggi: MSG FROM POSE ACTUAL POS e MSG FROM POSE ACTUAL SPEED. Il primo messaggio comunica la posizione della carrozzina in un sistema di riferimento assoluto, il secondo la velocità attuale della carrozzina. Qualora *PoseExpert* rilevi che *VisionExpert* non ha inviato messaggi per un certo periodo di tempo, invia il messaggio MSG FROM POSE UNKNOW ACTUAL POS, non contenente alcun valore.

Il modulo *PoseExpert* è in grado di ricevere messaggi da più moduli differenti (*MTiExpert* come mostrato in figura 6.3 o modulo *OdometryExpert* responsabile del calcolo della posizione tramite odometria). È poi compito

del *PoseExpert* integrare le informazioni ricevute per estrarre il risultato più corretto.

## 6.4 Integrazione

Durante la fase di integrazione si è dovuto adattare il modulo al sistema esistente su LURCH. In particolare il modulo è andato a sostituire il modulo di *VisionExpert* esistente.

I due moduli di localizzazione mirano, così, ad essere intercambiabili, utilizzando uno o l'altro a seconda dell'ambiente in cui si muove il robot e quindi il modo in cui i marker vengono posizionati nell'ambiente (orizzontali al soffitto o verticali alle pareti/altri supporti).

Fino a questo momento la posizione del marker veniva calcolata rispetto al sistema di riferimento camera. Per poter utilizzare il modulo sul robot è necessario che la posizione del marker sia calcolata rispetto al sistema di riferimento della carrozzina, per poi essere trasportata nuovamente nel sistema di riferimento mondo. È pertanto necessario effettuare delle rototraslazioni per passare dal sistema camera a quello carrozzina. Le trasformazioni eseguite sono specificate nell'immagine 6.4 .

Nonostante queste modifiche effettuate, il modulo è stato facilmente integrabile, in quanto è stato sviluppato tenendo conto delle regole imposte da MRT. La comunicazione tra i moduli tramite messaggi viene gestita facilmente con il framework DCDT, spiegato nei capitoli precedenti.

La webcam è stata installata sul retro della carrozzina e punta nella direzione contraria al movimento.

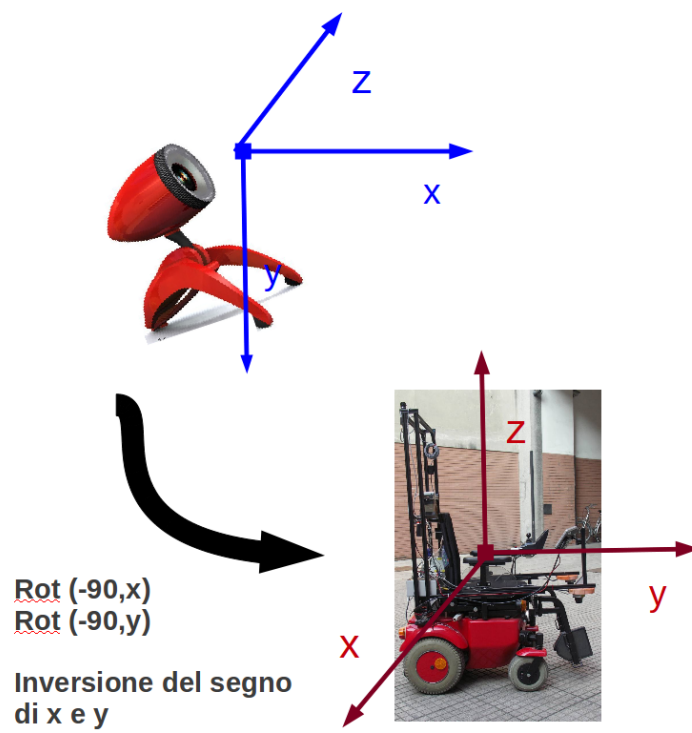


Figura 6.4: Passaggio dal sistema di riferimento camera al sistema di riferimento carrozzina

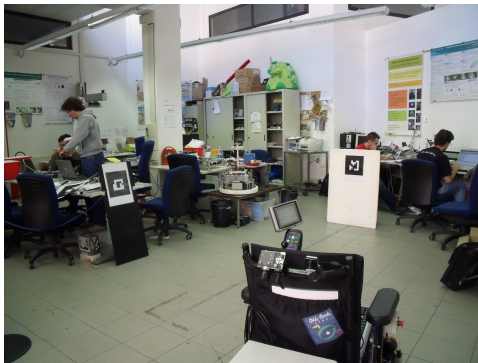
## 6.5 Test effettuati

Una volta integrato il modulo nel resto del sistema, si è ritenuto opportuno testare l'effettiva funzionalità del robot con il nuovo innesto sw.

Obiettivo dei test è sia quello di verificare il comportamento sperimentale del modulo sviluppato, sia quello di verificare che la carrozzina abbia lo stesso comportamento che aveva con il VisionExpert preesistente. Ottenere quest'ultima cosa significherebbe avere un modulo di visione e localizzazione completamente trasparente al sistema e quindi facilmente intercambiabile.

### 6.5.1 Ambiente di test

I test sono stati effettuati all'interno dell'AIRLab, laboratorio del Politecnico di Milano, a Lambrate.



(a)



(b)

### 6.5.2 Pianificazione delle prove

Sono state pianificate 2 tipologie di prove, una per effettuare un'analisi quantitativa e una per un'analisi qualitativa.

1. **Percorso 1 - analisi quantitativa** : è stato selezionato un percorso rettilineo, con il marker in piena visibilità. L'obiettivo è quello di analizzare la precisione dei dati estratti e calcolati da *VisionExpert*.
2. **Percorso 2 - analisi qualitativa** : il percorso selezionato è più articolato in quanto l'obiettivo di testare l'integrazione del modulo all'interno del sistema esistente.



### Percorso 1 - analisi quantitativa

**Posizionamento marker** Il test è stato effettuato in un'area rettangolare di dimensioni 5,10 x 3,90 m, priva di ostacoli. Su uno dei due lati corti del rettangolo è stato posto un marker, come mostrato nella figura 6.5

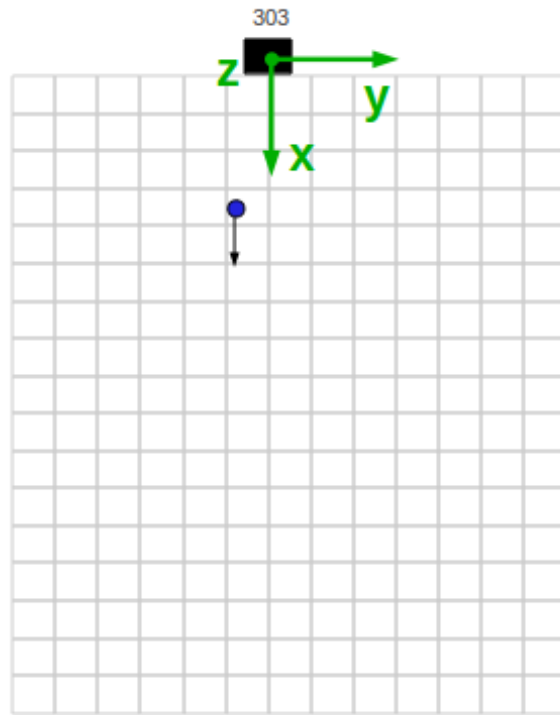


Figura 6.5: Ambiente di test e posizionamento marker

**Percorsi effettuati** Il percorso effettuato è stato un percorso rettilineo dal punto di coordinate -20 cm in y e +1 m in x fino al punto -20 cm in y e +4 m in x, come mostrato in figura 6.6

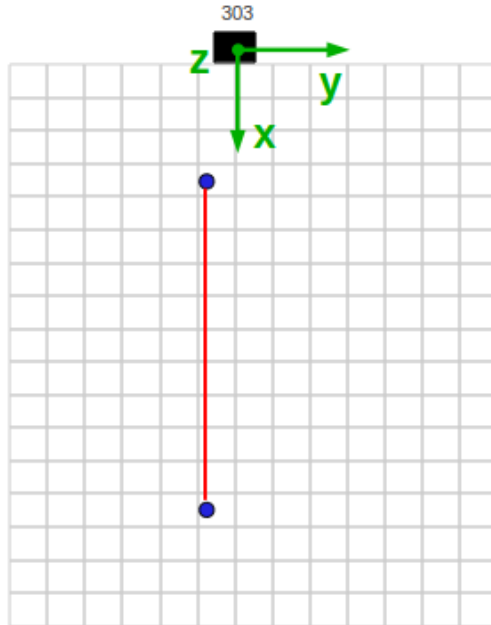


Figura 6.6: Percorsi di test

**Risultati ottenuti** Dall'analisi dei file di log generati dalla carrozzina è possibile analizzare le posizioni calcolate dal modulo *VisionExper* e inviate al resto del sistema. Dal file di log si vede che il marker è stato riconosciuto nel 95% dei frame acquisiti dalla webcam durante il percorso.

Nel grafico 6.7 sono riportate le posizioni calcolate durante il percorso e suddivise nelle componenti x (tracciato blu) e y (tracciato rosso).

**Analisi coordinata X** Analizzando il tracciato dello spostamento lungo l'asse x, si può suddividere il grafico in tre aree principali:

- Parte 1: parte di grafico orizzontale in cui la carrozzina è ferma a distanza 1,1 m dal marker.
- Parte 2: il grafico presenta un andamento monotono crescente dal valore di 1,1 m fino al valore 4m. La carrozzina si sta muovendo lungo il percorso indicato.

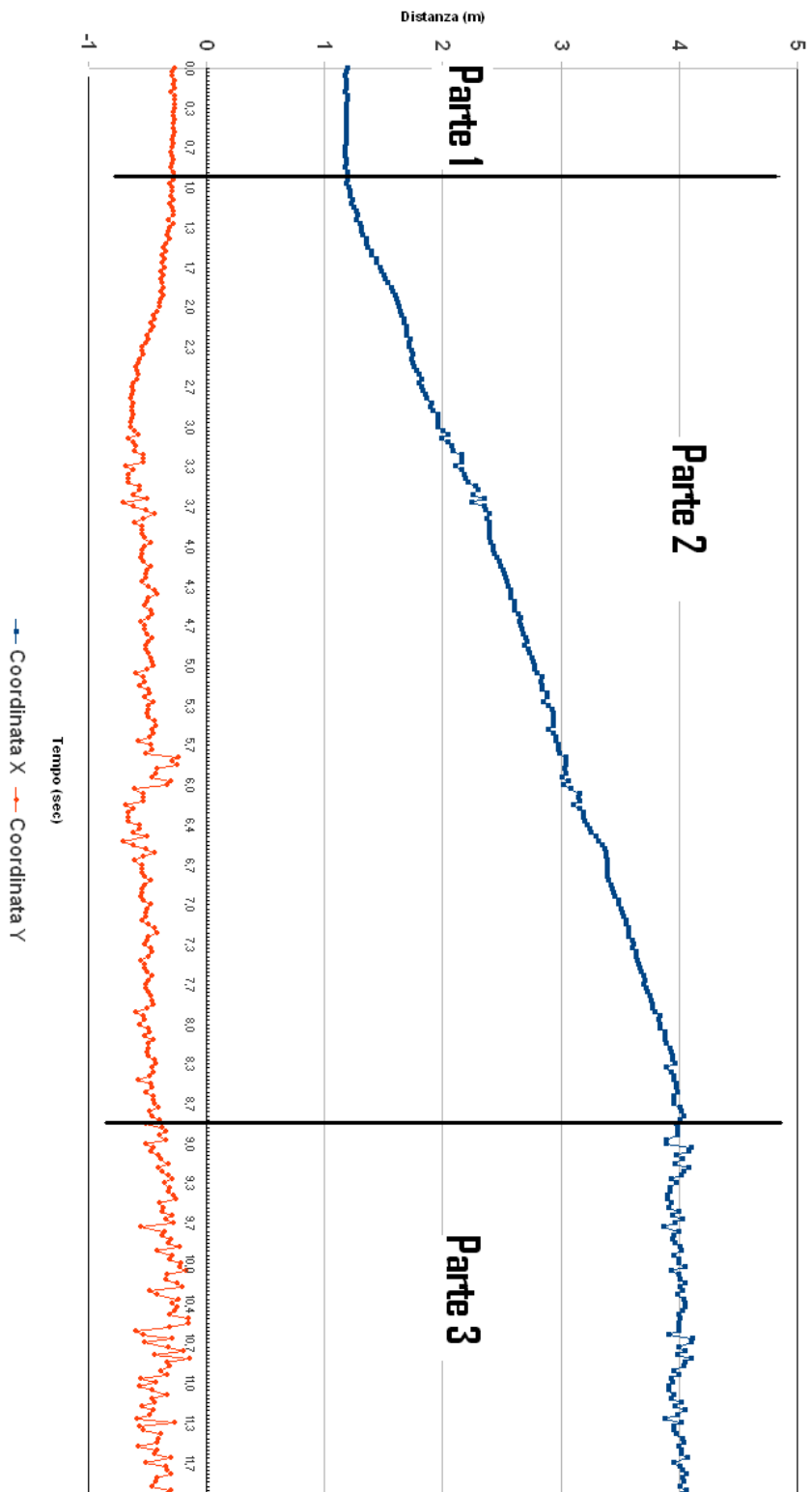


Figura 6.7

- Parte 3: il grafico ritorna orizzontale attorno al valore 4 m. La carrozzina ha raggiunto l'obiettivo e si è fermata.

**Parte 1 e parte 3** Queste due parti del grafico rappresentano i momenti in cui la carrozzina è stata ferma: la parte 1 rappresenta la fase iniziale mentre la parte 3 la fase finale. A un primo colpo d'occhio è facilmente individuabile una differenza tra i due tracciati.

Il grafico nella parte 1 è più definito e affetto da minor rumore rispetto al tracciato della parte 3, in cui i dati oscillano maggiormente. Nella tabella seguente sono riportati i valori di media, varianza e deviazione standard, calcolati su questi due tratti di grafico.

Parte grafico	Valor medio	Varianza	Deviazione std
1	1,17655 m	0,0000257 $m^2$	0,00507 m
3	3,99005 m	0,002861 $m^2$	0,05349 m

Di seguito sono riportati i grafici (6.8 e 6.9) dettagliati della parte 1 e 3 in cui sono indicati la serie dei valori acquisiti e il loro valore medio.

