POLITECNICO DI MILANO
Corso di Laurea in Ingegneria Informatica
Dipartimento di Elettronica e Informazione

Sviluppo di una carrozzina autonoma d'ausilio ai disabili motori

AI & R Lab
Laboratorio di Intelligenza Artificiale
e Robotica del Politecnico di Milano

Relatore: Ing. Matteo Matteucci
Correlatore: Ing. Davide Migliore
Correlatore: Ing. Giulio Fontana

Tesi di Laurea Specialistica di:
Simone Ceriani, matricola 679817

Anno Accademico 2006-2007

Sommario

Il presente lavoro si colloca nell'ambito della robotica mobile, disciplina che si occupa di studiare e progettare veicoli autonomi dotati di funzionalità "intelligenti", ovvero di caratteristiche che li rendono in grado di operare e muoversi senza bisogno d'intervento umano o riducendo i comandi che l'uomo deve impartire. In particolare si sono applicate le tecniche della robotica mobile allo sviluppo di una carrozzina d'ausilio ai disabili motori, estendendo le funzionalità di una carrozzina elettrica commerciale al fine di studiare come avvicinarla alle esigenze degli utenti che trovano difficoltà nell'utilizzare i normali ausili alla mobilità. Lo scopo della presente tesi è stato progettare e realizzare un prototipo di carrozzina elettrica che renda disponibili funzionalità aggiuntive rispetto a quelle offerte dai prodotti presenti sul mercato, partendo proprio da una carrozzina commerciale a cui apportare modifiche. Le modifiche apportate riguardano la creazione di un circuito d'interfaccia che permette di comandare la carrozzina tramite l'uso di un computer, l'interfacciamento di sensori per il rilevamento ostacoli e lo sviluppo del software di controllo che gestisce e governa il moto della carrozzina permettendo comportamenti di guida assistita e guida autonoma. La guida assistita permette di evitare collisioni con ostacoli, mentre la guida autonoma permette alla carrozzina di pianificare ed eseguire percorsi, localizzandosi nell'ambiente grazie a una telecamera e a degli algoritmi di posizionamento basati su landmark artificiali passivi. Le funzionalità sono state testate in ambienti semplici e controllati: il prototipo di carrozzina dalle funzionalità estese realizzato è risultato ben progettato e corrispondente agli obiettivi proposti.

Ringraziamenti

Innanzitutto vorrei ringraziare il Prof. Matteo Matteucci per avermi dato la possibilità di lavorare a questo progetto. Insieme a lui desidero ringraziare l'Ing. Davide Migliore e l'Ing. Giulio Fontana che hanno contribuito a coordinare i lavori, proponendo soluzioni sempre nuove per affrontare i problemi incontrati. Inoltre vorrei ringraziare il Prof. Andrea Bonarini per avermi fatto conoscere l'AirLab e le possibilità di progetto offerte e per essersi sempre interessato allo sviluppo del lavoro e ai risultati che mano a mano venivano raggiunti. Grazie per il clima cordiale e fraterno ma allo stesso tempo costruttivo e professionale nel quale ho potuto lavorare, mi è stato di grande aiuto; grazie per la fiducia gratuitamente riposta in me e nelle mie capacità.

Con la conclusione di questa tesi mi considero come giunto all'arrivo di una tappa, ma ecco che subito un nuovo percorso si apre, ancora tutto da scoprire. Per il sostegno che sempre mi hanno dato e che anche in futuro so che mi daranno, per la pazienza con cui mi accompagnano e per l'amore che gratuitamente mi hanno donato e mi insegnano a donare voglio ringraziare i miei genitori, Mario e Angela. Grazie anche a mia sorella Cristina e a tutti gli altri parenti che mi hanno sempre mostrato il loro affetto.

Desidero ringraziare anche tutti coloro che hanno condiviso e condivideranno più o meno intensamente con me un tratto del mio cammino, siete tutti importanti e ognuno di voi mi ha dato molto, contribuendo ad arricchire e riempire ogni giorno il tesoro di esperienze che conservo gelosamente e che giorno dopo giorno acquista sempre più peso e valore. Spazio per citare tutti non c'è, ma desidero ricordare gli amici dell'oratorio e i don che ci accompagnano, con i quali è bello condividere sia i momenti spensierati che le riflessioni e le preghiere. Agli amici più stretti va un ringraziamento particolare per avermi sopportato in questi ultimi periodi in cui ero un po' "assente". Grazie ai ragazzi del gruppo 18-19enni a cui faccio catechismo, forse non lo sapete, ma ricevo da voi molto di più di quello che vi do. Grazie a tutti gli educatori dei gruppi di catechismo con cui ho collaborato e col-

laboro, ognuno di voi è per me esempio e occasione di confronto. Grazie ai componenti della Banda, perchè la musica è un collante gioioso che esprime molte più emozioni delle parole. Grazie ai compagni di corso di questi ultimi anni, alcune strade si sono divise, ma la serenità con cui ci siamo aiutati e confrontati mi è d'esempio per la vita. Grazie ai compagni di viaggio negli interminabili tragitti delle Ferrovie Nord, è bello sapere di trovare sempre (o quasi sempre) qualcuno con cui passare il tempo. Grazie a tutti gli incontri casuali della vita di tutti i giorni, forse così casuali in fondo non sono; grazie a tutte le persone (studenti, dottorandi, professori) che sono passate dall'AirLab in questi mesi contribuendo a creare un clima caloroso ed accogliente (al punto che qualcuno lo definisce il "Centro Sociale AirLab"), in cui è facile sentirsi a proprio agio, trovare collaborazione, ridere e scherzare senza però perdere di vista gli obiettivi "professionali" che ognuno ha.

Si potrebbero scrivere tante altre cose, ricordare episodi e avvenimenti, ridere e scherzare sui fatti accaduti, ma forse è meglio farlo di persona, dove oltre che con le parole si può comunicare con gli sguardi, con i gesti, con le emozioni e i sentimenti. Per questo lascio ora lo spazio alla parte "tecnica" di questo lavoro, sperando di avere l'occasione di poter esprimere nuovamente il mio ringraziamento non solo con le parole scritte qui.

Indice

Capitolo 1

Introduzione

Il presente lavoro si colloca nell'ambito della robotica mobile, disciplina che si occupa di studiare e progettare veicoli autonomi dotati di funzionalità "intelligenti", ovvero di caratteristiche che li rendono in grado di operare senza bisogno d'intervento umano o riducendo i comandi che l'uomo deve impartire. In particolare si sono applicate le tecniche della robotica mobile allo sviluppo di una carrozzina d'ausilio ai disabili motori, estendendo le funzionalità di una carrozzina elettrica al fine di studiare come avvicinarla alle esigenze degli utenti che trovano difficoltà nell'utilizzare le normali carrozzine esistenti.

Lo scopo della presente tesi è stato progettare e realizzare una carrozzina elettrica che rendesse disponibili funzionalità aggiuntive rispetto a quelle offerte dai prodotti presenti sul mercato, partendo proprio da una carrozzina commerciale a cui apportare modifiche. In particolare si è scelto di estendere le capacità della carrozzina con funzionalità di guida semiautonoma (o assistenza alla guida) e di guida autonoma. L'assistenza alla guida è una funzionalità che permette di muovere la carrozzina in sicurezza, ad esempio, evitando collisioni, aggirando ostacoli e aiutando l'utente in alcune manovre. La guida autonoma permette alla carrozzina di muoversi automaticamente verso una destinazione specificata in un ambiente noto, pianificando e seguendo un percorso adeguato.

Le motivazioni di questo lavoro nascono dal desiderio di superare le forti limitazioni di usabilità che le carrozzine elettriche attualmente in commercio presentano nei confronti degli utenti affetti da specifici problemi motori. Alcuni disturbi (quali la spasticità) limitano infatti fortemente la capacità di comandare una carrozzina elettrica tradizionale, e possono dunque precludere ogni possibilità di movimento autonomo a chi ne è affetto, influendo sullo stato di benessere fisico e psicologico delle persone, che dipende fortemente

dalla possibilità di essere autonomi negli spostamenti e nei movimenti, come illustrato in [42] [46] [20]. Nel corso degli anni sono stati studiati, progettati e commercializzati numerosi ausili che possono aiutare le persone con problematiche motorie a conseguire uno stato di indipendenza e una condizione di autonomia. Tra di essi, l'ausilio che si è dimostrato maggiormente utile alla categoria di persone afflitta da problemi motori è la carrozzina, sia essa a comando manuale o motorizzata. In particolare, la carrozzina elettrica assume un ruolo fondamentale nell'assicurare la mobilità a coloro che sono affetti da disabilità motorie.

Purtroppo, alcune categorie di disabili motori non sono in grado di utilizzare correttamente e agevolmente i dispositivi di comando presenti sulle carrozzina elettriche attualmente in commercio; il presente lavoro vuole contribuire al superamento di questa limitazione. La ricerca scientifica in ambito robotico si è spesso interessata alle problematiche di questa categoria di persone, cercando di sviluppare carrozzine con funzionalità tipiche dei robot mobili. In [38] [42] [54] sono illustrate le caratteristiche salienti di decine di carrozzine dalle funzionalità estese progettate e realizzate dagli anni novanta fino a oggi. Molti di questi prototipi sono dotati di funzionalità evolute e interessanti, tra cui evitamento ostacoli e guida autonoma, come, ad esempio, nei lavori descritti in [29] [11] [37] [7]; altri rivolgono la loro attenzione principalmente all'estensione delle modalità con cui è possibile comandare la carrozzina, utilizzando, ad esempio, sistemi di riconoscimento vocale [28] o rilevazione dei movimenti della testa [23] [32].

Molti di questi rappresentano tuttavia vere e proprie piattaforme robotiche mobili adattate al trasporto di un utente. Essi assumono dunque il ruolo di utili prototipi da laboratorio, ma difficilmente sarebbe possibile immaginarne un uso pratico in situazioni reali, a causa di vistosi limiti in termini di praticità e costo. Ad esempio, è facile immaginare che quando l'accesso al sedile di guida della carrozzina è reso difficoltoso dalla presenza di sensori ingombranti o da altri dispositivi, l'uso della stessa risulta problematico. Inoltre in molti progetti sono stati utilizzati sensori e componenti molto costosi, impedendo di fatto un possibile sviluppo di tali progetti verso prodotti effettivamente commercializzabili. In questo lavoro di tesi si è modificata una carrozzina elettrica commerciale applicando le tecniche della robotica mobile per realizzare un prototipo in grado di offrire funzionalità "intelligenti", ma dotato di caratteristiche di usabilità e costo comparabili con quelle di una carrozzina elettrica tradizionale.

Oggetto di questo lavoro è stata la progettazione e realizzazione di un circuito che, collegato alla scheda di controllo interna al joystick di comando, comunica al computer la posizione della leva del joystick e permette di con-

trollare il movimento della carrozzina. La carrozzina è stata poi equipaggiata con un computer a basso costo alimentato dalle batterie della carrozzina stessa. Tale computer, caratterizzato da dimensioni molto ridotte e bassi consumi elettrici è stato utilizzato per l'esecuzione del software di controllo basato su logica fuzzy. Questo software è stato realizzato secondo il principio della *composizione gerarchica informata* (*HIC*) [3] ed è basato su una architettura multiagente realizzata grazie a un framework apposito (DCDT, [33]). Il software risulta estremamente modulare e quindi facilmente modificabile ed estendibile.

Gli elementi hardware e software dedicati all'implementazione delle funzionalità estese della carrozzina elettrica non precludono nè modificano la possibilità di guida manuale della stessa, e dunque appaiono del tutto trasparenti all'utente che non desideri servirsene. Essi consentono inoltre il controllo della carrozzina tramite un dispositivo di comando alternativo, costituito da un joypad senza fili, che permette di controllare la carrozzina anche a distanza: può ad esempio risultare utile per richiamare la carrozzina da una posizione di parcheggio verso il letto dell'utente, per consentire la salita a bordo di quest'ultimo.

Successivamente si è implementata la funzionalità di assistenza alla guida che aiuta l'utente in situazioni critiche e pericolose, evitando collisioni e assistendo l'utente in manovre come l'aggiramento degli ostacoli o il movimento parallelo al profilo di un muro. Per fornire al calcolatore le informazioni sensoriali necessarie per l'esecuzione di tali compiti, sono stati installati a bordo della carrozzina due scanner laser. Un punto di forza del sistema realizzato risiede nell'autonomia con cui la carrozzina determina quale sia l'azione corretta da intraprendere. Ad esempio, il sistema di controllo sceglie automaticamente se arrestarsi in presenza di un ostacolo o se modificare la traiettoria seguita in modo da evitarlo, basandosi sulle informazioni rilevate riguardo all'ostacolo stesso.

Come ultima estensione si è progettato e sviluppato un sistema di guida autonoma, che permette di pianificare ed eseguire percorsi in ambienti conosciuti dal sistema, ovvero di cui sia disponibile una mappa metrica bidimensionale e in cui è possibile conoscere in tempo reale la posizione della carrozzina. I percorsi pianificati sono costituiti da una lista di via point che vengono inseguiti in successione.

Per supportare la guida autonoma, dopo aver analizzato alcuni sistemi di localizzazione indoor commerciali ([48] e [19]) e averli valutati come poco vantaggiosi a causa del costo eccessivo, si è progettato e sviluppato un sistema di localizzazione per ambienti indoor basato su landmark artificiali passivi rilevati con l'uso di tecniche di visione artificiale. In particolare si

sono sfruttati i sistemi di *fiducial marker*, noti sia in applicazioni robotiche [15] sia in applicazioni di realtà aumentata [2] [17] per costruire un sistema di localizzazione che permette di conoscere la posizione e l'orientamento della carrozzina in tre gradi di libertà rispetto a un sistema di riferimento assoluto utilizzando marker posizionati sul soffitto dell'ambiente. Il sistema di localizzazione realizzato si è rivelato robusto e affidabile, oltre che molto economico e facilmente scalabile su vaste zone.

Durante il movimento autonomo della carrozzina il sistema di riconoscimento ed evitamento degli ostacoli rimane attivo: esso collabora con il sistema di guida autonoma per garantire non solo il raggiungimento dell'obiettivo, ma anche una navigazione sicura in ambiente.

Il prototipo realizzato ha mostrato che la realizzazione di una carrozzina elettrica dotata di funzionalità avanzate e "intelligenti" è un obiettivo non lontano e che le scelte progettuali ed implementative effettuate sono valide: nella sua versione attuale (suscettibile di miglioramenti e già predisposta ad essi) la carrozzina è infatti in grado di evitare collisioni, di assistere l'utente nell'aggiramento degli ostacoli e di compiere semplici percorsi con movimenti autonomi in ambienti controllati.

Grazie al presente lavoro si sono gettate solide basi di sviluppo sulle quali è possibile intraprendere azioni di affinamento ed estensione, sia considerando la possibilità di aggiungere ulteriori sensori , sia perfezionando le funzionalità già presenti al fine di creare un prototipo sufficientemente evoluto da poter essere sottoposto al giudizio di utenti disabili in ambienti reali. Altri sviluppi potranno riguardare l'integrazione di sistemi di comando basati sul riconoscimento vocale, sul tracciamento della motilità oculare o su interfacce Brain-Computer Interface (BCI) che permetterebbero a un numero ancora maggiore di utenti disabili di aumentare le loro capacità e possibilità di muoversi autonomamente.

La presente tesi è piuttosto corposa per assolvere all'esigenza di documentare dettagliatamente le scelte progettuali e realizzative e permettere di comprendere nei particolari il funzionamento del sistema realizzato. Infatti il prototipo realizzato è stato concepito in modo da poter fungere da base per successivi ampliamenti e perfezionamenti, e dunque una completa documentazione delle sue caratteristiche é particolarmente necessaria. La struttura con cui è organizzata questa tesi è la seguente:

- Nel Capitolo 2 viene illustrata l'importanza degli ausili alla mobilità e vengono presentate le caratteristiche salienti di alcune soluzioni proposte in letteratura per estendere la funzionalità delle carrozzine elettriche.

- Nel Capitolo 3 viene presentata una panoramica riguardante i sensori e le tecniche della robotica mobile che meglio si adattano all'estensione delle funzionalità di una carrozzina elettrica.

- Nel Capitolo 4 viene descritto il progetto della la carrozzina elettrica oggetto di questo lavoro, identificando gli obiettivi da raggiungere e le componenti utilizzate per il loro perseguimento.

- Nel Capitolo 5 viene presentato il sistema di localizzazione per ambienti indoor basato su tecniche di visione artificiale sviluppato in questa tesi.

- Nel Capitolo 6 viene illustrata la struttura del software di controllo realizzato per la carrozzina elettrica, specificandone il comportamento e l'architettura.

- Nel Capitolo 7 sono analizzati e commentati i risultati sprimentali ottenuti, sia dal punto di vista della realizzazione del prototipo che dal punto di vista del comportamento del controllore nello svolgere i compiti di assistenza alla guida e di guida autonoma. tracciando anche alcune possibili linee di estensione del sistema.

- Nel Capitolo 8 sono tracciate le conclusioni del lavoro svolto e ne vengono proposte le linee di sviluppo, sia in termini di miglioramento delle funzionalità già realizzate che di identificazione di nuove funzionalità aggiuntive.

Capitolo 2

Ausili alla mobilità e tecnologia

In questo capitolo si affronta il problema degli ausili alla mobilità necessari ad alcune categorie di persone. Vengono presentate le principali carrozzine elettriche intelligenti realizzate in ambito di ricerca, le linee in cui le funzionalità di questi ausili sono state estese e le loro caratteristiche salienti.

2.1 Ausili alla mobilità

Per gli esseri umani, la possibilità di muoversi autonomamente e in modo agevole è essenziale per il conseguimento di uno stato di benessere fisico e psicologico, come evidenziato da numerosi studi [42]. Ad esempio, nello sviluppo cognitivo dei bambini il movimento favorisce l'apprendimento dei concetti di relazione spaziale, senza i quali i normali processi di apprendimento e sviluppo delle facoltà intellettive possono risultare rallentati e limitati [46]. Negli anziani, invece, la difficoltà crescente di compiere movimenti induce un processo involutivo che peggiora le condizioni psico-fisiche della persona.

Dal punto di vista fisico le maggiori difficoltà di movimento limitano il numero e la durata delle attività che una persona riesce a svolgere, anche fino a indurla a rinunciare a espletare i bisogni fisiologici e non soddisfare le necessità nutrizionali, con conseguente peggioramento delle sua salute. Dal punto di vista psicologico l'impossibilità di muoversi agevolmente comporta difficoltà di socializzazione, accrescendo lo stato di solitudine e favorendo l'insorgere di stati d'ansia e depressione. In [20] si evidenzia che il 31% delle

persone con difficoltà motorie soffrono di crisi d'ansia e stati depressivi, contro il 4% delle persone senza difficoltà motorie.

L'uso di carrozzine manuali o motorizzate aumenta le possibilità di movimento e favorisce lo sviluppo di circoli virtuosi che portano le persone con impedimenti fisici a un sostanziale stato di benessere psico-fisico. Le carrozzine motorizzate si guidano solitamente con un joystick e sono rivolte soprattutto alle persone che non sono in grado di imprimere alle ruote la forza necessaria per muovere una carrozzina manuale o per le quali questo compito risulterebbe troppo gravoso.

Un discreto numero di persone che potrebbero usufruire dei benefici degli ausili alla mobilità non è però in grado di utilizzare al meglio le normali carrozzine presenti in commercio. Possiamo, ad esempio, individuare in questa categoria spastici, distonici e parkinsoniani e in generale le persone affette da disturbi che alterano la capacità di compiere movimenti in modo accurato o che causano movimenti involontari. Inoltre risulta rischioso affidare veicoli semoventi a ipovedenti e a coloro che oltre a problemi fisici presentano deficit mentali.

Al fine di aumentare il grado di autonomia di queste persone, alleggerendo inoltre il compito di chi li assiste, fin dai primi anni '80 sono state studiate in ambito accademico numerose soluzioni per approntare delle carrozzine con più funzionalità rispetto a quelle normalmente presenti sul mercato. Tali funzionalità non sono da considerarsi migliorative in senso assoluto, ma solo in relazione alla particolare categoria di persone che abbiamo considerato. Per le persone che riescono agevolmente a utilizzare in sicurezza le normali carrozzine elettriche, le funzionalità aggiuntive possono risultare oltre che superflue anche scomode.

2.2 Carrozzine elettriche e robotica mobile

Nel corso degli anni sono state progettate e realizzate numerose carrozzine elettriche con funzionalità estese. Le modifiche sono state apportate in diverse direzioni, con diversi scopi e con funzionalità offerte differenti. Nel progettare carrozzine con potenzialità superiori rispetto a quelle normalmente presenti sul mercato, è possibile seguire tre vie:

- Creare un robot mobile con un sedile che possa ospitare una persona (e.g. Véichule Autonome pour Handicapé Moteur (VAHM) [37] [7], Mister Ed [11]).
- Utilizzare una carrozzina motorizzata commerciale come base e cambiare i dispositivi elettronici di controllo dei motori con un apparato

progettato “ad hoc” (e.g. NavChair [28], Office wheelchair with high Maneuverability and Navigational Intelligence (Omni) [6], Mobility Aid for elderly and disabled people (MAid) [36], SENARIO [21]).

- Utilizzare una carrozzina motorizzata commerciale come base creando dei dispositivi che si interfacciano con i bus proprietari (e.g. Smart Wheelchair Assistance Module (SPAM) [43], Hephaestus [44], Tin-Man [34], Siamo [32]).

2.2.1 Estensione delle funzionalità delle carrozzine elettriche

L’aggiunta di un sedile a un robot mobile è stata una delle prime strade percorse, ma risulta dispendiosa e richiede un notevole sforzo per la progettazione e realizzazione della struttura meccanica e della motorizzazione.

L’uso di una carrozzina commerciale come base permette di mantenere tutte le funzionalità che non sono legate direttamente al movimento della carrozzina, ma che la rendono differente da una “sedia a motore”. Le carrozzine elettriche presenti sul mercato sono studiate per servire al meglio la persona disabile che le occuperà. Il sedile è oggetto di studi di ergonomia, in quanto deve garantire comfort anche per utilizzi prolungati nel tempo. Su alcuni modelli il sedile è a sua volta dotato di attuatori che permettono, con l’uso dello stesso joystick di guida, di adattarlo alla postura della persona che lo occupa senza necessità di intervento manuale. L’uso di una base commerciale è la strada più praticata e le modifiche riguardano la sostituzione del sistema di controllo dei motori con uno creato “ad hoc”. Il joystick originale della carrozzina viene tipicamente estromesso dal controllo nella guida, oppure il suo utilizzo è mutuamente esclusivo con l’uso delle funzionalità aggiuntive previste, che saranno accessibili solo con l’uso di un diverso dispositivo di comando.

La progettazione di dispositivi che si interfacciano con i bus dati già presenti sulla carrozzina permette un grado elevato di integrazione con le funzionalità di base disponibili e permette di realizzare un sistema che è compatibile con tutte le carrozzine basate sullo stesso tipo di bus dati. Le difficoltà sono legate alla conoscenza delle specifiche elettriche del bus dati, delle informazioni che si trasmettono su di esso e del protocollo che governa lo scambio dei messaggi. Spesso i bus dati sono proprietari e le specifiche non accessibili, con conseguente necessità di effettuare operazioni di “reverse-engeenering” per ricostruire le informazioni necessarie ad interfacciare apparati esterni con il bus.

Intercettando i segnali analogici prodotti dalla leva del joystick prima che essi siano campionati e codificati per essere immessi sul bus dati è possibile

interfacciarsi al bus dati in modo indiretto. Lo studio di questi segnali e la produzione di circuiti d'interfaccia è solitamente più semplice dell'analisi del protocollo del bus dati, anche se lega strettamente il circuito di interfaccia al particolare modello di joystick utilizzato.

2.2.2 La carrozzina elettrica come robot mobile

Lo scopo primario della costruzione di una carrozzina con funzionalità aumentate sta nel fornire ausili alla mobilità più efficienti e disponibili ad un pubblico più vasto per diminuire le barriere e superare gli ostacoli che si presentano nel loro percorso di integrazione sociale. Dal punto di vista prettamente scientifico, però, una carrozzina elettrica risulta essere un buon veicolo adatto ad applicazioni e studi di robotica mobile. Le caratteristiche principali di una carrozzina elettrica vista come robot mobile sono:

- È progettata per muoversi in ambienti reali, interni o esterni.
- Sviluppa velocità di spostamento sufficientemente elevate.
- È agile nei percorsi tortuosi e permette manovre agevoli in spazi stretti.
- È dotata di una struttura robusta sulla quale si possono montare dispositivi e sensori.
- L'alimentazione a batterie è ben dimensionata e può essere sfruttata per alimentare gli accessori che si vogliono aggiungere alla carrozzina.

2.2.3 Funzionalità offerte

Nel panorama delle carrozzina elettriche con funzionalità estese si possono individuare due grosse categorie: carrozzine semiautonome e autonome. In questo lavoro non saranno trattate le possibili estensioni delle carrozzina dal punto di vista meccanico, come, ad esempio, la costruzione di apparati supplementari che diano alla carrozzina la possibilità di salire le scale o l'installazione di sospensioni attive che garantiscano stabilità anche su terreni accidentati, analizzate in [13].

Le carrozzine semiautonome hanno come obiettivo principale quello di assistere l'utente nella guida, implementando funzioni come l'evitamento ostacoli, il passaggio attraverso porte o l'avvicinamento a oggetti. Il percorso da seguire è completamente a discrezione dell'utente, che lo comunica attraverso il dispositivo di comando (o i dispositivi) disponibile sulla carrozzina.

Le carrozzine autonome offrono funzionalità molto simili a quelle solitamente implementate nei robot autonomi. L'utente è chiamato a specificare una destinazione e il sistema di controllo si preoccupa di pianificare il percorso ed eseguirlo. Per svolgere questo tipo di attività è necessario conoscere e rappresentare la mappa del luogo in cui si opera o disporre di un sistema di riconoscimento dei luoghi raggiunti, ad esempio, basandosi su landmark attivi o passivi. Nell'eseguire il percorso è auspicabile che il sistema di evitamento ostacoli sia attivo. Considerando la difficoltà di integrare le informazioni conosciute sull'ambiente con quelle che si acquisiscono mano a mano (come gli ostacoli imprevisti), è raro che nelle funzionalità offerte sia prevista la possibilità di ripianificare percorsi integrando le nuove informazioni sull'ambiente.

Come è facile immaginare, è possibile far interagire la modalità semiautmatica con quella automatica, lasciando all'utente la scelta o inserendo dei metodi automatici di cambio modalità.

2.2.4 Dispositivi di comando

I dispositivi di comando tradizionali di una carrozzina elettrica sono il joystick e una serie di pulsanti direzionali. A questi sono stati aggiunti in ambito di ricerca altre interfacce di comando come

- Comando vocale, grazie alla disponibilità di hardware e software sul mercato (e.g. SENARIO [21]).
- Tracciamento dell'attività oculare tramite l'uso di elettrodi o telecamere (e.g. Siamo [32]).
- Riconoscimento dell'orientamento e posizione della testa (e.g Madarasz wheelchair [29] e Intelligent Wheelchair System dell'università di Osaka [23]).

Si intuisce facilmente che questi metodi di comando non sono precisi e rapidi come quello attuabile tramite un joystick. Ad esempio, con un'interfaccia di riconoscimento vocale, sarà possibile eseguire comandi a intervalli di tempo dell'ordine dei secondi; inoltre, bisogna porre molta attenzione ai casi in cui i comandi non vengono riconosciuti correttamente o vengono del tutto ignorati.

I metodi e i dispositivi di comando alternativi devono dunque essere accoppiati con sistemi di identificazione degli ostacoli che garantiscano la sicurezza dell'utente dalle conseguenze di comandi non riconosciuti o interpretati male (e.g. NavChair [28]).

2.2.5 Sensori per il rilevamento di ostacoli

Per poter sviluppare funzionalità di supporto alla sicurezza del movimento della carrozzina, come evitamento ostacoli o approccio sicuro a oggetti, è necessario introdurre sensori di distanza e prossimità. I più utilizzati ed economici sono i dispositivi di misura basati su ultrasuoni (*Sonar*) e su luce infrarossa (*IR*).

Il sonar permette di rilevare oggetti che riflettono l'onda sonora emessa e, calcolando il "tempo di volo", stima la distanza dell'oggetto incontrato. I problemi principali si riscontrano con i materiali fonoassorbenti e con superfici che non riflettono nella direzione di provenienza il suono, risultando così invisibili al sensore. I sonar sono solitamente utilizzati in numero elevato e montati in modo da coprire l'intera area di interesse, anche se questa configurazione può provocare false rilevazioni dovute alla ricezione da parte di un sensore dell'eco di un'onda generata da un altro sensore (fenomeno del *crosstalk*, rif. Sezione 3.1.2).

I sensori di distanza a infrarossi emettono luce al posto di suoni e hanno dunque difficoltà nel rilevare superfici che assorbono la luce infrarossa. Anche le superfici trasparenti o rifrattive sono ingannevoli per un sensore IR.

La categoria di sensori più evoluta e precisa è quella degli scanner laser (*Laser Range Finder, LRF*). Uno scanner laser è costituito da un sensore di misura della distanza montato su una struttura rotante. Il sensore laser emette un fascio di luce coerente e misura la distanza della superficie che riflette il fascio emesso misurando il tempo trascorso tra emissione e ricezione. Dopo ogni rilevazione la struttura rotante compie un movimento che permette di cambiare l'orientamento del fascio laser emesso. In questo modo è possibile effettuare una scansione planare e conoscere le distanze rilevate con ogni orientamento del laser. L'angolo di scansione è solitamente di 180 °, ma varia in base ai modelli presenti sul mercato. Il costo di un sensore laser è elevato, ma, a differenza dei sensori precedentemente presentati, è sufficiente usare uno o due sensori laser per garantire la copertura della zona interessata intorno alla carrozzina (e.g. MAid [36] e SENARIO [21]). Per ulteriori dettagli sui sensori laser e per una panoramica su quelli presenti sul mercato si rimanda alla Sezione 3.1.

Un ultimo metodo di rilevamento ostacoli è dato dall'uso della visione artificiale. Questo comporta l'uso di telecamere e software di analisi di immagini. I costi sono contenuti rispetto a quelli dei sensori laser, ma le difficoltà maggiori sono legate ai metodi e algoritmi per il corretto riconoscimento degli ostacoli e alla rilevazione delle loro distanze. Data la forte

espansione del ramo della visione artificiale, è probabile che nei prossimi anni si ottengano buoni risultati con questa tecnologia, che al momento è confinata al riconoscimento di landmark per la localizzazione (e.g. MAid [36]) o al tracciamento dei movimenti della testa per il comando (e.g. Intelligent Wheelchair System [23])

2.2.6 Software di controllo

Per poter analizzare e utilizzare le informazioni prodotte dai sensori, integrandole con la volontà espressa dall'utente tramite il dispositivo di comando e controllare il movimento della carrozzina è prassi comune installare a bordo del veicolo un computer che svolga la funzione di centralizzatore delle informazioni e da produttore del controllo.

Il software di controllo può essere sviluppato seguendo varie architetture studiate in letteratura. Tra queste troviamo:

- Reti neurali per la riproduzione di percorsi precedentemente appresi (e.g. Smart Wheelchair della Kanazawa University [40]).
- Regole di comportamento che governano le decisioni in base agli input (e.g. SPAM [43]).
- Architettura *Subsumption*, dove i comportamenti "primitivi" sono combinati per produrre comportamenti più complessi e di alto livello (e.g. Mister Ed [11]).

Le architetture sono solitamente strutturate a livelli. Ai livelli più bassi sono affidati i compiti puramente reattivi, mentre quelli più alti sono coinvolti in processi decisionali di ragionamento automatico. Per ulteriori dettagli si rimanda alla Sezione 3.4.

2.2.7 Localizzazione e posizionamento

Per garantire autonomia nella pianificazione dei percorsi, è necessario conoscere l'ambiente in cui la carrozzina si muove. Per fare questo è possibile utilizzare mappe metriche, che codificano le distanze e la posizione di ostacoli e muri, oppure mappe topologiche, che codificano le connessioni tra punti salienti. Un'altra possibilità è quella di preparare dei percorsi nell'ambiente con l'uso di tracce colorate o magnetiche che siano riconoscibili da sistemi di visione o sensori.

Qualora non si voglia legare il movimento a percorsi specifici come quelli offerti dall'uso di bande magnetiche o colorate, è necessario che la carrozzina,

o più in generale un robot mobile, sia equipaggiato con un sistema di localizzazione che gli permetta di conoscere la sua posizione nella mappa. Per fare ciò è possibile utilizzare dei landmark naturali o artificiali riconoscibili con sistemi di visione artificiale. I landmark naturali sono rappresentati da forme o caratteristiche salienti dell'ambiente distinguibili, come, ad esempio, gli spigoli su un soffitto o oggetti appesi alle pareti. Questi landmark si considerano naturali in quanto non sono posizionati appositamente, ma sono già parte dell'ambiente. L'uso di landmark naturali comporta più problemi di possibili ambiguità rispetto a quelli artificiali, che presentano solitamente caratteristiche che li rendono difficilmente confondibili. Una carrozzina autonoma che riconosce landmark naturali è CCPWNS [12] mentre l'uso di landmark artificiali è implementato, ad esempio, in Madarasz [29]

Un'uteriore possibilità per effettuare localizzazione è rappresentato dall'uso di "fari", che emettono segnali riconoscibili e distinguibili. Questo approccio è utilizzato ad esempio in MAid [36] e VAHM [37] [7].