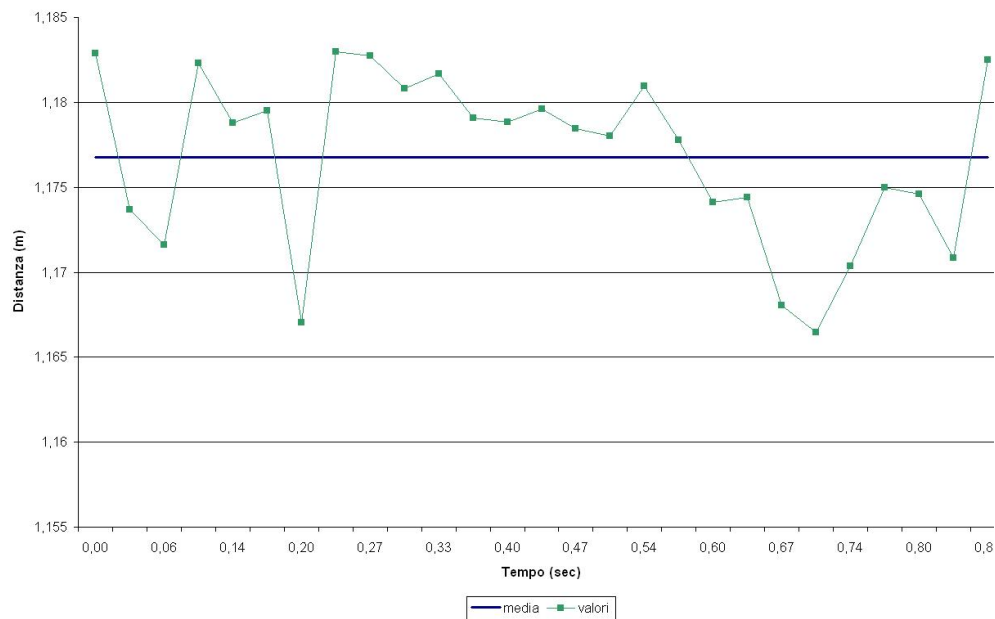


Figura 6.8: Parte 1 - Valori calcolati e media

Nella parte 1 i valori oscillano tra 1,181 m e 1,171 m, salvo in alcuni punti che raggiungono i valori di 1,167 m e 1,183 m. In ogni caso i dati

non si discostano molto dalla media. L'errore percentuale, calcolato come deviazione standard divisa per il valore medio, è pari a 0,43%. Si può quindi affermare che il sistema a questa distanza è robusto, in quanto garantisce una buona affidabilità e ripetibilità.

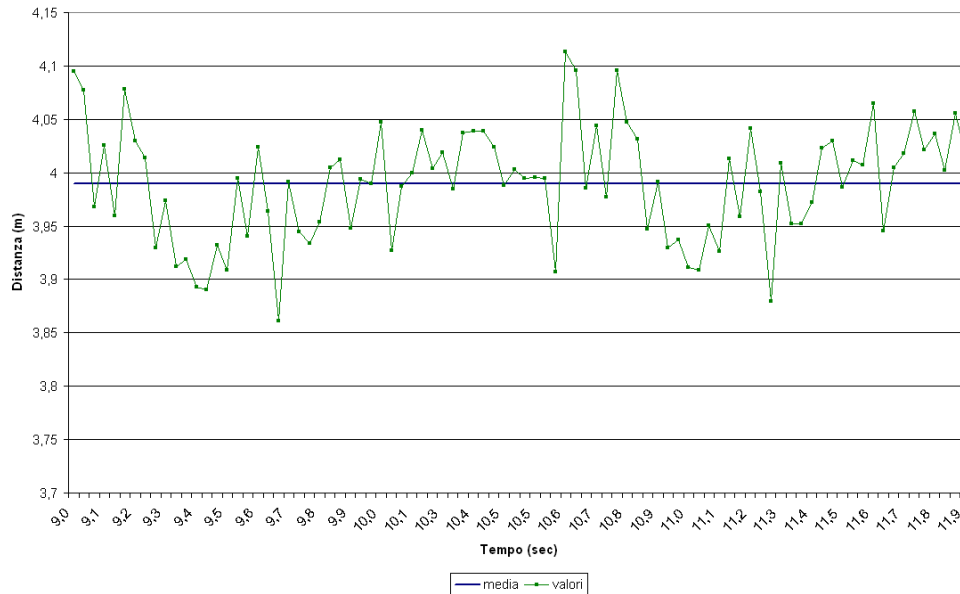


Figura 6.9: Parte 3 - Valori calcolati e media

Nella parte 3 i valori si trovano tra 4,05 m e 3,93 m, salvo in alcuni punti che raggiungono i valori di 4,1 m e 3,86 m.

I valori oscillano di più rispetto alla parte precedente: qui si ha uno scarto di 10 cm tra valore medio minimo e massimo, mentre nella parte 1 si ha uno scarto di 1 cm. L'errore percentuale in questo caso è comunque pari a 1,34%. Questa differenza è dovuta al fatto che la maggior distanza rende il riconoscimento del marker più difficoltoso: più l'immagine è lontana più viene vista sfocata dalla webcam ed è quindi più difficile per l'ARToolKitPlus riconoscere esattamente i contorni (tant'è che dai test svolti in palestra si è visto che la distanza massima che garantisce il riconoscimento dei marker è pari a 6m). Nell'immagine 6.10 è possibile notare come gli oggetti in primo piano siano nitidi e ben definiti, mentre quelli sullo sfondo siano sfocati e sia difficile delineare bene i contorni.

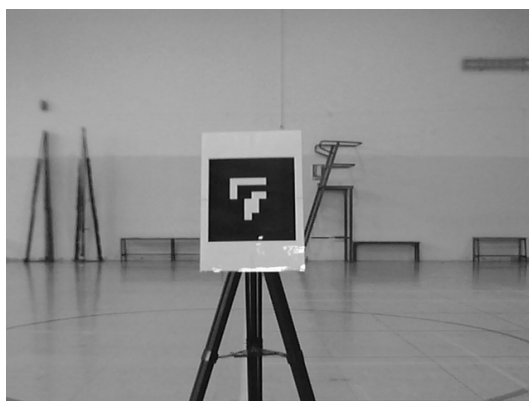


Figura 6.10: Differenza di messa a fuoco tra immagini vicine e lontane

**Parte 2** Passiamo ora a considerare la parte 2 del tracciato, cioè quella in cui la carrozzina è in movimento.

Di seguito viene analizzato il comportamento tenuto dalla carrozzina prendendo come riferimento un moto rettilineo uniforme, come rappresentato nella figura 6.11

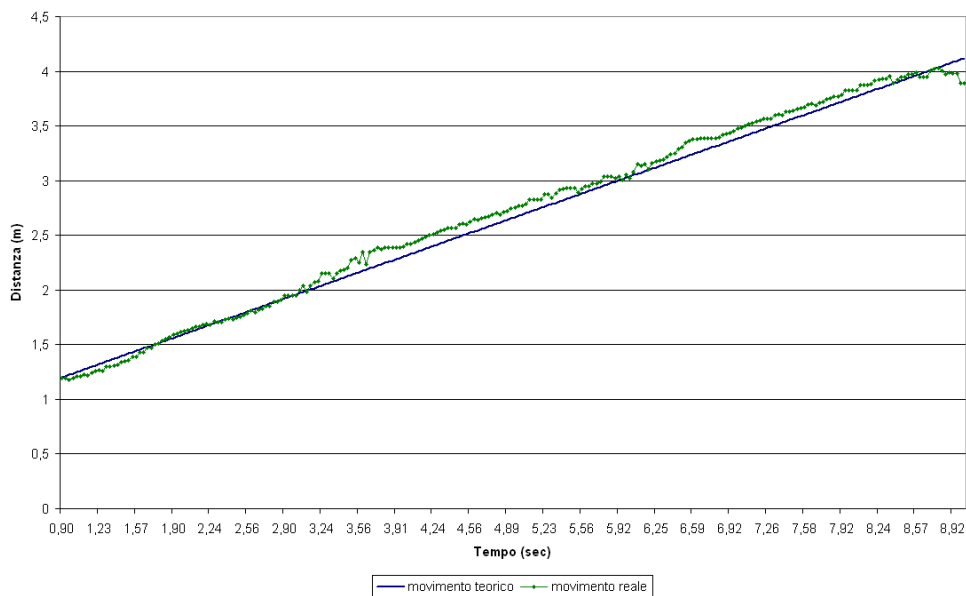


Figura 6.11: Parte 2 - Carrozzina in movimento

Notiamo che il moto tenuto dalla carrozzina è assimilabile a un moto rettilineo uniforme: se, infatti, trascuriamo la fase di accelerazione iniziale, necessaria per portare la carrozzina dalla velocità nulla alla velocità di regime, notiamo che tutto il tracciato consiste in una linea pressoché parallela a

quella della velocità stimata con moto rettilineo uniforme. L'errore medio rispetto alla retta teorica si assesta su un valore pari a 4 cm, con valori massimi di scarto pari a 16 cm. La velocità tenuta dalla carrozzina è pari a 0,34 m/s.

**Coordinata Y** Viene analizzato ora il percorso fatto dal punto di vista dello spostamento lungo l'asse Y. Di seguito, in figura 6.12, è riportato il grafico indicante i valori calcolati e il valor medio.

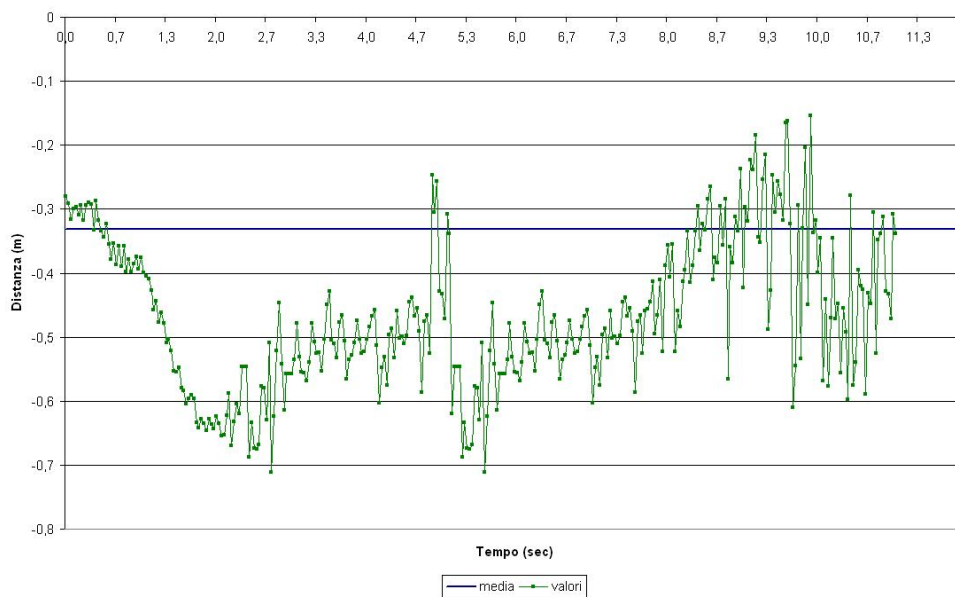


Figura 6.12: Coordinata Y - Valori calcolati e media

Il valore medio è pari a -0,331 m e la deviazione standard è pari a 0,100 m. Questo valore così elevato di deviazione standard è spiegabile analizzando il grafico e osservando il comportamento tenuto dalla carrozzina durante lo spostamento: osservando la carrozzina muoversi si è visto come nel primo tratto abbia compiuto uno spostamento laterale rispetto al percorso rettilineo teorico in quanto le ruote non erano posizionate diritte. La carrozzina ha dovuto fare una curva per raddrizzarle per poi ripuntare verso il punto di arrivo desiderato. Si trova una corrispondenza di questo comportamento guardando anche il grafico 6.12: la carrozzina prima della partenza si trova in coordinate -0,3m, poi non appena si muove la y diminuisce fino a raggiungere il valore -0,7m (ruote completamente raddrizzate), per poi progressivamente riavvicinarsi al valore di partenza -0,3m.

Anche nel caso della componente y si può notare un aumento progressivo del rumore nel segnale all'aumentare della distanza. Ciò è spiegabile con le stesse motivazioni esposte per la componente X.

**Frame rate** Dall'analisi dei file di log si è potuto anche calcolare il tempo di acquisizione e analisi dei frame. Mediamente un frame viene acquisito ogni 30 ms, rispettando le tempistiche utilizzate dal modulo *VisionExpert* precedente,.

## **Percorso 2 - analisi qualitativa**

**Posizionamento marker** L'area in cui si è effettuato il test è la stessa usata nel test del percorso 1: un'area rettangolare di dimensioni 5,10 x 3,90 m, priva di ostacoli. In questo test l'area è stata attrezzata con quattro marker con ID differente, posizionati uno per ogni lato del quadrato, come mostrato di seguito.

**Percorsi effettuati** Una volta mappato l'ambiente di test con i marker, sono stati scelti 3 punti raggiungibili dalla carrozzina (i punti A, B e C nella figura 6.13) e sono stati creati dei percorsi di test tra questi tre punti. Ogni punto del percorso rappresentato sul grafico è in corrispondenza dell'asse verticale della telecamera.

I marker sono stati posizionati in modo tale da garantire la visione di un marker in buona parte del percorso da effettuare. Alcuni punti sono stati volontariamente lasciati liberi da marker per testare l'integrazione con il modulo dell'odometria. Avendo posizionato la webcam nella parte posteriore della carrozzina (con verso contrario al moto della carrozzina), la visione dei marker attesa sarà la seguente:

- Nel tratto da B ad A i marker che il sistema inquadrerà saranno il 64
- Nel tratto da A a C i marker 313 e 303.
- Nel tratto da C a B i marker 313 e 303.

Si è voluto inoltre testare il funzionamento degli altri moduli del sistema a fronte dell'inserimento del nuovo modulo. Si è voluta testare la funzione di *ObsacleAvoidance*, ovvero il comportamento che la carrozzina tiene quando, durante il suo movimento, si trova davanti ostacoli inaspettati e non previsti. Per testare ciò abbiamo posto un ostacolo (non mappato nell'ambiente) davanti alla carrozzina mentre era in moto e abbiamo osservato il comportamento.



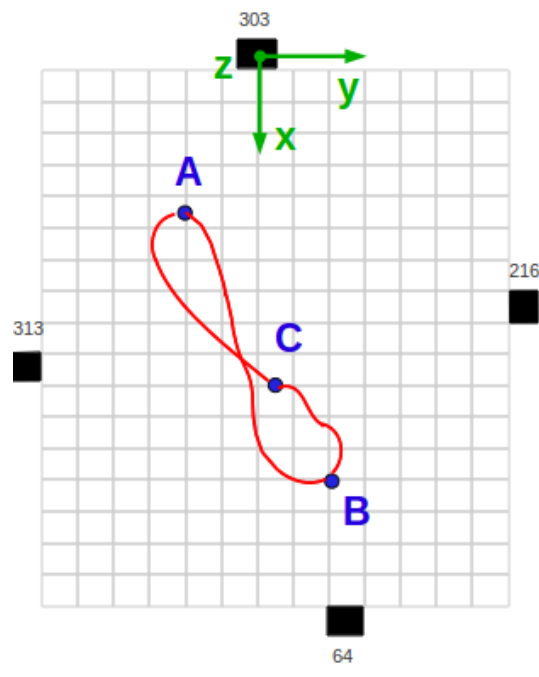


Figura 6.13: Percorsi di test

**Risultati ottenuti** I test effettuati hanno dato risultati soddisfacenti, infatti la carrozzina raggiunge tutte le posizioni ordinate. ciò sta a significare che è stata raggiunta una corretta integrazione del modulo di *VisionExpert* con le altre funzionalità della carrozzina, quali l'odometria. I dati sulla posizione, calcolati in modo congiunto dall'odometria e dal modulo di visione, hanno garantito un corretto raggiungimento dell'obiettivo.

Nella figura 6.14 sono evidenziati in rosso i tratti del percorso in cui la localizzazione è stata effettuata tramite odometria, in quanto nessun marker si trovava nel cono di visione della webcam. I tratti colorati in verde, invece, sono quelli in cui i marker erano visibili alla telecamera.

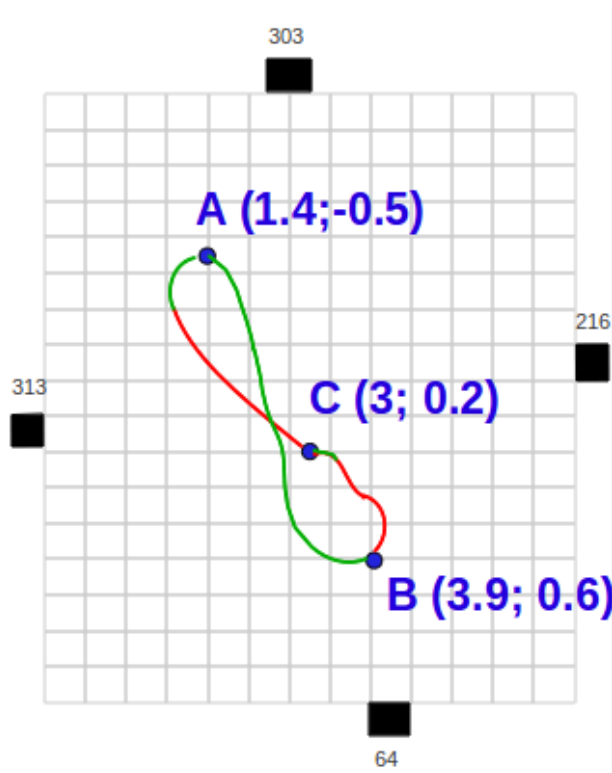


Figura 6.14: Visione effettuata tramite odometria e riconoscimento di marker

È possibile notare che lo spostamento dal punto B al punto A è stato compiuto in piena visibilità di marker. Solo parzialmente sono stati riconosciuti i marker nel tratto di percorso da A a C e da C a B. In questi tratti quindi la localizzazione è stata svolta, in buona parte, dall'odometria. Di seguito vengono rappresentate, sotto forma di grafico, le posizioni calcolate dal modulo *VisionExpert* nei tratti di visibilità.

Nel tratto dal punto B al punto A i valori calcolati dal modulo di visione sono quelli rappresentati nel grafico seguente(6.15).

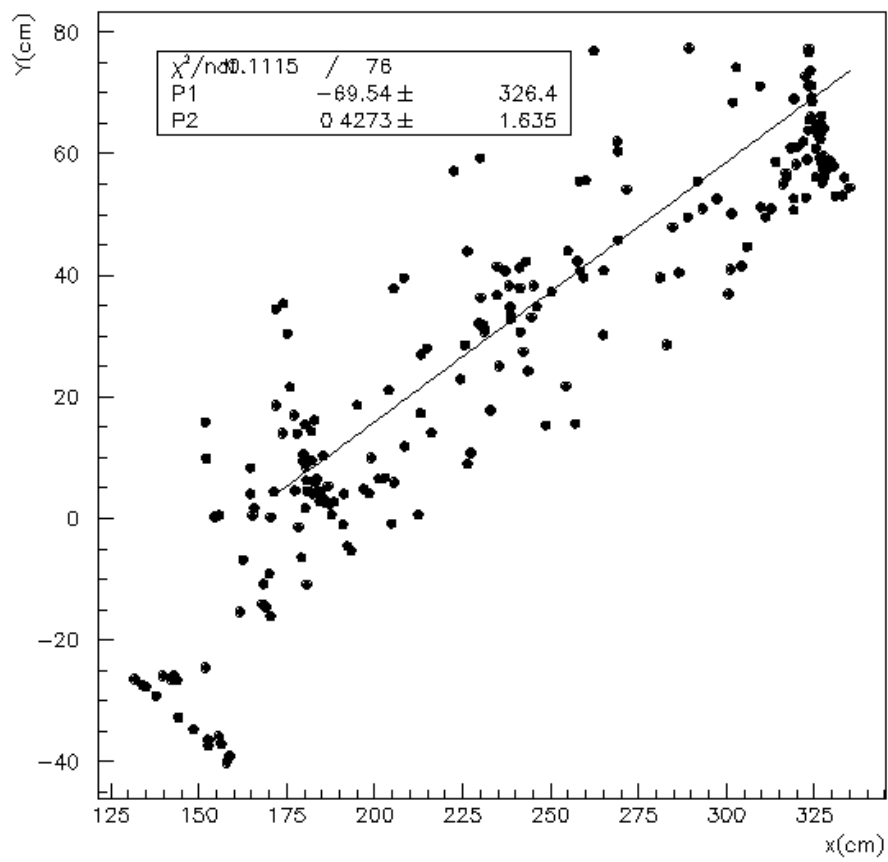


Figura 6.15: Valori calcolati da VisionExpert e media

È stata ricavata la retta di interpolazione e gli scarti di ogni valore rispetto ad essa. Questi ultimi sono rappresentati nel grafico 6.16.

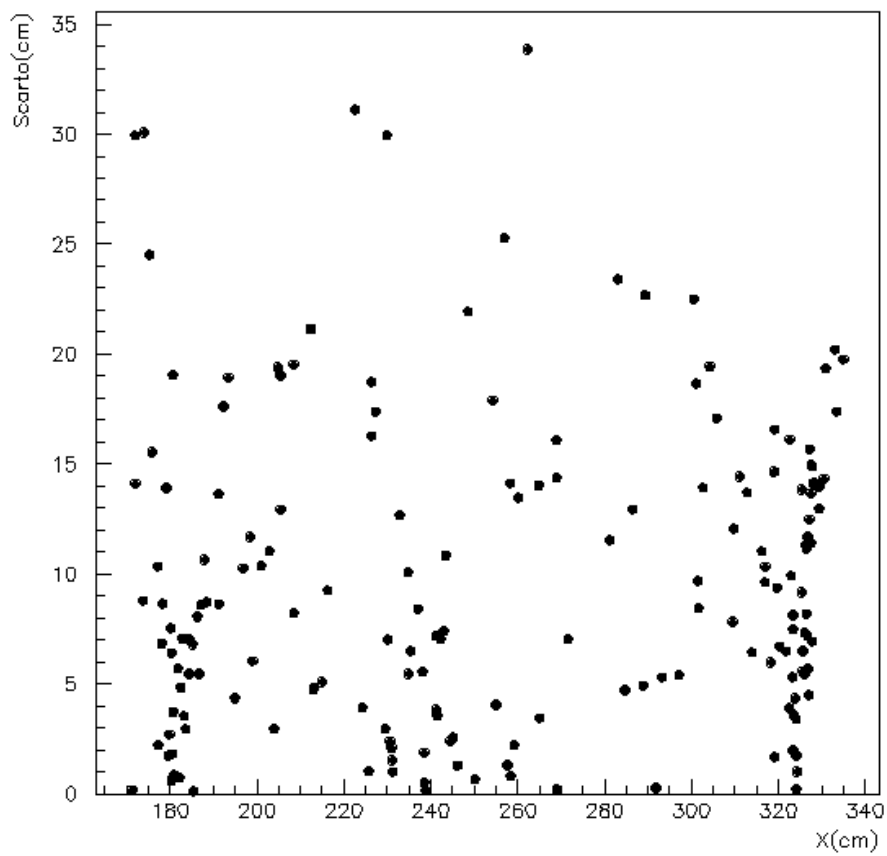


Figura 6.16: Scarti rispetto al valore medio

Dal grafico è possibile notare che la maggior parte degli scarti sono compresi tra 0 cm e 15 cm. Solo alcuni alcuni valori isolati raggiungono i 20-30cm. In ogni caso il valor medio degli scarti è pari a 11 cm, valore che può essere ritenuto accettabile per gli obiettivi di questo sistema.

Come è stato evidenziato nella figura 6.14, il secondo tratto in cui, parzialmente, vengono riconosciuti i marker è quello dal punto A al punto C (la curva iniziale) e la partenza dal tratto C al tratto B. Di seguito (grafico 6.17) sono rappresentati i valori calcolati in questi tratti di percorso.

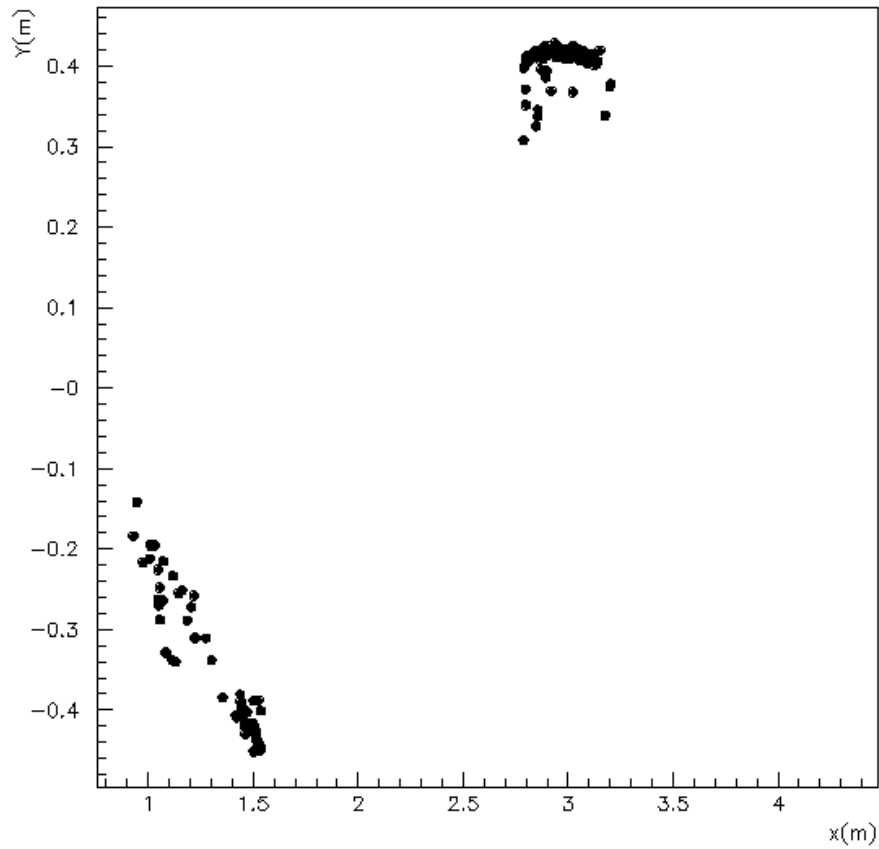


Figura 6.17: Scarti rispetto al valore medio

Viene ora analizzato in particolare il tratto iniziale del percorso tra A e C in termini di retta interpolante e di scarti dei valori (6.18 e 6.19).

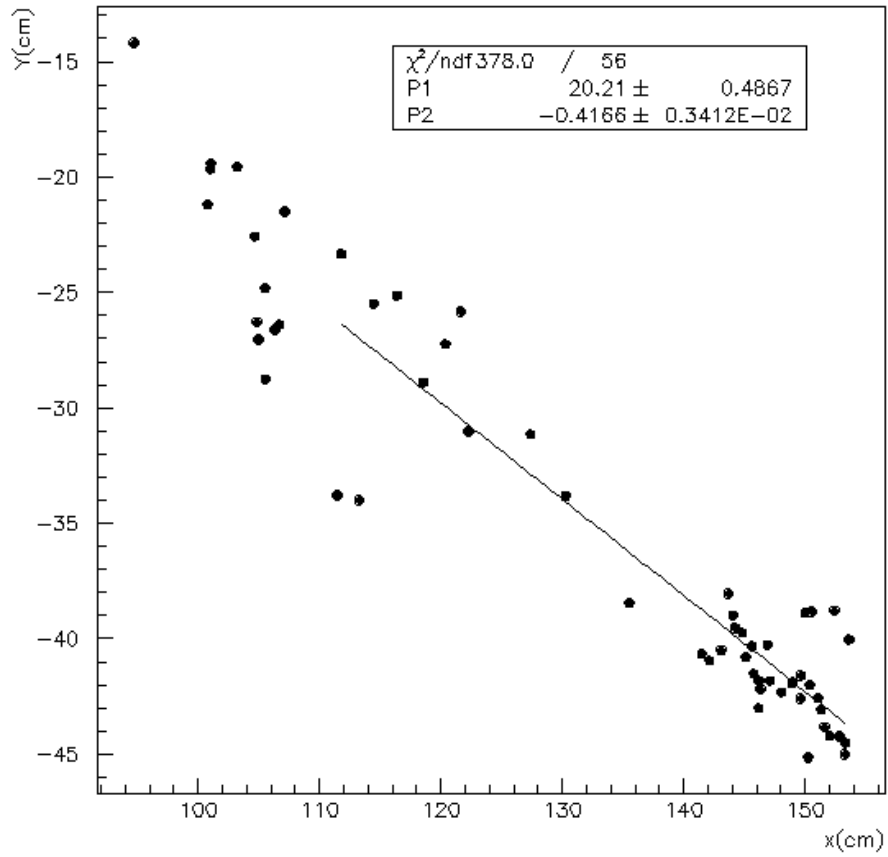


Figura 6.18: Scarti rispetto al valore medio

Dal grafico degli scarti si può notare come questi siano bassi e contenuti: hanno infatti un valore medio di 2,53 cm.