Un sistema di posizionamento assoluto può essere utilizzato per garantire la stabilità dei sistemi di posizionamento relativo. È infatti noto che per effettuare la localizzazione basandosi su informazioni inerziali o odometriche è necessario disporre di un sistema di azzeramento dell'errore, che diverge al crescere dello spazio percorso.

2.3 Progetti esistenti

I primi prototipi di carrozzine elettriche con funzionalità estese risalgono ai primi anni '80. Nel corso degli anni ne sono state sviluppate molte con caratteristiche molto differenti per quanto riguarda il tipo di sensori utilizzati, le interfacce di comando per la guida e le funzionalità offerte. Di seguito sono brevemente elencati alcuni progetti e le loro peculiarità, una lista più completa si può trovare in [38], [42] e [54].

SMART ALEC (Stanford, USA, 1980-1990) è la prima carrozzina semiautonoma. È basata su una struttura commerciale modificata, equipaggiata con encoder sulle ruote e sonar per il rilevamento di ostacoli. Il sistema di guida è basato sul rilevamento della posizione della testa dell'utente, effettuato sempre con l'uso dei sonar. Le funzionalità offerte sono l'evitamento delle collisioni, il mantenimento di distanze prefissate da un muro, utile per lo spostamento nei corridoi, e inseguimento di obiettivi.

Madarasz wheelchair (Università dell'Arizona, USA, 1986) [29] è la prima carrozzina autonoma. È equipaggiata con sonar e sistemi di visione

artificiale per il riconoscimento di landmark artificiali. I movimenti da eseguire sono specificati in un linguaggio apposito.

Mister Ed (IBM, USA, 1990) [11] è un robot mobile con un sedile su di esso. Il software è basato su architettura subsumption con comportamenti primitivi come l'attraversamento di porte, mantenimento della distanza da un muro e inseguimento di un target.

VAHM (Università di Metz, Francia, 1992-2004) [37] [7], acronimo di Véichule Autonome pour Handicapé Moteur, è basato su un robot mobile con un sedile. L'architettura di controllo è a tre livelli e rende possibile la navigazione autonoma basata su mappa interna, evitamento ostacoli e mantenimento della distanza costante da un muro. La scelta della modalità operativa è lasciata all'utente. La mappa interna è multipla (metrica e topologica). Per la pianificazione del percorso da seguire utilizza fari a luce infrarossa. È stato realizzato un secondo prototipo, con le stesse funzionalità, basato su una carrozzina commerciale modificata.

NavChair (Università del Michigan, USA, 1993-2002) [28] si basa su una carrozzina commerciale con modifiche al sistema di controllo dei motori. Offre funzionalità di evitamento collisioni e un gruppo di comportamenti per compiti specifici, come il passaggio per porte o il mantenimento della distanza costante da un muro. I comandi possono essere impartiti grazie a un sistema di riconoscimento vocale.

TinMan (Kipr, USA, 1994-1999) [34] rappresenta una serie di prototipi basati su carrozzine elettriche. Il primo prototipo utilizzava un dispositivo meccanico per muovere la leva del joystick, i successivi sono stati sviluppati modificando il sistema elettronico di controllo dei motori di carrozzine commerciali. Le funzionalità offerte sono l'evitamento di collisioni e la navigazione autonoma.

CCPWNS (Università di Notre Dame, USA, 1994-2000) [12], acronimo di Computer Controlled Power Wheelchair Navigation System, permette di riprodurre percorsi precedentemente appresi dal sistema. Il sistema di visione identifica landmark. Non prevede nessun tipo di evitamento ostacoli.

SENARIO (Finlandia, 1995-1998) [21] è una carrozzina commerciale modificata. Permette l'interazione nella guida dell'utente con il sistema di evitamento ostacoli. La navigazione autonoma è gestita con

una mappa interna. La localizzazione è affidata a una rete neurale. L'architettura software è di tipo distribuito.

OMNI (Università di Hagen, Germania, 1995-1999) [6], acronimo di Office wheelchair with high Maneuverability and Navigational Intelligence, è una carrozzina elettrica omnidirezionale commerciale modificata. Le funzionalità sono organizzate in forma gerarchica: evitamento ostacoli semplice, modi operativi per compiti specifici (distanza costante da muri, attraversamento porte) e navigazione autonoma.

Rolland I e II (Università di Bremen, Germania, 1997-2002) [24] [25] [26] sono state oggetto di numerose evoluzioni e studi. Sono basate su una carrozzina commerciale (Meyra Genius 1.522) che dispone già di interfacce per il comando in velocità e la misura della velocità con encoder sulle ruote. Il primo prototipo prevedeva navigazione autonoma basata su landmark artificiali e odometria abbinata all'evitamento di collisioni con sonar, IR e bumpers. Le modalità operative erano numerose (distanza costante da un muro, attraversamento di porte) ed era possibile eseguire percorsi appresi.

Nel secondo prototipo sono presenti solo sonar, ma l'algoritmo di evitamento ostacoli è più sofisticato. I comportamenti base con cui insegnare le traiettorie sono la rotazione sul posto e il mantenimento della distanza costante da un muro. Nella modalità semiautonoma la carrozzina modifica la sua velocità in base alla presenza degli ostacoli. La Figura 2.1 mostra la carrozzina Rolland II e la cintura di sonar che avvolge l'intera struttura della carrozzina.

MAid (Germania, 1998-2003) [36], acronimo di Mobility Aid for elderly and disabled people, è una carrozzina commerciale modificata nella parte di controllo dei motori. Ha due modalità operative: Narrow–Area Navigation (NAN) e Wide–Area Navigation (WAN). Nella modalità NAN permette di navigare da un punto di partenza ad un goal, nella modalità WAN è in grado anche di identificare ed evitare oggetti mobili nell'ambiente. Successivamente è stata aggiunta la possibilità di inseguire oggetti in movimento.

Intelligent Wheelchair System (Osaka University, Giappone, 1998-2003) [23] interpreta i movimenti della testa con l'uso di una telecamera rivolta verso il sedile. Una seconda telecamera, rivolta verso l'esterno, permette di inseguire un obiettivo e di interpretare i comandi anche quando l'utente non è a bordo della carrozzina. La volontà dell'utente

viene confrontata con la possibilità di muoversi in una certa direzione, basandosi sia su una mappa topologica sia sulle rilevazioni dei sonar. La localizzazione è affidata a un sistema di integrazione dei dati odometrici. Grazie all'uso della telecamera esterna il sistema è in grado di riconoscere la presenza dei pedoni e muovere la carrozzina in modo da evitare le collisioni.

Hephaestus (TRAC Labs, USA, 1999-2002) [44] prevede comportamenti di evitamento ostacoli. È compatibile con più modelli di carrozzine anche di diverse case costruttrici e non richiede nessuna modifica al sistema di controllo dei motori.

Siamo (Università di Alcala, Spagna, 1999-2003) [32] è stata utilizzata con vari dispositivi di input: controllo vocale, riconoscimento di movimenti della testa, movimento oculare rilevato con elettrodi. Prevede un sistema di evitamento ostacoli con scanner laser e sensori a infrarossi in grado di riconoscere anche avvallamenti o pendii ripidi.

Smart Wheelchair (Università di Kanazawa, Giappone, 2000) [40] è in grado di localizzarsi utilizzando dei radiofari. Grazie alla conoscenza della posizione permette la navigazione automatica. Può apprendere e riprodurre percorsi grazie all'uso di reti neurali. Non implementa nessun sistema di identificazione e evitamento ostacoli.

SPAM (Sciences, USA, 2003-2004) [43], acronimo di Smart Wheelchair Assistance Module, è sviluppata come accessorio per motorizzare carrozzine manuali. È compatibile con numerosi modelli di carrozzine manuali. La funzionalità offerta è l'evitamento delle collisioni. L'architettura software è basata su regole di comportamento.

Rolland III (Università di Bremen, Germania, 2005) [31] ha due scanner laser posizionati al livello del terreno, uno in direzione frontale e uno rivolto all'indietro. Le ruote sono dotate di encoders per la conoscenza dell'odometria. Il sistema di visione si basa su una telecamera omnidirezionale per la ricerca di punti salienti nell'ambiente.

Figura 2.1: Carrozzine con funzionalità estese realizzate (in ordine da sinistra a destra e dall'alto in basso): TinMan I, TinMan II, Intelligent Wheelchair System, OMNI, NavChair, Rolland II

Capitolo 3

Panoramica sulla robotica mobile

In questo capitolo vengono presentati i sensori e le tecniche tipiche della robotica mobile che meglio si adattano all'estensione delle funzionalità di una carrozzina elettrica. Tra di essi è possibile citare sensori per il rilevamento degli ostacoli, tecniche di localizzazione per ambienti indoor, telecamere e visione artificiale, pianificatori di percorsi e architetture software per la realizzazione di controllori robotici. Sviluppare in questa sede un'analisi completa dello stato dell'arte della robotica mobile sarebbe alquanto complesso e lungo, di conseguenza in questa sezione sono descritte solo le tecniche e i sensori che sono risultati utili o che hanno ispirato lo sviluppo del presente lavoro.

3.1 Sensori per il rilevamento di ostacoli

Nella Sezione 2.2.5 sono stati introdotti i principali tipi di sensori utilizzati per il riconoscimento degli ostacoli presenti nell'ambiente. In questa sezione si introducono le tecniche utilizzate per la misura della distanza e vengono presentati gli *scanner laser*, di cui viene anche fornita una panoramica dei prodotti disponibili sul mercato.

3.1.1 Misurazione della distanza

Per misurare distanze tra un punto e una superficie si possono usare tre approcci base, come descritto in [5]:

- Misura del tempo di volo (*Time-of-flight*, TOF) tra l'istante di emis-

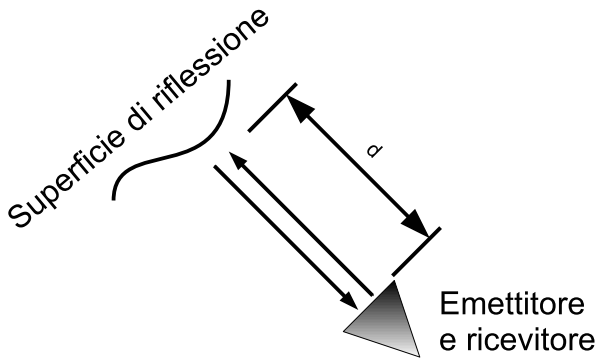


Figura 3.1: Principio di funzionamento di un sistema di misura della distanza a tempo di volo

sione di un impulso di energia e il suo ritorno alla sorgente dopo aver subito la riflessione da parte di una superficie.

- Misura dello sfasamento (*phase-shift measurement* o *phase-detection*) tra un'onda continua emessa e quella riflessa.
- Sensori basati su radar a modulazione di frequenza, ovvero sulla rilevazione del cambio di frequenza tra l'onda emessa e quella restituita.

3.1.2 Misura di distanza a tempo di volo

Molti sensori attuali usano il metodo della misura del tempo di volo per misurare la distanza di un oggetto che produce un eco del segnale emesso (Figura 3.1). Il tempo che intercorre tra l'emissione del segnale e il suo ritorno è pari al doppio dello spazio diviso per la velocità dell'onda, da cui si ricava facilmente la distanza dalla superficie:

$$t = \frac{2d}{v} \qquad d = \frac{tv}{2} \tag{3.1}$$

L'impulso emesso può essere un ultrasuono, un'onda radio o un impulso luminoso. A seconda del tipo di segnale emesso, la velocità di propagazione varia notevolmente: il suono in aria si propaga a circa 0.3 m/ms, la luce a 0.3 m/ns. La complessità dei sensori e dei circuiti elettrici di controllo e conteggio del tempo dipende notevolmente dal tipo di segnale utilizzato. Con alcuni tipi di segnale, in particolare con i laser, è possibile garantire che il percorso seguito dopo la riflessione sia identico a quello di andata, ovvero non sia influnzato dalle caratteristiche geometriche della superficie. È quindi possibile porre il generatore del segnale e il ricevitore molto vicini

o sovrapposti, evitando così di dover inserire nella logica di controllo un sistema di triangolazione per il calcolo delle distanze. Questa particolarità garantisce misure più accurate a distanze elevate, dove i sistemi basati sulla triangolazione sono meno accurati.

Cause d'errore

Tra le cause d'errore più comuni che possono inficiare la bontà della misura di un sistema a tempo di volo si evidenziano:

- Variazione della velocità di propagazione dell'onda.
- Errori nella rilevazione dell'istante di tempo di arrivo dell'eco.
- Errori dovuti al sistema di conteggio del tempo.
- Interazione tra l'onda emessa e la superficie riflettente.

La velocità di propagazione della luce è considerabile costante, mentre quella del suono è fortemente influenzata dalla temperatura dell'ambiente e, in modo meno marcato, dall'umidità.

Gli errori nella rilevazione dell'istante di tempo in cui l'onda riflessa arriva al rilevatore sono dovuti alla degradazione dei fronti del segnale impulsivo. È dunque difficile stabilire la soglia che identifica l'effettiva ricezione del segnale. Se si usa un meccanismo a soglia fissa si identificano in maniera errata le superfici più riflettenti, in quando il loro segnale giunge al rilevatore non degradato e viene quindi calcolata una distanza inferiore a quella reale. Per ovviare a questo problema si utilizzano soglie variabili in base al tempo che è intercorso dall'invio del segnale.

Il conteggio del tempo in un sistema di misura di distanza basato su TOF è critico soprattutto quando il segnale utilizzato è luminoso o elettromagnetico. I circuiti di controllo e misura del tempo devono essere in grado di misurare intervalli inferiori ai nanosecondi, soprattutto se si desiderano misure precise su distanze brevi.

Quando un'onda, sia essa acustica o elettromagnetica, incontra una superficie, una parte di essa viene riflessa, una parte assorbita dal materiale e in alcuni casi una parte attraversa la superficie stessa. In base al tipo di materiale e all'angolo di incidenza l'onda riflessa può essere molto degradata, al punto da non essere riconosciuta dal sensore. In questo caso la distanza misurata risulta erroneamente infinita, ovvero non viene evidenziata la presenza di alcun oggetto sulla traiettoria seguita dall'onda.

Un fenomeno che si verifica soprattutto con i sonar e le onde acustiche è il *crosstalk* o *diafonia* (Figura 3.2). Esso avviene quando un'onda emessa

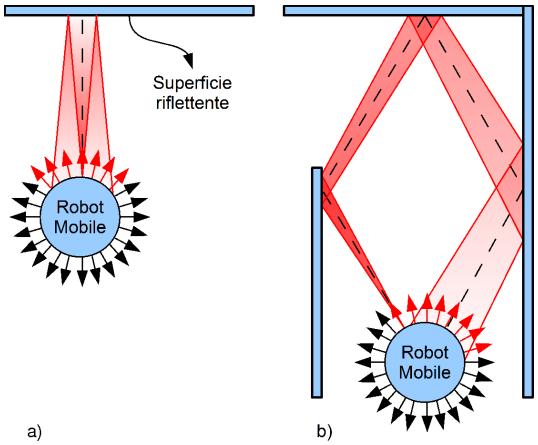


Figura 3.2: Fenomeno del crosstalk nella misura delle distanze con i sistemi a tempo di volo, (a) crosstalk diretto e (b) crosstalk indiretto

viene riflessa in una direzione tale per cui viene ricevuta da un altro sensore. Il metodo più comune per evitare questo tipo di errore consiste nel ripetere la misura e nel fornire come distanza rilevata un valore sintetico, nel caso più banale la media, delle rilevazioni.

3.1.3 Misuratori di distanza a tempo di volo laser

I sistemi di misura della distanza laser a tempo di volo, detti anche *laser radar* o *lidar*, fecero la loro comparsa nel 1970 presso i Jet Propulsion Laboratory di Pasadena (California). La precisione dei primi prototipi era di qualche centimetro per misure comprese tra 1 e 5 metri. Per permettere di misurare le distanze rilevate in più direzioni sono stati introdotti i *laser scanner*. Il principio di funzionamento è molto semplice e consiste nel montare l'emettitore laser e il ricevitore su un asse ruotante. In questo modo è possibile effettuare la scansione delle distanze su un piano, solitamente con una limitazione sull'escursione angolare del motore.

I sensori presenti sul mercato sono numerosi e con caratteristiche differenti. Per poter realizzare un confronto significativo tra i prodotti di queste case sarebbe necessario impostare degli esperimenti standard e valutare le prestazioni di ogni sensore. In questo contesto è sufficiente paragonare le caratteristiche tecniche indicate dal costruttore, allo scopo di fornire una panoramica dei sistemi disponibili. Gli scanner laser presi in considerazione sono prodotti da SICK e Siemens (Germania) e Hokuyo (Giappone). Per

Figura 3.3: Laser scanner commerciali (in ordine da sinistra a destra e dall'alto in basso): Sick LMS200, LMS211, LMS221, LMS291, LMS400, PeCo, Oem e Pds; Siemens LS4; Hokuyo UGR-04LX

ulteriori informazioni si rimanda a [41] e ai siti web di SICK[1] Hokuyo[2] e Siemens[3]

I dati ritenuti significativi per il confronto sono:

- Angolo di apertura della scansione.
- Risoluzione angolare.
- Minima distanza rilevabile.
- Range tipico di lavoro.
- Massima distanza rilevabile.

[1] http://www.sick.com
[2] http://www.hokuyo-aut.jp
[3] http://www.siemens.com

- Risoluzione.
- Accuratezza.
- Frequenza di scansione.
- Peso.
- Dimensioni.
- Caratteristiche elettriche.

Nelle Tabelle 3.1 e 3.2 sono riportate le caratteristiche dei modelli analizzati. SICK è la casa più attiva nel mercato degli scanner laser. I suoi prodotti di maggior successo sono LMS 200 e LMS 291, utilizzati in molti progetti di robotica mobile [4]. Siemens si rivolge soprattutto ad un mercato industriale e il suo scanner laser (LS4) è alla base di sistemi di automazione e sicurezza. Hokuyo produce uno scanner laser decisamente differente dai concorrenti: è il più piccolo e leggero (ingombro di 50 x 50 x 70 mm, peso 160 grammi), offre un buon range angolare di scansione con una risoluzione angolare discreta anche se non ha le stesse prestazioni, valutate in termini di distanza massima rilevabile e frequenza di scansione, della famiglia dei SICK.

Nella Tabella 3.3 sono riportate le caratteristiche della luce emessa e la classe di sicurezza dagli scanner laser. Il significato delle classsi di sicurezza è il seguente:

Classe 1: Intrinsecamente sicuri, si possono osservare ad occhio nudo senza danno.

Classe 2: Non intrinsecamente sicuri ma non creano notevoli problemi, in quanto i normali tempi di reazione dell'occhio umano fanno si che la retina non rimanga esposta per un tempo sufficiente da risultare pericoloso.

Classe 3A: Il laser non deve essere osservato.

Classe 3B: La visione diretta del fascio non è sicura.

Classe 4: Comprende i laser più potenti e pericolosi, come quelli utilizzati nel taglio dei metalli.

[4] Sul sito http://www.sick.com sono mostrate molte vetture partecipanti al Darpa Grand Challenge equipaggiate con scanner laser SICK

Modello		Range angolare (Risoluzione)	Distanza minima,	tipica,	massima	Risoluzione	Accuratezza
SICK	LMS 200	180° (0.25°, 0.5°, 1°)	0 m	10 m	30 m	10 mm	±15 mm
	LMS 211	100° (0.25°, 0.5°, 1°)	2 m	30 m	72.5 m	10 mm	±35 mm
	LMS 221	180° (0.25°, 0.5°, 1°)	2 m	30 m	80 m	10 mm	±35 mm
	LMS 291	180° (0.25°, 0.5°, 1°)	2 m	30 m	80 m	10 mm	±35 mm
	LMS 400	70° (0.1°...1°)	1 m	3 m	–	1 mm	±4 mm
	LD OEM	360° (0.125°)	0.5 m	24 m	100 m	3.9 mm	±25 mm
	LD PDS	360° (0.125°)	0.5 m	24 m	100 m	3.9 mm	±25 mm
	LD PeCo	90° (–)	2.5 m	5.5 m	–	–	–
		90° (–)	5.5 m	15 m	–	–	–
HOKUYO	URG-04LX	240° (0.36°)	0.002 m	4 m	5.6 m	–	10mm o 1%
SIEMENS	LS4	190° (0.36°)	0 m	4 m	50 m	5mm	–

Tabella 3.1: Caratteristiche tecniche dei più comuni scanner laser (1)

Modello		Frequenza scansioni	Peso	Dimensioni	Caratteristiche elettriche
SICK	LMS 200	75 Hz	4.5 Kg	156 x 155 x 210 mm	24V DC, typ. 2A, max 6A
	LMS 211	75 Hz	9 Kg	265 x 351 x 202 mm	24V DC, typ. 2A, max 6A
	LMS 221	75 Hz	9 Kg	351 x 265 x 194.5 mm	24V DC, typ. 2A, max 6A
	LMS 291	75 Hz	9 Kg	156 x 155 x 210 mm	24V DC, typ. 2A, max 6A
	LMS 400	150...500 Hz	2.3 Kg	129.9 x 179 x 106.7 mm	24V DC, max 1A
	LD OEM	5...20 Hz	3.2 Kg	222 x 120.5 x 115 mm	24V DC, max 1.5A
	LD PDS	5...20 Hz	3.2 Kg	222 x 120.5 x 115 mm	24V DC, max 1.5A
	LD PeCo	10 Hz	6 Kg	365 x 350 x 180 mm	24V DC, max 1.5A
HOKUYO	URG-04LX	10 Hz	0.16 Kg	50 x 50 x 70	5V DC, typ 0.5 A, max 0.8 A
SIEMENS	LS4	25 Hz	2 Kg	130 x 195 x 135 mm	24V DC, 0.3 A

Tabella 3.2: Caratteristiche tecniche dei più comuni scanner laser (2)

Modello		Tipo di luce emessa	Classe di sicurezza
SICK	LMS 200	Infrarossa	Classe 1
	LMS 211		
	LMS 221		
	LMS 291		
	LMS 400	Visibile ($\lambda = 650nm$)	Classe 2
	LD OEM	Infrarossa ($\lambda = 905nm$)	Classe 1
	LD PDS		
	LD PeCo		
HOKUYO	URG-04LX	Infrarossa ($\lambda = 785nm$)	Classe 1
SIEMENS	LS4	Infrarossa ($\lambda = 905nm$)	Classe 1

Tabella 3.3: Tipo di luce emessa e classe di sicurezza dei più comuni scanner laser

Anche se la classe 2 di sicurezza non comporta rischi oggettivi, è preferibile usare laser di classe 1, che sono assolutamente sicuri. Inoltre è buona prassi rendere l'iterazione tra un sensore e l'ambiente non rilevabile all'uomo, quindi è preferibile utilizzare laser infrarossi che non sono visibili all'occhio umano. L'unico modello che non rispetta le caratteristiche ricercate è il SICK LMS 400, mentre tutti gli altri sono di classe 1 con emissione di luce infrarossa.

3.2 Posizionamento in ambienti indoor

Nella Sezione 2.2.7 è stato brevemente introdotto il problema della localizzazione e del posizionamento. Si introducono ora alcuni metodi utilizzati per il posizionamento in ambienti indoor, senza il quale non sarebbe possibile rendere autonoma la pianificazione e l'esecuzione di percorsi.

Non esiste un metodo di posizionamento che sia in assoluto migliore di altri e spesso è necessario combinare più metodi per ottenere stime della posizione affidabili. Si possono però individuare due categorie in cui suddividere i metodi utilizzati:

- Posizionamento relativo.
- Posizionamento assoluto.

Il posizionamento relativo è ottenibile con l'integrazione nel tempo di informazioni odometriche o inerziali. E' ben noto in letteratura ([5]) che questo metodo soffre del problema della crescita senza limiti dell'errore. È necessario quindi introdurre dei sistemi di "azzeramento" della posizione,

Figura 3.4: Il sensore inerziale XSens MTi

ovvero deve essere periodicamente disponibile un informazione sulla posizione assoluta del robot mobile, che serva da nuovo punto di partenza per il sistema di posizionamento relativo.

Tipicamente i sistemi che si basano su informazioni odometriche ottengono le informazioni sulla rotazione delle ruote o sull'angolo di sterzata con l'uso di encoders. Il costo di questi dispositivi è solitamente basso. In alternativa si possono utilizzare i sistemi inerziali, principalmente basati su giroscopi e accelerometri. Misurando le velocità di rotazione attorno agli assi e le accelerazioni lungo gli assi di un sistema di riferimento solidale con tali dispositivi, si può risalire alla posizione e alla velocità tramite integrazione nel tempo. I costi sono più elevati rispetto ai sistemi di odometria basati su encoder, ma un sensore inerziale è autocontenuto, ovvero può essere collocato su un qualunque robot senza modifiche alla struttura dei motori o delle ruote, come invece richiesto dai sistemi odometrici tradizionali. Per una panoramica sui principi di funzionamento di encoders, giroscopi e accelerometri e sulle tecnologie costruttive si rimanda a [5].

XSens MTi

Il sensore MTi (Figura 3.4), prodotto da XSens, è un esempio di sistema di rilevazione di dati inerziali miniaturizzato. I dati che è in grado di rilevare sono:

- l'orientamento in tre dimensioni rispetto al sistema di riferimento assoluto terrestre, grazie alla rilevazione dell'intensità del campo magnetico.
- la velocità di rotazione intorno ai tre assi del sistema di riferimento solidale con il sensore, rilevata con l'uso di giroscopi.
- l'accelerazione lungo i tre assi del sistema di riferimento solidale con il sensore, rilevata con accelerometri.

Dimensioni	58 x 58 x 22 mm
Peso	50 g
Alimentazione	da 4.5 a 30 V
Potenza richiesta	360 mW
Interfaccia	RS232 o USB
Frequenza rilevazioni	512 Hz velocità e accelerazioni
	120 Hz orientamento assoluto
Massima velocità rotazione rilevabile	±300 deg/s
Massima accelerazione rilevabile	±50 m/s^2
Massima campo magnetico rilevabile	±750 mGauss

Tabella 3.4: Caratteristiche tecniche del sensore XSens MTi

Le dimensioni di questo sensore sono molto contenute (58 x 58 x 22 mm), così come il suo peso (50 g). I dati di velocità di rotazione e accelerazione sono disponibili ad una frequenza massima di 512 Hz, quelli di orientamento ad un massimo di 120 Hz. In Tabella 3.4 sono riportate le caratteristiche tecniche del sensore MTi, per ulteriori informazioni si rimanda a [53] e al sito del costruttore[5].

3.2.1 Posizionamento assoluto

Un sistema di posizionamento assoluto trova applicazione sia come metodo di posizionamento sia come sistema di azzeramento e correzione di un sistema di posizionamento relativo. Le modalità con cui può essere effettuato sono le seguenti:

Fari attivi La posizione viene calcolata dal robot misurando le direzioni di incidenza dei segnali trasmessi da tre o più fari. I fari emettono solitamente luce o onde radio e devono essere posizionati in luoghi noti nell'ambiente.

Landmark artificiali In luoghi noti nell'ambiente vengono posizionati degli oggetti, bidimensionali o tridimensionali, che sono facilmente distinguibili con sistemi di visione artificiale. Le proprietà geometriche dei landmark possono essere sfruttate per calcolare la distanza da essi, oppure la posizione può essere stimata con metodi di triangolazione come nel caso dei fari attivi. Il riconoscimento dei landmark e la stima della posizione rispetto a essi può risultare computazionalmente oneroso.

[5] http://www.xsens.com

Landmark naturali Si utilizzano dei punti salienti dell'ambiente, senza necessità di preparare l'ambiente in cui il robot andrà a operare. L'affidabilità di questo metodo è inferiore a quella basata sui landmark artificiali e comunque è richiesta una fase di addestramento in cui riconoscere le feature da associare ai luoghi nell'ambiente. Un esempio di applicazione di questa tecnica di localizzazione è il robot Minerva [47].

Modello dell'ambiente Con questa tecnica è possibile confrontare le informazioni acquisite dai sensori del robot con un modello dell'ambiente noto. Grazie a questa corrispondenza è possibile determinare la posizione del robot. È inoltre possibile estendere la mappa iniziale utilizzando le nuove informazioni acquisite, con tecniche che vanno sotto il nome di *Simultaneus Localization and Mapping (SLAM).*

Tag Mobili attivi e sensori passivi sulle pareti. Un tag attivo è un dispositivo che emette un segnale identificativo (ad esempio un ultrasuono) che possa essere recepito da sensori in posizione nota. In base agli istanti di tempo in cui il segnale emesso dai tag viene recepito dai diversi sensori, è possibile calcolarne la posizione ed eventualmente anche l'orientamento.

3.2.2 StarGazer

Un esempio di sistema di rilevamento della posizione basato su landmark artificiali è StarGazer. I landmark sono superfici planari in grado di riflettere luce infrarossa, tipicamente posizionate sul soffitto. Grazie a un proiettore di raggi infrarossi accoppiato a una telecamera (Figura 3.5), è possile risalire al codice del landmark visibile e alla posizione relativa della telecamera rispetto ad esso. Con questo sistema è possibile coprire zone di diametro variabile tra i 2.5 m e i 5 m con un singolo tag (a seconda della distanza del soffitto). La massima distanza del soffitto dal sistema proiettore e telecamera è di 6 m. Il numero di tag differenti è 32 nella versione base e 4096 in quella destinata a coprire ampi spazi. A seconda della distanza dal soffitto sono proposti diversi tipi di tag. Alcune caratteristiche tecniche sono riportate in Tabella 3.5 e un esempio d'uso è visibile in Figura 3.6. Per ulteriori dettagli si rimanda a [19] e al sito del costruttore[6].

[6] http://www.hagisonic.com

Figura 3.5: Il proiettore ad infrarossi e la telecamera di StarGazer

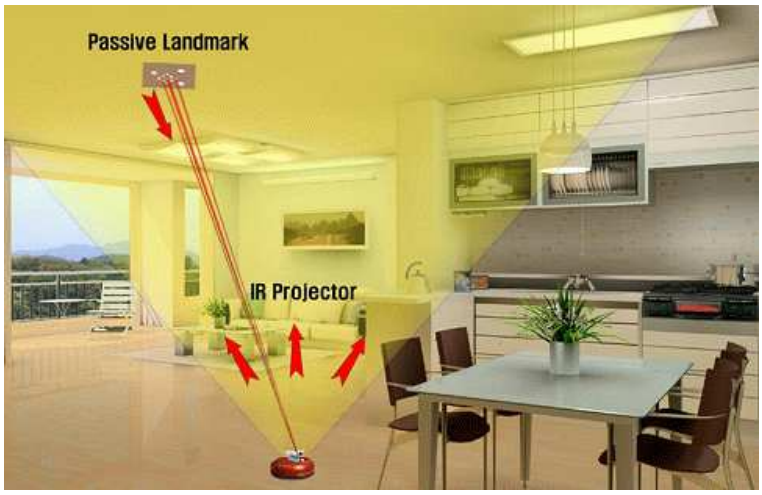


Figura 3.6: Esempio d'uso di StarGazer

Interfaccia	UART(TTL 3.3V), 115.200bps
Dimensioni	50 x 50 x 28mm
Protocollo	basato sul codice ASCII
Frequenza rilevazione	10-20 Hz
Range di localizzazione per singolo landmark	diametro da 2.5 a 5 m (con soffitto a distanza da 2 a 6 m)
Ripetibilità	2 cm
Risoluzione dell'angolo	1.0 grado
Tipi di landmark	Tipo 1: $1.2 \leq h \leq 2.9$ m Tipo 2: $2.9 \leq h \leq 4.0$ m Tipo 3: $4.0 \leq h \leq 6.0$ m
Tipi di landmark (classificati per numero di codici disponibili)	HL1: 32 (per spazi normali) HL2: 4,096 (per spazi estesi)
Caratteristiche elettriche	5 V: 300 mA, 12 V: 70 mA

Tabella 3.5: Caratteristiche tecniche di StarGazer

3.2.3 Ubisense

È un prodotto commerciale per la localizzazione basato su tag mobili attivi e sensori fissi sulle pareti. I tag mobili emettono segnali a banda ultralarga (*UltraWideBand*, UWB), ovvero impulsi di energia in radiofrequenza di durata estremamente ridotta (nell'ordine dei nanosecondi o picosecondi). Il principale vantaggio degli UWB rispetto alle normali onde radio è dato dalla possibilità di distinguere facilmente quali segnali sono corretti rispetto a quelli che giungono da riflessioni multiple.