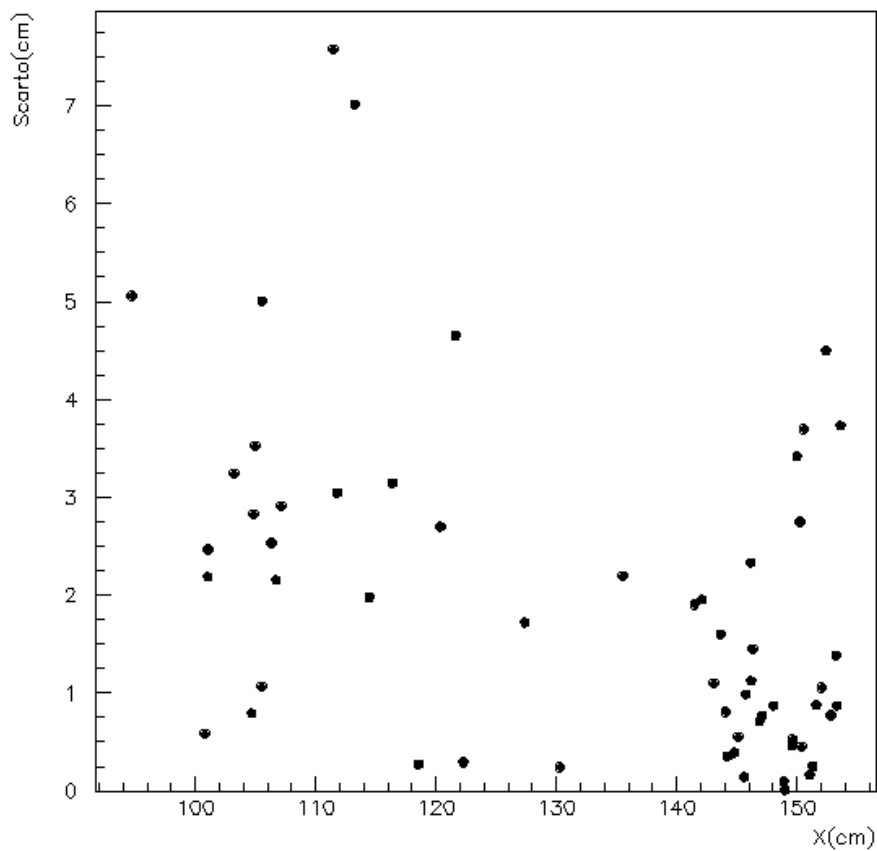


Figura 6.19: Scarti rispetto al valore medio

Se si analizzano i valori medi degli scarti calcolati nei due tratti del percorso, si può notare che essi differiscono di un ordine di grandezza: il primo vale 15 cm, mentre il secondo 2,5 cm. Questa maggiore dispersione dei dati attorno al valor medio interpolato può essere spiegata guardando la distanza tra webcam e marker. Nel tratto da B ad A la camera arriva a trovarsi a una distanza di 4 m circa dal marker, mentre nel tratto da A a C non supera il metro e mezzo. Come già analizzato nei test precedenti e preliminari si ha una maggiore dispersione dei dati e una minor precisione all'aumentare della distanza tra camera e marker.

In ogni caso i valori di scarto calcolati sono contenuti e adatti alla situazione di utilizzo del sistema.

Per quanto riguarda l'integrazione con la funzione di `ObstacleAvoidance` non si sono notate differenze rispetto al sistema con il modulo pre-esistente. Non appena la carrozzina arriva in prossimità dell'ostacolo si blocca evitando di urtare lo stesso. Dopo che l'ostacolo è stato rimosso la carrozzina riprende

il percorso che stava compiendo in precedenza.

In altri casi invece la carrozzina può provare a ri-pianificare il percorso in modo tale da evitare l'ostacolo senza interrompersi.

L'inserimento di un nuovo modulo nel sistema non ha quindi creato delle modifiche nei comportamenti standard della carrozzina. La sostituzione del nuovo modulo di *VisionExpert* risulta quindi trasparente al sistema.



## Capitolo 7

# Conclusioni e sviluppi futuri

### 7.1 Conclusioni

Con questo lavoro è stato possibile dimostrare che è possibile utilizzare le librerie ARToolKit Plus per creare un modulo di localizzazione posizionando i marker in posizione verticale.

Tale modulo è risultato essere completamente intercambiabile con il modulo di localizzazione precedentemente sviluppato per LURCH e già funzionante. In tal modo è possibile utilizzare la carrozzina con marker posizionati sia in modo verticale (su pareti o simili), sia in modo orizzontale (sul soffitto). L'introduzione nel sistema LURCH di questo modulo non compromette le funzionalità dell'intero sistema. Un esempio è dato dalle funzionalità di *ObstacleAvoidance*: si ha lo stesso comportamento sia con il modulo di localizzazione verticale sia con quello orizzontale.

La scelta della grandezza dei marker è strettamente dipendente dalle dimensioni dell'ambiente in cui il robot si deve muovere; nei test preliminari svolti in palestra si è infatti visto che con un marker di dimensioni 15x15 cm il riconoscimento ottimale del marker era garantito fino a una distanza di 3m. Ingrandendo il marker fino alla dimensione di 25x25 cm, la distanza di riconoscimento ottimale è diventata di 5m. Pertanto a seconda dell'ambiente di utilizzo del sistema di localizzazione, è buona cosa scegliere la dimensione del marker più opportuna. Da tenere ben presente quando si utilizzano/posizionano i marker in un ambiente è la questione dei riflessi: occorre posizionare i marker in posizioni il più possibile prive di riflessi di luce, in quanto questi provocano modifiche nell'immagine e quindi un non riconoscimento del marker.

## 7.2 Sviluppi futuri

**Nuovi test** Possibili sviluppi futuri di questo modulo possono riguardare l'effettuazione di nuovi test. I test svolti finora sono serviti solamente per testare l'integrazione del modulo effettuata all'interno di un sistema già esistente e funzionante. In futuro è possibile effettuare nuovi percorsi più realistici, simulando in modo più accurato il comportamento che il robot può tenere all'interno di fiere o ambienti più complessi. È possibile testare dei percorsi a S, come quelli mostrati in figura 7.1.

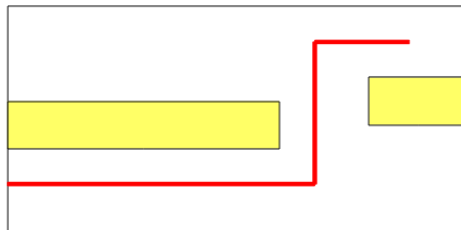


Figura 7.1: Possibili percorsi di test

Per far ciò occorre, però, prima migliorare altri moduli del sistema in grado ad esempio di creare la mappa dell'ambiente e identificare i marker presenti in maniera automatica.

**Aggiunta di telecamere** Collegata alla pianificazione di percorsi più complessi, c'è la possibilità di aggiungere al sistema più telecamere, in modo tale da garantire un angolo di vista maggiore e quindi una maggiore probabilità di inquadrare un marker durante la navigazione.

L'aggiunta di più telecamere potrebbe semplificare l'operazione di mappatura dell'ambiente con marker (diminuzione del numero di marker da utilizzare), proprio grazie all'angolo di vista più grande di cui si dispone.

Un esempio di utilizzo di più telecamere in simultanea può essere utilizzato in una pianificazione di percorso come quella rappresentata nell'immagine 7.2.

In un percorso di questo tipo si può infatti pensare di predisporre una telecamera su ogni lato della carrozzina, in modo tale da essere in grado di analizzare marker posti sia alla sua destra che alla sua sinistra. ciò potrebbe portare vantaggi nelle parti di percorso in cui occorre effettuare curve di 90°: il marker posizionato su un lato può uscire dall'angolo di vista della telecamera e il sistema di localizzazione si troverebbe privo di input finché un nuovo marker non viene inquadrato. Questo periodo di stand-by potrebbe essere ridotto grazie all'aggiunta della nuova telecamera.

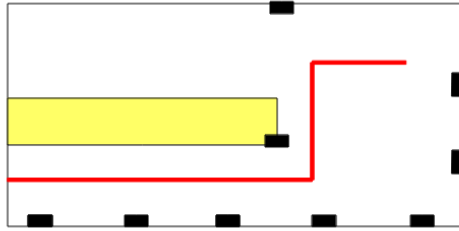


Figura 7.2: Possibili percorsi di test con l'utilizzo di 2 telecamere

**Future integrazioni** Il modulo può essere utilizzato su qualunque robot indoor che necessiti di un sistema di localizzazione. L'ambiente in cui viene usato il robot deve però poter essere attrezzato e modificato con gli opportuni marker. Un esempio di possibile integrazione può essere effettuata sul robot E-2? attualmente in via di sviluppo presso l'AirLab del politecnico di Milano.



Figura 7.3: E-2?

L'ambiente applicativo di tale robot è quello dello fiere, quindi un luogo chiuso e di dimensioni finite in cui possibile aggiungere i marker. Non è stato possibile testare fin da ora il modulo su questo robot, in quanto non sono ancora disponibili per E-2? alcuni moduli essenziali per la navigazione e la pianificazione di percorsi.

# Bibliografia

- [1] Ceriani, S., *Sviluppo di una carrozzina autonoma d'ausilio ai disabili motori*, Tesi di laurea, Politecnico di Milano, AIRLab, Artificial Intelligence and Robotics Laboratory.
- [2] Dalli, M., *Sviluppo di un sistema di controllo basato su odometria per una carrozzina robotica*, Politecnico di Milano, AIRLab, Artificial Intelligence and Robotics Laboratory.
- [3] Giussani, C., *Middleware-Guida per l'utente*.
- [4] Malagò, L., *MRT User Manual*, Politecnico di Milano, Dept. of Electronics and Information, AIRLab, Artificial Intelligence and Robotics Laboratory.
- [5] Bonarini-Matteucci-Restelli, *MRT: Robotics Off-the-Shelf with the Modular Robotic Toolkit*, Politecnico di Milano, Department of Electronics and Information.
- [6] Intel, *Open source computer vision library - reference manual*
- [7] Zanella, S., *Autolocalizzazione di robot mobile tramite riconoscimento di feature visive*, Università degli studi di Genova, Facoltà di Ingegneria
- [8] Hirokazu Kato-Mark Billingham-Ivan Poupyrev *ARToolKit*, Hiroshima City University, Human Interface Technology Laboratory University of Washington
- [9] Zingone, R., *Sistema di visione multicamera per il coordinamento di squadre di robot mobili*, Università degli studi di Siena, Facoltà di Ingegneria

# Appendice A

## Appendice

### A.1 Dati risultanti dai test preliminari

Di seguito sono riportati i dati ottenuti durante le misurazioni effettuate utilizzando sia il marker piccolo sia il marker grande.

Nella prima parte dell'allegato sono riportate la media delle matrici ottenute dall' ARToolKitPlus, nella seconda parte sono riportate le matrici indicanti la posizione della camera nel sistema mondo (matrici ottenute tramite i calcoli descritti nel paragrafo 4.3.3).

Le aree della tabella evidenziata in rosso sono le aree in cui non sono state effettuate prove (area fuori dal campo visivo della telecamera). Le aree in arancione sono le aree in cui sono state effettuate prove e in cui nessun marker è stato riconosciuto.

#### A.1.1 Posizione marker ricavata da ARToolKitPlus

I dati riportati nelle tabelle sono così strutturati:

- **Tabella misure effettuate:** riporta un 1 nelle posizioni in cui sono state effettuate delle prove di riconoscimento, degli 0 nei punti in cui non sono state effettuate prove (zone fuori dal campo di vista/di riconoscimento).
- **Tabella riconoscimento marker:** riporta il valore della percentuale di marker riconosciuti nella posizione
- **Tabella errore posizione coordinata x:** ogni cella riporta l'errore medio della posizione del marker lungo l'asse x nel sistema di riferimento camera, associato al marker posizionato in quelle coordinate
- **Tabella errore posizione coordinata z:** ogni cella riporta l'errore medio della posizione del marker lungo l'asse z nel sistema di riferimento camera, associato al marker posizionato in quelle coordinate

- **Tabella errore angolo con asse z:** ogni cella riporta il valore dello scostamento angolare dal valore teorico dell'asse z

Marker grande



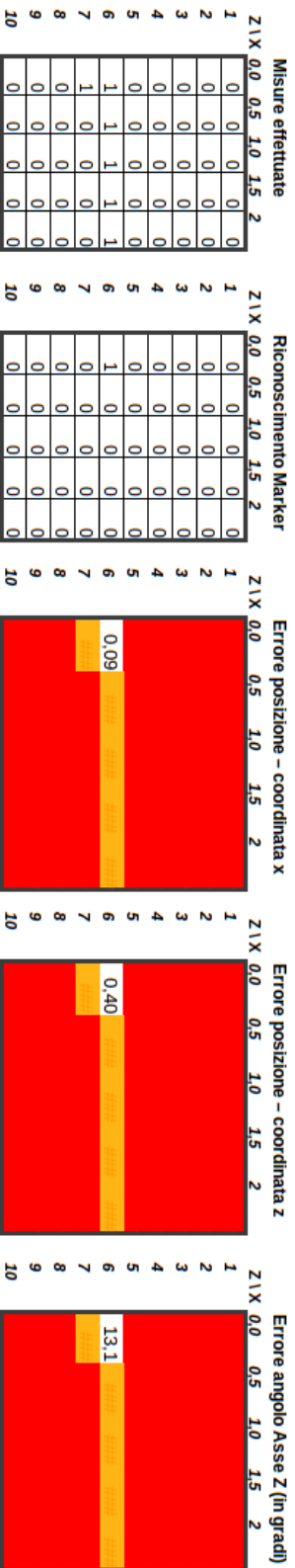
**Inclinazione 0° attorno a asse X**  
**Inclinazione 0° attorno a asse Y**

Misure effettuate		Riconoscimento Marker		Errore posizione - coordinata x		Errore posizione - coordinata z		Errore angolo Asse Z (in gradi)												
Z	X	0.0	0.5	1.0	1.5	2	Z	X	0.0	0.5	1.0	1.5	2	Z	X	0.0	0.5	1.0	1.5	2
1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
2	0	0	0	0	0	0	2	0	0	0	0	0	0	2	0	0	0	0	0	0
3	0	0	0	0	0	0	3	0	0	0	0	0	0	3	0	0	0	0	0	0
4	0	0	0	0	0	0	4	0	0	0	0	0	0	4	0	0	0	0	0	0
5	0	0	0	0	0	0	5	0	0	0	0	0	0	5	0	0	0	0	0	0
6	1	1	1	1	1	1	6	1	1	0.8	1	1	1	6	0.40	0.39	0.38	0.34	0.31	0.25
7	1	0	0	0	0	0	7	1	0	0	0	0	0	7	0.27	0.43	0.78	-2.10	-4.43	-20.7
8	0	0	0	0	0	0	8	0	0	0	0	0	0	8						
9	0	0	0	0	0	0	9	0	0	0	0	0	0	9						
10	0	0	0	0	0	0	10	0	0	0	0	0	0	10						

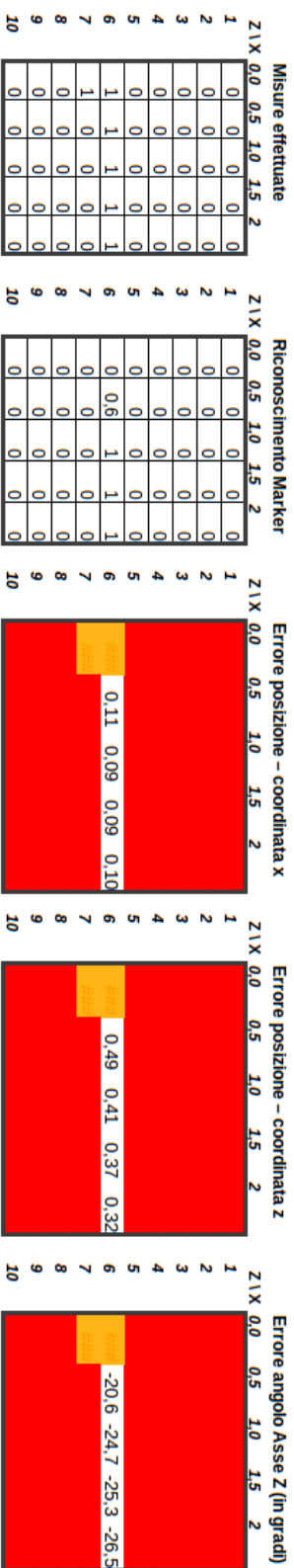
**Inclinazione 0° attorno a asse X**  
**Inclinazione 15°dx attorno a asse Y**

Misure effettuate		Riconoscimento Marker		Errore posizione - coordinata x		Errore posizione - coordinata z		Errore angolo Asse Z (in gradi)												
Z	X	0.0	0.5	1.0	1.5	2	Z	X	0.0	0.5	1.0	1.5	2	Z	X	0.0	0.5	1.0	1.5	2
1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
2	0	0	0	0	0	0	2	0	0	0	0	0	0	2	0	0	0	0	0	0
3	0	0	0	0	0	0	3	0	0	0	0	0	0	3	0	0	0	0	0	0
4	0	0	0	0	0	0	4	0	0	0	0	0	0	4	0	0	0	0	0	0
5	0	0	0	0	0	0	5	0	0	0	0	0	0	5	0	0	0	0	0	0
6	1	1	1	1	1	1	6	0	0.6	1	1	1	1	6	0.20	0.37	0.28	0.18		
7	1	0	0	0	0	0	7	0	0	0	0	0	7							
8	0	0	0	0	0	0	8	0	0	0	0	0	0	8						
9	0	0	0	0	0	0	9	0	0	0	0	0	0	9						
10	0	0	0	0	0	0	10	0	0	0	0	0	0	10						

**Inclinazione 0° attorno a asse X**  
**Inclinazione 15°sx attorno a asse Y**



**Inclinazione 0° attorno a asse X**  
**Inclinazione 30°dx attorno a asse Y**



Inclinazione 0° attorno a asse X  
Inclinazione 30°sx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	1	1	1	1	1
7	1	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	1	0,6	0,4	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Errore posizione - coordinata x

Z \ X	0,0	0,5	1,0	1,5	2
1					
2					
3					
4					
5					
6	0,12	0,16	0,19		
7					
8					
9					
10					

Errore posizione - coordinata z

Z \ X	0,0	0,5	1,0	1,5	2
1					
2					
3					
4					
5					
6	0,39	0,41	0,43		
7					
8					
9					
10					

Errore angolo Asse Z (in gradi)

Z \ X	0,0	0,5	1,0	1,5	2
1					
2					
3					
4					
5					
6	2,75	0,04	-0,20		
7					
8					
9					
10					

Inclinazione 15° attorno a asse X  
Inclinazione 0° attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	1	1	0	0
6	1	1	1	1	1
7	1	1	1	1	1
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	1	0,8	0	0
6	0,8	1	1	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Errore posizione - coordinata x

Z \ X	0,0	0,5	1,0	1,5	2
1	0,01				
2	0,03	0,03			
3	0,06	0,05	0,05		
4	0,04	0,06	0,08		
5	0,05	0,09	0,13		
6	0,06	0,09	0,12		
7					
8					
9					
10					

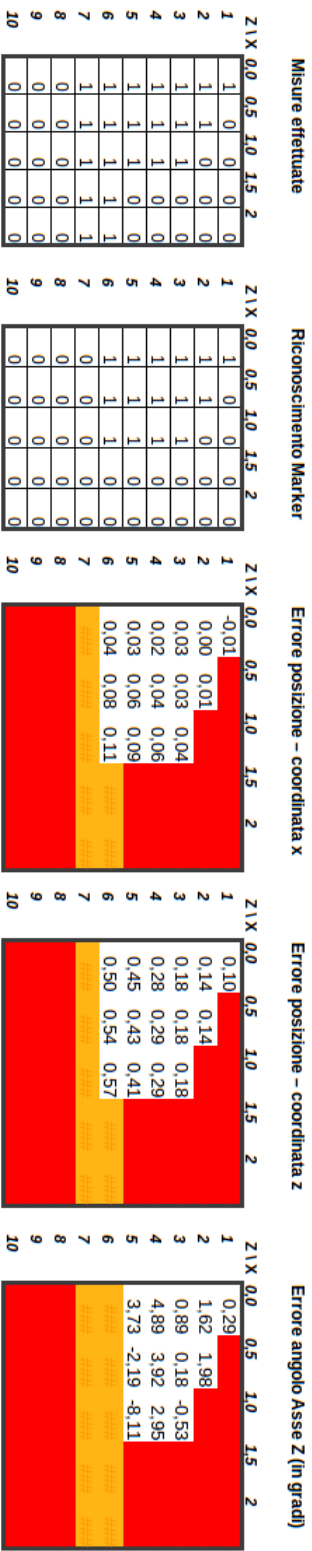
Errore posizione - coordinata z

Z \ X	0,0	0,5	1,0	1,5	2
1	0,10				
2	0,13	0,13			
3	0,14	0,14	0,14		
4	0,36	0,31	0,26		
5	0,28	0,36	0,44		
6	0,41	0,42	0,43		
7					
8					
9					
10					

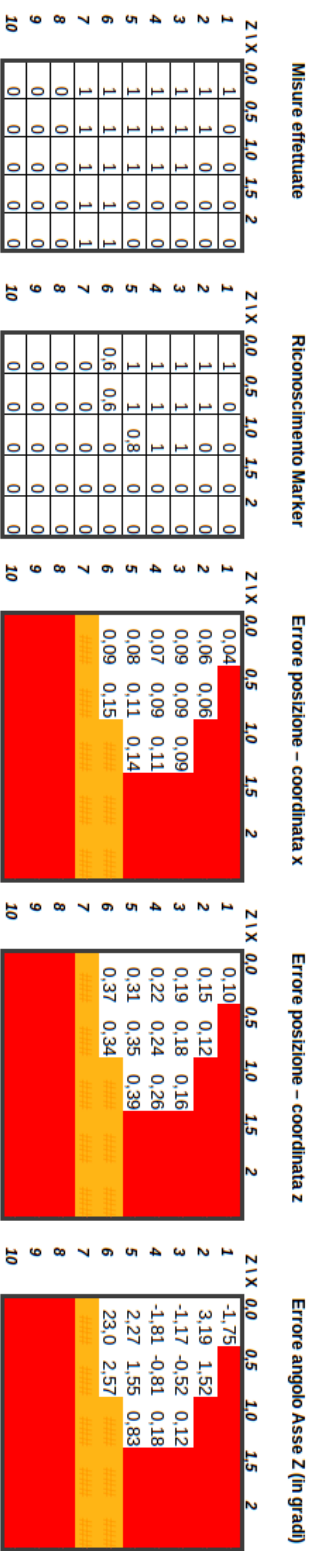
Errore angolo Asse Z (in gradi)

Z \ X	0,0	0,5	1,0	1,5	2
1	0,03				
2	1,83	2,04			
3	0,48	-0,35	-1,18		
4	0,78	1,32	1,86		
5	0,77	3,93	7,10		
6					
7					
8					
9					
10					

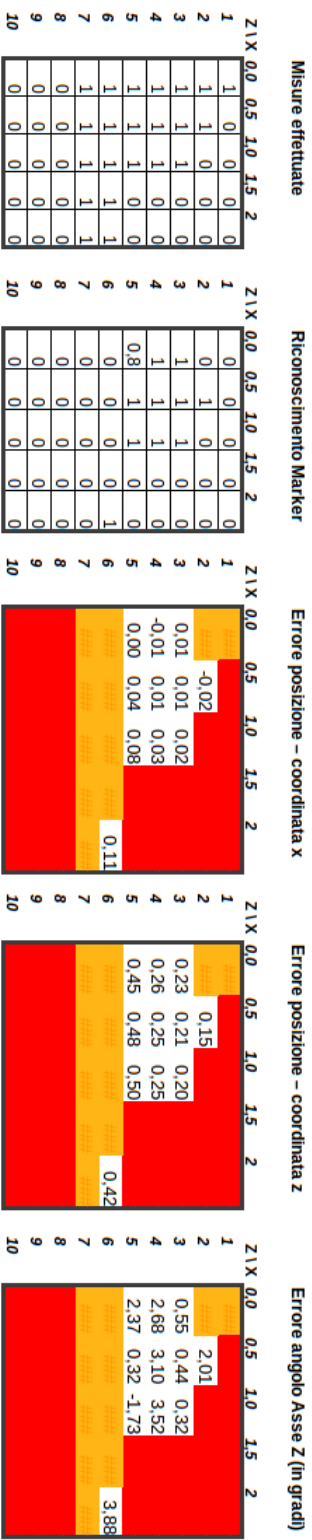
**Inclinazione 15° attorno a asse X**  
**Inclinazione 15°dx attorno a asse Y**



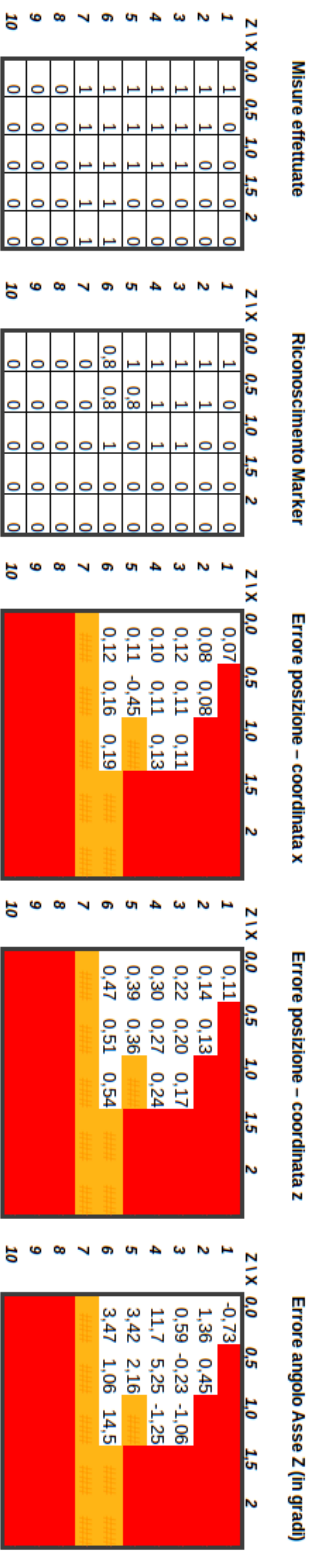
**Inclinazione 15° attorno a asse X**  
**Inclinazione 15°sx attorno a asse Y**



**Inclinazione 15° attorno a asse X**  
**Inclinazione 30°dx attorno a asse Y**

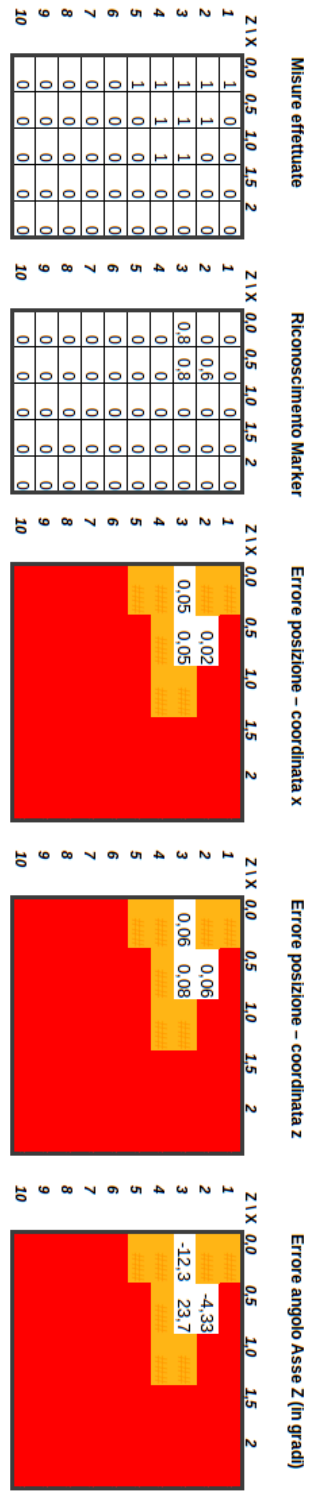


**Inclinazione 15° attorno a asse X**  
**Inclinazione 30°sx attorno a asse Y**

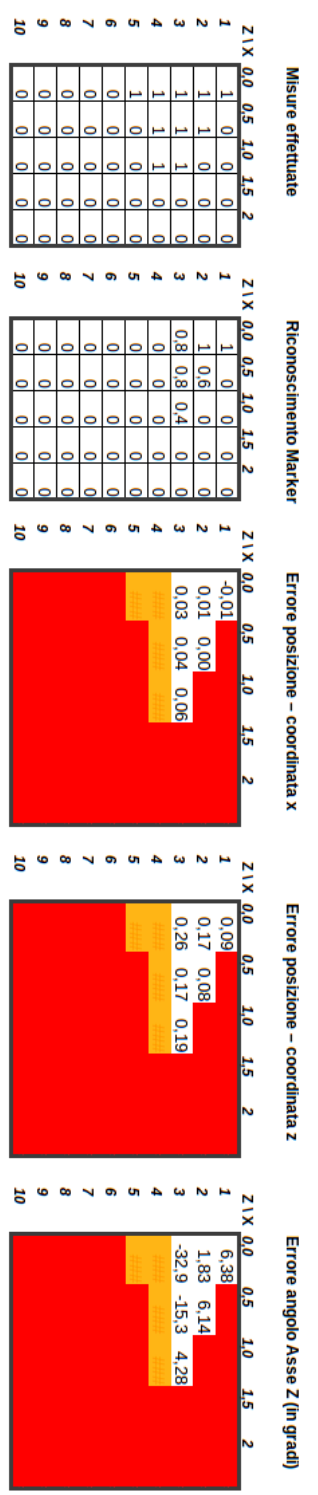


Marker piccolo

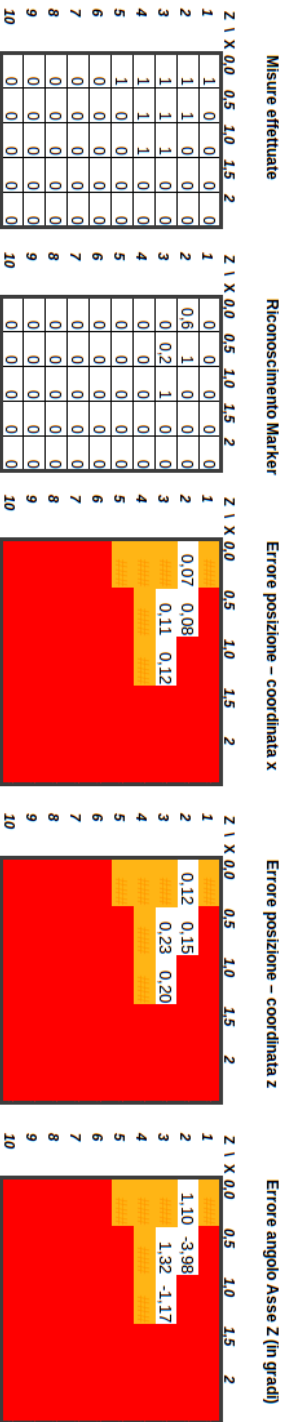
**Inclinazione 0° attorno a asse X**  
**Inclinazione 0° attorno a asse Y**