I sensori posizionati nell'ambiente ricevono i segnali emessi dai tag e ne stimano la posizione utilizzando due algoritmi:

- Time Differential Of Arrival (TDOA) che calcola la posizione del tag analizzando gli istanti di tempo in cui il segnale emesso dal tag viene ricevuto dai vari sensori
- Angle Of Arrival (AOA) che stima l'angolo da cui arriva il segnale emesso dal tag.

L'uso combinato dei due metodi permette di rilevare la posizione anche quando il tag è visibile da soli due sensori, diminuendo quindi il numero di sensori necessari per coprire interamente l'ambiente interessato. La posizione rilevata con questo sistema presenta una deviazione standard di circa 15cm, che può crescere in condizioni sfavorevoli.

L'installazione del sistema richiede un notevole sforzo per il posizionamento dei sensori. La precisione con cui si posizionano e si calibrano i sensori influisce fortemente sulla precisione del sistema. L'area che si può coprire con l'uso di 4 sensori è di circa 400m^2. Per ulteriori informazioni si rimanda al sito del costruttore[7] e [48] [49] [50].

3.3 Pianificazione di percorsi

La risoluzione del problema della pianificazione del moto dei robot (*robot motion planning*) è uno degli elementi fondamentali per la creazione di robot autonomi. L'obiettivo generale della pianificazione del moto dei robot è quello di trovare tecniche mediante le quali un robot possa automaticamente determinare un percorso da una posizione iniziale a una finale. Questa definizione del problema è del tutto generale ed è valida sia per robot mobili sia per manipolatori. Le condizioni aggiuntive che governano la ricerca di un percorso sono:

[7] http://www.ubisense.net

- Evitare urti con ostacoli presenti nello spazio di lavoro.
- Prevedere un margine di sicurezza nell'evitare gli ostacoli.
- Garantire la realizzabilità del percorso considerando i vincoli cinematici del robot.
- Ottenere un percorso il più possibile breve e rapido.

La pianificazione si occupa solo del calcolo del percorso e non dell'inseguimento, rimandando al controllore del robot il compito di attuare i movimenti necessari ed eventualmente comunicare la necessità di una nuova pianificazione del percorso.

Il problema della pianificazione di percorsi è stato largamente affrontato e molte soluzioni sono state proposte. In questa sezione sono presentate le linee generali delle soluzioni proposte, per approfondimenti si rimanda a [35] e [27].

3.3.1 Classificazione dei pianificatori

È possibile individuare tre categorie di soluzioni al problema della pianificazione:

- Metodi basati su grafi (*roadmap*).
- Scomposizione in celle (*cell decomposition*).
- Campi di potenziale (*potential field*).

I metodi basati su grafi riducono la mappa a un grafo che mantiene solo le informazioni salienti dei collegamenti esistenti tra le zone. I metodi basati su scomposizione in celle dividono la mappa in celle che possono essere occupate o libere e la ricerca del percorso avviene dunque su una griglia. I metodi basati sui campi di potenziale usano un concetto di energia potenziale di attrazione e repulsione. Affidando un potere attrattivo al punto di goal e repulsivo agli ostacoli è possibile creare percorsi con le caratteristiche desiderate.

3.3.2 Metodi basati su grafi

Un primo metodo di ricerca di percorsi su grafi consiste nello scegliere un insieme di punti nella mappa dell'ambiente in modo che essi non siano nello spazio occupato dagli ostacoli. Questi punti rappresentano i nodi del grafo. Gli archi tra i nodi vengono generati secondo un criterio di visibilità tra

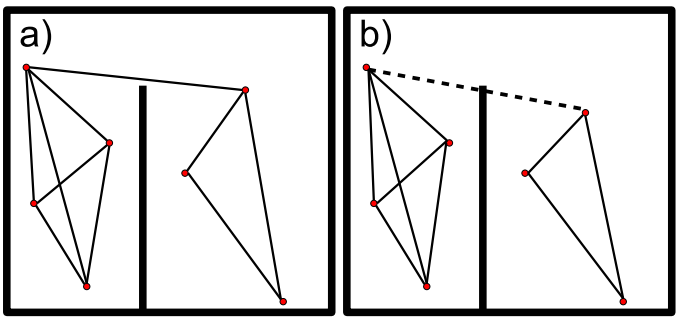


Figura 3.7: Grafo di visibilità su una mappa di esempio. Si può notare con la differente scelta dei punti può compromettere la raggiungibilità di una zona. Nella figura b) l'arco tratteggiato non fa parte del grafo di visibilità a causa dell'intersezione con un ostacolo. Il grafo risulta non connesso e diviso in due componenti

i punti corrispondenti ai nodi: se per andare dal nodo i al nodo j con un segmento rettilineo non si incontrano ostacoli allora l'arco i-j è parte del grafo di visibilità, ovvero è una "strada" percorribile per raggiungere j partendo da i o viceversa. Una volta che il grafo è interamente costruito, la ricerca di percorsi usa metodi di ricerca di cammini su grafi. La mappa nel suo complesso viene ignorata, se ne mantiene solo una descrizione topologica delle connessioni.

Il metodo con cui si campionano i punti nella mappa per costruire il grafo influisce fortemente sulle possibilità di realizzare percorsi. Se i punti sono scelti in modo casuale è possibile che si verifichi, soprattutto in presenza di passaggi stretti, una situazione come quella di Figura 3.7, in cui due zone della mappa possono risultare raggiungibili (Figura 3.7a) o non raggiungibili (Figura 3.7b).

3.3.3 Scomposizione in celle

Con la scomposizione in celle la mappa viene partizionata in tante zone tali che ogni zona non si intersechi con nessun altra e l'unione dello spazio coperto dalle singole zone copra tutta la mappa. In linea di principio la forma e le dimensioni delle celle possono essere di qualunque natura, ma per semplicità si utilizzano principalmente forme poligonali regolari. Ogni cella, oltre alle informazioni sulla sua posizione, dimensione, forma e connessioni con le celle adiacenti, mantiene anche uno stato che può assumere due valori: libera o occupata. Una cella è occupata quando il suo spazio è intersecato da uno o più ostacoli. La ricerca di un percorso da un punto di partenza ad

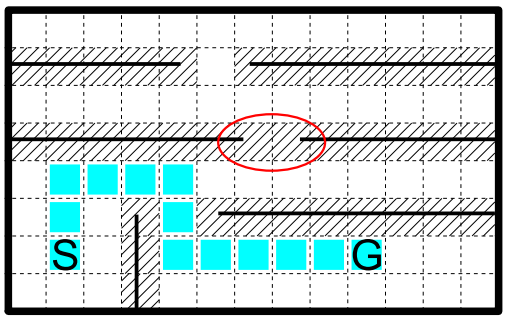


Figura 3.8: Scomposizione in celle quadrate ed esempio di percorso calcolato da un punto di inizio (S) e un punto di fine (G). Le celle occupate sono tratteggiate. Si può notare che la zona evidenziata in rosso presenta un possibile passaggio reale che la granularità delle celle considera invece non praticabile.

un punto destinazione avverrà con una ricerca su grafo tra celle adiacenti libere.

È facile immaginare che il numero delle celle influenza la rapidità di costruzione della mappa e il numero di strade da esplorare nella ricerca di un percorso. Più alto è il numero delle celle più sono numerosi i controlli di intersezione con gli ostacoli da effettuare per valutare lo stato delle celle. Più sono le celle, più il numero di passaggi intermedi che un percorso considera è elevato. Un numero troppo elevato di celle comporta tempi di calcolo molto lunghi. Un numero di celle troppo basso può rendere non connesse due zone della mappa che in realtà lo sono (Figura 3.8). Per ovviare a questo problema sono state sviluppate soluzioni che utilizzano celle di diverse dimensioni a seconda della zona della mappa che si considera: celle più piccole sono utili su zone più ostruite e celle più grandi su zone completamente libere.

3.3.4 Campi di potenziale

Il metodo dei campi di potenziale prevede di assegnare al punto di destinazione un potenziale attrattivo e agli ostacoli un potenziale repulsivo. La forma del potenziale assegnato deve garantire la località, ovvero l'effetto repulsivo di un ostacolo deve esaurirsi a una certa distanza dall'ostacolo stesso. Anche il potenziale attrattivo gode della stessa proprietà, ma il suo decadimento deve essere più lento, per garantire che anche partendo da un punto lontano nella mappa l'attrazione verso il goal sia significativa. La funzione potenziale complessiva terrà conto di tutte le componenti repulsive e di quella attrattiva. L'algoritmo che guida il movimento del robot è solitamente semplice e basato su metodi matematici noti di minimizzazione come la di-

scesa del gradiente. Il problema principale legato all'uso di metodi basati su campi potenziali è dato dai possibili minimi locali che si possono incontrare lungo il percorso. In queste situazioni l'algoritmo di discesa del gradiente termina la sua esecuzione, ma il punto raggiunto non è il minimo globale posto in corrispondenza del goal. Numerose soluzioni sono state proposte e sono allo studio per ovviare a questo problema.

3.4 Architetture software per il controllo

Il problema del controllo nel campo della robotica mobile è stato studiato estesamente da diversi autori e nel corso degli anni varie architetture sono state proposte. Le tre principali sono descritte in [18] e sono:

- Gerarchica.
- Reattiva.
- Ibrida.

Una nuova architettura che conserva i pregi delle tre architetture proposte, ma che usa un meccanismo differente per la gestione dei comportamenti è stata proposta in [3] seguendo il principio della *Composizione Gerarchica Informata (HIC)* e sarà trattata nella Sezione 3.6.

3.4.1 Architettura gerarchica

L'architettura *gerarchica* è stata proposta verso la fine degli anni sessanta e l'approccio da essa utilizzato è considerato classico nel campo dell'Intelligenza Artificiale. Questo paradigma è stato il primo a essere utilizzato per la progettazione di controllori robotici ed è basato sulla scomposizione *verticale* del problema di controllo in una serie di unità funzionali. Ogni unità ha il compito di provvedere all'esecuzione di un'attività specifica. Come visibile in Figura 3.9, le attività sono svolte in modo sequenziale e il coordinamento tra esse è dunque implicito nella struttura dell'architettura stessa. L'attuazione sequenziale crea però un legame stretto tra i livelli, in quanto l'azione di controllo di un livello non può essere prodotta prima del termine delle attività dei livelli precedenti, con un conseguente ritardo inserito tra una percezione e l'attuazione di un comando. Il tempo di esecuzione di un ciclo di controllo è dato dalla somma dei tempi di esecuzione dei singoli moduli, tra i quali si può sicuramente individuare come computazionalmente molto oneroso quello di pianificazione. Un punto debole di questa architettura è

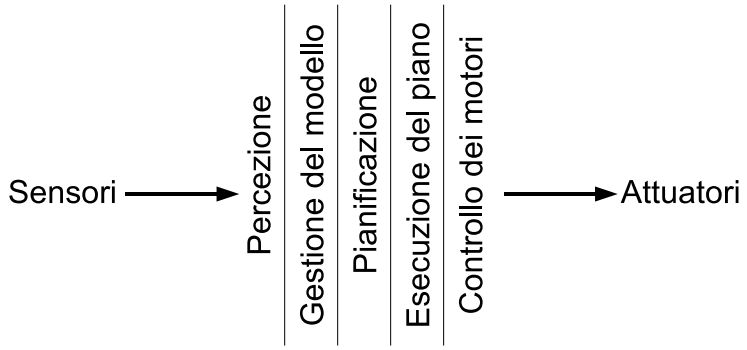


Figura 3.9: Scomposizione verticale di un controllore robotico, tipica dell'architettura gerarchica

costituito dalla sua rigidità, infatti il malfunzionamento di un singolo modulo impedisce la corretta esecuzione di tutto il sistema. La lentezza con cui gli ingressi provocano cambiamenti nelle uscite rende questa architettura adatta ad ambienti statici e predicibili.

3.4.2 Architettura reattiva

L'architettura *reattiva*, a differenza di quella gerarchica, cerca di estromettere il pianificatore dal controllo, per rendere più rapido e lineare il flusso delle informazioni tra il modulo di percezione e quello di attuazione. Il controllore non ragiona su un modello del mondo, ma dispone di un certo numero di moduli elementari (*comportamenti*). Ciascun comportamento reagisce direttamente a un sottoinsieme di informazioni sensoriali e propone delle azioni per di raggiungere uno scopo elementare. I comportamenti sono progettati indipendentemente gli uni dagli altri e la loro esecuzione è parallela. Il comportamento complessivo del sistema è ottenuto dall'interazione dei singoli comportamenti elementari.

Agendo direttamente sulle informazioni sensoriali, questa architettura risulta estremamente rapida nel reagire ai cambiamenti dell'ambiente. Il raggiungimento di obiettivi di alto livello non può però essere garantito, in quanto non esiste un modello sul quale verificare che le azioni proposte portino effettivamente a un risultato. Per lo stesso motivo l'ottimalità della successione di azioni compiute dal sistema non è assicurata.

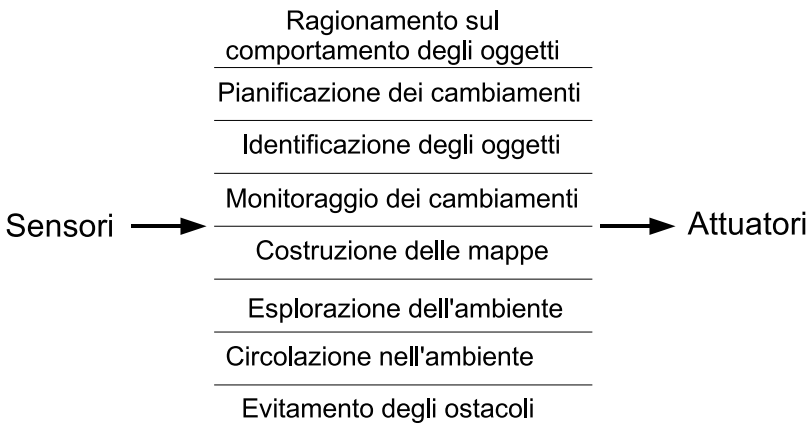


Figura 3.10: Scomposizione orizzontale di un controllore robotico, tipica dell'architettura reattiva

Estensioni dell'architettura reattiva

L'architettura reattiva è estremamente dinamica e adatta ad ambienti in rapida evoluzione, ma l'assenza di un modulo di pianificazione rende limitato il numero di comportamenti globali esprimibili. Alcune estensioni sono state proposte, come la *subsumption architecture* (la cui scomposizione orizzontale del controllore è illustrata in Figura 3.10) di Brooks [8] e l'architettura a *moduli di competenza* di Maes [30]. Si cita per completezza l'architettura proposta da Arkin con il nome *Motor Schema* [1], senza però trattarne ulteriormente la struttura.

L'architettura subsumption è un'architettura reattiva basata sulla scomposizione del comportamento globale desiderato in comportamenti elementari organizzati a livelli. I livelli più alti sono quelli con grado di astrazione più alto (e.g. ragionamento sul mondo, previsione del cambiamento dell'ambiente), i livelli più bassi si occupano dei compiti più basilari (e.g. evitare collisioni). Tutti i comportamenti sono eseguiti in parallelo, quindi tutti gli obiettivi che i comportamenti hanno sono perseguiti in modo concorrente. Ogni singolo comportamento è realizzato come una macchina a stati finiti estesa (AFSM). I comportamenti sono collegati tra loro in quanto le uscite di un modulo possono essere ingresso di altri moduli. Un modulo può sopprimere l'uscita di un altro (ad esempio un modulo che svolge il compito di evitare ostacoli può inibire in situazioni critiche le uscite di tutti i moduli che propongono movimenti potenzialmente pericolosi). Analogamente anche gli ingressi dei moduli possono essere soppressi da altri moduli, ma in

questo caso il valore di ingresso al modulo può essere fornito da chi effettua la soppressione. Ogni livello è costituito da più AFSM collegate tra loro e l'insieme di tutti i livelli costituisce la struttura di controllo.

Il potere espressivo dell'architettura subsumption è elevato, grazie alla possibilità di coordinare i comportamenti, ma la progettazione della rete risulta molto complessa e i moduli risultano strettamente interconnessi e progettati l'uno per l'altro.

L'architettura a moduli di competenza di Maes permette di associare ad ogni modulo un livello di attivazione che ne indica l'importanza rispetto agli altri. Il livello di attivazione è stabilito dinamicamente in base alla situazione, ovvero a variabili di stato o a input del controllore. Ogni modulo ha delle precondizioni, che specificano quando quel particolare comportamento è attuabile, e delle postcondizioni, che regolano le interazioni con gli altri moduli. L'architettura così realizzata risulta rigida, perchè ogni modulo deve essere a conoscenza dell'insieme dei moduli presenti per poter gestire correttamente l'interazione con essi.

3.4.3 Architettura ibrida

L'architettura *ibrida* nasce dal tentativo di integrare le due architetture precedenti in una struttura a livelli. I livelli sono tipicamente tre. Il più basso è responsabile dei comportamenti elementari puramente reattivi e collega direttamente informazioni sensoriali e attuatori. Lo scopo di questo livello è quello di reagire a situazioni urgenti e impreviste, come la presenza di ostacoli vicini al robot. Le proposte dei livelli superiori possono essere modificate o ignorate.

Il livello intermedio è costituito da comportamenti simili a quelli del livello più basso, ma i dati di input non provengono solo dai sensori, ma anche dal pianificatore del livello più alto. Questo livello è responsabile del conseguimento degli obiettivi elementari proposti dal pianificatore, ma l'esecuzione dei comandi proposti è subordinata alle decisioni del livello sottostante.

Il terzo livello è tipicamente riservato ad un pianificatore e comunica al livello intermedio le singole azioni del piano da svolgere. Il *feedback* dei livelli inferiori verso i livelli più alti permette di gestire i casi in cui sia necessaria una ripianificazione delle attività. La struttura tipica di questa architettura è illustrata in Figura 3.11. Una caratteristica fondamentale di questo approccio, non ottenibile con un'architettura reattiva, è la possibilità di creare piani di esecuzione non banali, grazie alla presenza di un modulo di pianificazione e allo stesso tempo di garantire tempestività in situazioni urgenti.

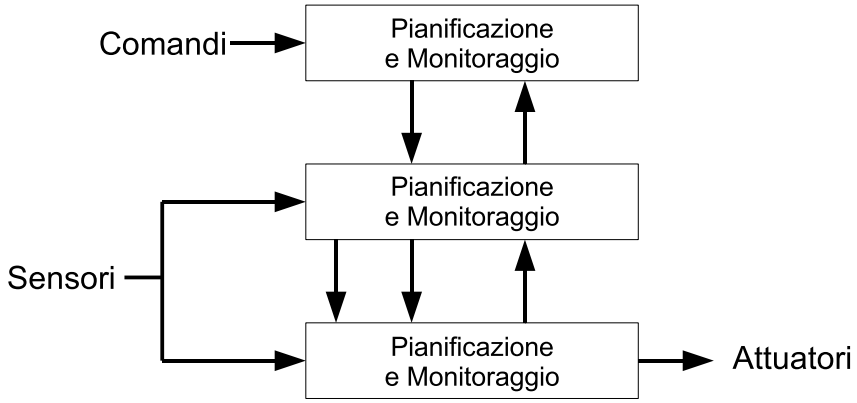


Figura 3.11: Scomposizione a tre livelli di un controllore robotico ibrido

3.5 Logica Fuzzy

La *logica fuzzy* fu introdotta per la prima volta da Zadeh nel 1965 [55] ed è stata largamente utilizzata nella realizzazione di controllori robotici. Il termine fuzzy è traducibile in italiano con "sfumato" o "nebuloso" ed è proprio questo il concetto che sta alla base della logica fuzzy. Nella logica classica il valore di un predicato può essere vero o falso (e.g., "oggi fa freddo" può assumere valore vero o falso), con valore convenzionale 0 e 1. Nella logica fuzzy il valore di verità di un predicato può assumere un valore nel campo reale tra 0 e 1, quindi il confine tra un'affermazione vera o falsa non è netto, ma più "sfumato".

Questa modellizzazione del concetto di verità è più attinente alle situazioni reali rispetto alla logica classica. Considerando il predicato d'esempio utilizzato in precedenza, non è possibile stabilire il valore di verità assoluto al concetto di "freddo", mentre è possibile darne una valutazione in termini più sfumati, come ad esempio "è abbastanza vero che oggi fa freddo", dove con abbastanza vero si intende che è più vera l'affermazione "oggi fa freddo" rispetto a "oggi non fa freddo". Gli elementi che costituiscono la logica fuzzy sono i seguenti:

- Insiemi fuzzy.
- Relazioni tra insiemi fuzzy.
- Operazioni su insiemi fuzzy.
- Predicati fuzzy.

- Regole fuzzy.

3.5.1 Insiemi Fuzzy

Per introdurre la struttura degli insiemi fuzzy si fa riferimento alla seguente definizione di insieme classico: un insieme I è definito da una funzione caratteristica μ_I che valutata in corrispondenza di un elemento x può assumere solo i valori 0 o 1:

$$\mu_I(x) = \begin{cases} 0 & \text{se} \quad x \notin I \\ 1 & \text{se} \quad x \in I \end{cases}$$

L'estensione di questa definizione agli insiemi fuzzy prevede che la funzione caratteristica μ_I assuma valori reali compresi tra 0 e 1, esprimendo cioè un grado di appartenenza all'insieme I dell'elemento x. Un insieme fuzzy è dunque una generalizzazione del concetto di insieme. Un insieme canonico può essere espresso con insiemi fuzzy utilizzando la funzione caratteristica a due valori precedentemente illustrata.

3.5.2 Relazioni tra insiemi Fuzzy

Le relazioni definite tra insiemi fuzzy sono l'uguaglianza e il contenimento. Un insieme fuzzy I è uguale a J se e solo se la loro funzione caratteristica coincide puntualmente:

$$I = J \quad \Leftrightarrow \quad \forall x \quad \mu_I(x) = \mu_J(x)$$

Un insieme fuzzy I è sottoinsieme di J se e solo se ogni elemento x dell'universo ha un grado di appartenenza all'insieme I al più uguale al grado di appartenenza di x all'insieme J:

$$I \subseteq J \quad \Leftrightarrow \quad \forall x \quad \mu_I(x) \leq \mu_J(x)$$

3.5.3 Operazioni tra insiemi Fuzzy

Le operazioni definite su insiemi fuzzy sono intersezione, unione e complemento, come per gli insiemi classici. Queste operazioni non sono però definite in modo univoco, come invece avviene nel caso degli insiemi classici. Faremo riferimento ai soli operatori proposti da Zadeh:

$$\begin{aligned} \mu_{A \cap B}(x) &= min(\mu_A(x), \mu_B(x)) \\ \mu_{A \cup B}(x) &= max(\mu_A(x), \mu_B(x)) \\ \overline{\mu_A}(x) &= 1 - \mu_A(x) \end{aligned}$$

3.5.4 Predicati fuzzy

Un predicato è un letterale a cui si associa un valore di verità. Detto P un predicato atomico e a una costante, il valore di verità di $P(a)$, indicato con $\langle P(a) \rangle$ è dato da $\mu_P(a)$, dove con $\mu_P(x)$ si indica la funzione caratteristica di un insieme fuzzy associato al predicato P. Il valore di verità di un predicato composto è calcolato tramite ricorsione, utilizzando le operazioni precedentemente definite, con i valori di verità delle sue singole componenti:

$$\begin{aligned} \langle \neg Q \rangle &= 1 - \langle Q \rangle \\ \langle Q \wedge R \rangle &= min(\langle Q \rangle, \langle R \rangle) \\ \langle Q \vee R \rangle &= max(\langle Q \rangle, \langle R \rangle) \end{aligned}$$

3.5.5 Regole fuzzy

Una regola fuzzy è una struttura formata da un'antecedente e da un conseguente. L'antecedente è un predicato fuzzy (semplice o composto) mentre il conseguente specifica dei valori fuzzy da associare a opportune variabili di uscita. Un esempio di regola fuzzy è

$$SE \quad \text{TargetADestra} \quad ALLORA \quad \text{RuotaADestra}$$

3.6 BRIAN e Mr.BRIAN

Mr.BRIAN (Multilevel Ruling BRIAN) [3], estensione di BRIAN (BRIAN Reacts by Inferring ActioNs) [4] è un sistema di controllo di robot basato su comportamenti che permette di ridurre la complessità di progettazione e garantire la riusabilità e la modularità dei comportamenti grazie all'uso della *composizione gerarchica informata* presentata in [3]. Il cuore del sistema è la logica fuzzy, presentata nella sezione precedente, con la quale si descrivono i comportamenti desiderati e le modalità di selezione del comportamento da utilizzare. Sviluppata presso il Politecnico di Milano, questa architettura è stata utilizzata in alcuni progetti del gruppo di Intelligenza artificiale e Robotica, come ad esempio Milan Robocup Team[8] e FollowMe [45].

3.6.1 BRIAN

Brian è un'architettura di controllo basata su comportamenti che prevede un sistema di coordinamento per la scelta dei comportamenti attivi e per la fusione dei risultati, basato su condizioni descritte in termini di variabili di stato e rilevazioni sensoriali.

[8] http://robocup.elet.polimi.it/MRT

Comportamenti fuzzy

Un comportamento fuzzy è costituito da un insieme di regole fuzzy (vedi Sezione 3.5.5). Il raggruppamento di più regole in un comportamento è basato su una modellizzazione dell'attività che tale comportamento è destinato a svolgere. Ad esempio, un comportamento di evitamento ostacoli raggruppa tutte le regole che, considerando i valori dei predicati sulla posizione degli ostacoli e della direzione desiderata del robot, agiscono in modo da modificare la traiettoria per evitare collisioni e raggiungere comunque l'obiettivo.

Controllo fuzzy

Il processo che permette di ragionare sui dati di input per produrre un output che rappresenta l'azione di controllo proposta è il seguente:

1. Fuzzyficazione dei dati di ingresso.
2. Valutazione del valore di verità dei predicati.
3. Scelta dei comportamenti da attivare.
4. Valutazione delle regole dei singoli comportamenti.
5. Fusione dei risultati.
6. Defuzzyficazione dei risultati.

La spiegazione in dettaglio di ogni parte del processo di ragionamento è la seguente:

1. Fuzzyficazione dei dati di ingresso

BRIAN permette di associare a ogni variabile di ingresso un insieme di insiemi fuzzy. Ad esempio, per la variabile *Distanza* possiamo associare gli insiemi fuzzy LONTANO, MEDIO, VICINO, raggruppati a loro volta nell'insieme TIPODISTANZA. La forma delle funzioni caratteristiche dei singoli insiemi fuzzy può essere rettangolare (definendo dunque insiemi tipici della logica classica), ad impulso (definendo insiemi discreti), triangolare, trapezoidale, a rampa in salita o in discesa e triangolo diviso (Figura 3.12). La fuzzyficazione dei dati consiste nell'associare a un particolare valore di distanza fornito il valore di verità di ogni predicato atomico associato agli insiemi fuzzy contenuti in TIPODISTANZA.

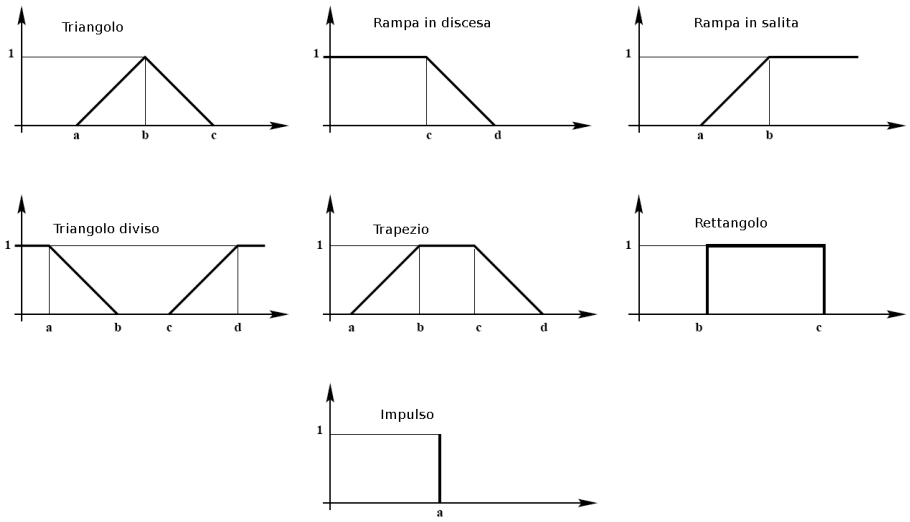


Figura 3.12: Funzioni caratteristiche per la definizione di insiemi fuzzy previste in BRIAN

2. Valutazione del valore di verità dei predicati

Una volta calcolati i valori dei predicati atomici in base ai valori delle variabili di ingresso, è necessario calcolare il valore di verità dei predicati complessi con il metodo ricorsivo illustrato nella Sezione 3.5.4. La possibilità di definire predicati fuzzy composti semplifica la costruzione di strutture complesse. Ad esempio è possibile definire un predicato

$$\begin{aligned}\text{NessunOstacolo} = (\neg\text{OstacoloNord}) \wedge (\neg\text{OstacoloOvest}) \\ \wedge(\neg\text{OstacoloEst}) \wedge (\neg\text{OstacoloSud})\end{aligned}$$

riassumendo così in un unico predicato un concetto più articolato.

3. Scelta dei comportamenti da attivare

BRIAN introduce un meccanismo di attivazione dei comportamenti che permette di descrivere tramite regole fuzzy quali sono le condizioni che determinano se un certo comportamento è attuabile o meno. Ad esempio, un comportamento che permette di muovere il robot al fine di raggiungere un punto nello spazio è attuabile se la posizione attuale è nota e se il punto da raggiungere è stato specificato. Le regole che governano l'attivazione dei comportamenti sono dette di *CANDO*, in quanto stabiliscono se un comportamento può essere attuato.

4. Valutazione delle regole dei singoli comportamenti

Una volta valutati quali sono i comportamenti attivi, BRIAN valuta le

singole regole che definiscono il particolare comportamento. Nel caso in cui l'antecedente risulti vero, ovvero il valore di verità associato è superiore ad una certa soglia, l'azione proposta dalla regola viene aggiunta all'insieme delle azioni proposte dai comportamenti attivi e pesata con il valore di verità associato all'antecedente delle regola stessa. Se ad esempio un comportamento avesse le seguenti due regole:

SE TargetLontano *ALLORA* Velocità VELOCE
SE TargetVicino *ALLORA* Velocità MEDIA

e i valori di verità associati alle regole fossero rispettivamente 0.4 e 0.6, le azioni proposte dal singolo comportamento considerato per la variabile Velocità risulterebbero VELOCE 0.4 e MEDIA 0.6.

5. Fusione dei risultati

Quando più comportamenti sono attivi si pone il problema di fondere i risultati in un valore sintetico da fornire agli attuatori. BRIAN utilizza anche in questo caso regole fuzzy che permettono di specificare il grado di motivazione di un comportamento. Un comportamento può essere attivo, ma poco desiderato. Si pensi, ad esempio, al caso in cui siano attivi due comportamenti: uno per il raggiungimento di un target e l'altro per l'evitamento di ostacoli. Mano a mano che un ostacolo si avvicina al robot il comportamento di evitamento ostacoli deve assumere priorità rispetto a quello di inseguimento del target. Le regole che permettono di specificare quando un comportamento è desiderato sono dette regole di *WANT* e il loro valore di verità funge da peso nella fusione dei risultati. Ad esempio, se il comportamento di evitamento ostacoli propone velocità in direzione frontale nulla e il comportamento di inseguimento del target propone velocità massima in avanti, le regole WANT associate permettono di spostare il risultato maggiormente verso una o l'altra proposta.

6. Defuzzyficazione dei risultati

Questa fase è duale rispetto a quella di fuzzyficazione dei dati. Essa permette infatti di tradurre le azioni proposte dai comportamenti in valori numerici utilizzabili dagli attuatori. Come nel caso della fuzzyficazione vengono definiti degli insiemi fuzzy e viene specificato un tipo di dato fuzzy che raggruppa più insiemi fuzzy. Le variabili di uscita sono associate a un tipo di dato fuzzy e i valori proposti dai comportamenti sono legati alle singole funzioni di attivazione. Ad esempio, possiamo ipotizzare di definire una variabile Velocità e associarla al tipo fuzzy TVELOCITÀ composto dalle funzioni caratteristiche

LENTA, MEDIA, VELOCE. Le funzioni caratteristiche utilizzabili come uscita sono solo di tipo impulsivo e la defuzzyficazione è basata sulla media pesata dei valori proposti. Se, ad esempio, i valori proposti per Velocità fossero 0.4 MEDIA e 0.6 VELOCE, con funzioni caratteristiche di MEDIA e VELOCE impulsive centrate rispettivamente su 50 e 100 si otterrebbe come dato sintetico defuzzyficato un valore di $0.4 \cdot 50 + 0.6 \cdot 100 = 80$ da inviare agli attuatori.

Un vantaggio dell'architettura di controllo sviluppabile con BRIAN è il legame lasco che si instaura tra i comportamenti: a differenza delle architetture presentate nella Sezione 3.4, i vari comportamenti sono specificati in modo indipendente l'uno dall'altro. La conoscenza della totalità dei comportamenti presenti nel sistema è nota solo al modulo che gestisce le regole di CANDO e WANT. La progettazione del sistema di controllo avviene in modo modulare e riusabile, in quanto ogni singolo comportamento risulta indipendente dagli altri presenti nel sistema. Un difetto individuabile in questo approccio è dato dalla possibilità di fondere i risultati di due comportamenti che risultano contrastanti: se un comportamento sceglie di andare a destra per evitare un ostacolo e un comportamento di inseguimento target propone di avanzare si otterrà un movimento in diagonale che potrebbe compromettere la riuscita dell'evitamento ostacoli. Questo problema viene superato dallo sviluppo di BRIAN in Mr.BRIAN.

3.6.2 Mr.BRIAN

Mr.BRIAN (Multilevel Ruling BRIAN) è il successore di BRIAN e mantiene tutte le caratteristiche di BRIAN, introducendo però la possibilità di strutturare i comportamenti in una gerarchia. Mr.BRIAN supera il problema della ambiguità della fusione dei risultati di BRIAN in quanto permette di stabilire una scala di priorità dei comportamenti. Un comportamento di livello i riceve in ingresso oltre ai dati di input, i valori di uscita del livello $i-1$ e può agire su di essi sia effettuando le sue proposte, sia cancellando le proposte (o alcune proposte) del livello precedente. Considerando l'esempio precedente che creava un comportamento non desiderato in BRIAN, in Mr.BRIAN è possibile posizionare al livello più basso (priorità più bassa) il comportamento di inseguimento target e al livello superiore il comportamento di evitamento ostacoli. Quando l'inseguimento target propone di avanzare in presenza di un ostacolo frontale, il comportamento al livello superiore può cancellare la proposta del livello inferiore e proporre una rotazione per evitare l'ostacolo.

Il ciclo di controllo di Mr.BRIAN, evoluzione di quello di BRIAN, risulta il seguente:

1. Fuzzyficazione dei dati di ingresso
2. Valutazione dei valori di verità dei predicati
3. Scelta dei comportamenti da attivare (regole CANDO)
4. Valutazione delle regole dei comportamenti del livello i-esimo
5. Fusione dei risultati
6. Defuzzyficazione dei risultati
7. Se i era l'ultimo livello: fine del ciclo di controllo
   Altrimenti: incremento del valore di i e ripresa dal punto 1

Si noti che nel momento in cui si riprende il ciclo dal punto 1 non è necessario fuzzyficare nuovamente tutti i valori e rivalutare tutti i predicati. La fuzzyficazione riguarda solo le uscite del livello precedente che si trasformano in ingressi e i predicati da valutare sono solo quelli che coinvolgono le variabili d'uscita del livello precedente.

Mr.BRIAN presenta molti aspetti vantaggiosi rispetto alle architetture classiche presentate nella Sezione 3.4. Innanzitutto l'uso della logica fuzzy permette di descrivere i comportamenti in un linguaggio di alto livello e con un elevato grado di astrazione rispetto ai dati sensoriali. L'architettura si rifà in larga parte a quella reattiva in quanto tutti i comportamenti, a parità di livello, vengono eseguiti in parallelo. Il problema della fusione dei risultati è superato dall'inserimento della gerarchia dei comportamenti, senza imporre la rigidità della struttura e la lentezza dell'architettura gerarchica e superando il problema della dipendenza dei comportamenti e moduli presente nelle architetture di Brooks e Maes. La struttura a livelli della gerarchia di controllo permette di semplificare la progettazione dei comportamenti: essi sono progettati in modo indipendente, tenendo conto solo delle azioni proposte dal livello inferiore come ingresso aggiuntivo ai dati sensoriali. Le condizioni di CANDO e WANT garantiscono la flessibilità nell'attuazione dei comportamenti in base alle condizioni esterne, alle variabili di stato e agli obiettivi da raggiungere.

Capitolo 4

Progetto di una carrozzina robotica

In questo capitolo si illustra tutta la fase di progetto della carrozzina dalle funzionalità avanzate. Partendo dall'analisi dei requisiti e degli obiettivi da raggiungere sono presentate tutte le scelte fatte: i sensori e la componentistica hardware utilizzata, lo studio del montaggio e posizionamento di ogni componente a bordo della carrozzina, il progetto del circuito di interfaccia tra il sistema di controllo della carrozzina e il computer e le caratteristiche richieste al software di controllo.

4.1 Analisi dei requisiti

Lo scopo del presente lavoro è progettare e realizzare una carrozzina dalle funzionalità estese rispetto a quelle normalmente presenti in commercio. La carrozzina non è da realizzare ex-novo, ma a partire da un modello commerciale a cui apportare modifiche e aggiunte. Le funzionalità che si ritiene utile sviluppare per favorire l'uso delle carrozzine elettriche da parte di un maggior numero di utenti sono:

- Guida con diversi dispositivi di comando.
- Guida assistita.
- Movimento autonomo.

L'introduzione di nuovi dispositivi di comando facilita la guida della carrozzina da parte di persone che non sono in grado di utilizzare agevolmente il comune joystick. Uno degli obiettivi di questo lavoro è quello di progettare

un'interfaccia standard che permetta l'introduzione di nuovi dispositivi di comando in modo semplice e rapido. La presenza di più dispositivi di comando rende necessario lo sviluppo di un meccanismo di mutua esclusione che permetta la guida con un solo dispositivo alla volta e allo stesso tempo il cambio automatico del dispositivo di comando.

La guida assistita ha lo scopo di aiutare l'utente nel comando della carrozzina con funzionalità come evitamento ostacoli, evitamento collisioni e correzione della traiettoria. L'utente può essere assistito sia per aumentare o garantire la sua sicurezza e incolumità, sia per migliorare la sua attitudine alla guida, aiutandolo quindi nelle manovre che risultano a lui più difficili. Si pensi, ad esempio, a quanto può essere difficoltoso per un utente afflitto da tremori o altri movimenti accidentali e involontari delle mani guidare una carrozzina elettrica lungo un corridoio. Il rischio di compiere brusche sterzate che lo portino a collisioni con le pareti sarebbe elevato e un sistema di guida assistita potrebbe intervenire mantenendo una distanza di sicurezza da un muro, garantendo contemporaneamente la sicurezza dell'utente e la realizzazione di una traiettoria più fluida rispetto a quella proposta dall'utente.

La possibilità di far muovere autonomamente la carrozzina elettrica permette di sgravare totalmente l'utente dal compito della guida. Tipicamente chi usa carrozzine elettriche è portato a muoversi per la maggior parte del tempo negli stessi ambienti, come, ad esempio, l'abitazione, il luogo di lavoro o la scuola. Una carrozzina in grado di muoversi autonomamente in questi ambienti risparmierebbe, soprattutto alle persone che trovano difficoltà nella guida, la fatica di compiere gli spostamenti abituali, permettendo di specificare solo l'obiettivo da raggiungere, e.g., cucina, sala stampa, bagni, etc..

4.1.1 Obiettivi

Per poter realizzare le funzionalità illustrate nella sezione precedente, è necessario tradurre i requisiti funzionali in obiettivi più semplici che supportino la fase di progettazione e sviluppo. In particolare è necessario

- Scegliere i sensori e la componentistica hardware da utilizzare per perseguire gli obiettivi proposti.
- Progettare e realizzare una struttura fisica che ospiti i sensori e i dispositivi che si vogliono utilizzare.
- Progettare e implementare un'architettura di controllo modulare che

permetta di realizzare funzionalità di guida assistita e di guida autonoma e che supporti l'uso di più dispositivi di comando.

- Provare la carrozzina in ambienti controllati al fine di analizzare e valutare il comportamento delle funzionalità aggiuntive realizzate, identificandone i pregi e i possibili miglioramenti.

I sensori e la componentistica hardware da utilizzare sono scelti in base agli obiettivi da raggiungere. La scelta dei sensori da utilizzare è valutata sia con criteri assoluti, ovvero che riguardano le caratteristiche generali del sensore, sia con criteri specifici che riguardano la possibilità di montare agevolmente e in posizione utile i sensori stessi. La componentistica hardware deve garantire prestazioni di calcolo adeguate e consumi ridotti, per garantire elevata autonomia.

La struttura fisica ha lo scopo di creare una base esterna su cui applicare tutti i componenti fisici (e.g., sensori, circuiti di interfaccia, telecamere ...) che si vogliono aggiungere alla carrozzina. Tale struttura deve intralciare il meno possibile l'accesso e l'uso della carrozzina da parte dell'utente, permettendo però una certa flessibilità nel posizionamento e nel montaggio dei componenti. L'aggiunta di dispositivi inizialmente non previsti deve essere agevole e deve comportare poche modifiche alla struttura complessiva.

Per realizzare funzionalità di guida assistita e autonoma e rendere agevole l'introduzione di dispositivi di comando differenti è necessario utilizzare un'architettura software di controllo che garantisca robustezza in situazioni critiche e modularità nell'inserimento di nuove componenti. La criticità è data dall'incertezza delle situazioni e dall'imprecisione delle rilevazioni sensoriali. Inoltre un sistema robusto deve essere in grado di non compromettere le funzionalità del controllo anche in presenza di malfunzionamenti o errori. La modularità dell'architettura è quindi fondamentale per lo sviluppo di funzionalità complesse, in quanto permette di implementare distintamente funzionalità semplici e lasciare che sia l'interazione tra di esse a sviluppare comportamenti complessi. Ad esempio, realizzando un modulo che gestisce l'evitamento ostacoli indipendentemente dal modulo che si interfaccia con il dispositivo di comando è possibile aggiungere nuovi dispositivi di comando in modo completamente trasparente al sistema di evitamento ostacoli, facilitando così l'espansione del sistema.

Il test del comportamento della carrozzina è volto alla valutazione e analisi delle funzionalità realizzate. Ad esempio, è significativo verificare come si comporta il sistema di assistenza alla guida e quali sono i punti di forza e i possibili miglioramenti da apportare. Grazie a queste valutazioni è

Figura 4.1: La carrozzina Otto Bock Rabbit utilizzata per questo lavoro

possibile indicare in modo preciso quali sono le linee di evoluzione da seguire e quali i risultati ottenuti.

4.1.2 Otto Bock Rabbit

La carrozzina commerciale a disposizione per lo sviluppo del sistema è una Rabbit prodotta dalla ditta tedesca Otto Bock[1]. Come visibile in Figura 4.1 è una carrozzina adatta sia ad ambienti indoor che outdoor, in quanto presenta due ruote posteriori di dimensioni generose e con pneumatici tassellati. La trazione è realizzata con due motori indipendenti che agiscono su ciascuna delle ruote posteriori, mentre le ruote anteriori sono libere e non comandabili. I motori assorbono una potenza di circa 200W ciascuno quanto utilizzati a piena potenza. L'alimentazione è fornita da due batterie da 12V e 70Ah collegate in serie e poste sotto il sedile di guida. Il sistema di controllo dei motori, di comando con il joystick e l'insieme di tutte le componenti elettroniche presenti sulla carrozzina sono parte della linea DX System prodotta dalla Dynamic Controls[2].

4.2 Sensori e componentistica hardware

Una volta stabilite le funzionalità di cui si vuole dotare la carrozzina è necessario individuare quali sono i sensori necessari e la componentistica hardware da utilizzare. I criteri che guidano la scelta dei sensori e della componentistica hardware sono riassumibili in:

[1] http://www.ottobock.com
[2] http://www.dynamiccontrols.com

Figura 4.2: Lo scanner laser Hokuyo URG-04LX

Economicità, relativamente alla classe di dispositivi a cui appartengono.

Prestazioni, valutate con la cifra di merito appropriata.

Rapporto qualità-prezzo, che coinvolge non solo le prestazioni del dispositivo ma anche qualità costruttiva, affidabilità, garanzia e supporto.

Minima invasività, in quanto i dispositivi devono disturbare il meno possibile l'utente che si trova sulla carrozzina. Questo criterio si può tradurre in svariati modi in base all'ambito di applicazione: se si considera un dispositivo posto nelle vicinanze della postazione di guida è possibile rendere minima la sua invasività miniaturizzandolo. Se si considera un dispositivo le cui dimensioni non arrecano disturbi all'utente, si valutano fattori come la rumorosità o la produzione di vibrazioni.

Consumo elettrico, per garantire un'autonomia elevata alla carrozzina elettrica. Sono preferibili dispositivi a basso consumo, ma qualora fosse necessario utilizzare dispositivi che richiedono alimentazione elettrica tale da ridurre in modo drastico l'autonomia della carrozzina, si considererà la possibilità di installare delle batterie supplementari.

Facilità d'implementazione, in quanto è preferibile utilizzare dispositivi con interfacce di collegamento e di alimentazione standard rispetto a sistemi che richiedono un elevato grado di conoscenza o particolari accorgimenti tecnici per essere interfacciati.

La scelta dei sensori e dei componenti da utilizzare è legata agli obiettivi prefissati. Per realizzare un sistema che assista l'utente nella guida, permettendo di evitare collisioni e ostacoli, è indispensabile avere dei sensori di misura della distanza. Al fine di dotare la carrozzina di autonomia è

necessario un sistema di localizzazione, che permetta di conoscere in tempo reale la posizione della carrozzina all'interno di una mappa. L'elaborazione dei dati sensoriali è affidata a un calcolatore che dovrà garantire un buon compromesso tra prestazioni, consumo di corrente e ingombro. Per permettere l'interazione dell'utente con il software di controllo è necessario disporre di periferiche di input/output che non sono necessariamente quelle normalmente utilizzate (e.g., tastiera, mouse, etc.).

4.2.1 Sensori di distanza

Nella Sezione 3.1 sono stati introdotti i principali sensori di misura della distanza e le caratteristiche salienti. Al fine di ottenere una scansione il più possibile completa dell'ambiente che circonda la carrozzina in movimento e degli ostacoli presenti, si è scelto di utilizzare due scanner laser Hokuyo URG-04LX (Figura 4.2), che permettono un range di scansione planare di 240° con risoluzione di 0.36° e distanza massima rilevabile di 5.6m. Le caratteristiche di questo sensore che maggiormente hanno influito sulla sua scelta sono: il prezzo contenuto rispetto agli altri sensori sul mercato, le dimensioni e il peso di gran lunga inferiori a quelle dei concorrenti e il basso consumo di corrente. Il collegamento degli scanner laser al computer avviene tramite l'interfaccia USB e l'alimentazione richiesta dai sensori è di 5V. Per ulteriori informazioni tecniche sullo scanner laser URG-04LX e su altri modelli presenti in commercio si rimanda alla Sezione 3.1.3.

4.2.2 Sensori inerziali e giroscopi

La volontà di estendere le funzionalità della carrozzina elettrica intervenendo nel modo meno invasivo possibile sulle componenti della carrozzina stessa ha portato a non considerare la progettazione di un sistema di odometria. Infatti, l'installazione di encoder sulle ruote o sui motori comporta modifiche sostanziali alla struttura della carrozzina, legando strettamente il progetto a un particolare modello di carrozzina elettrica. I sensori inerziali sono invece autocontenuti, ovvero possono essere utilizzati per misurare accelerazioni lineari e velocità rotazionali semplicemente applicando il sensore alla carrozzina, senza creare vincoli fisici con gli organi in movimento. Si è scelto di utilizzare il sensore XSens MTi, visibile in Figura 4.3 e presentato nella Sezione 3.2, che fornisce i dati della velocità rotazionale intorno ai tre assi del sistema di riferimento solidale con il sensore, l'accelerazione lungo gli stessi tre assi e l'orientamento del sensore rispetto a un sistema di riferimento assoluto terrestre. Il sensore si collega al computer tramite una porta USB, da cui preleva anche l'alimentazione necessaria. I dati forniti sono sovrabbon-

Figura 4.3: Il sensore XSens MTi

danti rispetto a quelli necessari per il controllo del moto di una carrozzina, ma la disponibilità di questo sensore ha indotto a non acquistarne uno che offrisse solo le prestazioni strettamente necessarie.

Considerando di montare il sensore con asse z rivolto verso il basso e x nella direzione frontale della carrozzina, si possono considerare come dati utili l'accelerazione lungo gli assi x e y e la velocità di rotazione intorno all'asse z. Tutti gli altri dati sono superflui in quanto l'accelerazione lungo l'asse z sarà pari alla gravità terrestre (si noti che il sensore rileva la forza che agisce sull'asse, da cui, nota la massa mobile, ricava l'accelerazione), mentre le velocità di rotazione intorno agli assi x e y sono sempre nulle in quanto la carrozzina mantiene sempre fisso il suo orientamento rispetto al terreno (supposto piano ed orizzontale). I dati del magnetometro possono risultare utili per semplificare il posizionamento del sensore, permettendo di correggere i valori di velocità e accelerazione in base all'orientamento relativo del sensore rispetto al sistema di riferimento terrestre. Sarebbe possibile, ad esempio, eliminare eventuali componenti di accelerazione dovute alla gravità terrestre rilevate erroneamente a causa di un non perfetto posizionamento del sensore. Lo studio di sistemi di correzione basati su questo principio richiede però una lunga fase di analisi dei dati e caratterizzazione del sensore e richiederebbe di utilizzare tutti i dati di accelerazione e velocità disponibili. La volontà di rendere il sistema il piú possibile semplice ha indotto perciò a non considerare i dati di orientamento assoluti forniti dal sensore. Individuati come dati strettamente necessari la velocità di rotazione intorno all'asse z e le accelerazioni lungo x e y sarà possibile, una volta dimostrata la validità del sistema realizzato, cercare sul mercato un sensore che offra solo i dati necessari, con un risparmio economico notevole.

Figura 4.4: La telecamera UniBrain Fire-I400 priva di obiettivo

4.2.3 Sistemi di localizzazione

Per conoscere la posizione della carrozzina all'interno di una mappa in un ambiente indoor, rendendo così possibile lo sviluppo di un controllore che sia in grado di compiere dei movimenti autonomi, è necessario approntare un sistema di localizzazione. Si è scelto di utilizzare un sistema di localizzazione basato su landmark artificiali e tecniche di visione artificiale. Il principio di funzionamento del sistema di localizzazione proposto e il confronto con altri sistemi sono illustrati nel Capitolo 5. Per questo sistema è necessario che sulla carrozzina sia posizionata una telecamera che inquadri il soffitto e che le immagini prodotte dalla telecamera siano acquisite da un computer e analizzate da un opportuno software. Per facilitare l'interfacciamento con il computer e con i software di acquisizione immagini si è scelto di utilizzare una telecamera digitale Fire-I400 prodotta dalla UniBrain[3], visibile in Figura 4.4. Il collegamento con il computer avviene con la porta FireWire (IEEE 1394), da cui la telecamera preleva anche l'alimentazione necessaria. Il CCD da 1/4" della telecamera permette di rilevare immagini a una risoluzione di 640x480 pixel a colori a una frequenza massima di 30 fotogrammi al secondo. Si è scelto di montare sul sistema un'ottica grandangolare con distanza focale di 3.5mm, che permette di inquadrare una zona di circa 1.5x1 metri a una distanza di 1.5 metri.

4.2.4 Computer di bordo

Per raccogliere ed elaborare i dati sensoriali e produrre l'azione di controllo per comandare la carrozzina è necessario disporre di un'unità di calcolo. Dovendo montare il computer a bordo della carrozzina particolare attenzione deve essere posta nella scelta dell'hardware per quanto riguarda dimensioni e consumo di corrente. Per il progetto RAWSEEDS[4] il Politecnico di Mi-

[3] http://www.unibrain.com
[4] http://www.rawseeds.org

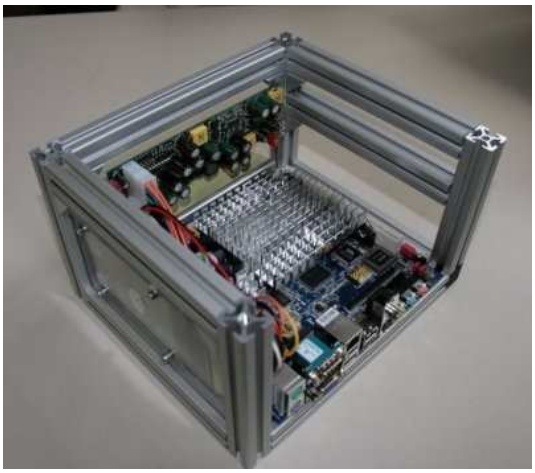


Figura 4.5: Il computer di bordo "PCBrick", montato in una struttura in profilati di alluminio

lano ha assemblato un computer x86 compatibile a cui è stato dato nome PCBrick. È basato su una scheda VIA EN15000 che rispetta lo standard mini-ITX (170x170mm) e, grazie a un ampio dissipatore metallico, necessita solo di una piccola ventola per il raffreddamento che risulta assolutamente silenziosa. Le interfacce messe a disposizione da questa scheda madre sono numerose, tra cui: 6 porte USB 2.0, supporto per una porta FireWire, scheda Audio a 6 canali integrata, connettore Gigabit LAN Ethernet, video VGA e due porte PS/2. Il processore utilizzato è un VIA C7 con frequenza di clock da 1.5GHz. Alla scheda madre sono collegati 1 GB di ram DDR2 e un hard disk da 2.5" da 80GB a 7200 giri al minuto. Il PCBrick è alimentato in corrente continua e, grazie a una scheda DC/DC, accetta tensioni di alimentazioni comprese tra 6V e 24V. La potenza massima richiesta in condizioni di massimo carico del processore è di 25W. La scheda madre e tutti i componenti citati sono montati in una struttura di profilati di alluminio di dimensioni 225x195x135mm. Il sistema operativo installato su PCBrick è Linux, nella distribuzione Gentoo con interfaccia grafica Gnome.

Nell'ambito del progetto RAWSEEDS è stato dimostrato che è semplice e molto efficace utilizzare più PCBrick collegati in rete per gestire il calcolo in parallelo. In fase di progettazione si suppone di utilizzare un solo PCBrick rimandando alla verifica delle prestazioni del sistema la valutazione sulla necessità o meno di incrementare la potenza di calcolo.

4.2.5 Periferiche

Le normali periferiche di input e output di cui dispone un computer sono difficilmente utilizzabili su un veicolo mobile come una carrozzina elettrica. Infatti per utilizzare agevolmente un mouse serve un piano di appoggio, mentre una tastiera risulta eccessivamente ingombrante per essere posizionata su di una carrozzina senza intralciare i movimenti dell'utente. Considerando la categoria di utenti che possono trovare beneficio nell'uso di una carrozzina dalle funzionalità estese è inoltre importante constatare che le normali periferiche potrebbero risultare difficili o addiritura impossibili da utilizzare. Bisogna inoltre considerare che la presenza di un computer a bordo di una carrozzina è principalmente legata alla gestione del moto della carrozzina stessa e al convogliamento e analisi delle informazioni sensoriali. Le operazioni che l'utente deve svolgere con il computer sono molto limitate e riguardano, ad esempio, la selezione di una destinazione per il movimento autonomo. Si è scelto quindi di utilizzare un monitor di piccole dimensioni con funzionalità touch screen e altoparlante integrato. Con un dispositivo di questo tipo è possibile creare interfacce utente semplici, costituite da pochi pulsanti grafici di grandi dimensioni che possono essere selezionate con la pressione di un dito sullo schermo, rendendo del tutto superflua la necessità di utilizzare le classiche periferiche di input. Il monitor scelto per questo scopo è il 700YYV prodotto dalla Xenarc[5]. Ha una diagonale di 7" con risoluzione nativa di 800x480 ed è dotato di un altoparlante integrato. L'alimentazione che il monitor accetta è compresa tra 11 e 24V e il suo consumo di corrente usando l'alimentazione a 24V è di 0.74A.

4.2.6 Dispositivi di comando

Un dispositivo molto semplice che permette di comandare la carrozzina a distanza e quindi, ad esempio, di farla guidare dall'utente anche quando non è a bordo del veicolo o di lasciarla guidare a qualcuno che lo assiste, è il joypad senza fili RumblePad2 prodotto da Logitech[6], visibile in Figura 4.6. Questo dispositivo si interfaccia al computer tramite un modulo di ricezione collegato a una porta USB, ha una portata di circa 10 metri e le batterie che lo alimentano garantiscono un'autonomia di circa 100 ore. Il joypad mette a disposizione dell'utente due controlli analogici costituiti da una levetta mobile su un range angolare di 360 °, uno digitale con quattro frecce direzionali e 12 pulsanti digitali di cui due accessibili con la pressione delle levette analogiche.

[5] http://www.xenarc.com
[6] http://www.logitech.com

Figura 4.6: Il Joypad Logitech RumblePad2 senza fili

La presenza di molti pulsanti e di due leve analogiche e una digitale rende questo joypad molto versatile. Si è scelto di utilizzare solo alcuni dei pulsanti disponibili. In paricolare i pulsanti numerati da 1 a 4 funzionano da selettori della velocità massima raggiungibile. La pressione del pulsante 1 riduce il valore di fondo scala al 40%, il 2 al 60%, il 3 all'80% e il 4 mantiene inalterato il fondo scala. La guida avverrà con uno solo dei controlli disponibili. La selezione di una delle levette analogiche avviene con la pressione del pulsante associato alla leva stessa. Qualora si preme la leva che si sta utilizzando per il comando, il controllo passa ai pulsanti digitali. All'avvio del sistema il controllo è del sistema digitale. Per utilizzare, ad esempio, la leva analogica di sinistra è necessario premerla. Una nuova pressione della stessa leva sposta il controllo al pannello digitale, mentre una pressione della leva destra sposta il controllo sulla leva analogica di destra. In fase di progetto si è preferito non assegnare compiti predefiniti agli altri pulsanti, lasciandoli liberi per funzioni di debug e di supporto allo sviluppo del prototipo.

4.2.7 Derivazione delle alimentazioni

Tutti i dispositivi presentati sono attivi e necessitano di alimentazione elettrica. Alcuni di essi, come la telecamera UniBrain FireI400 o il sensore XSens MTi, prelevano l'alimentazione necessaria al loro funzionamento direttamente dalla porta del computer a cui sono collegati. Altri, come il monitor o i sensori laser, necessitano di una alimentazione separata. La carrozzina, come descritto nella Sezione 4.1.2, è dotata di due batterie da

12V da 70Ah collegate in serie che rendono dunque disponibile una tensione di alimentazione di 24V. L'autonomia garantita dalle batterie permette di alimentare tutti i dispositivi senza bisogno di introdurre batterie aggiuntive. Utilizzando le sole batterie della carrozzina elettrica è possibile utilizzare il caricabatterie fornito con la carrozzina stessa per riportare in efficienza tutto il sistema e si garantisce che finchè la batterie della carrozzina hanno una carica accumulata sufficiente al movimento, anche il sistema che realizza le funzionalità aggiuntive è fuzionante. Utilizzando alimentazioni separate la procedura di carica risulterebbe più complessa e sarebbe difficile garantire che entrambi i sottosistemi siano sempre alimentati correttamente e quindi contemporaneamente funzionanti.

La tensione di 24V può essere prelevata direttamente dalle batterie e distribuita in parallelo al computer e al monitor, che accettano una tensione di alimentazione di 24V. I sensori laser hanno bisogno di un'alimentazione di 5V, per la cui produzione è necessario disporre di un convertitore DC/DC. Si è scelto il dispositivo PRS54-7 prodotto da PowerOne[7] che, prelevando una tensione in input compresa tra 7V e 40V produce un output a 5V con un erogazione massima di corrente pari a 4A e con efficienza circa dell'80%.

Per rendere facilmente accessibili le due tensioni di alimentazione disponibili è opportuno realizzare un punto comune di alimentazione con dei connettori standard. La tensione di 24V viene derivata in parallelo dalle batterie e portata a 4 connettori standard e al convertitore DC/DC. L'uscita a 5V è distribuita in parallelo a 6 connettori. Il collegamento alle batterie è realizzato con un connettore, così da dare la possibilità di scollegare facilmente le batterie e utilizzare un'alimentazione differente come, ad esempio, un alimentatore da banco. È prevista la possibilità di interrompere l'alimentazione con un sezionatore.

La presenza della tensione è segnalata da un led e l'intero sistema è protetto da un fusibile. Il dimensionamento del fusibile deve garantire protezione da guasti e corto circuiti. I sensori laser hanno un consumo di potenza di circa 2.5W ciascuno, che considerando il rendimento dell'80% del convertitore DC/DC, comportano un consumo complessivo di 6.25W sull'alimentazione a 24V. Il computer consuma al massimo 25W e il monitor 18W circa. La potenza complessiva massima richiesta è di circa 50W, ovvero con una tensione di 24V la corrente assorbita supera di poco i 2A. Al fine di evitare rotture accidentali del fusibile si è considerato un fattore di sicurezza di circa 2.5 volte, utilizzando un fusibile da 5A. Lo schema del sistema di derivazione delle alimentazioni è riportato in Figura 4.7.

[7] http://www.power-one.com

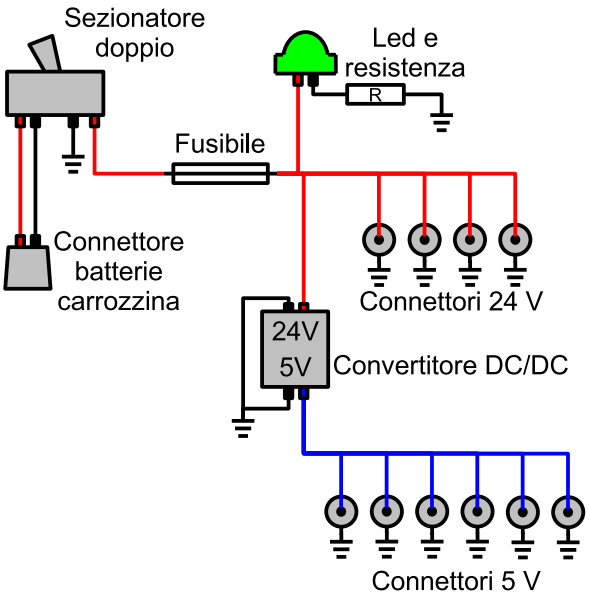


Figura 4.7: Progetto della derivazione delle alimentazioni

4.3 Interfacciare la carrozzina e il computer[8]

Creare un interfaccia che permetta di guidare la carrozzina tramite i comandi inviati da un computer è un passo fondamentale per l'estensione delle funzionalità offerte da un carrozzina elettrica. La possibilità di convogliare le informazioni rilevate dai sensori e di decidere un'azione di controllo con un software residente sul computer permette di realizzare le funzionalità di guida assistita e movimento autonomo di cui si vuole dotare la carrozzina. La presenza di un circuito di interfaccia semplifica anche il collegamento di dispositivi di comando diversi dal joystick della carrozzina: tutti i dispositivi di comando che possono essere collegati alle interfacce standard di un computer o adattati a esse possono prendere parte al controllo del moto della carrozzina. Ad esempio, sarebbe possibile comandare la carrozzina con dispositivi come trackball, joypad, tavolette grafiche e anche software di riconoscimento vocale.

Nei progetti analizzati nella Sezione 2.3 sono stati utilizzati principal-

[8]La progettazione e la realizzazione del circuito di interfaccia tra la carrozzina e il computer è stata svolta in collaborazione con Marco Dalli nell'ambito del corso "Laboratorio di Intelligenza Artificiale e Robotica". Per maggiori dettagli si rimanda a [10] e [9]

mente due metodi per permettere di comandare la carrozzina con un computer. Una delle soluzioni più praticate consiste nel modificare o sostituire il circuito di controllo dei motori della carrozzina e interfacciarsi direttamente ai motori elettrici con uno circuito di controllo realizzato ad hoc. Questa modalità estromette il joystick originale della carrozzina dal controllo nella guida, oppure rende il suo utilizzo mutuamente esclusivo con l'uso delle funzionalità aggiuntive previste, che saranno accessibili solo con l'uso di un diverso dispositivo di comando. Una seconda modalità prevede di interfacciarsi direttamente al bus dati che trasporta le informazioni dal joystick al sistema di controllo dei motori. Questa operazione permette, rispetto alla modifica del circuito di controllo, di creare dei dispositivi compatibili con tutte le carrozzine elettriche con bus dati identico e necessita di modifiche molto meno invasive alla struttura della carrozzina. Per fare ciò è però necessario conoscere quali sono le specifiche elettriche del bus dati e il protocollo che governa lo scambio di informazioni o, qualora queste non fossero disponibili, effettuarne un reverse engineering.

Un'altra possibilità per comandare la carrozzina da un computer è interfacciarsi al bus dati in modo indiretto. Questo è possibile intercettando i valori di tensione analogici generati dalla leva del joystick, qualora questi siano facilmente accessibili. Creando un circuito di interfaccia che produce tali valori di tensione è possibile interfacciarsi al bus dati in modo assolutamente trasparente, in quanto il sistema di controllo rileverà la presenza solo del joystick. Questa soluzione riduce la generalità del circuito di interfaccia, in quanto esso risulta compatibile sono con i modelli di joystick che presentano gli stessi segnali di tensione analogica. Joystick differenti che operano sullo stesso bus dati potrebbero non essere compatibili con il sistema proposto.