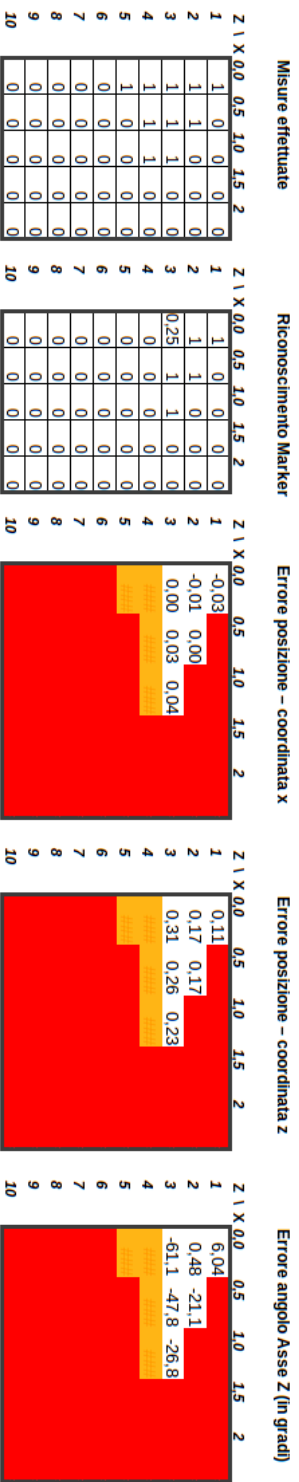
**Inclinazione 0° attorno a asse X**  
**Inclinazione 15°dx attorno a asse Y**



Inclinazione 0° attorno a asse X  
Inclinazione 15°sx attorno a asse Y

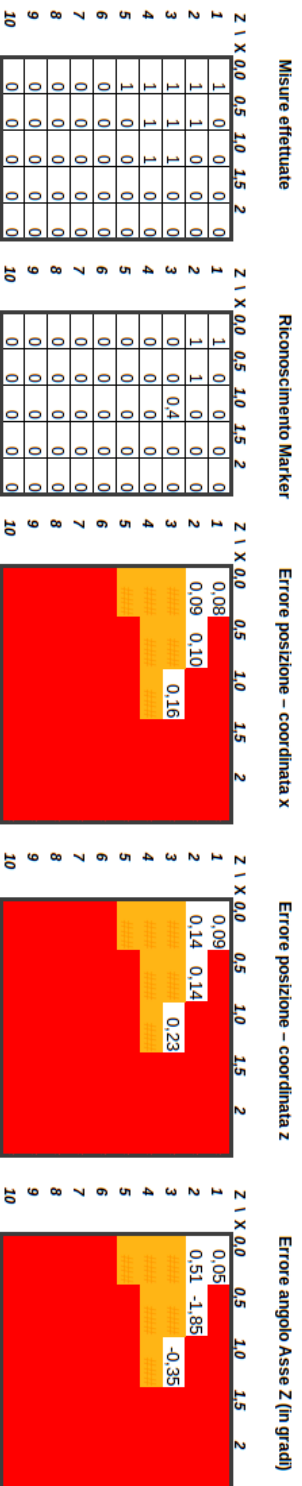


Inclinazione 0° attorno a asse X  
Inclinazione 30°dx attorno a asse Y

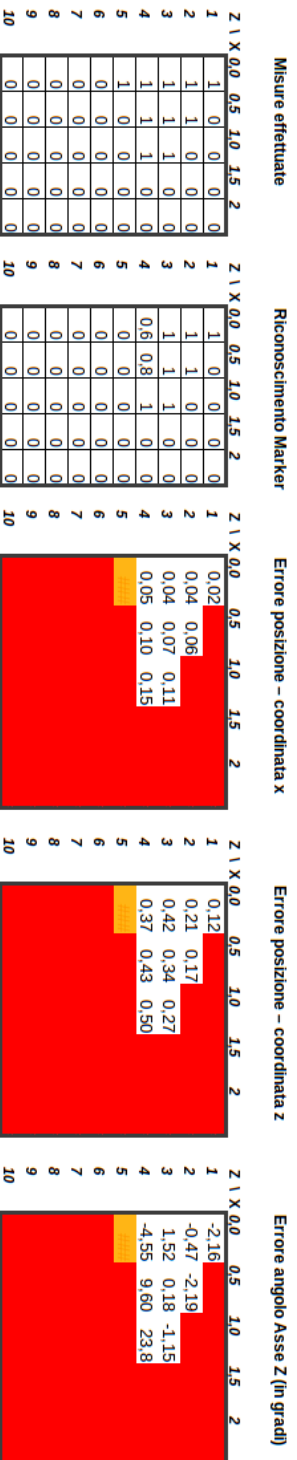




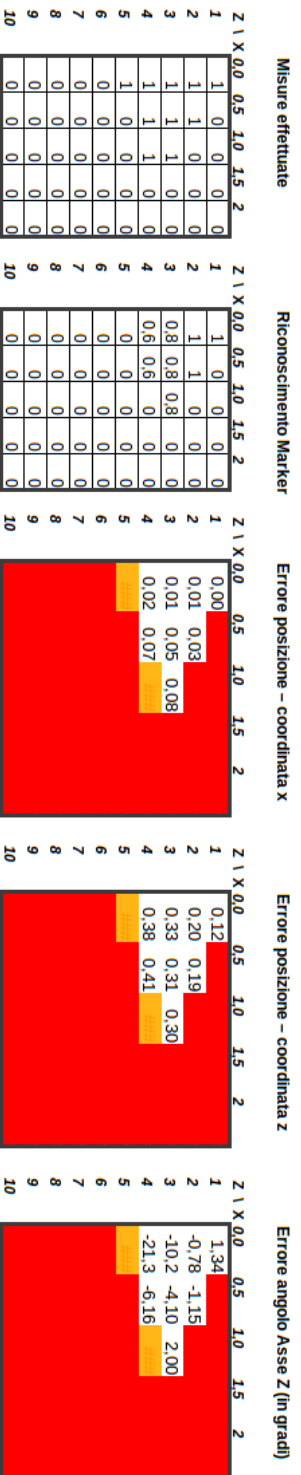
Inclinazione 0° attorno a asse X  
 Inclinazione 30°sx attorno a asse Y



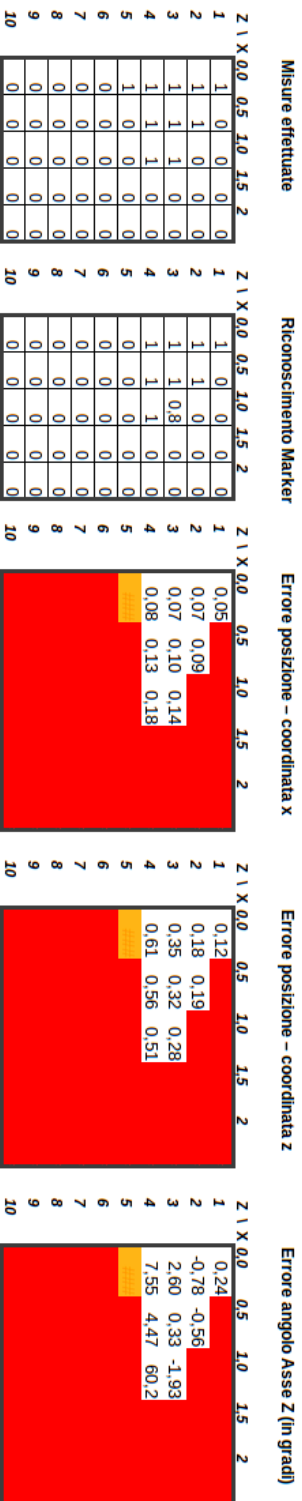
Inclinazione 15° attorno a asse X  
 Inclinazione 0° attorno a asse Y



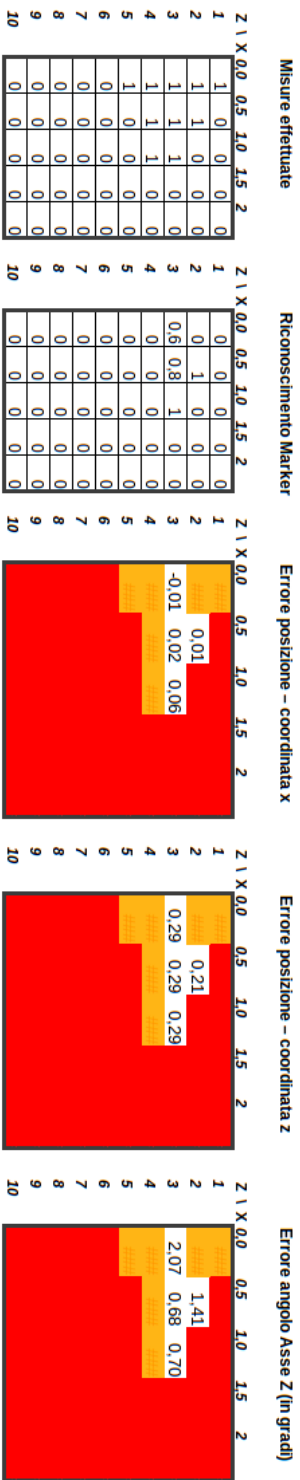
**Inclinazione 15° attorno a asse X**  
**Inclinazione 15°dx attorno a asse Y**



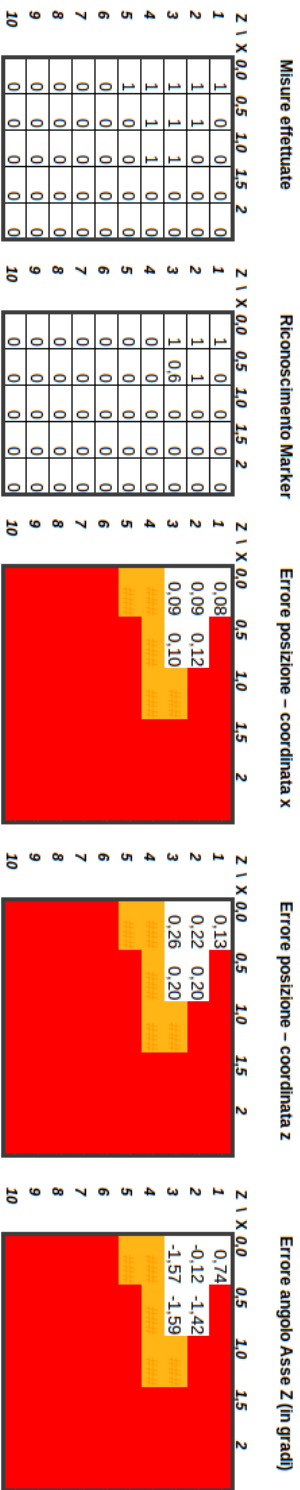
**Inclinazione 15° attorno a asse X**  
**Inclinazione 15°sx attorno a asse Y**



Inclinazione 15° attorno a asse X  
 Inclinazione 30°dx attorno a asse Y



Inclinazione 15° attorno a asse X  
 Inclinazione 30°sx attorno a asse Y



### A.1.2 Tabelle del posizionamento e della rotazione della telecamera nel sistema di riferimento mondo

I dati riportati nelle tabelle sono così strutturati:

- **Tabella misure effettuate:** riporta un 1 nelle posizioni in cui sono state effettuate delle prove di riconoscimento, degli 0 nei punti in cui non sono state effettuate prove (zone fuori dal campo di vista/di riconoscimento)
- **Tabella riconoscimento marker:** riporta il valore della percentuale di marker riconosciuti nella posizione
- **Tabella coordinata x webcam:** ogni cella riporta il valore della coordinata x della telecamera nel sistema di riferimento mondo, associata al marker posizionato in quelle coordinate
- **Tabella coordinata z webcam:** ogni cella riporta il valore della coordinata z della telecamera nel sistema di riferimento mondo, associata al marker posizionato in quelle coordinate
- **Tabella componente x del vettore z telecamera e tabella componente z del vettore z telecamera:** componenti nel piano z-x del vettore z (asse ottico della telecamera) da cui ricavare lo scostamento angolare dal valore teorico
- **Tabella angolo con asse z:** ogni cella riporta il valore dello scostamento angolare dal valore teorico dell'asse z

Marker grande

**Inclinazione 0° attorno a asse X**  
**Inclinazione 0° attorno a asse Y**

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	1	1	1	1	1
7	1	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	1	1	0,8	1	1
7	1	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	-0,09	-0,07	-0,05	-0,34	-0,63
7					-2,63
8					
9					
10					

Coordinata Z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	0,42	0,41	0,40	0,27	0,14
7					-0,23
8					
9					
10					

Componente X del versore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	0,00	0,00	-0,01	0,03	0,08
7					0,35
8					
9					
10					

Componente Z del versore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	-1,00	-1,00	-1,00	-1,00	-1,00
7					-0,93
8					
9					
10					

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	-0,27	0,26	0,79	-1,81	-4,40
7					-20,6
8					
9					
10					

Inclinazione 0° attorno a asse X  
 Inclinazione 15° dx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	1	1	1	1	1
7	1	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0,6	1	1	1
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X webcam

Z \ X	0,0	0,5	1,0	1,5	2,0
1					
2					
3					
4					
5					
6	0,47	-1,28	0,27	0,32	
7					
8					
9					
10					

Coordinata Z webcam

Z \ X	0,0	0,5	1,0	1,5	2,0
1					
2					
3					
4					
5					
6	0,19	0,14	0,16	0,17	
7					
8					
9					
10					

Componente X del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	50,00	0,01	0,18	0,10	-0,06
7	50,00				
8					
9					
10					

Componente Z del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	50,00	-0,99	-0,99	-1,00	-0,99
7	50,00				
8					
9					
10					

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	50,00	-0,38	-10,6	-5,64	3,55
7	50,00				
8					
9					
10					