La carrozzina su cui è stato effettuato il presente lavoro è equipaggiata con i sistemi di controllo prodotti dalla Dynamic Controls[9] della serie DX System. Le specifiche del bus dati non sono disponibili al pubblico e le richieste di informazioni per motivi di ricerca scientifica non sono state accolte. La Dynamic Controls distribuisce il modulo DX-ACC4 che permette di comandare la carrozzina con 4 segnali, uno per ogni direzione di marcia. Questo dispositivo è venduto a un prezzo piuttosto elevato (alcune centinaia di euro), ma le funzionalità offerte sono molto limitate, in quanto non è possibile regolare la velocità del comando, ma solo impostare un comando digitale (e.g., avanti, indietro). Inoltre il modulo DX-ACC4 non nasce per essere collegato a un computer, quindi è comunque necessario progettare un

[9] http://www.dynamiccontrols.com

circuito di interfaccia. A causa dell'eccessivo costo e delle scarse funzionalità offerte da questo sistema, si è reso evidente che le modalità di interfaccia percorribili per interfacciare un computer e la carrozzina sono ridotte alla sostituzione dei circuiti di controllo dei motori o l'interfacciamento indiretto al bus dati. La prima soluzione presenta notevoli difficoltà in quanto richiede il progetto e la realizzazione di un circuito di potenza per comandare i motori elettrici di una carrozzina. Questa soluzione è difficilmente riutilizzabile su carrozzine diverse e le modifiche da apportare alla carrozina sono numerose. La modifica del joystick permette di interfacciarsi indirettamente al bus dati ed è più praticabile, in quanto non richiede la progettazione di circuiti di potenza e non richiede modifiche sostanziali e particolarmente invasive sulla carrozzina. Un'analisi preliminare della struttura del joystick ci ha confermato che questa soluzione era praticabile ed era effettivamente più vantaggiosa.

4.3.1 Progetto di un circuito di interfaccia

L'analisi del joystick ha messo in luce la presenza di un connettore a 7 piedini, visibile in Figura 4.8, che permette di accedere facilmente ai segnali di tensione della leva. Con l'uso di un oscilloscopio è stato possibile misurare la tensione continua presente sui piedini del connettore, giungendo alle seguenti conclusioni:

- Due piedini hanno la funzione di portare l'alimentazione di 5V e il riferimento di massa alla leva.
- Un piedino non è connesso e questo è facilmente individuabile dalla mancanza di un filo elettrico di collegamento.
- I restanti quattro piedini producono segnali di tensione che descrivono la posizione della leva. Muovendo la leva lungo la direttrice avanti-indietro i valori di tensione che variano sono quelli di due soli piedini, mentre i restanti due sono associati al movimento destra-sinistra. I valori di tensione prodotti sui piedini sono differenziali: infatti, considerando il movimento avanti-indietro e i segnali di tensione associati, quando la leva è in posizione centrale la tensione su entrambi i piedini è pari a 2.5V riferiti a massa. Quando si muove la leva avanti, uno dei due segnali sale fino a 3.75V e l'altro scende fino a 1.25V. Quando si muove la leva indietro il comportamento è opposto, quindi il range massimo di variazione relativa tra i due segnali è di 5V. Il comportamento dei piedini associati al movimento destra-sinistra è analogo.

Figura 4.8: Scheda elettronica del joystick della carrozzina con evidenziato il connettore della leva.

Si noti che la presenza dell'alimentazione di 5V sul connettore permette di alimentare anche il circuito di interfaccia che si intende realizzare.

Scollegando il connettore è possibile creare un circuito di interfaccia che ciclicamente

- "Legge" la posizione della leva e le comunica al computer.
- Interpreta i comandi del computer e li presenta all'altro capo del connettore sotto forma di valori di tensione appropriati.

Per interfacciare la carrozzina al computer è necessario creare una corrispondenza tra i valori di tensione e i valori digitali, di modo che si possa comunicare al computer la posizione della leva, mentre per comandare la carrozzina è necessario convertire i valori digitali in tensioni analogiche appropriate. Considerando queste necessità e la ciclicità con cui le operazioni devono essere svolte si è cercato un microprocessore che potesse fare da base di sviluppo per il circuito e che contenesse il maggior numero di moduli hardware utili all'applicazione da realizzare. I convertitori analogico-digitale (ADC) e l'interfaccia per la comunicazione seriale RS232 sono presenti in molti microprocessori della famiglia PIC prodotta da Microchip[10], mentre nessun microprocessore della famiglia PIC mette a disposizione un numero adeguato di convertitori digitale-analogico (DAC). Si è scelto di utilizzare il microprocessore PIC18F452 e i convertitori digitale-analogico MCP4822, anch'essi prodotti dalla Microchip e perfettamente compatibili con i microprocessori PIC. Utilizzando l'aritmetica nativa di questo processore (a 8

[10] http://www.microchip.com

bit) è possibile descrivere la posizione della leva con valori digitali compresi tra 0 a 255, con una risoluzione nella conversione analogico-digitale di circa 20mV. La risoluzione della conversione digitale-analogica è equivalente. Per l'interfacciamento con il computer è necessario introdurre il chip MAX232 prodotto dalla Maxim[11] che converte i valori di tensione dell'interfaccia seriale disponibile sul PIC (0–5V) a quella adatta al computer (a ±12V). Lo schema elettrico del circuito progettato è riportato in Figura 4.9, per ulteriori dettagli si rimanda a [10] e [9].

In una versione definitiva e stabile della carrozzina autonoma si può supporre che la carrozzina si muova solo quando il computer è acceso e il software di controllo in esecuzione. A livello di prototipo è però consigliabile seguire un approccio più conservativo, che permetta di guidare la carrozzina sempre, anche quando il pc non è acceso o il software di controllo non funzioni a dovere. Si è quindi scelto di dotare il circuito di due modalità di funzionamento: *manuale* e *automatica*. Nella modalità manuale è possibile guidare la carrozzina con il normale joystick, quindi la presenza del circuito è del tutto trasparente e il computer non può in alcun modo controllare la carrozzina. Nella modalità automatica è invece il computer a controllare il movimento della carrozzina. La modalità in uso è comunicata tramite due led di colore diverso e il cambio della modalità operativa si effettua tramite la pressione di un pulsante.

All'accensione del circuito, che corrisponde all'accensione della carrozzina elettrica, la modalità impostata è quella manuale. Il passaggio alla modalità automatica è effettuabile solo azionando il pulsante predisposto, mentre il passaggio alla modalità manuale può avvenire sia con la pressione dello stesso pulsante, sia tramite la richiesta del computer o qualora il computer non invii al circuito comandi per un tempo superiore a un timeout prefissato. In questo modo si garantisce un minimo livello di sicurezza: se si passa alla modalità automatica, ma il computer non è acceso o non comunica correttamente i comandi, dopo un breve periodo di tempo la carrozzina viene automaticamente riportata in modalità manuale. Allo stesso modo si garantisce che, senza la volontà dell'utente, il computer non possa prendere il controllo del moto della carrozzina.

Il software da realizzare per il microprocessore PIC segue il seguente flusso:

1. Leggere i valori in ingresso agli ADC e sottrarli a coppie per ottenere due valori assoluti che rappresentano la posizione assoluta della leva lungo le due direttrici.

[11] http://www.maxim-ic.com

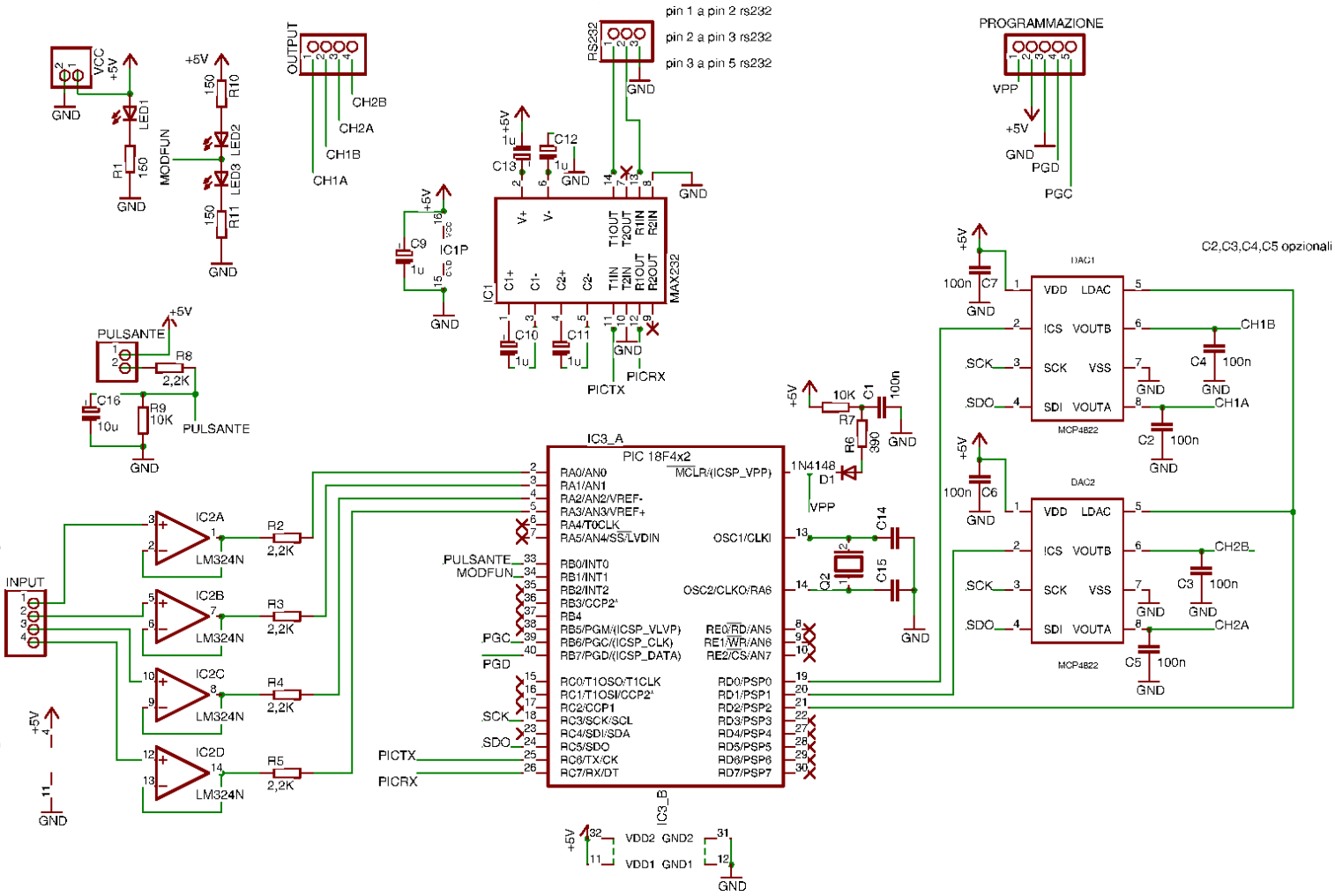


Figura 4.9: Circuito di interfaccia tra joystick della carrozzina e computer

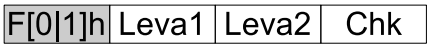


Figura 4.10: Trama dei messaggi scambiati tra il computer e il circuito.

2. Comunicare al PC i valori calcolati al punto 1.
3. Se la modalità è automatica, porre in uscita l'ultimo comando ricevuto dal PC.
4. Se la modalità è manuale, porre in uscita i valori calcolati al punto 1.
5. Leggere i dati eventualmente comunicati dal PC.
6. Controllare la scadenza del timeout di validità dei comandi.
7. Controllare eventuali pressioni del pulsante.
8. Aggiornare la modalità di funzionamento in base agli eventi rilevati ai punti 5, 6 e 7.
9. Ripetere dal punto 1.

Il ciclo è ripetuto ogni 20ms, quindi con una frequenza di 50Hz. Questa frequenza garantisce di rilevare tutti i movimenti imposti sulla leva dall'utente, che, come è stato accertato tramite prove sperimentali, non superano una frequenza di 10Hz e allo stesso modo permettono un controllo fluido da parte del computer.

4.3.2 Protocollo di comunicazione

I dati che devono scambiarsi il circuito e il computer di controllo sono:

- Posizione della leva sul primo asse.
- Posizione della leva sul secondo asse.
- Modalità di funzionamento.

Le informazioni sono le stesse nei due versi, ma la loro interpretazione è differente: i dati trasmessi dal computer al circuito hanno il significato di comando, mentre quelli trasmessi dal circuito verso il computer hanno scopo informativo.

La trama prevista per lo scambio dei i dati è molto semplice e composta da 4 byte (Figura 4.10). Il primo è un campo che svolge la funzione di sincronizzazione e allo stesso tempo comunica la modalità di funzionamento. I

valori ammissibili per questo campo sono F0h, che indica la modalità manuale e F1h, che indica la modalità automatica. Il secondo e il terzo byte indicano la posizione della leva rispettivamente dell'asse avanti-indietro e destra-sinistra del joystick o il comando per la carrozzina, in base al verso in cui fluisce la trama. I valori sono interi senza segno compresi tra 0 e 255, con il valore centrale (128) che corrisponde allo "zero logico", ovvero alla condizione in cui la leva è in posizione centrale. Il quarto byte è un campo di controllo ed è calcolato come somma a modulo 255 dei tre byte iniziali del messaggio. Se la verifica del valore del checksum fallisce il circuito rifiuta il comando del computer. Un comportamento analogo si può ottenere sul computer con i dati trasmessi dal circuito. Si noti che, in base alle specifiche individuate per il cambio della modalità di funzionamento, il circuito ignora tutte le trame che sono trasmesse dal computer quando è in modalità manuale. Se la modalità di funzionamento è automatica, l'invio da parte del computer di una trama che specifica il funzionamento manuale è interpretata come la volontà di passare al funzionamento manuale. Se la modalità è manuale, il passaggio alla modalità automatica è possibile solo su richiesta esplicita dell'utente.

Per ulteriori informazioni sul funzionamento del circuito e sulle modifiche da apportare al joystick della carrozzina si rimanda a [10] e [9].

4.4 Progetto di una struttura di supporto

Tutti i componenti necessari alla realizzazione di una carrozzina elettrica dalle funzionalità estese individuati nella Sezione 4.2 devono essere collocati a bordo della carrozzina stessa. Montare singolarmente ogni singolo componente sulla carrozzina è scomodo e qualora fosse necessario riportare la carrozzina in condizioni originali o spostare i componenti su una nuova carrozzina, sarebbe necessario scollegare ogni singolo componente e smontarlo. È molto più comodo progettare una struttura di supporto che sia applicabile alla carrozzina in modo semplice e con poche operazioni. Tale struttura deve essere progettata in funzione della forma e delle possibilità di fissaggio offerte dalla carrozzina stessa. I criteri che guidano il progetto di questo struttura sono:

Comodità per l'utente. La struttura non deve modificare la postura dell'utente e non deve intralciare l'accesso alla carrozzina.

Semplicità di montaggio. La struttura deve essere facilmente applicabile e rimovibile per permettere di riportare alle condizioni originali la carrozzina. Questa caratteristica garantisce che nel caso siano necessari

interventi di manutenzione essi non siano complicati eccessivamente dalla presenza di una struttura aggiuntiva. La semplicità di montaggio riguarda anche il fissaggio dei componenti alla struttura. Più la struttura offre possibilità di posizionare liberamente sensori e apparati hardware su di essa, più sarà facile posizionare nel modo più appropriato i componenti previsti e aggiungere nuovi componenti qualora si rivelino necessari.

Considerando che la carrozzina è piuttosto ingombrante e che deve essere manovrabile in spazi ristretti è necessario fare in modo che tutto ciò che si monta sulla carrozzina non sporga dalla carrozzina stessa, almeno per quanto riguarda la larghezza e la profondità, per non intralciarne il movimento. In altezza è possibile eccedere in quanto questo non crea problemi nel normale movimento. La larghezza massima della carrozzina, rilevata in prossimità delle ruote posteriori, è di circa 0.6m e la lunghezza, considerando anche i poggiapiedi anteriori, è circa di 1m.

Tra i componenti individuati nella Sezione 4.2 gli unici che devono essere posizionati in modo particolare sono i due scanner laser, la telecamera e il monitor touch screen. La posizione e l'orientamento dei laser influisce molto sugli ostacoli che essi possono rilevare. Se si posizionano i laser vicino a parti della carrozzina o vicino al corpo dell'utente, il range angolare non occluso può risultare molto ridotto. Anche per quanto riguarda la telecamera, che come detto deve inquadrare il soffitto, la possibilità di occlusioni varia in base al suo posizionamento. Il monitor touch screen deve essere posizionato in modo da essere ben visibile e raggiungibile dalla posizione di guida. Tutti gli altri componenti sono da fissare alla struttura solo per questioni di utilità, ovvero per renderne agevole il montaggio e la rimozione.

4.4.1 Posizionamento dei laser

I sensori laser scandiscono una zona planare di circa 240 ° e, considerando di montarli parallelamente al terreno su cui si muove la carrozzina, per garantire di rilevare il maggior numero di ostacoli è necessario che essi siano il più in basso possibile, compatibilmente con la struttura della carrozzina. Particolare attenzione deve essere posta nel garantire che la carrozzina stessa o la presenza del corpo dell'utente non ostruisca le rilevazioni. Per diminuire ulteriormente l'altezza da terra dei sensori si è ritenuto opportuno montarli con la base rivolta verso l'alto.

Montando i laser in prossimità delle ginocchia dell'utente è possibile garantire che la misura effettuata con i laser non sarà disturbata dalla presenza dell'utente stesso e della struttura della carrozzina. Considerando che il ran-

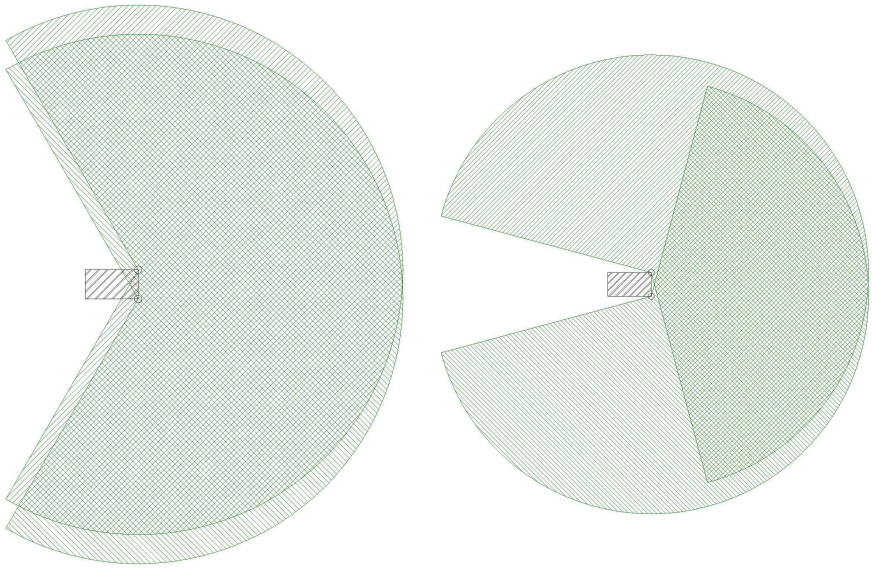


Figura 4.11: Confronto tra le zona coperta dalla scansione dei sensori laser Hokuyo URG-04LX in due configurazioni differenti.

ge angolare di scansione è di 240 °, si è preferito ruotare i sensori in modo da evitare di misurare inutilmente distanze in zone che potrebbero essere ostruite dalla presenza dell'utente o dalla struttura delle carrozzina e allo stesso tempo garantire di coprire il più possibile anche le zone laterali. La distanza tra i laser è imposta dalla larghezza della carrozzina nella zona in cui i laser sono posizionati, ed è pari a circa 550mm. Un confronto tra la zona di scansione ottenuta utilizzando i sensori con lo zero rivolto in direzione frontale rispetto alla carrozzina e utilizzando i laser ruotati di 45 ° e −45 ° è mostrato in Figura 4.11. I vantaggi dati dal posizionamento dei laser ruotati non riguardano solo la maggiore zona di copertura, ma anche la sovrapposizione della zona di scansione centrale, che permette, qualora un laser fallisca una scansione, di avere una sicurezza data dalla presenza dell'altro laser. È evidente che la zona retrostante alla carrozzina e le zone laterali posteriori non sono per nulla coperte dalla scansione. Considerando però che la direzione principale di marcia consiste in movimenti frontali il problema di coprire la zona posteriore con sensori per il rilevamento ostacoli non viene trattato in questo lavoro, ma rimandato eventualmente a sviluppi successivi.

Per garantire che urti accidentali non danneggino i sensori laser si è

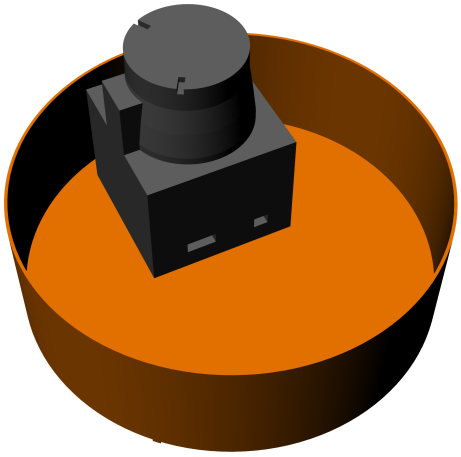


Figura 4.12: Progetto della struttura di protezione del sensore laser Hokuyo URG-04LX.

progettata una struttura di protezione che accoglie il sensore e i cavi di alimentazione e trasferimento dati. È significativo considerare che i danni al sensore possono essere provocati anche da urti indiretti, come nel caso di torsioni dei cavi che si ripercuotono sui sensori stessi. La progettazione e la successiva realizzazione di una struttura di protezione per i sensori laser è stata semplificata dall'uso di un tappo a vite in PVC per tubi dell'acqua comunemente utilizzato in edilizia. Il diametro esterno del tappo è di circa 135mm e internamente presenta una superficie piatta su cui è possibile montare il sensore laser e fissare saldamente i cavi. Eventuali urti accidentali vanno a impattare sulla robusta parete verticale del tappo, evitando di torcere e piegare i cavi o di danneggiare direttamente il sensore. Il progetto del posizionamento del sensore all'interno del tappo a vite è mostrato in Figura 4.12. Si noti che il sensore non è posizionato al centro del piano, ma di lato, per permettere di alloggiare comodamente il cavo USB e il cavo di alimentazione.

4.4.2 Posizionamento della telecamera

La telecamera deve inquadrare il soffitto, ma non è sufficiente posizionarla in modo che il suo obiettivo sia direzionato verso l'alto. Infatti, se la telecamera viene posizionata in basso sulla carrozzina, è possibile che la presenza di persone nell'ambiente in cui si muove la carrozzina ostruisca in parte o completamente la zona inquadrata. Per minimizzare questo problema è necessario posizionare la telecamera a un'altezza di circa 1.5m dal suolo, così da minimizzare la possibilità di occlusioni. La telecamera risultà così più

vicina al soffitto da inquadrare, garantendo una migliore definizione dell'immagine a scapito però di un minor campo visivo. Una struttura di questa altezza per montare la telecamera garantisce comunque il passaggio agevole della carrozzina negli ambienti, in quanto tipicamente i soffitti delle normali abitazioni sono alti almeno 2.5m e le porte almeno 2m.

4.4.3 Progetto della struttura

Si è scelto di costruire la struttura di supporto per i componenti da aggiungere alla carrozzina con i profilati di alluminio prodotti dalla ditta Item[12], in quanto offrono un vasto assortimento di componenti per il montaggio. La disponibilità dei modelli CAD dei prodotti ha permesso di progettare dettagliatamente la struttura prima della sua realizzazione. La struttura non deve supportare carichi gravosi, si è scelto quindi di utilizzare i profilati di sezione 20x20mm e i relativi componenti di fissaggio (squadrette angolari, viti e tasselli specifici,etc.).

La carrozzina dispone di una zona posteriore libera, retrostante al sedile del guidatore, su cui è possibile montare una struttura di supporto. La presenza del telaio in acciaio della carrozzina permette di fissare facilmente la struttura. Nella spiegazione del progetto della struttura di supporto si fa riferimento alla Figura 4.13 e alle sue parti identificate da numeri.

Si è utilizzato un singolo profilato di alluminio per la base (1) di dimensioni 80x600x20mm, a cui sono sono stati collegati quattro profilati di lunghezza 1060mm (2) per formare un parallelepipedo a sviluppo verticale con misure di 1060x80x490mm. Il parallelepipedo è chiuso superiormente da due profilati da 80mm (3) e da due di 490mm (4). La telecamera deve essere posizionata in cima a questa struttura, in posizione centrale (5). La larghezza del parallelepipedo è stata stabilita in base alla larghezza del sedile, di modo che fosse possibile far passare dei profilati orizzontali su cui fissare i sensori laser circa all'altezza delle ginocchia dell'utente. Si sono utilizzati due profilati da 600mm per ogni lato. Il primo profilato (6) è fissato a uno degli spigoli del parallelepipedo, mentre il secondo (7) è parzialmente sovrapposto al primo profilato. La sovrapposizione parziale permette di regolare la lunghezza complessiva del sistema costituito dai due profilati, rendendo possibile modificare facilmente la posizione dei sensori laser (8), che sono montati in fondo all'ultimo tratto del profilato inseriti nel loro supporto protettivo. Il monitor (9) può essere posizionato indifferentemente a destra o a sinistra dell'utente su uno dei profilati che supportano i laser.

Per dare rigidità alla struttura si è utilizzato un altro profilato da 490mm

[12] http://www.item.info

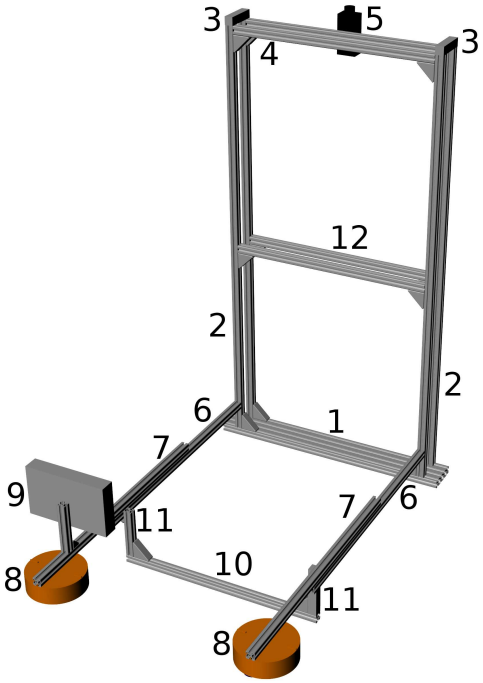


Figura 4.13: Progetto CAD della struttura di supporto in profilati di alluminio

(10) che collega le due barre orizzontali dei sensori laser con due profilati verticali di 120mm (11). Quest'ultima parte della strutturra è bloccata al supporto per i poggiapiedi anteriori della carrozzina e garantisce che la struttura non possa compiere oscillazioni che potrebbero compromettere la buona lettura delle informazioni sensoriali dei laser e della telecamera.

Tutti gli altri componenti che devono essere alloggiati sulla struttura sono applicabili alla struttura progettata, ma il loro posizionamento non è stato studiato a priori, lasciando a considerazioni pratiche la scelta di posizionamento. Si noti che è possibile inserire all'interno del parallelepipedo ulteriori profilati da 490mm che svolgono la funzione di "mensole" di supporto per alloggiare quanto necessario (12). Si prevede di alloggiare il sensore inerziale e giroscopico XSens MTi su una di queste mensole in posizione centrale e di fissare il computer di bordo alla base della struttura.

La struttura descritta è stata progettata in modo da avere un impatto trascurabile sull'accesso al sedile di guida, sulla comodità e sul comfort dell'utente.

4.5 Software di controllo

Il compito di un software di controllo di robot mobili è quello di integrare le informazioni disponibili e scegliere i comandi da inviare al sistema di attuazione per perseguire degli obiettivi. Ad esempio, utilizzando le informazioni sugli ostacoli presenti, il software di controllo deve proporre valori di attuazione tali da evitare le collisioni o evitare gli ostacoli seguendo un percorso alternativo. Nella Sezione 3.4 sono state descritte alcune architetture proposte il letteratura per strutturare controllori robotici. In particolare nella Sezione 3.6 è stata presentata l'architettura di controllo basata sulla composizione di comportamenti proposta in [3] e [4]. Tale architettura, basata sulla definizione di comportamenti fuzzy, permette di definire facilmente il sistema che governa il controllo del robot, garantendo un elevato grado di flessibilità per estensioni successive. La disponibilità del codice sorgente del controllore, la presenza di ampia documentazione e l'elevato grado di stabilità raggiunto grazie all'uso in numerosi progetti (tra cui Milan Robocup Team[13] e [45]), hanno portato a scegliere questa architettura per il software di controllo della carrozzina.

Prima di progettare e realizzare il software che effettua il controllo della carrozzina è necessario analizzare quali sono i compiti specifici, almeno ad alto livello, a cui esso deve assolvere. Innanzitutto il software deve comunicare con il circuito di interfaccia in modo da comandare la carrozzina e acquisire la posizione della leva. Anche i comandi imposti con il joypad devono essere recepiti e interpretati in modo da comandare correttamente la carrozzina. Per quanto riguarda i dati rilevati dai sensori, la sola acquisizione spesso non è sufficiente, ma sono necessarie fasi di analisi dei dati o di modellizzazione. In particolare, per le immagini rilevate con la telecamera è necessario svolgere l'analisi dell'immagine per rilevare il marker e calcolare la posizione della carrozzina nel sistema di riferimento assoluto, mentre per i dati provenienti dai sensori laser è necessario produrre un modello sintetico che permetta di riconoscere facilmente la presenza degli ostacoli. Un altro compito molto importante per poter sviluppare un sistema di guida autonoma della carrozzina è la possibilità di pianificare percorsi su una mappa, per la quale è necessario disporre di un modulo di pianificazione. Inoltre il software deve assolvere ad altri compiti di secondaria importanza per quanto riguarda il controllo, ma utili per il debugging e lo sviluppo del sistema, come la produzione di log o la visualizzazione on-line del comportamento del sistema.

A causa nella numerosità e della diversità dei compiti che il software deve

[13] http://robocup.elet.polimi.it/MRT

svolegere è indispensabile prevedere che alcune operazioni siano eseguite in parallelo. Sarebbe impossibile infatti gestire tutte le operazioni individuate in modo rigido con un ciclo di polling. Per garantire l'esecuzione parallela è necessario sviluppare un'applicazione multi-processo o multi-thread. La scelta in fase progettuale è caduta sui thread ed è dettata soprattutto dall'eccessivo overhead di memoria per la gestione dei processi. La necessità di far comunicare i vari thread può essere gestita in diversi modi. Il più semplice, preso in considerazione in una prima fase, è costituito dal meccanismo "a lavagna", in cui ogni modulo scrive le sue informazioni in uno spazio di memoria condivisa, gestendo opportunamente la sincronizzazione tra gli accessi. Questa tecnica non permette però di gestire facilmente la separazione del software su più unità di calcolo, mentre un meccanismo basato, ad esempio, sullo scambio di messaggi TCP-IP lo renderebbe più agevole. Si è scelto di utilizzare un framework software, chiamato DCDT, che è stato studiato appositamente per realizzare applicazioni multiagente basate sui thread con comunicazione basata su messaggi TCP-IP.

Il Capitolo 6 si occuperà interamente della progettazione e dell'architettura del software di controllo, in questo momento è sufficiente introdurre il modulo DCDT che è alla base, insieme a Mr.BRIAN, della progettazione e della realizzazione dell'intera architettura software.

4.5.1 DCDT

Nell'ambito del progetto Milan Robocup Team è stato utilizzato DCDT (Device Communities Development Toolkit) [33], un framework software che facilita l'implementazione di agenti software che possono comunicare e scambiarsi informazioni. DCDT si occupa anche dell'esecuzione degli agenti, infatti ogni agente creato in DCDT, detto *membro*, è un thread che viene eseguito all'interno di una *Agorà*, ovvero un processo che raccoglie più membri e ne gestisce l'esecuzione. 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 perido 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.

La comunicazione tra agenti è realizzata mediante pacchetti TCP/IP e può avvenire sia all'interno dell'Agorà sia tra diverse Agorà residenti su macchine differenti. 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. È 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.

Capitolo 5

Localizzazione con landmark artificiali

In questo capitolo si introduce un sistema di localizzazione per ambienti indoor basato su telecamere e landmark artificiali. Sono illustrati sia gli aspetti legati alla progettazione logica che all'effettiva realizzazione e implementazione del sistema. Le problematiche riscontrate nello sviluppo hanno comportato una serie di modifiche al progetto iniziale. Tali modifiche sono presentate nell'ordine in cui sono state affrontate.

5.1 Approcci alla localizzazione indoor

Un sistema di localizzazione è 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. Altre proprietà desiderabili sono la facilità di installazione e d'uso oltre alla scalabilità su vaste zone. Un esempio di sistema di localizzazione disponibile su scala mondiale è il GPS, ma il suo segnale non è in genere rilevabile all'interno degli edifici, quindi tale sistema non è adatto alla 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 [48] [49] [50] e StarGazer [19], presentati nei dettagli rispettivamente nelle Sezioni 3.2.2 e 3.2.3. 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 alimenta-

zione 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 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 [47]. 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, come ad esempio in [39].

5.2 Fiducial Marker

I sistemi basati su fiducial marker sono utilizzati in svariati ambiti, dalla robotica [15] alle applicazioni di realtà aumentata [2] [17]. Un marker è solitamente costituito da una superficie planare di forma nota con un contenuto informativo. In questo lavoro sono stati presi in considerazione i sistemi ARToolKit [22], ARTag e ARToolKitPlus [51]. 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

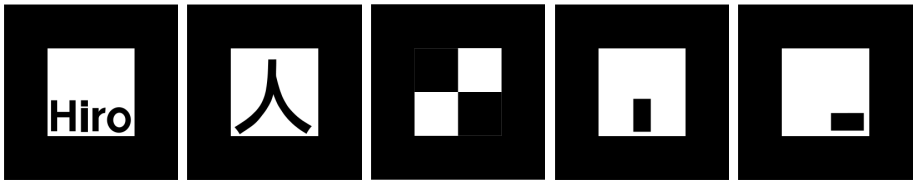


Figura 5.1: Pattern base di ARToolKit

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.

In [16] sono confrontati due sistemi di riconoscimento di fiducial marker: ARTag e ARToolKitPlus. Predecessore di entrambi è ARToolKit che è stato largamente utilizzato in sistemi di realtà aumentata e interfacce per l'interazione tra l'uomo e il computer (*Human Computer Interaction, HCI*), come in [52]. Per questo motivo è utile cominciare la descrizione dei sistemi partendo da ARToolKit stesso. La presenza del prefisso *AR* nel nome di tutti i sistemi citati indica che il loro sviluppo è stato principalmente legato a studi di realtà aumentata (*Augmented Reality*), come in [2] e [17].

5.2.1 ARToolKit

I marker di ARToolKit sono di forma quadrata con un marcato bordo nero. 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 sogliatura 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. L'aggiunta di pattern personalizzati è possibile registrando un certo numero di immagini del nuovo pattern, che costituiranno la base per il confronto. Alcuni pattern base forniti da ARToolKit sono visibili in Figura 5.1.

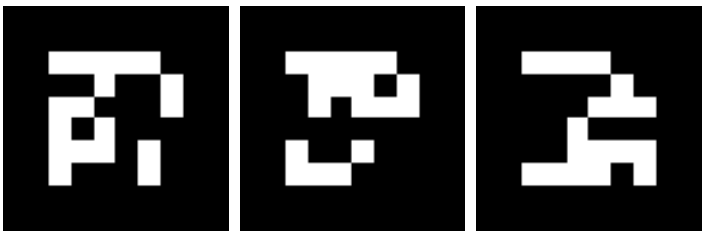


Figura 5.2: Alcuni pattern di ARTag

ARToolKit presenta due grossi problemi che ne limitano la precisione e il campo d'uso:

- Il meccanismo di estrazione dell'immagine binaria basato su singola soglia non si adatta a condizioni di luce variabile.
- L'uso della correlazione come unico meccanismo di identificazione del pattern comporta numerosi errori nel riconoscimento dei pattern, sia in termini di falsi positivi sia di riconoscimento errato del contenuto informativo del marker.

A questi si aggiunge l'imprecisione della procedura per la creazione di nuovi tag e la scarsa disponibilità di tag di base. Un'altra limitazione è rappresentata dall'impossibilità di utilizzare strumenti standard per la calibrazione della telecamera (e.g., Camera Calibration Toolbox for Matlab[1]), ma di doversi affidare a un software proprietario.

5.2.2 ARTag

ARTag usa marker quadrati con bordo nero e contenuto informativo basato sul codice a barre bidimensionale Datamatrix[2] (alcuni pattern sono visibili in Figura 5.2). Rispetto ad ARToolKit usa un diverso sistema di identificazione del bordo del marker, non più basanto su un meccanismo a singola soglia, ma su rilevazione di contorni. ARTag è più versatile in condizioni di luce variabile e riconosce correttamente il marker anche in presenza di occlusioni parziali dei bordi. L'uso di un codice digitale per la specifica del contenuto informativo del pattern riduce sensibilmente il numero di errori nel riconoscimento dei pattern e rende disponibile un elevato numero di tag differenti.

[1] http://www.vision.caltech.edu/bouguetj/calib_doc
[2] standard ISO/IEC16022

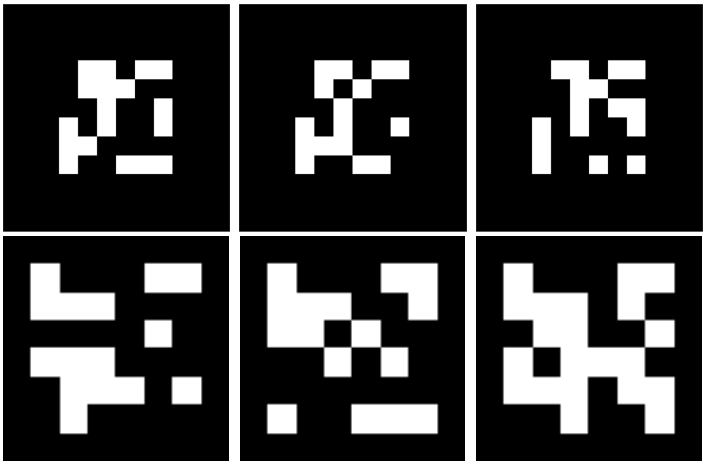


Figura 5.3: Pattern di ARToolKitPlus, tre di tipo simple ID e tre BCH ID

5.2.3 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. 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 come nel caso di ARTag). I marker predefiniti con contenuto digitale sono di due tipi: *simple ID* con 512 marker disponibili e *BCH ID* con 4096 marker (alcuni esempi sono riportati in Figura 5.3).

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 matri-

ce di rototraslazione (*Robust Planar Pose Tracking, RPP*) che porta dal sistema di riferimento telecamera al sistema di riferimento tag.

ARToolKitPlus è portabile anche su dispositivi mobili come palmari e telefoni cellulari, grazie anche alla possibilità di eseguire i calcoli in virgola fissa. L'algoritmo RPP non è disponibile sui dispositivi mobili, in quanto richiede una precisione di calcolo non raggiungibile con sistemi a virgola fissa. Attualmente è in sviluppo un nuovo strumento, nominato Studierstube Tracker[3], che ha molte funzionalità in più rispetto ad ARToolKitPlus. Purtroppo questo tool non è attualmente disponibile al pubblico.

5.2.4 Localizzazione con fiducial marker

L'obiettivo della localizzazione è quello di conoscere, con una certa affidabilità e precisione, la posizione di un robot, o più in generale di un oggetto, rispetto a un sistema di riferimento assoluto. D'altra parte i tre sistemi bassati su fiducial marker presentati permettono, data un immagine in cui è visibile un marker, di conoscere la posizione e l'orientamento di tale marker in 6 gradi di libertà (*degree of freedom*, DOF) nel sistema di riferimento telecamera. Montando in posizione fissa la telecamera e posizionando i marker sull'oggetto mobile, i sistemi basati su fiducial marker permettono di trovare direttamente la posizione dell'oggetto. Purtroppo questa soluzione risulta poco scalabile, infatti, se l'ambiente da coprire è molto vasto e diviso in zone è necessario utilizzare più telecamere per mantenere sempre in vista l'oggetto mobile. L'uso di più telecamere comporta costi aggiuntivi e un notevole sforzo di calibrazione del sistema per ottenere le relazioni tra le posizioni delle singole telecamere e un unico sistema di riferimento assoluto.

Per coprire zone vaste è più vantaggioso utilizzare una telecamera mobile e posizionare i marker nell'ambiente. Il costo dei marker è molto basso e in seguito saranno illustrate alcune procedure automatiche per la calibrazione del sistema, ovvero per la creazione delle relazioni tra posizione dei marker e sistema di riferimento assoluto.

5.3 Posizione assoluta con telecamera mobile

Supponendo di disporre di marker fissi nell'ambiente e di una telecamera posta su un oggetto mobile, i sistemi di fiducial marker permettono di conoscere la posizione e l'orientamento del marker nel sistema di riferimento della telecamera. Per risalire alla posizione e all'orientamento dell'oggetto

[3] http://studierstube.icg.tu-graz.ac.at/handheld_ar/stbtracker.php

mobile su cui è montata la telecamera in un sistema di riferimento assoluto è necessario:

- Conoscere la posizione e l'orientamento di ogni marker rispetto al sistema di riferimento assoluto.
- Conoscere la posizione e l'orientamento della telecamera rispetto a un punto fissato dell'oggetto mobile (e.g. il baricentro, il centro cinematico del robot, etc.).

Queste informazioni sono sufficienti per calcolare la posizione e l'orientamento dell'oggetto mobile rispetto al sistema di riferimento assoluto in un qualsiasi istante in cui in un'immagine ripresa dalla telecamera sia visibile almeno un marker. Da tale immagine si ricavano il codice identificativo del marker visibile (i) e la matrice $T^C_{M_i}$ di rototraslazione che descrive la posizione e l'orientamento del marker i nel sistema di riferimento solidale con la telecamera.

È possibile calcolare la relazione $T_C^{M_i}$ che esprime la posizione e l'orientamento della telecamera nel sistema di riferimento solidale con il marker i tramite inversione della matrice di rototraslazione di partenza:

$$T_C^{M_i} = (T^C_{M_i})^{-1} \tag{5.1}$$

La relazione che lega il sistema di riferimento solidale con il marker i al sistema di riferimento assoluto, identificata da $T^W_{M_i}$, permette di calcolare la posizione e l'orientamento della telecamera nel sistema di riferimento assoluto (T^W_C):

$$T^W_C = T^W_{M_i} \cdot T_C^{M_i} \tag{5.2}$$

Come ultimo passaggio è possibile conoscere la posizione assoluta del sistema di riferimento scelto sull'oggetto grazie alla relazione T^C_R che descrive la posizione e l'orientamento del sistema di riferimento posto sull'oggetto rispetto alla telecamera:

$$T^W_R = T^W_C \cdot T^C_R \tag{5.3}$$

Sostituendo in 5.3 le Equazioni 5.2 e 5.1 si ottiene

$$T^W_R = T^W_{M_i}(T^C_{M_i})^{-1} \cdot T^C_R \tag{5.4}$$

Il primo e l'ultimo termine dell'equazione di cui sopra sono costanti, in quanto stabiliscono relazioni rigide tra sistemi di riferimento. Il termine centrale varia in base al posizionamento relativo di telecamera e marker i-esimo.

L'Equazione 5.4 permette dunque di conoscere la posizione assoluta di un oggetto mobile in base a:

- Ccodice identificativo del marker visibile in un immagine.
- Matrice di rototraslazione che ne descrive la posizione e l'orientamento rispetto alla telecamera rilevata dal sistema di fiducial marker.
- Informazioni statiche memorizzate in fase di calibrazione del sistema.

5.3.1 Procedura di localizzazione

L'algoritmo che permette di analizzare un immagine acquisita da una telecamera per calcolare la posizione dell'oggetto mobile su cui è montata la telecamera è quindi il seguente:

1. Acquisizione di un'immagine dalla telecamera.
2. Processo di analisi svolto dal sistema di fiducial marker.
   Se non vengono rilevati marker nell'immagine comunica l'impossibilità di stabilire la posizione. Altrimenti memorizza la matrice $T^C_{M_i}$ e l'identificativo i del marker.
3. Controlla se la matrice $T^W_{M_i}$ è disponibile tra quelle memorizzate nel sistema. Se non è disponibile comunica l'impossibilitá di stabilire la posizione. Altrimenti calcola T^W_R (Equazione 5.3).

5.4 Relazioni tra sistemi di riferimento

Nella sezione precedente si è supposto che le relazioni $T^W_{M_i}$ fossero note. Queste relazioni sono in genere a sei gradi di libertà: tre per la traslazione che avviene tra i sistemi di riferimento e tre per la rotazione intorno agli assi. Misurare manualmente questi parametri in modo preciso non è banale. Una semplificazione si può effettuare posizionando i marker secondo uno schema geometrico molto preciso. Utilizzando, ad esempio, una griglia che mantiene i marker a distanze prefissate e orientamento noto è possibile risalire facilmente alla posizione e rotazione di ogni marker rispetto a un sistema di riferimento assoluto. Un approccio di questo tipo è però difficilmente applicabile a un ambiente non strutturato, in quanto si potrebbero verificare casi in cui la posizione dei marker deve adattarsi alla forma e alle strutture presenti nell'ambiente (e.g. una porta che passa da un locale ad un altro, un livello di altezza differente del soffitto). È necessario dunque garantire una certa flessibilità nel posizionamento dei marker e allo stesso tempo semplicità nella creazione delle relazioni con il sistema di riferimento assoluto.

5.4.1 Relazioni dirette

Sfruttando il sistema di fiducial marker è possibile creare una catena di relazioni successive che permette di risalire alla relazione di ciascuno di essi con il sistema di riferimento assoluto basandosi sulla conoscenza della relazione tra un solo marker di base e il sistema di riferimento assoluto (eventualmente coincidente con il sistema di riferimento solidale con il marker di base stesso).

Se si posizionano il marker i e il marker j in modo che sia possibile inquadrare entrambi in una stessa immagine, è possibile ottenere dal sistema di fiducial marker sia la matrice $T^{C}_{M_i}$ sia la matrice $T^{C}_{M_j}$. La relazione tra il marker i e il marker j è quindi data da:

$$T^{M_i}_{M_j} = (T^{C}_{M_i})^{-1} \cdot T^{C}_{M_j} \tag{5.5}$$

Supponendo di conoscere un'unica relazione tra il sistema di riferimento assoluto e il sistema solidale con il marker i è possibile calcolare la relazione tra il sistema di riferimento assoluto e un qualunque marker j che possa essere inquadrato insieme al marker i tramite la seguente equazione:

$$T^{W}_{M_j} = T^{W}_{M_i} \cdot T^{M_i}_{M_j} \tag{5.6}$$

5.4.2 Relazioni indirette

Per estendere la zona coperta dal sistema di localizzazione è inevitabile che si verifichi la condizione in cui alcuni marker non possono essere inquadrati insieme al marker di base i. È però possibile posizionare tali marker in modo che essi possano essere inquadrati almeno insieme ad un altro marker e ricorsivamente costruire le relazioni che portano fino al marker base. Supponendo di disporre di un immagine in cui sono visibili il marker j e il marker k, di una seconda immagine contenente il marker i e il marker j e della relazione $T^{W}_{M_i}$ è possibile seguire il seguente procedimento:

1. Calcolare $T^{M_j}_{M_k}$ con l'Equazione 5.5 dalla prima immagine.
2. Calcolare $T^{M_i}_{M_j}$ con l'Equazione 5.5 dalla seconda immagine.
3. Calcolare $T^{W}_{M_j}$ con l'Equazione 5.6
4. Calcolare $T^{W}_{M_k}$ con l'Equazione 5.6 come

$$T^{W}_{M_k} = T^{W}_{M_j} \cdot T^{M_j}_{M_k} \tag{5.7}$$

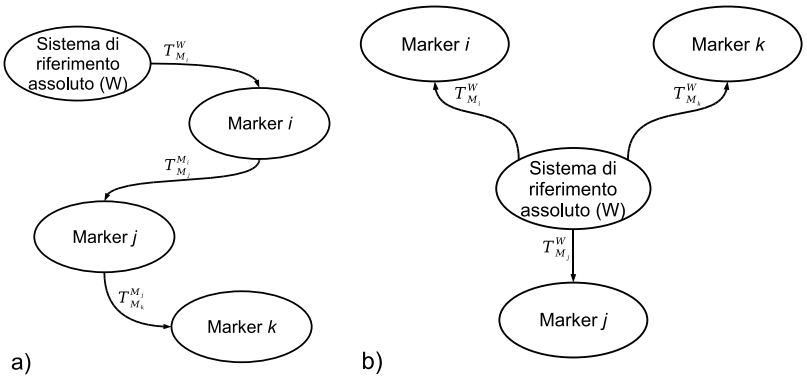


Figura 5.4: a) Albero delle relazioni indirette; b) riduzione a stella

Iterando questo procedimento è possibile in linea di principio coprire zone di vaste dimensioni e creare in modo automatico la mappa dei marker nel sistema di riferimento assoluto. La struttura minima che descrive le relazioni indirette che permettono di giungere al sistema di riferimento assoluto è quella di un albero, ovvero di un grafo aciclico in cui due nodi sono connessi esattamente da un cammino. Con l'applicazione ricorsiva delle equazioni descritte è possibile ridurre l'albero a una stella con al centro il sistema di riferimento assoluto, come mostrato in Figura 5.4.

Si noti che gli archi inversi, non mostrati in figura, possono essere calcolati ove necessario con la relazione $T_{M_i}^{M_j} = (T_{M_j}^{M_i})^{-1}$.

La relazione tra il sistema di riferimento assoluto e il marker base $(T_{M_i}^{W})$ deve essere specificata senza procedure automatiche, ma è facile individuare delle posizioni in cui porre il marker base che permettano di semplificare questa relazione. Ad esempio, se si posiziona un marker sul soffitto in un angolo di una stanza è possibile definire un sistema che giace sul pavimento come sistema assoluto che sia semplicemente traslato di una quantità pari all'altezza della stanza rispetto al sistema di riferimento solidale con il marker.

5.4.3 Relazioni multiple

Un caso precedentemente non considerato, ma sicuramente non infrequente, è quello in cui la struttura più adatta a descrivere le relazioni tra i marker non sia un albero ma un grafo connesso. È infatti possibile che siano disponibili le relazioni tra tre marker differenti j,k,m come $T_{M_k}^{M_j}$, $T_{M_m}^{M_j}$, $T_{M_m}^{M_k}$.

Da queste relazioni è possibile creare tutte le relazioni inverse, ovvero $T_{M_j}^{M_k}$, $T_{M_j}^{M_m}$, $T_{M_k}^{M_m}$. Si supponga inoltre di conoscere la relazione dal sistema

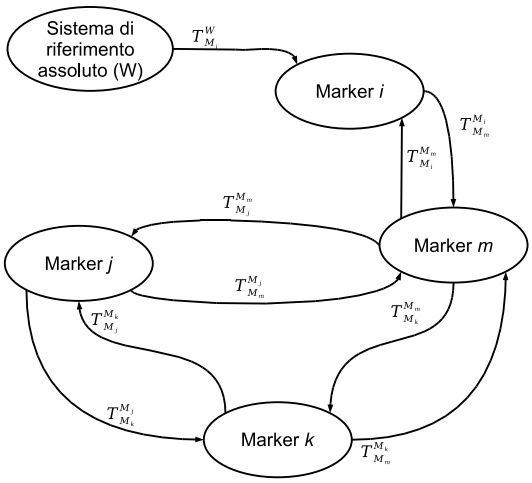


Figura 5.5: Esempio di grafo delle relazioni con quattro marker

di riferimento assoluto al marker base i ($T^{W}_{M_i}$) e dal marker i al marker m ($T^{M_m}_{M_i}$).

In questa configurazione, schematizzata in Figura 5.5, è possibile individuare più modi per creare la relazione che va dal sistema di riferimento assoluto a quello del marker k. Lo stesso vale per il marker j. In linea teorica sarebbe sufficiente estrarre un albero di copertura (anche casuale) del grafo e calcolare le relazioni con il sistema di riferimento assoluto sull'albero. Purtroppo, a causa del rumore che si può presentare nei dati rilevati in modo automatico, la scelta dell'albero di copertura non è ininfluente e può compromettere la precisione del sistema di posizionamento. In caso di grafi molto complessi, in cui il numero di passaggi intermedi per risalire al sistema assoluto è alto, l'effetto di accumulazione dell'errore potrebbe rendere inaffidabile il sistema di localizzazione. A questo livello di astrazione, in cui non si è ancora fatta nessuna considerazione sul tipo di rumore che può affliggere la stima delle matrici di rototraslazione rilevata con sistemi di fiducial marker, non è possibile ipotizzare quali siano le modalità migliori per scegliere l'albero di copertura del grafo. L'algoritmo proposto per la scelta automatica dell'albero di copertura è illustrato nella Sezione 5.6.4.

5.5 Realizzazione del sistema di localizzazione

Nelle sezioni precedenti si è dimostrata la realizzabilità di un sistema di posizionamento assoluto basato sull'uso di fiducial marker e sono stati pre-

sentati tre sistemi software di analisi di fiducial marker. Prima di realizzare effettivamente il sistema di localizzazione sono state effettuate alcune considerazioni al fine di scegliere quale strumento di gestione dei fiducial marker utilizzare. Per verificare se la precisione del sistema sarebbe stata sufficiente per realizzare un sistema robusto e affidabile sono state effettuate prove preliminari in condizioni controllate.

5.5.1 Scelta del sistema di gestione dei fiducial marker

Nella Sezione 5.2 sono state analizzate le caratteristiche di ARToolKit, ARTag e ARToolKitPlus. L'uso di ARToolKit è stato subito escluso in quanto tutte le funzionalità offerte da questo sistema sono presenti anche in ARToolKitPlus. ARToolKit si è dimostrato però un valido sistema di prova, in quanto il kit di sviluppo, disponibile con licenza GPL[4], è fornito con una buona documentazione e con molti esempi d'uso.

ARToolKitPlus, che deriva da ARToolKit anche se non ne condivide il codice, è nato per essere usato da progetti interni al team di sviluppo e non con lo specifico intento di essere distribuito al pubblico. La documentazione a corredo e gli esempi forniti sono di bassa qualità, anche se l'esperienza acquisita con ARToolKit permette di comprenderne in fretta le caratteristiche salienti.

La libreria di ARTag è distribuita con una licenza a termine e il codice sorgente non è disponibile; oltretutto non è al momento disponibile alcuna licenza commerciale. Anche se ARTag presenta alcune caratteristiche interessanti e innovative, come il riconoscimento di marker anche se parzialmente occlusi, è stato scartato a favore di ARToolKitPlus, distribuito in modo gratuito, a tempo illimitato e corredato di codice sorgente.

5.5.2 Posizionamento dei marker e della telecamera

Il buon funzionamento del sistema è influenzato dalla scelta della posizione dei marker rispetto alla telecamera. Innanzitutto nel posizionare i marker è necessario garantire che si possa inquadrare ogni marker almeno insieme ad un altro, al fine di garantire la connessione del grafo delle relazioni tra i sistemi di riferimento descritto nella Sezione 5.4.3. Un buon posizionamento dei marker deve mantenere bassa la probabilità che i marker siano nascosti da altri oggetti presenti nell'ambiente o da persone che si muovono, di modo che i dati di posizionamento siano disponibili per il maggior tempo possibile. Le posizioni della telecamera e dei marker valutate sono state:

[4] http://www.gnu.org/copyleft/gpl.html

- Telecamera frontale e marker posizionati sulle pareti verticali della stanza.
- Telecamera rivolta verso il basso e marker sul pavimento.
- Telecamera rivolta verso l'alto e marker sul soffitto.

La soluzione con telecamera frontale e marker sulle pareti è fortemente soggetta a occlusioni in quanto persone od oggetti nella stanza possono coprire in tutto o in parte i marker. Come ulteriore problema, se l'ambiente fosse di grandi dimensioni sarebbe necessario utilizzare marker molto grandi per far si che la telecamera possa riconoscerli anche da lontano. Sarebbero inoltre necessari anche marker più piccoli, in quanto avvicinandosi alle pareti della stanza i marker di grosse dimensioni uscirebbero facilmente dall'inquadratura della telecamera. Il caso di marker sul pavimento risolve il problema di coprire zone vaste, in quanto è possibile posizionare i marker uniformemente su tutta la superficie di movimento. Anche in questa configurazione è però molto facile che i marker possano essere oscurati dalla presenza di persone od oggetti nell'ambiente. La configurazione più vantaggiosa è la terza, in cui si usano i marker sul soffitto garantendo la possibilità di coprire zone ampie e riducendo la probabilità delle occlusioni.

In conclusione si è scelto di utilizzare una telecamera rivolta verso l'alto e di posizionare i marker sul soffitto. Il sistema che si va a sviluppare non è però rigido, e permette di posizionare i marker anche su soffitti a più livelli. Inoltre è comunque possibile posizionare marker sulle pareti verticali, a patto che siano sufficientemente elevati da essere inquadrati dalla telecamera.

5.5.3 Prove effettuate

Per poter utilizzare ARToolKitPlus è stato necessario calibrare la telecamera. Questa operazione è stata effettuata utilizzando Camera Calibration Toolbox for Matlab[5] che permette di stimare sia la matrice fondamentale della telecamera sia i parametri di correzione della distorsione radiale dell'ottica. Per valutare le prestazioni di ARToolKitPlus è stata utilizzata una struttura che mantentiene la telecamera rivolta verso il terreno (Figura 5.6). Il marker è stato stampato su un foglio di carta comune. Spostando il marker sul pavimento si simula la configurazione reale con marker sul soffitto e telecamera mobile, semplificando però la realizzazione delle prove. Il marker utilizzato è quadrato con lato di 160mm. Marker più piccoli risultano difficilmente visibili da distanze superiori al metro, mentre marker più grandi sono

[5] http://www.vision.caltech.edu/bouguetj/calib_doc

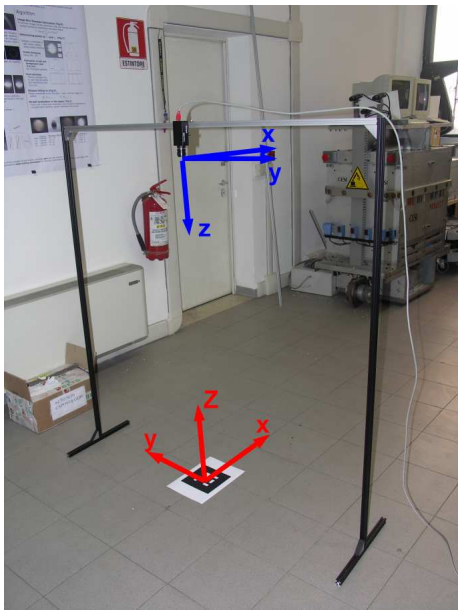


Figura 5.6: Struttura di test per il sistema di localizzazione

difficilmente riproducibili con stampanti comuni, in quanto è buona prassi lasciare un abbondante margine bianco intorno al bordo nero del marker per favorire l'identificazione del contorno.

Non è scopo di questo lavoro proporre dei metodi di test per valutare la precisione delle misure effettuate e analizzare in modo approfondito le condizioni che migliorano o peggiorano le prestazioni dei sistemi di localizzazione basati su fiducial marker, ma alcuni test preliminari sono stati svolti per analizzare la bontà del sistema in uso.

Sistema di riferimento telecamera

Inizialmente si è svolto un test di ripetibilità, ovvero sono state registrate più immagini mantenendo ferme sia la telecamera che il marker. In linea teorica, analizzando queste immagini si dovrebbero ottenere sempre gli stessi risultati, ma ciò non è garantito a causa della presenza di rumore. Tale rumore è dovuto, ad esempio, a vibrazioni impercettibili della struttura o variazioni delle condizioni di luce con cui si effettua la ripresa. Inoltre, la condizione di perpendicolarità tra il marker e la telecamera favorisce gli errori nella stima della rotazione relativa tra il sistema di riferimento solidale con il marker e quello solidale con la telecamera. È infatti sufficiente rilevare erroneamente, anche di pochi pixel, uno dei quattro angoli del marker per

introdurre un errore sull'orientamento dei marker evidente. I dati valutati sono la posizione e la rotazione del marker rispetto al sistema di riferimento della telecamera e sono forniti dalla libreria di gestione dei fiducial marker in una matrice omogenea di rototraslazione che ha la seguente forma:

$$T_M^C = \begin{pmatrix} u_1 & v_1 & w_1 & t_x \\ u_2 & v_2 & w_2 & t_y \\ u_3 & v_3 & w_3 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.8}$$

La posizione del marker in coordinate telecamera è espressa direttamente dall'ultima colonna della matrice. L'estrazione degli angoli di rotazione α intorno all'asse z, β intorno a y e γ intorno a x espressi con il sistema areonautico di *yaw*, *pitch* e *roll*, si ricava dalle seguenti relazioni:

$$\begin{cases} u_1 = \cos\alpha\cos\beta \\ u_2 = \sin\alpha\cos\beta \\ u_3 = -\sin\beta \\ v_1 = \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \\ v_2 = \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma \\ v_3 = \cos\beta\sin\gamma \\ w_1 = \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ w_2 = \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ w_3 = \cos\beta\cos\gamma \end{cases} \tag{5.9}$$

Il marker è stato posizionato come visibile in Figura 5.7 e sono state analizzate 350 immagini mantenendo il sistema nella stessa condizione. La posizione media del marker rispetto al sistema di riferimento della telecamera è risultata $x = 9.23$mm, $y = 8.96$mm, $z = 1456.87$mm. La deviazione standard che caratterizza questi valori medi è rispettivamente di 0.07mm, 0.01mm, 0.94mm. Per quanto riguarda l'orientamento si è rilevato $\alpha = 91.38\,^\circ$, $\beta = -8.76\,^\circ$, $\gamma = 170.90\,^\circ$, con rispettive deviazioni standard di $0.03\,^\circ$, $0.22\,^\circ$, $0.28\,^\circ$. Si nota facilmente che i dati più rumorosi (considerando l'errore in valore assoluto) sono quelli relativi alla posizione in z e all'orientamento rispetto agli assi x e y. Sono state effettuate altre prove statiche di ripetibilità posiziondo il marker nei quattro angoli dell'immagine (Figura 5.8), con i risultati riportati in Tabella 5.1. Il numero di immagini utilizzate è variabile, ma sempre superiore alle 300 unità. Questi test confermano che i valori più rumorosi sono quelli di posizione z e di orientamento β e γ, evidenziando però che la forte distorsione dovuta all'obiettivo grandangolare

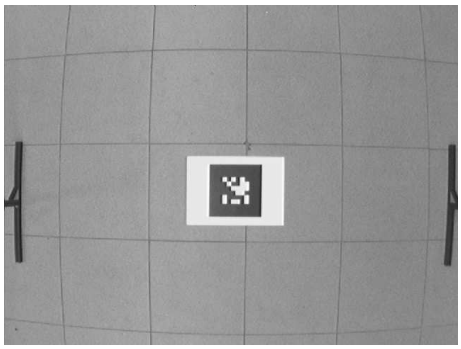


Figura 5.7: Marker in posizione centrale rispetto alla telecamera

influisce sulla precisione dei dati rilevati con i marker ai bordi dell'immagine. Il caso con marker nell'angolo Nord Est dell'immagine presenta deviazioni standard elevate e peggiori rispetto alle altre configurazioni anche per quanto riguarda i valori di posizione x e y. Si può però notare dalla Figura 5.8 che il marker è stato appositamente posizionato al limite dell'immagine, a differenza degli altri casi in cui si è mantenuto un certo margine. Nelle Figure 5.9 e 5.10 sono riportate le nuvole di punti che rappresentano le posizioni rilevate per il marker in posizione centrale e nei quattro angoli considerati rispettivamente in due dimensioni (considerando le sole coordinate x e y) e in tre dimensioni (aggiungendo anche la coordinata z).

Un'ulteriore prova è stata compiuta muovendo il marker lungo un percorso di forma rettangolare con lati di circa 600mm e 690mm (alcune immagini sono visibili in Figura 5.11). Con riferimento alla Figura 5.12 che riporta le posizioni del marker rispetto al sistema di riferimento solidale con la telecamera nelle sole coordinate x e y si può notare come il percorso risulti ben definito. Osservando invece la Figura 5.13 in cui è stata aggiunta la coordinata z si può notare come questa sia decisamente più rumorosa (sempre considerando gli errori in valore assoluto), delle altre due considerate, come verificato precedentemente per i casi statici.