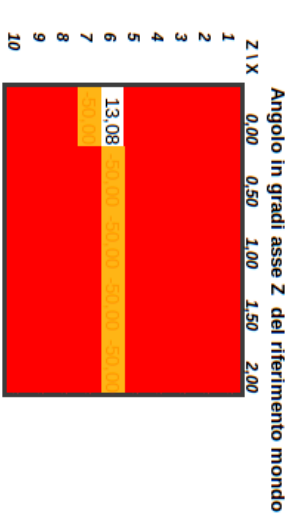
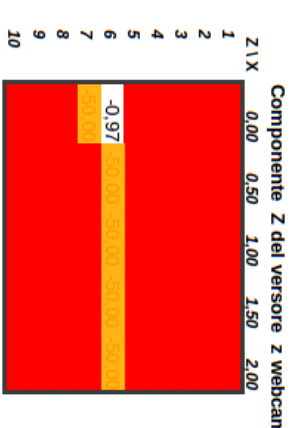
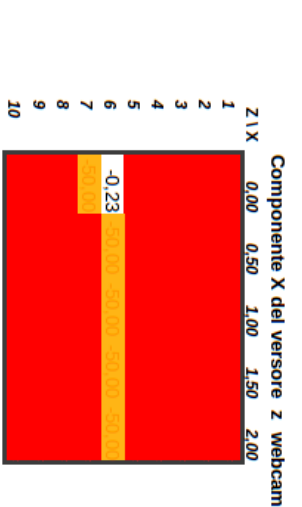
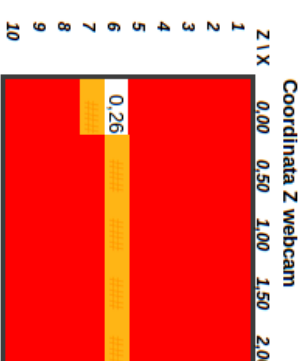
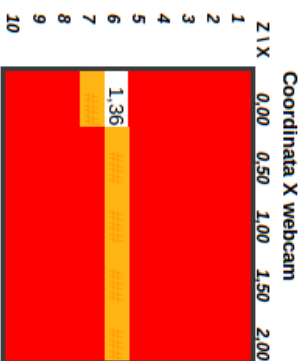
Inclinazione 0° attorno a asse X  
 Inclinazione 15° Sx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	1	1	1	1	1
7	1	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	1	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0





Inclinazione 0° attorno a asse X  
 Inclinazione 30° dx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	1	1	1	1	1
7	1	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0,6	1	1	1
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X webcam

Z \ X	0,0	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6			-2,77	-2,94	-2,66
7					
8					
9					
10					

Coordinata Z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6			-0,36	-0,21	-0,86
7					
8					
9					
10					

Componente X del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6			0,16	0,45	0,42
7					
8					
9					
10					

Componente Z del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6			-0,99	-0,98	-0,99
7					
8					
9					
10					

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6			-9,17	-24,7	-23,2
7					
8					
9					
10					

Inclinazione 0° attorno a asse X  
 Inclinazione 30° Sx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	1	1	1	1	1
7	1	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	1	0,6	0,4	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	0,19	0,00	-0,19		
7					
8					
9					
10					

Coordinata Z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	0,40	0,41	0,43		
7					
8					
9					
10					

Componente X del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	-0,05	-0,02	0,00	0,00	0,00
7					
8					
9					
10					

Componente Z del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	-1,00	-1,00	-0,99	-0,99	-0,99
7					
8					
9					
10					

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2					
3					
4					
5					
6	2,78	1,41	0,02	0,00	0,00
7					
8					
9					
10					

**Inclinazione 15° attorno a asse X**  
**Inclinazione 0° attorno a asse Y**

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	1	1	0	0
6	1	1	1	1	1
7	1	1	1	1	1
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	1	0,8	0	0
6	0,8	1	1	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,01				
2	0,03	0,04			
3	-0,05	-0,09	-0,12		
4	0,02	0,03	0,05		
5	0,01	0,29	0,56		
6	0,01	1,29	2,58		
7					
8					
9					
10					

Coordinata Z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,08				
2	-0,19	-0,20			
3	-0,38	-0,35	-0,32		
4	-0,24	-0,34	-0,44		
5	-0,19	-0,25	-0,31		
6	-0,70	-0,19	0,33		
7					
8					
9					
10					

Componente X del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	0,00				
2	-0,03	-0,03			
3	0,00	0,01	0,02		
4	-0,01	-0,02	-0,03		
5	-0,01	-0,07	-0,12		
6	-0,01	-0,21	-0,40		
7					
8					
9					
10					

Componente Z del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,88				
2	-0,88	-0,87			
3	-0,86	-0,88	-0,89		
4	-0,89	-0,87	-0,86		
5	-0,94	-0,90	-0,87		
6	-0,85	-0,88	-0,91		
7					
8					
9					
10					

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1	0,02				
2	1,89	2,26			
3	0,20	-0,62	-1,42		
4	0,87	1,50	2,15		
5	0,80	4,37	8,20		
6	0,71	13,23	23,96		
7					
8					
9					
10					

## Inclinazione 15° attorno a asse X Inclinazione 15° dx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	1	1	0	0
6	1	1	1	1	1
7	1	1	1	1	1
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	1	1	0	0
6	1	1	1	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	0,04	0,10	-0,01		
2	0,10	0,10	-0,01		
3	0,08	0,04	-0,01		
4	0,42	0,33	0,23		
5	0,42	-0,16	-0,75		
6	0,43	-0,46	-1,35		
7					
8					
9					
10					

Coordinata Z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,07	-0,13	-0,28		
2	-0,15	-0,13	-0,28		
3	-0,29	-0,28	-0,28		
4	-0,28	-0,24	-0,20		
5	-0,12	0,06	0,24		
6	-0,41	-0,08	0,25		
7					
8					
9					
10					

Componente X del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,02	-0,05	-0,01		
2	-0,04	-0,02	-0,01		
3	-0,03	-0,02	-0,07		
4	-0,10	-0,08	-0,07		
5	-0,08	0,02	0,12		
6	-0,07	0,06	0,19		
7					
8					
9					
10					

Componente Z del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,89	-0,90	-0,89		
2	-0,90	-0,90	-0,89		
3	-0,88	-0,89	-0,89		
4	-0,89	-0,90	-0,90		
5	-0,92	-0,95	-0,99		
6	-0,88	-0,93	-0,98		
7					
8					
9					
10					

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1	1,33	3,06	0,38		
2	2,73	3,06	0,38		
3	1,96	1,16	0,38		
4	6,32	5,23	4,15		
5	4,96	-1,21	-6,88		
6	4,52	-3,57	-10,7		
7					
8					
9					
10					

Inclinazione 15° attorno a asse X  
 Inclinazione 15° Sx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	0	0	0	0
3	1	1	0	0	0
4	1	1	0	0	0
5	1	1	0	0	0
6	1	1	1	1	1
7	1	1	1	1	1
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	1	0,8	0	0
6	0,6	0,6	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,10				
2	0,02	-0,05			
3	-0,23	-0,19	-0,15		
4	-0,31	-0,25	-0,19		
5	0,01	-0,09	-0,19		
6	2,30	1,35			
7					
8					
9					
10					

Coordinata Z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,07				
2	-0,14	-0,16			
3	-0,24	-0,25	-0,26		
4	-0,41	-0,36	-0,31		
5	-0,49	-0,38	-0,28		
6	-0,08	-0,41			
7					
8					
9					
10					

Componente X del versore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	0,05				
2	-0,04	-0,01			
3	0,04	0,03	0,02		
4	0,05	0,03	0,01		
5	-0,02	-0,01	0,00		
6	-0,37	-0,17			
7					
8					
9					
10					

Componente Z del versore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,89				
2	-0,90	-0,89			
3	-0,89	-0,89	-0,89		
4	-0,88	-0,88	-0,89		
5	-0,88	-0,88	-0,89		
6	-0,92	-0,90			
7					
8					
9					
10					

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-3,15				
2	2,46	0,47			
3	-2,60	-1,84	-1,08		
4	-3,47	-2,19	-0,93		
5	1,37	0,54	-0,29		
6	21,95	10,46			
7					
8					
9					
10					

**Inclinazione 15° attorno a asse X**  
**Inclinazione 30° dx attorno a asse Y**

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	1	1	0	0
6	1	1	1	1	1
7	1	1	1	1	1
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	0,8	1	1	0	0
6	0	0	0	0	1
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X Webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2		0,18			
3	0,14	0,14	0,13		
4	0,38	0,40	0,41		
5	0,42	0,17	-0,08		
6				0,61	
7					
8					
9					
10					

Coordinata Z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2		-0,11			
3	-0,17	-0,18	-0,19		
4	-0,33	-0,31	-0,29		
5	-0,17	-0,10	-0,02		
6					
7					
8					
9					
10					-0,38

Componente X del versore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	0,996				
2		-0,06			
3	-0,04	-0,04	-0,04		
4	-0,08	-0,09	-0,09		
5	-0,07	-0,03	0,01		
6	0,00	-0,09	-0,09	-0,09	-0,10
7	0,00	-0,09	-0,09	-0,09	-0,09
8					
9					
10					

Componente Z del versore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1					
2		-0,91			
3	-0,90	-0,91	-0,91		
4	-0,89	-0,89	-0,89		
5	-0,91	-0,92	-0,93		
6					-0,99
7					
8					
9					
10					

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1	0,00				
2		4,07			
3	2,53	2,39	2,26		
4	5,14	5,57	6,01		
5	4,51	2,06	-0,33		
6	0,00	-0,09	-0,09	-0,09	6,51
7	0,00	-0,09	-0,09	-0,09	-0,09
8					
9					
10					

**Inclinazione 15° attorno a asse X**  
**Inclinazione 30° Sx attorno a asse Y**

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	1	1	0	0
6	1	1	1	1	1
7	1	1	1	1	1
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	0,8	0	0	0
6	0,8	0,8	1	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X WebCam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,13	-0,15	-0,15	-0,15	-0,15
2	-0,12	-0,15	-0,29	-0,29	-0,29
3	-0,20	-0,25	-0,29	-0,29	-0,29
4	0,71	0,16	-0,39	-0,39	-0,39
5	-0,01	-0,12	-0,39	-0,39	-0,39
6	0,00	0,00	1,30	1,30	1,30
7	0,00	0,00	1,30	1,30	1,30
8	0,00	0,00	1,30	1,30	1,30
9	0,00	0,00	1,30	1,30	1,30
10	0,00	0,00	1,30	1,30	1,30

Coordinata Z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,03	-0,12	-0,12	-0,12	-0,12
2	-0,13	-0,17	-0,20	-0,20	-0,20
3	-0,13	-0,17	-0,20	-0,20	-0,20
4	0,30	0,01	-0,28	-0,28	-0,28
5	-0,31	-0,35	-0,35	-0,35	-0,35
6	-0,33	-0,38	1,13	1,13	1,13
7	-0,33	-0,38	1,13	1,13	1,13
8	-0,33	-0,38	1,13	1,13	1,13
9	-0,33	-0,38	1,13	1,13	1,13
10	-0,33	-0,38	1,13	1,13	1,13

Componente X del versore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	0,04	0,03	0,03	0,03	0,03
2	0,01	0,04	0,05	0,05	0,05
3	0,02	0,04	0,06	0,06	0,06
4	-0,20	-0,07	0,06	0,06	0,06
5	-0,02	-0,06	0,06	0,06	0,06
6	-0,03	-0,14	-0,26	-0,26	-0,26
7	-0,03	-0,14	-0,26	-0,26	-0,26
8	-0,03	-0,14	-0,26	-0,26	-0,26
9	-0,03	-0,14	-0,26	-0,26	-0,26
10	-0,03	-0,14	-0,26	-0,26	-0,26

Componente Z del versore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,91	-0,90	-0,90	-0,90	-0,90
2	-0,90	-0,90	-0,90	-0,90	-0,90
3	-0,91	-0,91	-0,90	-0,90	-0,90
4	-1,01	-0,96	-0,90	-0,90	-0,90
5	-0,89	-0,93	-0,90	-0,90	-0,90
6	-0,90	-0,98	-1,06	-1,06	-1,06
7	-0,90	-0,98	-1,06	-1,06	-1,06
8	-0,90	-0,98	-1,06	-1,06	-1,06
9	-0,90	-0,98	-1,06	-1,06	-1,06
10	-0,90	-0,98	-1,06	-1,06	-1,06

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-2,82	-1,65	-1,65	-1,65	-1,65
2	-0,67	-1,65	-3,26	-3,26	-3,26
3	-1,28	-2,27	-3,26	-3,26	-3,26
4	11,01	4,22	-3,56	-3,56	-3,56
5	1,59	3,69	-3,56	-3,56	-3,56
6	1,65	8,27	13,72	13,72	13,72
7	1,65	8,27	13,72	13,72	13,72
8	1,65	8,27	13,72	13,72	13,72
9	1,65	8,27	13,72	13,72	13,72
10	1,65	8,27	13,72	13,72	13,72

Marker piccolo



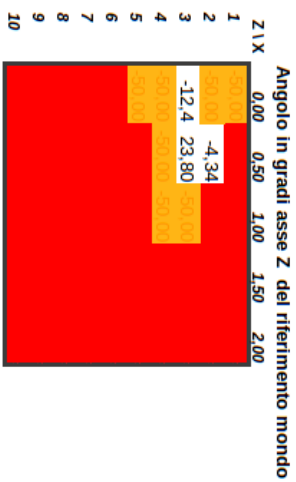
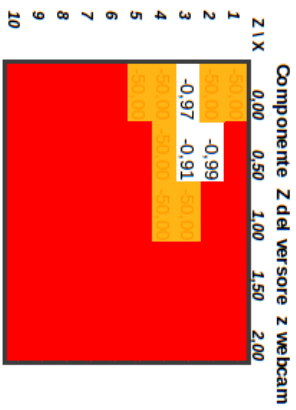
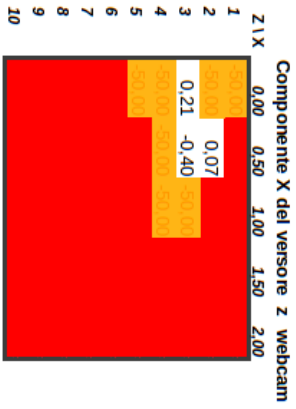
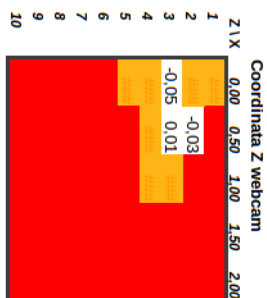
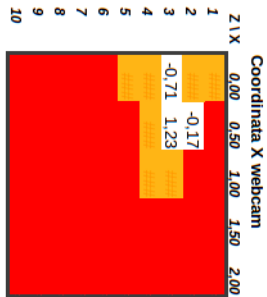
Inclinazione 0° attorno a asse X  
 Inclinazione 0° attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	0,6	0	0	0
3	0,8	0,8	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0



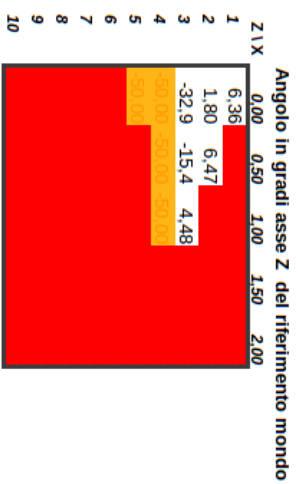
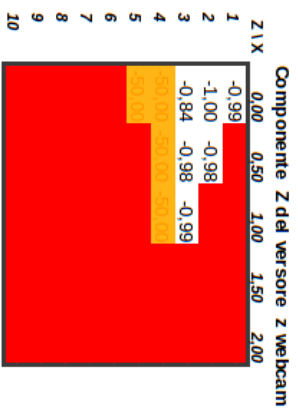
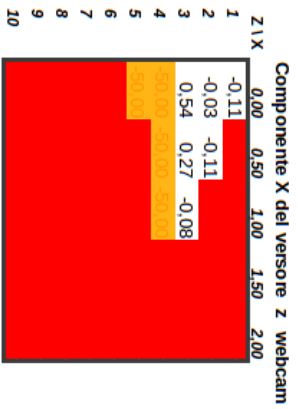
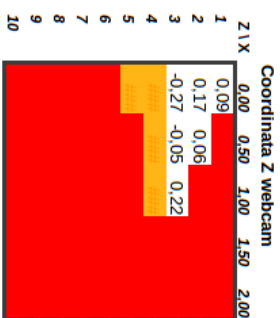
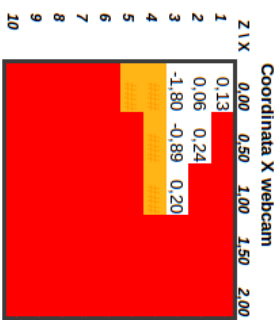
Inclinazione 0° attorno a asse X  
 Inclinazione 15°dx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	0,6	0	0	0
3	0,8	0,8	0,4	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0



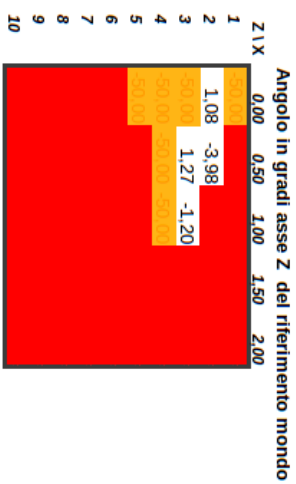
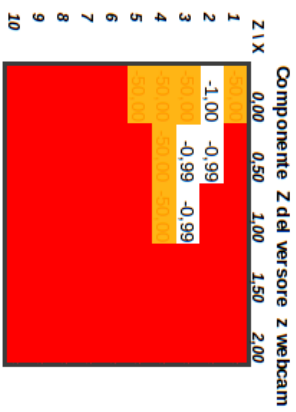
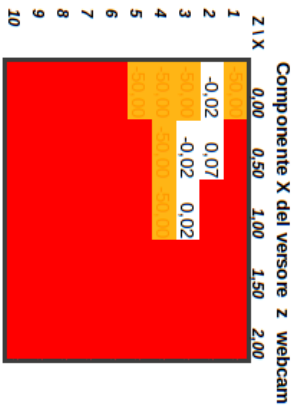
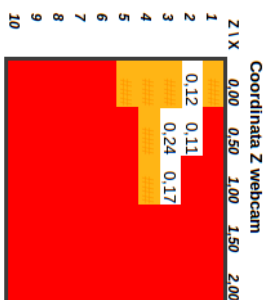
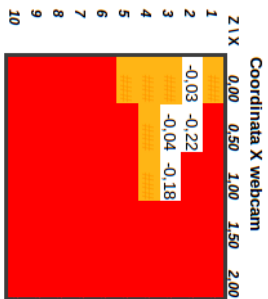
Inclinazione 0° attorno a asse X  
 Inclinazione 15°sx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0,6	1	0	0	0
3	0	0,2	1	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0



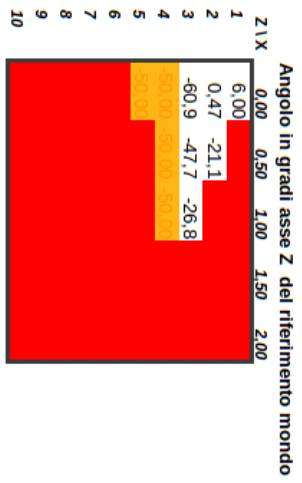
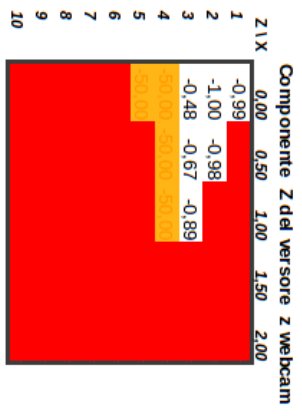
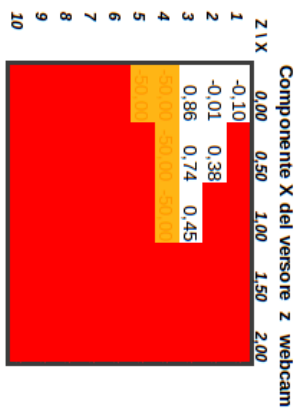
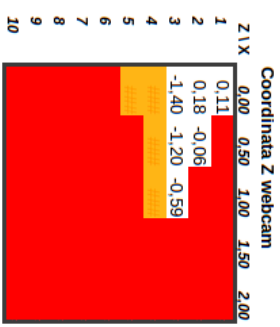
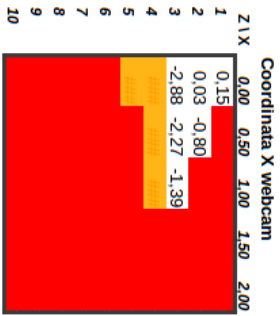
Inclinazione 0° attorno a asse X  
 Inclinazione 30°dx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	0,25	1	1	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0



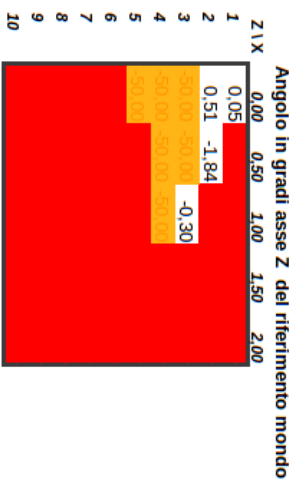
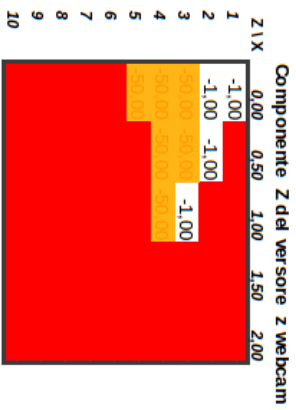
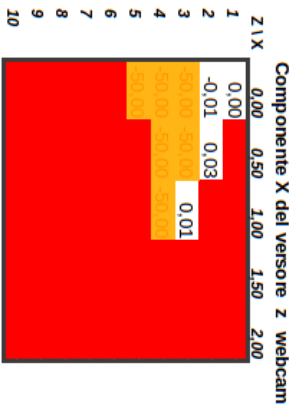
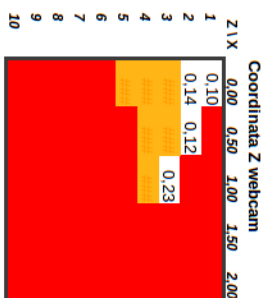
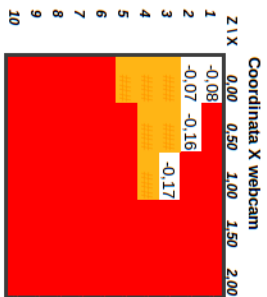
Inclinazione 0° attorno a asse X  
 Inclinazione 30°sx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	1	0
5	1	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	0	0,4	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0



Inclinazione 15° attorno a asse X  
 Inclinazione 0° attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	0,6	0,8	1	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,06				
2	-0,07	-0,14			
3	0,05	-0,06	-0,18		
4	-0,41	0,69	1,78		
5					
6					
7					
8					
9					
10					

Coordinata Z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,06				
2	-0,11	-0,13			
3	0,37	0,08	-0,20		
4	-0,46	0,12	0,70		
5					
6					
7					
8					
9					
10					

Componente X del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	0,04				
2	0,01	0,04			
3	-0,03	0,00	0,02		
4	0,08	-0,16	-0,41		
5					
6					
7					
8					
9					
10					

Componente Z del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,89				
2	-0,89	-0,90			
3	-1,00	-0,94	-0,89		
4	-0,84	-0,89	-0,94		
5					
6					
7					
8					
9					
10					

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-2,35				
2	-0,85	-2,50			
3	1,56	0,18	-1,38		
4	-5,47	10,49	23,60		
5					
6					
7					
8					
9					
10					

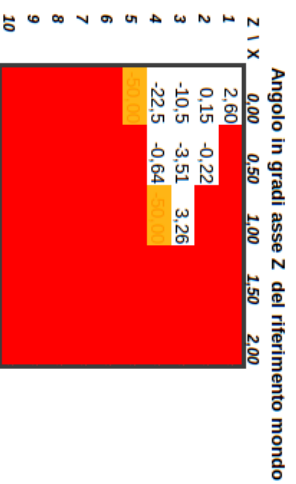
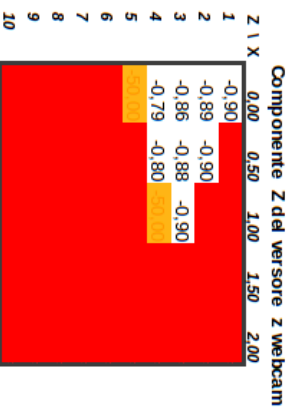
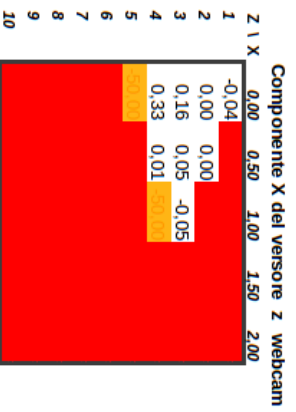
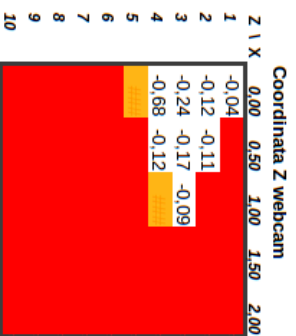
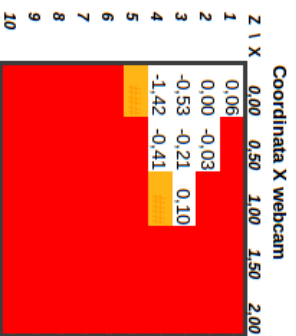
Inclinazione 15° attorno a asse X  
 Inclinazione 15°dx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2
1	-1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	-1	0	0	0	0
2	1	1	0	0	0
3	0,8	0,8	0,8	0	0
4	0,6	0,6	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0



Inclinazione 15° attorno a asse X  
 Inclinazione 15°sx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	0,8	0	0
4	1	1	1	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Coordinata X webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,07				
2	-0,15	-0,16			
3	0,01	-0,15	-0,31		
4	0,45	2,37	4,29		
5					
6					
7					
8					
9					
10					

Coordinata Z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,04				
2	-0,16	-0,10			
3	-0,03	-0,09	-0,14		
4	0,31	-0,19	-0,69		
5					
6					
7					
8					
9					
10					

Componente X del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	0,01				
2	0,03	0,03			
3	-0,03	0,01	0,05		
4	-0,12	-0,49	-0,86		
5					
6					
7					
8					
9					
10					

Componente Z del vettore z webcam

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,90				
2	-0,88	-0,90			
3	-0,91	-0,91	-0,90		
4	-0,95	-0,73	-0,51		
5					
6					
7					
8					
9					
10					

Angolo in gradi asse Z del riferimento mondo

Z \ X	0,00	0,50	1,00	1,50	2,00
1	-0,72				
2	-2,17	-1,65			
3	1,80	-0,70	-3,21		
4	7,20	33,84	59,20		
5					
6					
7					
8					
9					
10					



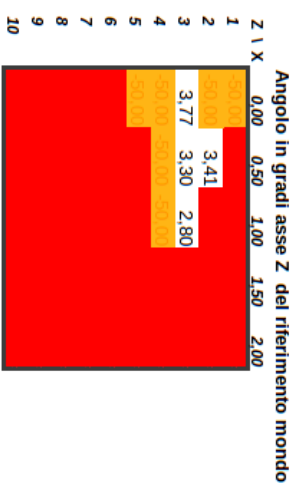
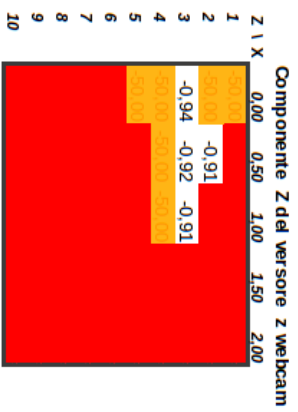
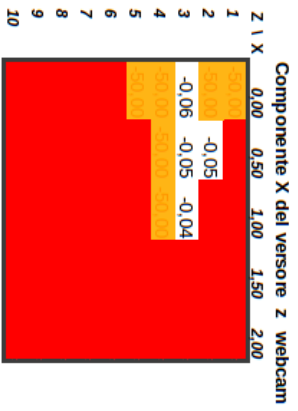
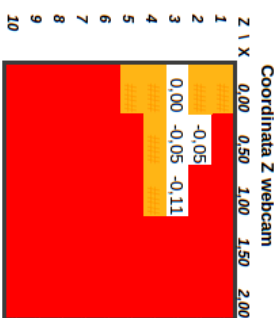
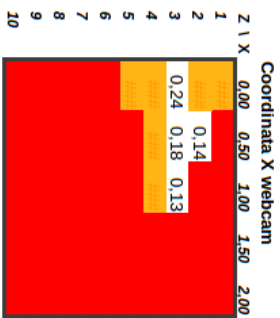
Inclinazione 15° attorno a asse X  
 Inclinazione 30°dx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	0	0	0	0	0
2	0	1	0	0	0
3	0,6	0,8	1	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0



Inclinazione 15° attorno a asse X  
 Inclinazione 30°sx attorno a asse Y

Misure effettuate

Z \ X	0,0	0,5	1,0	1,5	2
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	0	0
5	1	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Riconoscimento Marker

Z \ X	0,0	0,5	1,0	1,5	2,0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	0,6	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