Sistema di riferimento marker

Per conoscere la posizione della telecamera nel sistema di riferimento del marker è necessario invertire la matrice che descrive la posizione e l'orientamento del marker rispetto al sistema di riferimento della telecamera (cfr. Equazione 5.1). Data la particolare forma delle matrici di rototraslazione, per realizzare l'inversione della matrice illustrata nell Equazione 5.8, è

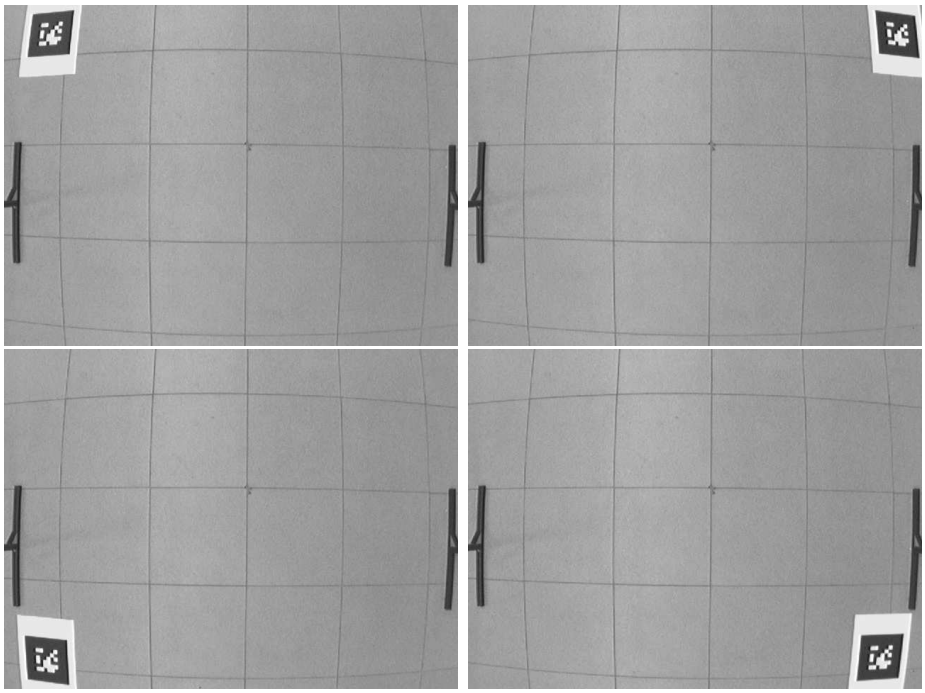


Figura 5.8: Marker negli angoli dell'immagine

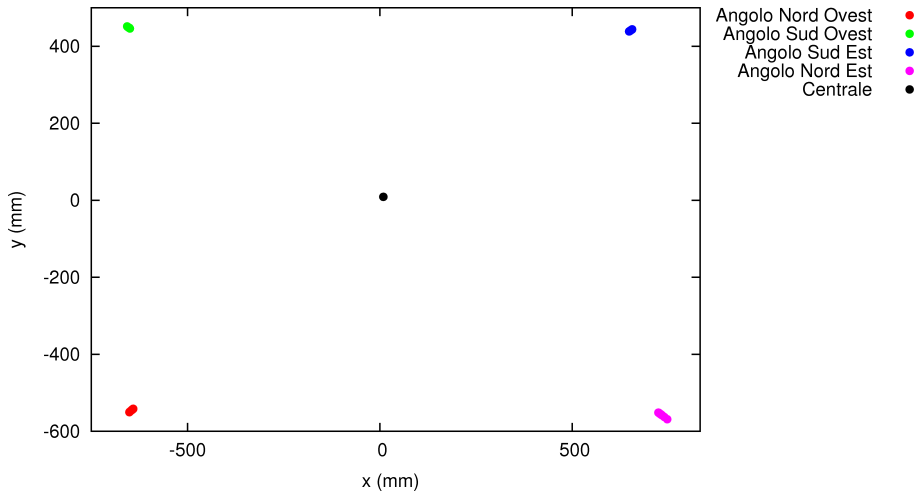


Figura 5.9: Posizione 2D del marker in coordinate telecamera nelle cinque posizioni considerate

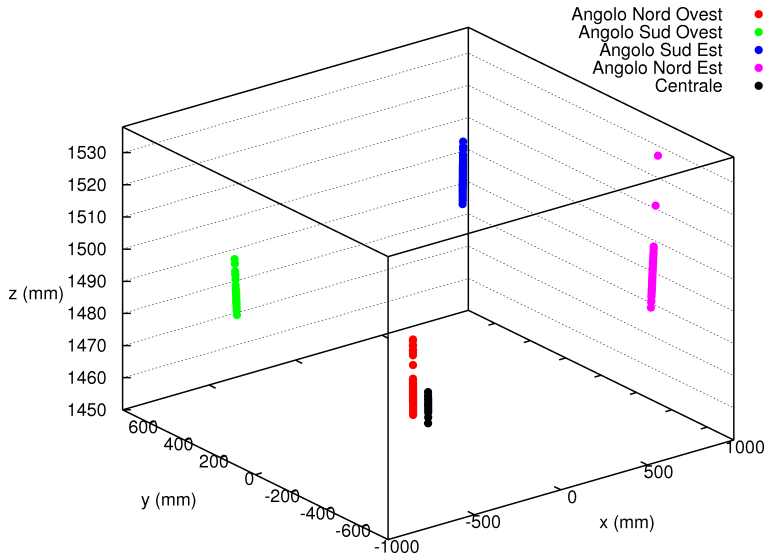


Figura 5.10: Posizione 3D del marker in coordinate telecamera nelle cinque posizioni considerate. Si noti che la scala dell'asse z è differente da quella degli altri assi per poter visualizzare meglio la distribuzione dei punti ottenuta

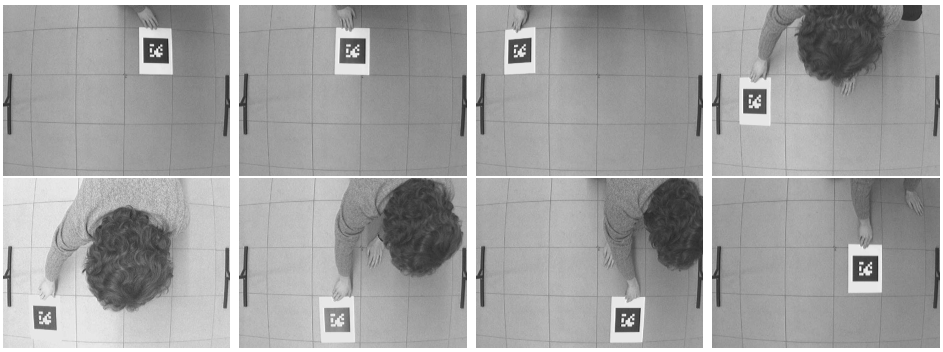


Figura 5.11: Alcune immagini del movimento rettangolare realizzato

Posizione Marker	x	y	z
Centrale	9.23 ± 0.07 mm	8.96 ± 0.01 mm	1456.87 ± 0.94 mm
Angolo Nord Ovest	−643.51 ± 1.91 mm	−543.79 ± 1.64 mm	1483.75 ± 4.41 mm
Angolo Nord Est	746.24 ± 4.75 mm	−583.73 ± 5.65 mm	1480.71 ± 7.68 mm
Angolo Sud Ovest	−651.98 ± 0.97 mm	447.94 ± 0.66 mm	1488.08 ± 2.28 mm
Angolo Sud Est	651.50 ± 1.25 mm	440.66 ± 0.86 mm	1504.84 ± 2.93 mm

Posizione Marker	α	β	γ
Centrale	91.38° ± 0.03°	−8.76° ± 0.22°	170.90° ± 0.28°
Angolo Nord Ovest	−175.43° ± 0.11°	−0.37° ± 0.23°	−175.52° ± 0.25°
Angolo Nord Est	−179.37° ± 0.07°	−0.90° ± 0.24°	−179.21° ± 0.18°
Angolo Sud Ovest	−178.09° ± 0.10°	−0.12° ± 0.15°	177.35° ± 0.26°
Angolo Sud Est	−177.84° ± 0.12°	−2.13° ± 0.25°	−179.08° ± 0.26°

Tabella 5.1: Posizione e orientamento del marker posto al centro e negli angoli dell'immagine rispetto alla telecamera

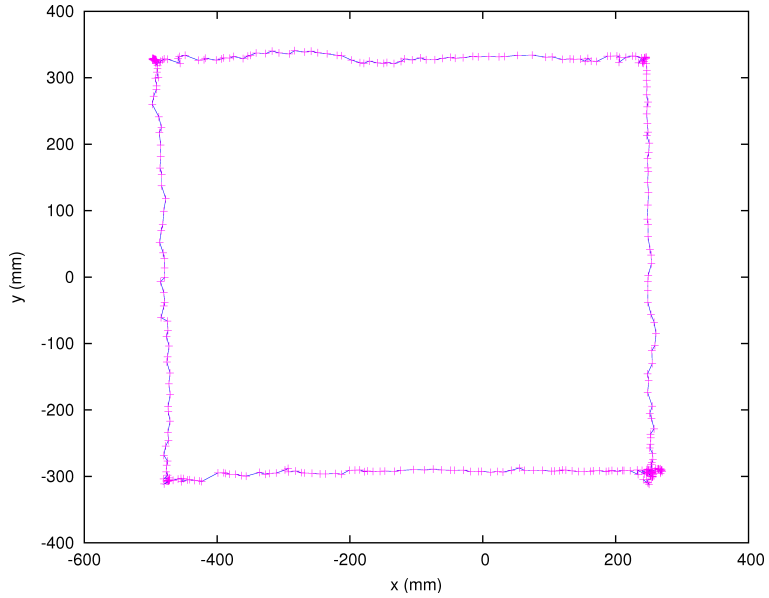


Figura 5.12: Posizione x e y del marker in coordinate telecamera in un percorso rettangolare

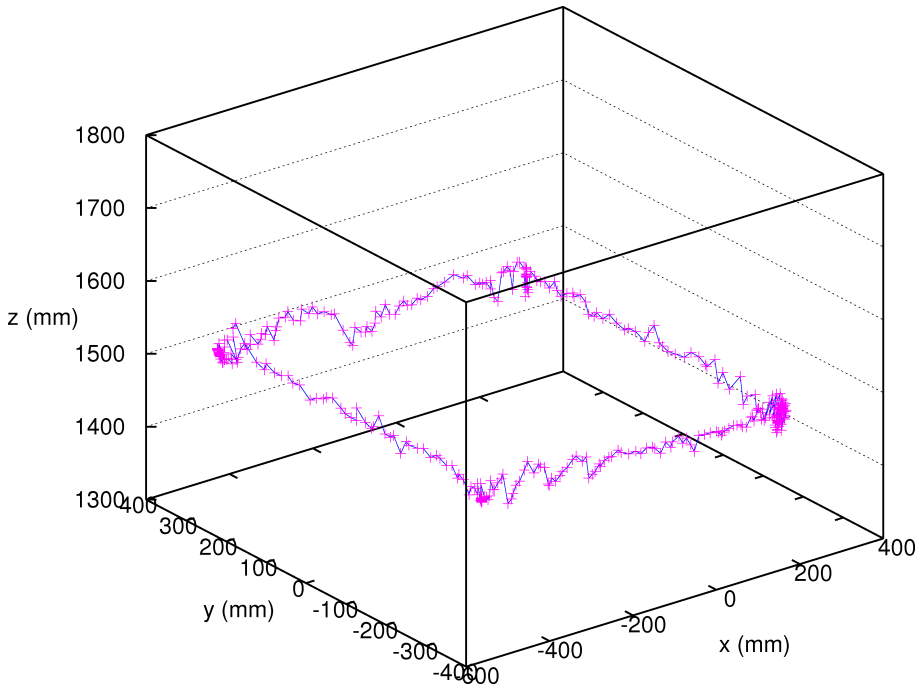


Figura 5.13: Posizioni x, y e z del marker in coordinate telecamera in un percorso rettangolare

sufficiente creare la seguente matrice:

$$T_C^M = \begin{pmatrix} u_1 & u_2 & u_3 & -([u_1, u_2, u_3] \cdot [t_x, t_y, t_z]) \\ v_1 & v_2 & v_3 & -([v_1, v_2, v_3] \cdot [t_x, t_y, t_z]) \\ w_1 & w_2 & w_3 & -([w_1, w_2, w_3] \cdot [t_x, t_y, t_z]) \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.10}$$

che è a sua volta una matrice di rototraslazione omogenea del tipo

$$T_C^M = \begin{pmatrix} \tilde{u}_1 & \tilde{v}_1 & \tilde{w}_1 & \tilde{t}_x \\ \tilde{u}_2 & \tilde{v}_2 & \tilde{w}_2 & \tilde{t}_y \\ \tilde{u}_3 & \tilde{v}_3 & \tilde{w}_3 & \tilde{t}_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.11}$$

L'ultima colonna di questa matrice rappresenta la posizione della telecamera nel sistema di riferimento solidale con il marker. È possibile estrarre gli angoli $\tilde{\alpha}$, $\tilde{\beta}$, $\tilde{\gamma}$ di yaw, pitch e roll che descrivono la rotazione sugli assi z, y e x del sistema di riferimento solidale con la telecamera rispetto al sistema di riferimento posto sul marker risolvendo il Sistema 5.9.

Si è calcolata la relazione che descrive la posizione e l'orientamento della telecamera rispetto al marker nei casi precedentemente analizzati con marker fermo nel centro e nei quattro angoli dell'immagine. La posizione nelle sole coordinate x e y della telecamera nel sistema di riferimento solidale con il marker è visualizzata in Figura 5.14 e nelle coordinate x, y e z in Figura 5.15. I dati di posizione e orientamento estratti sono riassunti in Tabella 5.2. È subito evidente che l'inversione della relazione introduce degli errori. L'analisi delle deviazioni standard dei dati di posizionamento confermano come la precisione del sistema risente fortemente del cambio di sistema di riferimento. Risultati molto peggiori si ottengono però invertendo la relazione per i dati relativi al percorso rettangolare. La posizione della telecamera rispetto al sistema di riferimento del marker è riportata nelle sole coordinate x,y in Figura 5.16 e nelle coordinate x,y e z in Figura 5.17. È evidente che l'inversione della matrice degrada notevolmente le rilevazioni, al punto da rendere irriconoscibile il percorso effettuato.

Investigando sulle cause che contribuiscono alla degradazione del risultato di posizionamento nel momento in cui si effettua l'inversione della matrice sono state valutate le seguenti due ipotesi:

- Problemi di approssimazione numerica nell'inversione della relazione.
- Errori nei valori della matrice da invertire.

La prima ipotesi è stata scartata in quanto l'operazione di inversione della matrice non si effettua con metodi generali, ma sfruttando la particolare

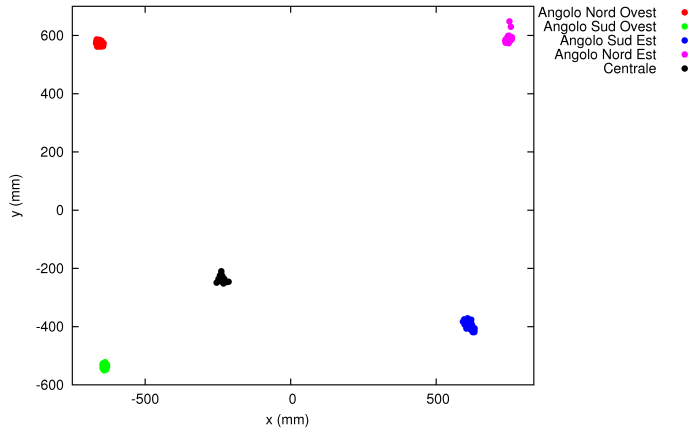


Figura 5.14: Posizione x e y della telecamera nel sistema di riferimento marker con il marker posto ai 4 angoli e in posizione centrale.

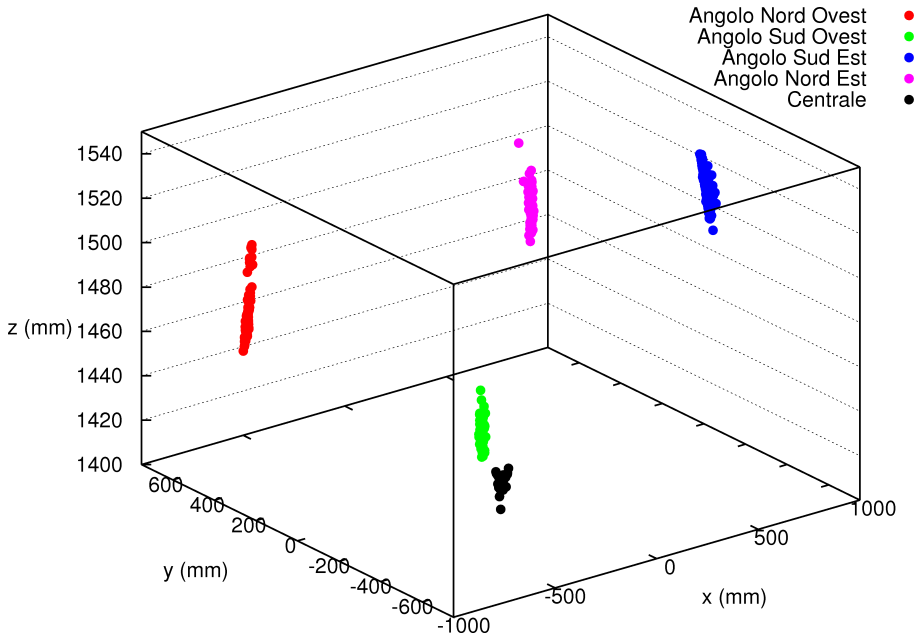


Figura 5.15: Posizione x, y e z della telecamera nel sistema di riferimento marker con il marker posto nei 4 angoli e in posizione centrale.

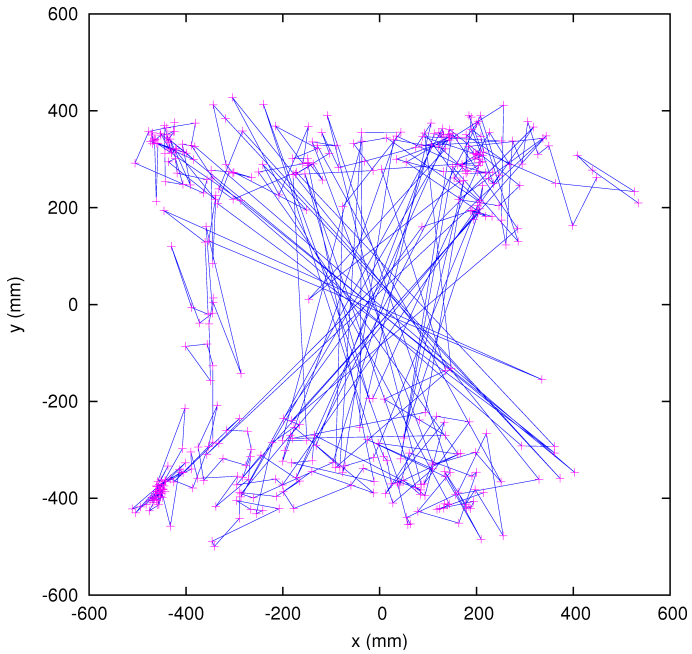


Figura 5.16: Posizione x e y della telecamera nel sistema di riferimento marker in un percorso rettangolare.

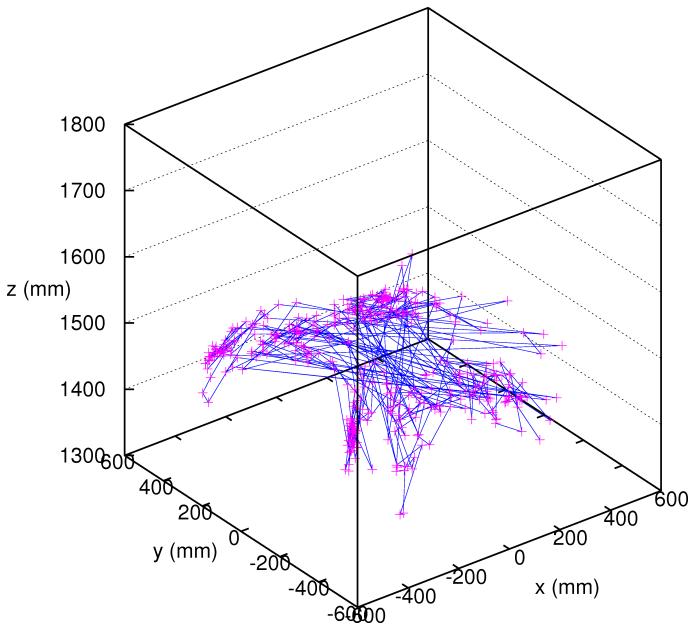


Figura 5.17: Posizione x, y e z della telecamera nel sistema di riferimento marker in un percorso rettangolare.

Posizione Marker	x	y	z
Centrale	-230.70 ± 5.37mm	-236.94 ± 7.13mm	1418.88 ± 1.27mm
Angolo Nord Ovest	-658.44 ± 4.56mm	575.40 ± 6.04mm	1465.13 ± 8.84mm
Angolo Nord Est	746.25 ± 4.75mm	583.73 ± 5.65mm	1480.71 ± 7.68mm
Angolo Sud Ovest	-639.78 ± 3.47mm	-537.62 ± 7.50mm	1463.45 ± 5.06mm
Angolo Sud Est	611.34 ± 6.69mm	-391.27 ± 7.25mm	1535.00 ± 5.71mm

Posizione Marker	α	β	γ
Centrale	$91.38° \pm 0.03°$	$-8.89° \pm 0.28°$	$171.02° \pm 0.22°$
Angolo Nord Ovest	$-179.43° \pm 0.11°$	$0.36° \pm 0.23°$	$-178.52° \pm 0.25°$
Angolo Nord Est	$-179.37° \pm 0.07°$	$-0.90° \pm 0.24°$	$-179.21° \pm 0.18°$
Angolo Sud Ovest	$-178.09° \pm 0.10°$	$0.21° \pm 0.16°$	$177.36° \pm 0.26°$
Angolo Sud Est	$-177.81° \pm 0.13°$	$2.09° \pm 0.24°$	$-179.00° \pm 0.27°$

Tabella 5.2: Posizione e orientamento della telecamera rispetto al marker posto al centro e negli angoli dell'immagine

forma delle matrici di rototraslazione (cfr. Equazione 5.10). Le uniche operazioni numeriche che si effettuano sono relative al calcolo dei valori dell'utilma colonna e, anche se coinvolgono moltiplicazioni e somme di numeri di ordine di grandezza differente (traslazioni nell'ordine delle migliaia di millimetri e valori nel range [-1,1]), non giustificano errori così grossolani. Prove pratiche hanno dimostrato che la moltiplicazione di una matrice di questo tipo per la sua inversa comporta errori trascurabili (nell'ordine di 10^{-15}) rispetto alla matrice identità attesa.

La seconda ipotesi si è dunque rilevata essere la causa della degradazione dei dati di posizionamento. Uno studio approfondito sugli effetti dell'errore complessivo che si genera a causa del rumore sui 6 parametri che specificano una matrice di rototraslazione è alquanto complesso. Come verifica dell'ipotesi della presenza del rumore è sufficiente considerare la variabilità dei dati di posizione nei casi statici riportata in Tabella 5.1. Dalla matrice illustrata nell'Equazione 5.10 si nota che il valore $\tilde{t_x}$ è combinazione lineare delle tre coordinate t_x, t_y e t_z con la prima colonna della matrice, che essendo composta da funzioni trigonometriche assume valori compresi tra 0 e 1. Dato che la configurazione tipica d'uso prevede che la coordinata z sia un valore nell'ordine dei metri (ovvero delle migliaia di millimetri), un piccolo errore nella stima dell'orientamento e della posizione del marker comporta grandi errori nell'inversione della matrice. Lo stesso tipo di errore influisce anche sul calcolo di $\tilde{t_y}$ e $\tilde{t_z}$. Un cambio di unità di misura non migliorerebbe i risultati, in quanto i problemi non sono di tipo numerico, ma di propagazione dell'errore: considerando, ad esempio, un errore ϵ_z che affligge la coordinata t_z, la posizione della telecamera in coordinate marker risulta

$$\begin{cases} \tilde{t}_{x_{\epsilon_z}} = -\left([u_1, u_2, u_3] \cdot [t_x, t_y, t_z + \epsilon_z]\right. \\ \tilde{t}_{y_{\epsilon_z}} = -\left([v_1, v_2, v_3] \cdot [t_x, t_y, t_z + \epsilon_z]\right. \\ \tilde{t}_{z_{\epsilon_z}} = -\left([w_1, w_2, w_3] \cdot [t_x, t_y, t_z + \epsilon_z]\right. \end{cases} \tag{5.12}$$

ovvero si introduce un errore sulle tre coordinate pari a

$$\begin{cases} |\tilde{t}_{x_{\epsilon_z}} - \tilde{t}_x| = \epsilon_z u_3 \\ |\tilde{t}_{y_{\epsilon_z}} - \tilde{t}_y| = \epsilon_z v_3 \\ |\tilde{t}_{z_{\epsilon_z}} - \tilde{t}_z| = \epsilon_z w_3 \end{cases} \tag{5.13}$$

Considerazioni analoghe si ottengono studiando l'effetto prodotto dal rumore su ϵ_x e ϵ_y su t_x e t_y. La valutazione degli effetti dell'errore ϵ_α, ϵ_β e ϵ_γ sull'orientamento è più complessa e non viene trattata, ma è facile intuire che sommando tutti gli effetti degli errori si può arrivare a degradare notevolmente la precisione della stima della posizione della telecamera nel sistema di riferimento marker.

5.6 Semplificazione da 6 a 3 dof

Le prove svolte con il sistema di localizzazione basato sull'uso dei dati forniti dalla libreria di gestione dei fiducial marker hanno rivelato che non è possibile stimare in modo preciso la posizione e l'orientamento della telecamera rispetto al marker, soprattutto quando le distanze tra i due sistemi di riferimento sono elevate. Limitandosi ai risultati finora ottenuti potrebbe sembrare impossibile utilizzare il sistema proposto per localizzare un oggetto mobile in un ambiente. È possibile però introdurre alcune semplificazioni che possono migliorare la precisione del sistema.

Considerando un sistema semplice in cui un oggetto mobile si sposta su un piano parallelo al marker, si nota che il problema della localizzazione non è a sei gradi di libertà, ma si riduce a tre gradi di libertà. Sono infatti sufficienti le coordinate $\tilde{t_x}$, $\tilde{t_y}$ e la rotazione $\tilde{\alpha}$ intorno all'asse z nel sistema di riferimento del marker per descrivere completamente la posizione e l'orientamento dell'oggetto. Si può inoltre affermare che i restanti tre gradi di libertà restano costanti. La distanza t_z del marker dal sistema di riferimento della telecamera e gli angoli β e γ di orientamento, che risultavano essere i più rumorosi, dovrebbero essere costanti, in quanto il moto relativo tra i due sistemi non li coinvolge.

La soluzione pratica per limitare la rumorosità dei dati forniti dall'analisi delle immagini con il sistema di fiducial marker consiste nel sostituire i valori β, γ e t_z nella matrici T_M^C con i valori costanti $\overline{\beta}$ e $\overline{\gamma}$ e $\overline{t_z}$. La nuova matrice che descrive la posizione del marker nel sistema di riferimento della telecamera risulta:

$$\overline{T}_M^C = \begin{pmatrix} \overline{u}_1 & \overline{v}_1 & \overline{w}_1 & t_x \\ \overline{u}_2 & \overline{v}_2 & \overline{w}_2 & t_y \\ \overline{u}_3 & \overline{v}_3 & \overline{w}_3 & \overline{t}_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.14}$$

Estendendo il problema al caso più generale in cui sono presenti più marker è necessario conoscere i valori costanti $\overline{\beta}_i$ e $\overline{\gamma}_i$ e $\overline{t}_{z_i}$ associati a ogni marker presente nel sistema. La stima della posizione della telecamera rispetto a un marker, noti i parametri costanti, segue quindi i seguenti passaggi:

1. Acquisizione della matrice $T_{M_i}^C$ dalla libreria di gestione dei fiducial marker.

2. Creazione della matrice $\overline{T}_{M_i}^C$ tramite sostituzione dei parametri costanti noti per quel particolare marker.

3. Calcolo di $\overline{T}_C^{M_i}$ dall'inversione di $\overline{T}_{M_i}^C$.

5.6.1 Stima delle costanti

Non è stato ancora affrontato il problema di come ricavare i valori costanti di $\overline{\beta}_i$ e $\overline{\gamma}_i$ e $\overline{t}_{z_i}$. Si potrebbe pensare di misurare manualmente i parametri costanti. Questo metodo, oltre a essere scomodo, rischia di essere impreciso, a meno di effettuare pesanti assunzioni sul posizionamento della telecamera. Oltretutto per poter misurare precisamente la distanza $\overline{t}_{z_i}$ del marker nel sistema di riferimento della telecamera sarebbe necessario conoscere precisamente la posizione del piano immagine della telecamera.

Supponendo che il rumore su β_i, γ_i e t_{z_i} sia a media nulla, è possibile utilizzare i valori medi di più rilevazioni della posizione del marker i come costanti $\overline{\beta}_i$ e $\overline{\gamma}_i$ e $\overline{t}_{z_i}$. Rilevando un certo numero di immagini in cui il marker i è visibile si calcolano le matrici $T^C_{M_i}[1 \dots n]$ da cui è possibile estrarre i valori $\beta_i[1 \dots n]$, $\gamma_i[1 \dots n]$ e $t_{z_i}[1 \dots n]$. Le medie di questi valori, indicate con $\hat{\beta}_i$ e $\hat{\gamma}_i$ e $\hat{t}_{z_i}$, rappresentano la stima delle costanti $\overline{\beta}_i$ e $\overline{\gamma}_i$ e $\overline{t}_{z_i}$ cercate. Non è necessario che la telecamera e il marker siano fermi per creare questi dati, ma è sufficiente che la telecamera e il marker sviluppino un movimento relativo tale per cui gli unici valori variabili siano t_{x_i}, t_{y_i} e α_i. Questa operazione va ripetuta per ogni marker che si vuole utilizzare nel sistema di localizzazione.

5.6.2 Risultati ottenuti

Per verificare se il metodo proposto è effettivamente applicabile si sono riutilizzati i dati del percorso rettangolare d'esempio precedentemente illustrato. Il risultato, limitandosi alle sono coordinate x e y è visibile in Figura 5.18 ed è ben riconoscibile la forma del percorso realizzato. Allo stesso modo anche l'inserimento della coordinata z conferma la bontà del risultato ottenuto, come mostrato in Figura 5.19. Per avere un'ulteriore conferma dell'affidabilità del metodo di correzione proposto si sono mantenuti costanti i valori $\hat{\beta}$ e $\hat{\gamma}$ e $\hat{t_z}$ e si sono realizzate altre prove di movimento del marker. In questo modo abbiamo potuto verificare con un metodo di *crossvalidazione* che i valori medi utilizzati sono generali e i risultati ottenuti non sono applicabili solo al singolo caso di studio, ma globalmente validi. Ulteriori conferme di questi risultati sono riscontrabili nel Capitolo 7 che mostra i risultati sperimentali di questo lavoro di tesi.

5.6.3 Procedure di localizzazione e riferimento tra i sistemi

Alla luce dei problemi riscontrati e del metodo individuato per risolverli delineato, le procedure di localizzazione e di creazione delle relazioni tra

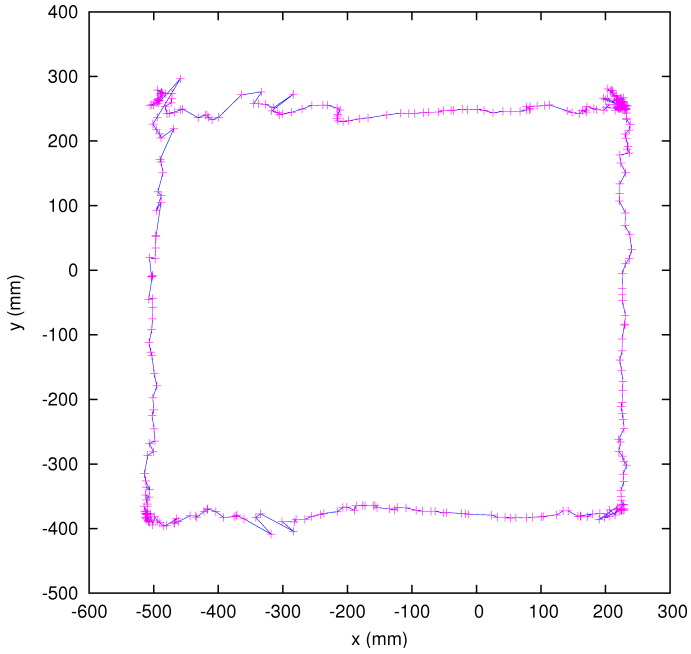


Figura 5.18: Posizione x e y della telecamera nel sistema di riferimento marker in un percorso rettangolare con correzione dei valori β, γ e t_z

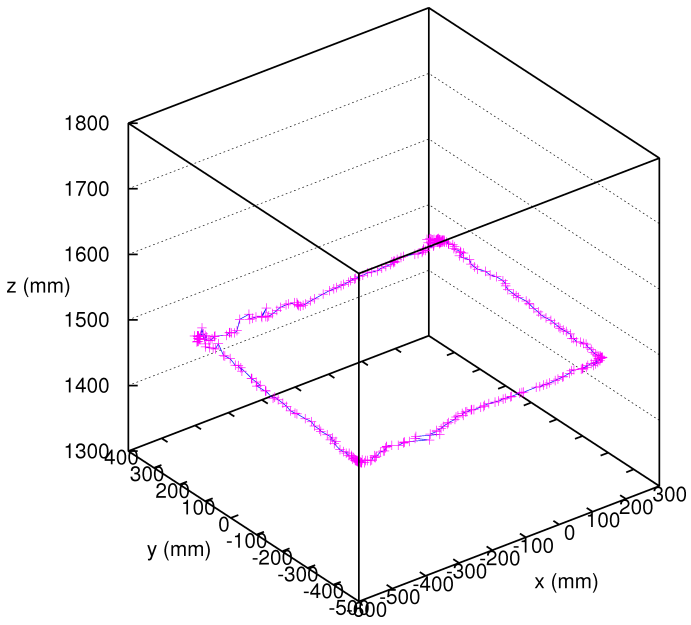


Figura 5.19: Posizione x e y della telecamera nel sistema di riferimento marker in un percorso rettangolare con correzione dei valori β, γ e t_z

marker e sistema assoluto illustrate nella Sezioni 5.3.1 e 5.4 devono essere opportunamente modificate.

La procedura di localizzazione di un oggetto mobile illustrata nella Sezione, una volta stimati i valori costanti $\hat{\beta}_i$ e $\hat{\gamma}_i$ e $\hat{t}_{z_i}$, diventa la seguente:

1. Acquisizione di un'immagine dalla telecamera.
2. Processo di analisi svolto dal sistema di fiducial marker: se non vengono rilevati marker nell'immagine comunica l'impossibilità di stabilire la posizione, altrimenti memorizza la matrice $T^C_{M_i}$ e l'identificativo i del marker.
3. Crea la matrice $\hat{T}^C_{M_i}$ con i dati $\hat{\beta}_i$, $\hat{\gamma}_i$ e $\hat{t}_{z_i}$.
4. Controlla se la matrice $\hat{T}^W_{M_i}$ è disponibile tra quelle memorizzate nel sistema: se non è disponibile comunica l'impossibilità di stabilire la posizione, altrimenti calcola $\hat{T}^W_R$ (Equazione 5.3).

La creazione della relazione diretta tra due marker i e j, supponendo noti i valori costanti $\hat{\beta}_i$, $\hat{\gamma}_i$, $\hat{t}_{z_i}$,$\hat{\beta}_j$, $\hat{\gamma}_j$ e $\hat{t}_{z_j}$ e avendo a disposizione un'immagine in cui sono presenti entrambi i marker si effettua sostituendo i valori costanti in entrambe le matrici. I passi da seguire sono dunque i seguenti:

1. Rilevazione della matrice $T^C_{M_i}$ fornita dal sistema di fiducial marker.
2. Rilevazione della matrice $T^C_{M_j}$ fornita dal sistema di fiducial marker.
3. Creazione di $\hat{T}^C_{M_i}$ con $\hat{\beta}_i$, $\hat{\gamma}_i$, $\hat{t}_{z_i}$.
4. Creazione di $\hat{T}^C_{M_j}$ con $\hat{\beta}_j$, $\hat{\gamma}_j$, $\hat{t}_{z_j}$.
5. Creazione di $\hat{T}^{M_i}_{M_j} = (\hat{T}^C_{M_i})^{-1} \cdot \hat{T}^C_{M_j}$.

Per migliorare la precisione delle relazioni tra i marker è consigliabile utilizzare più immagini in cui sono presenti sia il marker i che il j, calcolare un certo numero di matrici $\hat{T}^{M_i}_{M_j}[1 \dots n]$ ed estrarne i parametri $\hat{x}_{\hat{T}^{M_i}_{M_j}}[1 \dots n]$ $\dots \hat{\gamma}_{\hat{T}^{M_i}_{M_j}}[1 \dots n]$. La media dei singoli parametri permette di costruire la matrice $\hat{\hat{T}}^{M_i}_{M_j}$ che identifica la relazione media stimata tra il marker i e j.

Si noti che queste immagini, così come quelle necessarie per il calcolo delle relazioni tra i marker, devono essere catturate direttamente dal sistema mobile con la telecamera nella posizione in cui sarà realmente utilizzata per la localizzazione. È possibile utilizzare lo stessa sequenza di immagini sia per stimare i valori medi dei parametri di correzione che le relazioni tra i marker.

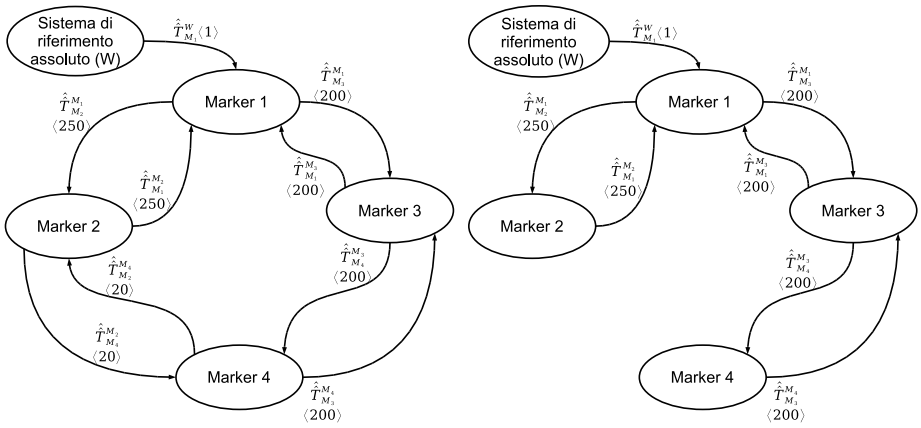


Figura 5.20: Grafo delle relazioni tra i sistemi di riferimento dei marker con archi pesati ed estrazione dell'albero di copertura di peso massimo.

L'unico vincolo è rappresentato dalla sequenza di esecuzione: come prima operazione è necessario stimare i parametri di correzione, successivamente è possibile stimare la posizione e l'orientamento relativo dei marker con il metodo appena illustrato.

5.6.4 Relazioni multiple tra marker

Nella Sezione 5.4.3 si è introdotta la configurazione a grafo delle relazioni tra marker. In linea teorica qualunque albero di copertura permette di giungere alle stesse relazioni tra sistema di riferimento assoluto e marker. Considerando però il rumore presente sui dati reali è necessario scegliere un albero che offra una certa garanzia sulla precisione delle relazioni costruite. Il grafo di Figura 5.5 può essere arricchito di informazioni utili a tale scopo. Ricordando che dalle matrici $T_{M_j}^{M_i}$ teoriche si è passati a usare $\hat{\hat{T}}_{M_j}^{M_i}$, un informazione utile è costituita dal numero n di matrici $\hat{\hat{T}}_{M_j}^{M_i}$ utilizzate per stimare la relazione media. Questa informazione può essere aggiunta all'arco della matrice $\hat{T}_{M_j}^{M_i}$. Nella scelta di un albero di copertura è opportuno evitare gli archi con peso basso, in quanto questo indica che la relazione tra i due marker è stata calcolata poche volte e le garanzie sulla sua precisione sono dunque inferiori in quanto la varianza della stima diminuisce all'aumentare del numero di campioni. Un albero di copertura a peso massimo rappresenta dunque una riduzione del grafo utilizzabile per la creazione delle relazioni tra il sistema di riferimento assoluto e i singoli marker. Un criterio che questo metodo trascura è però la lunghezza del percorso che collega il nodo di partenza con i nodi destinazione. Percorsi più lunghi, cioè che coinvolgono

più moltiplicazioni di matrici, fanno si che gli errori si accumulino maggiormente rispetto a percorsi brevi. Considerando però che le posizioni relative tra i marker sono piuttosto regolari, risulta poco probabile che si generino percorsi di lunghezze molto differenti.

Un certo numero di passaggi intermedi è dunque necessario. Un esempio di grafo da cui si estrae l'albero di copertura di peso massimo è mostrato in Figura 5.20. Il numero di relazioni su cui è stata effettuata la media è specificato tra parentesi angolari. Si noti che il peso associato alla relazione dal sistema assoluto al marker di base è ininfluente in quanto il suo arco è l'unico uscente dal nodo del sistema di riferimento assoluto, quindi farà sicuramente parte dell'albero di copertura.

Capitolo 6

Software di controllo

In questo capitolo viene analizzato e descritto il software di controllo della carrozzina elettrica dalle funzionalità estese realizzata. In una prima fase vengono descritte le scelte progettuali effettuate, successivamente vengono presentati i moduli che costituiscono l'applicazione e che realizzano il controllo della carrozzina.

6.1 Compiti del software di controllo

Il software di controllo costituisce una parte molto importante di un robot mobile. Infatti, le funzionalità che il robot è in grado di offrire non dipendono solo dalla sensoristica utilizzata e dagli attuatori, ma soprattuto dal modo in cui i dati vengono raccolti ed elaborati al fine di prendere decisioni.

In una carrozzina elettrica dalle funzionalità aumentate il software di controllo interagisce, oltre che con sensori e attuatori, anche con l'utente. Infatti il software di controllo deve essere in grado di interpretare la volontà dell'utente e gestire il moto della carrozzina in modo da garantire la sicurezza dell'utente che è a bordo di essa.

Nella Sezione 4.5 si è analizzato brevemente il software di controllo per la carrozzina elettrica e si è individuata la necessità di sviluppare un'applicazione multiagente, in cui ogni agente avesse la possibilità di scambiare informazioni con gli altri presenti nel sistema. Si è scelto di affidare la gestione dell'esecuzione degli agenti e la comunicazione a un framework software. Il framework scelto è DCDT ed è presentato nella Sezione 4.5.1. Inoltre si è identificata l'architettura di controllo chiamata Mr.Brian presentata nella Sezione 3.6 come più adatta allo sviluppo del controllore. Grazie alla

modularità che caratterizza questa architettura, il processo di progetto e la realizzazione del controllore risulta semplificato e facilmente estendibile.

Per descrivere l'architettura complessiva del software di controllo è necessario individuare i compiti specifici del software stesso. Innanzitutto è evidente che il software di controllo deve essere in grado di gestire tutti i dispositivi collegati alla carrozzina, ovvero il joystick originale della carrozzina, il joypad senza fili, i due scanner laser, la telecamera e il sensore inerziale e giroscopico. Le informazioni provenienti da tutti questi dispositivi sono utili per controllare il movimento della carrozzina stessa, considerando le diverse modalità di funzionamento in cui essa può operare.

È possibile riassumere i compiti del software come segue:

- Lettura dei dati relativi alla posizione della leva del joystick e impostazione del valore di controllo che permette di comandare la carrozzina. Infatti, il joystick originale della carrozzina rappresenta sia uno dei dispositivi con cui l'utente comanda il movimento della carrozzina stessa, sia l'interfaccia che permette di controllare il moto della carrozzina, grazie al circuito presentato nella Sezione 4.3. Il software di controllo deve essere in grado di comunicare con il circuito per poter comandare la carrozzina nel modo desiderato.

- Acquisizione dei comandi impostati con il joypad, che rappresenta un'altro dispositivo di input con il quale l'utente può comunicare la sua volontà al computer.

- Lettura dei dati relativi alle scansioni di ogni sensore laser, in modo da rendere disponibili le informazioni sugli ostacoli individuati e controllare la carrozzina al fine di compiere movimenti sicuri.

- Acquisizione di immagini dalla telecamera per calcolare la posizione assoluta della carrozzina nell'ambiente con il metodo presentato nel Capitolo 5.

- Acquisizione delle informazioni dal sensore inerziale e giroscopico XSens MTi al fine di conoscere lo stato della carrozzina, ovvero il modo in cui si muove.

- Pianificazione di percorsi per la guida automatica acquisendo dall'utente la destinazione da raggiungere.

- Esecuzione del percorso pianificato con l'obiettivo di raggiungere il punto di destinazione, controllando il moto della carrozzina in modo da seguire il percorso stabilito.

- Controllo e gestione di eventuali errori, guasti e malfunzionamenti per garantire la sicurezza dell'utente ed evitare situazioni pericolose.
- Visualizzazione dell'interfaccia grafica, punto di incontro tra l'utente e il software di controllo. Essa deve essere uno strumento utile per controllare il funzionamento del sistema e permettere all'utente di impartire alcuni comandi al software di controllo.
- Produzione di log dettagliati e facilmente analizzabili, utili sia in fase di sviluppo del sistema, per facilitare la ricerca di malfunzionamenti e problemi, sia in fase di analisi del comportamento del sistema stesso, in quanto permette di controllare le operazioni svolte dal sistema di controllo.

Come già accennato, il software di controllo è basato sul controllore gerarchico a comportamenti fuzzy Mr.Brian, presentato nella Sezione 3.6. La presenza di questo componente permette di realizzare il controllo della carrozzina concentrando tutte le informazioni sensoriali opportunamente modellizzate e codificate. Prima di descrivere l'architettura del software progettata e realizzata è opportuno introdurre alcune scelte progettuali effettuate.

6.1.1 Comunicazione tra i membri

DCDT utilizza una trama specifica per l'invio e la ricezione dei messaggi. Ogni messaggio deve essere infatti identificabile per permettere il funzionamento del meccanismo di iscrizione. Il formato del payload del messaggio è invece libero, ovvero è possibile spedire qualunque contenuto informativo. Nell'ambito del progetto MRT[1] sono state realizzate alcune estensioni a DCDT che si occupano di creare automaticamente il contenuto del messaggio codificato in XML. In particolare, ogni messaggio è contenuto in un campo `MESSAGE` caratterizzato dai seguenti attributi:

id: attributo numerico che specifica il tipo del messaggio.

sender: stringa che specifica il nome dell'host che invia il messaggio.

timestamp: specifica l'istante in cui il messaggio è stato inviato. È formato dai campi `SEC` e `USEC` preceduti da un underscore (_). Il timestamp misura il tempo trascorso dalla mezzanotte del 01/01/1970.

module: specifica il nome del membro che ha creato e inviato il messaggio.

[1] http://robocup.elet.polimi.it/MRT

Il contenuto del campo `MESSAGE` è libero e può essere specificato in qualsiasi modo all'atto della costruzione del messaggio. Per mantenere semplice il formato del contenuto si è scelto di utilizzare solo campi di nome `D`. Ogni campo contiene a sua volta una sola coppia di dati. Il primo elemento della coppia è una stringa che non contiene spazi e il secondo un numero reale. La coppia di dati rappresenta una consueta struttura di corrispondenza nome-valore. Ad esempio, un messaggio completo di tipo `15`, inviato da un membro di nome `A` residente sull'host `127.0.0.1` all'istante di tempo `1203435321.434723` contenente la coppia nome-valore `X 2.5` risulterebbe codificato così:

```
<MESSAGE id = 15 sender = 127.0.0.1
timestamp = _1203435321_434723 module = A>
<D>X 2.5 </D>
</MESSAGE>
```

Il parsing del campo `MESSAGE` è svolto dall'estensione di DCDT precedentemente presentata, mentre, data la struttura molto semplice del corpo del messaggio, è facile realizzare un parser che si occupi della lettura delle coppie nome valore di tutti i messaggi che saranno scambiati dal sistemi.

6.1.2 Pianificazione di percorsi

La pianificazione di percorsi è una attività fondamentale per dotare di autonomia un robot mobile e, in questo caso, una carrozzina elettrica con funzionalità estese. Si è scelto di integrare nel software di controllo della carrozzina il pianificatore Spike realizzato nel progetto FollowMe [45]. Non sono stati analizzati altri pianificatori in quanto Spike si è rivelato dopo alcuni test preliminari semplice da utilizzare e adatto allo scopo. Inoltre può essere considerato un progetto maturo, corredato di documentazione e codice sorgente.

Spike permette di definire una mappa metrica dell'ambiente. La mappa deve essere specificata staticamente con linee e cerchi che descrivono e delimitano l'ambiente. La mappa è statica, ovvero non può essere modificata dopo che è stata caricata dal pianificatore, tuttavia è possibile aggiungere dinamicamente ostacoli che permettono di aggiungere informazioni utili alla pianificazione. La suddivisione della mappa in celle per la ricerca del percorso è effettuata con l'uso di zone quadrate, il cui lato può essere impostato. Le celle sono collegate tra loro in base all'adiacenza dei lati che le costituiscono. La ricerca di un cammino da un punto di partenza a una destinazione è effettuata con il noto algoritmo A*, che permette di calcolare percorsi tra celle non occupate da ostacoli. I percorsi calcolati da A*

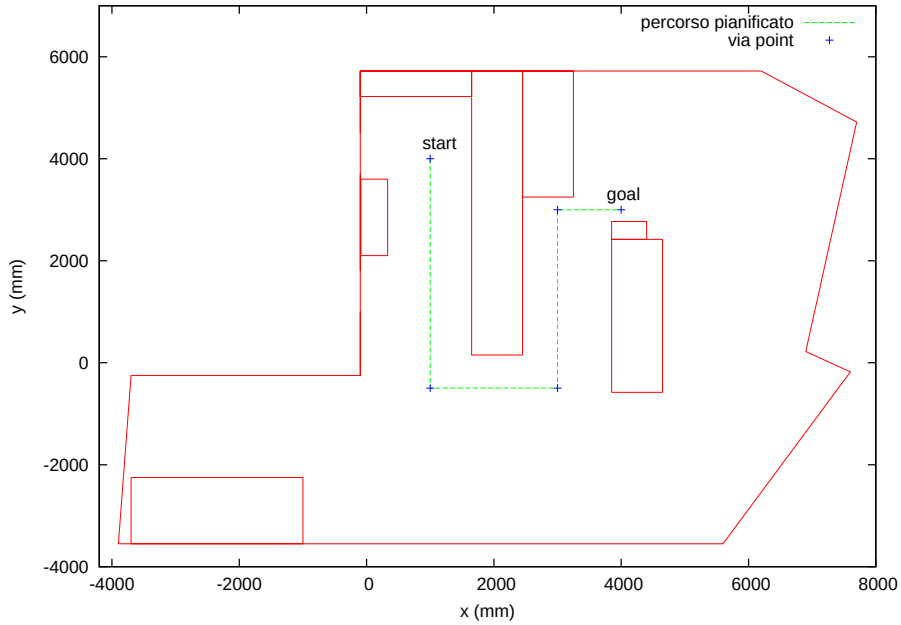


Figura 6.1: Esempio di percorso calcolato da Spike su una mappa

vengono semplificati ove possibile per evitare tratti eccessivamente segmentati. Il risultato dell'operazione di pianificazione è una lista di via point da seguire per realizzare il percorso oppure un'errore nel caso non sia possibile raggiungere il punto di destinazione. È importante sottolineare che il pianificatore non tiene conto in alcun modo né della cinematica del robot né del suo orientamento.

Un esempio di mappa con alcuni ostacoli e la lista di via point calcolata per raggiungere la destinazione specificata è mostrata in Figura 6.1.

6.2 Componenti software

Il software di controllo è basato sul framework DCDT, quindi i moduli che lo compongono sono dei membri di DCDT stesso. Considerando i compiti che il software di controllo deve svolgere, è impossibile presentare il sistema complessivo e spiegarne le funzionalità, mentre risulta più semplice esaminare quali sono le operazioni che il software deve svolgere nei casi semplici in cui è chiamato a operare. Nelle sezioni successive il sistema viene presentato gradualmente e la struttura del software si espande con l'aggiunta di nuovi membri fino ad arrivare a una definizione globale.

A titolo di riferimento sono elencati tutti i membri del framework DCDT

che sono stati realizzati nell'ordine in cui saranno presentati. I membri sono chiamati *esperti* in quanto si occupano di uno specifico compito e sono MotorExpert, BrianExpert, JoypadExpert, AudioExpert, HokuyoExpert, MTiExpert, SequencerExpert, PoseExpert, SpikeExpert, VisionExpert, ErrorExpert e GuiExpert.

Si ricorda che, grazie all'uso di DCDT, ogni esperto è in grado di scambiare informazioni con gli altri, con un meccanismo di publish/subscribe. In questa sezione analizziamo ciascun membro e le funzioni che deve svolgere per garantire il funzionamento del sistema di controllo della carrozzina e i messaggi che si attende di ricevere e produce. Si noti che l'identificativo numerico dei messaggi è per semplicità espositiva specificato con una stringa letterale.

Lo studio del software di controllo è stato eseguito immaginando il suo funzionamento durante il moto della carrozzina. Per chiarezza espositiva verranno presentati alcuni casi di studio che coprono tutte le funzionalità offerte dal software. Si suppone d'ora in avanti che il circuito di interfaccia del joystick con il PC sia in modalità automatica, ovvero che il moto della carrozzina sia governato dalla software di controllo stesso.

6.2.1 Guida con joystick

Il caso più semplice da analizzare è quello in cui si desidera guidare la carrozzina utilizzando il joystick posto sulla carrozina stessa. Supponiamo per ora che l'unico obiettivo sia quello di "copiare" i movimenti del joystick, senza integrare alcuna funzionalità "intelligente" (e.g., evitamento ostacoli). Gli esperti che intervengono in questo contesto sono due: MotorExpert e BrianExpert.

MotorExpert è il modulo che si occupa di realizzare la comunicazione con il circuito di interfaccia, ovvero è in grado di leggere le posizioni della leva del joystick e di comandare la carrozzina, seguendo il protocollo di comunicazione descritto nella Sezione 4.3.2. I messaggi a cui si iscrive questo membro sono quelli di tipo `MSG_FROM_BRIAN`, che contengono il comando da attuare sulla carrozzina, mentre i messaggi che invia sono `MSG_FROM_MOTION_BRIEF` e `MSG_FROM_MOTION_INFO`.

`MSG_FROM_BRIAN` contiene le informazioni necessarie per comandare la carrozzina espresse con tre coppie nome valore:

- *modecmd* comunica il modo di funzionamento da impostare sulla carrozzina. I valori ammissibili per questo campo sono solo 0 e 1. Il valore 0 indica che si intende chiedere alla carrozzina di passare alla modalità manuale, 1 indica di rimanere nella modalità automatica. Il

valore è tipicamente sempre impostato a 1, perchè qualora si richieda alla carrozzina di passare alla modalità manuale sarà impossibile ritornare alla modalità automatica senza la volontà dell'utente. Solo in situazioni critiche, come malfunzionamenti del sistema del controllo rilevabili dal sistema stesso, è possibile che sia necessario chiedere il passaggio alla modalità manuale.

- *fwrwcmd* indica il valore di attuazione da impostare sul circuito per il comando avati-indietro. Un valore pari a 0 indica la posizione di riposo, valori da 0 a +1 indicano la direzione "avanti" e da 0 a -1 indicano la direzione "indietro".

- *rxlxcmd* indica il valore di attuazione da impostare sul circuito per il comando destra-sinistra. Un valore pari a 0 indica la posizione di riposo, valori da 0 a +1 indicano "destra" e da 0 a -1 indicano "sinistra".

`MSG_FROM_MOTION_BRIEF` ha lo scopo di rendere disponibili le informazioni essenziali lette dal circuito di interfaccia con la carrozzina. Questo messaggio prevede tre coppie nome valore:

- *readmode* indica se il circuito di comando sta operando in modalità manuale (0) o automatica (1).

- *readfwrw* comunica il valore della posizione della leva del joystick sulla direzione avanti-indietro. I valori ammissibili vanno da -1 a 1, con 0 che indica la posizione di riposo della leva, i valori positivi indicano la leva in avanti e quelli negativi leva indietro.

- *readrxlx* comunica il valore della posizione della leva del joystick sulla direzione destra-sinsitra. I valori ammissibili vanno da -1 a 1, con 0 che indica la posizione di riposo della leva, i valori positivi indicano la leva spostata verso destra e quelli negativi leva a sinistra.

`MSG_FROM_MOTION_INFO` completa le informazioni del messaggio precedente con i seguenti dati:

- *readdatacount* indica il numero di trame lette dalla porta seriale.

- *readdatacounterr* indica il numero di errori riscontrati nelle trame.

- *countsend* indica il numero di trame di comando inviate al circuito.

Le informazioni di questo messaggio risultano utili per il log del funzionamento dell'applicazione software e per il debug, ma non sono strettamente necessarie per il controllo della carrozzina.

La sequenza di operazioni svolte ciclicamente da MotorExpert è la seguente:

1. Lettura dei dati relativi alla posizione della leva e invio dei messaggi di tipo `MSG_FROM_MOTION_BRIEF` e `MSG_FROM_MOTION_INFO`.

2. Controllo della disponibiltià di messaggi di tipo `MSG_FROM_BRIAN`; in caso affermativo è necessario aggiornare i valori di attuazione da inviare al circuito di interfaccia.

3. Se al punto 1 si è rilevato che il circuito è in modalità automatica invio del valore di attuazione.

I valori della posizione della leva comunicati dal circuito e quelli del comando inviati dal computer sono interi compresi tra 0 e 255, con 128 come valore della posizione di riposo. MotorExpert si occupa di convertirli secondo una banale corrispondenza lineare con i più intuitivi valori compresi tra -1 e 1. Si noti che per mantenere attiva la comunicazione con il circuito di controllo, il valore di attuazione va inviato periodicamente al circuito di interfaccia. Infatti il circuito è dotato di un meccanismo di sicurezza basato un timeout che, qualora non vengano ricevuti comandi di attuazione, imposta la modalità di funzionamento a "manuale". Questo non è un comportamento desiderabile e si è scelto quindi di inviare periodicamente il valore di attuazione, mantenendo il valore di attuazione precedente qualora non si ricevono messaggi che specificano nuovi valori, come descritto al punto 2. Al punto 3 si è invece introdotta una condizione sull'invio dei messaggi al circuito, ovvero che la modalità di funzionamento rilevata (comunicata dal circuito stesso nella trama letta al punto 1) sia automatica. È infatti inutile inviare trame di comando al circuito di comando quando questo è in modalità manuale, in quanto tali messaggi sarebbero ignorati.

Il membro BrianExpert è il cuore del controllo della carrozzina e gestisce l'interfaccia con il controllore a comportamenti fuzzy Mr.Brian. Considerando la sola funzionalità di guida con il joystick, è evidente che i dati di ingresso per il controllore fuzzy sono solo quelli specificati nel messaggio `MSG_FROM_MOTION_BRIEF` inviato dal MotorExpert, ovvero *readmode*, *readfwrw* e *readrxlx*. La fuzzyficazione di *readmode* avviene con il tipo di dato fuzzy `BOOLEANFLAG`, *readfwrw* con `FWRWSPEEDREAD` e *readrxlx* con `RXLXSPEEDREAD`. Gli insiemi fuzzy utilizzati sono illustrati in Figura 6.2. Le uscite del controllore sono i valori di *modecmd*, *fwrwcmd* e *rxlxcmd* che costituiscono il

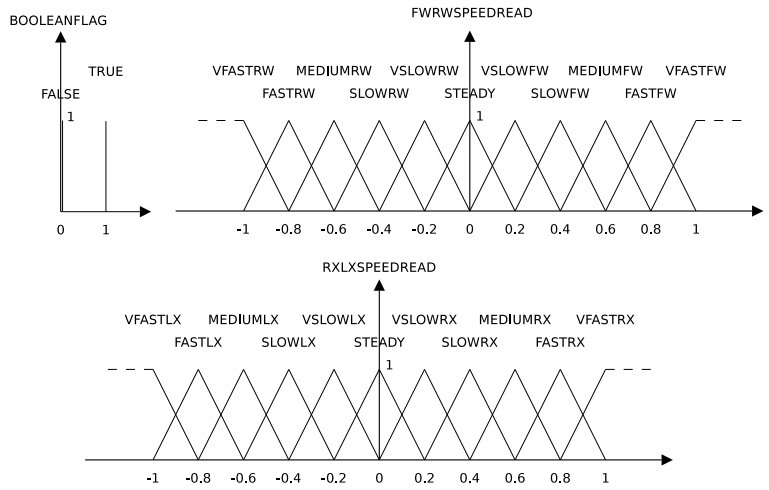


Figura 6.2: Insiemi fuzzy utilizzati per la fuzzyficazione dei dati

messaggio `MSG_FROM_BRIAN` precedentemente illustrato. La defuzzyficazione di *modecmd* avviene con l'insieme di insiemi fuzzy `BOOLEANFLAGCMD`, *fwrwcmd* con `FWRWSPEEDCMD` e *rxlxcmd* con `RXLXSPEEDCMD`. Gli insiemi fuzzy utilizzati sono visibili in Figura 6.3.

I predicati definiti sui dati di ingresso sono `Modereadauto` che assume valore vero quando *readmode* vale `TRUE`, `Modereadmanual` che è la negazione del predicato precedente; i predicati `FwRwSpeedSteady`, `FwSpeedVerySlow` ... `FwSpeedVeryFast`, `RwSpeedVerySlow` ... `RwSpeedVeryFast` sono associati ai valori *readfwrw* e altrettanti predicati analoghi sono associati a *readrxlx*.

Per la funzionalità ipotizzata fino a questo punto, ovvero di "copiare" le posizioni della leva del joystick e permettere quindi il movimento della carrozzina, è sufficiente utilizzare un solo comportamento, che prende il nome di `FollowJoystickMotion`. Le regole che specificano questo comportamento sono molto semplici ed intuitive, quindi se ne riportano solo alcune a titolo d'esempio:

```
(FwRwSpeedSteady) =>  (fwrwcmd   STEADY);
(FwSpeedVerySlow) =>  (fwrwcmd  VSLOWFW);
                  ...
      (LxSpeedFast) =>  (rxlxcmd   FASTLX);
(LxSpeedVeryFast) =>  (rxlxcmd  VFASTLX);
```

Il significato delle regole è intuitivo e permette di copiare i valori proposti dalla leva del joystick sui dati di uscita.

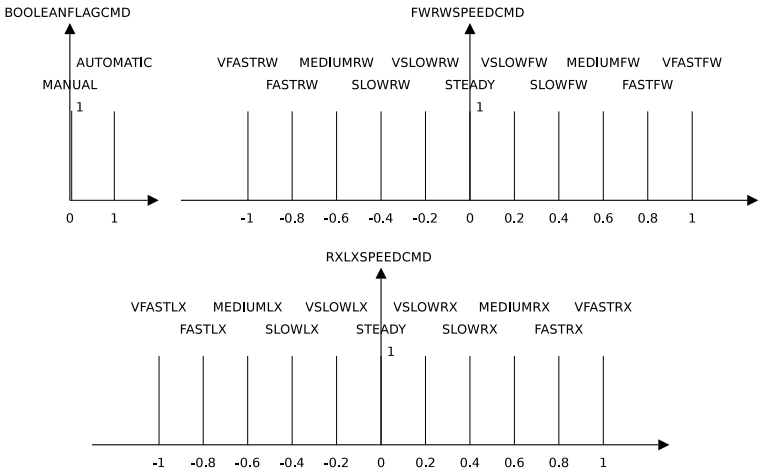


Figura 6.3: Insiemi fuzzy utilizzati per la defuzzyficazione delle uscite del controllore

In realtà è necessario introdurre un comportamento banale, che sarà attivo sempre durante il funzionamento del sistema, chiamato `ModeCmdAuto` ed è costituito da una sola regola:

$$(\texttt{Always}) => \quad (\texttt{modecmd AUTOMATIC});$$

Questo comportamento si occupa di mantenere la modalità di funzionamento automatica, in quanto il predicato `Always` è sempre vero.

Il membro BrianExpert svolge dunque i seguenti compiti:

1. Riceve i messaggi di ingresso di tipo `MSG_FROM_MOTION_BRIEF`.
2. Esegue il ciclo di controllo di Mr.Brian con i due comportamenti `FollowJoystickMotion` e `ModeCmdAuto` descritti.
3. Invia il messaggio `MSG_FROM_BRIAN` con l'azione di controllo calcolata al punto precedente.

Lo schema a blocchi del sistema composto dai membri BrianExpert e MotorExpert è mostrato in Figura 6.4, mentre un diagramma che illustra il flusso dei messaggi in un ciclo di controllo è illustrato in Figura 6.5.

6.2.2 Guida con joypad

La prima estensione considerata al sistema proposto consiste nel supportare l'interazione tra due dispositivi di comando, aggiungendo anche il joypad senza fili e il membro JoypadExpert che si occupa della sua gestione.

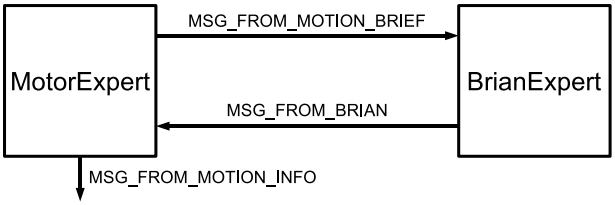


Figura 6.4: Schema a blocchi del sistema con i soli membri BrianExpert e MotorExpert

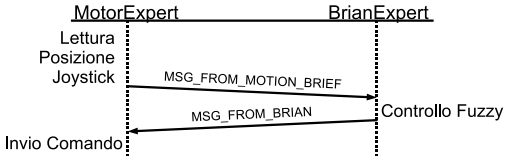


Figura 6.5: Flusso dei messaggi tra MotorExpert e BrianExpert

JoypadExpert si occupa di leggere lo stato del joypad e produrre messaggi di tipo `MSG_FROM_JOYPAD_BRIEF` e `MSG_FROM_JOYPAD_INFO`.

Il messaggio `MSG_FROM_JOYPAD_BRIEF` contiene due coppie nome valore che hanno il compito di indicare il valore per il comando della carrozzina proposto dal joypad:

- *joypadfwrw* indica la posizione della leva del joypad sull'asse avanti-indietro, compresa tra -1 e 1.
- *joypadrxlx* indica la posizione della leva del joypad sull'asse destra-sinsitra, compresa tra -1 e 1.

Il messaggio `MSG_FROM_JOYPAD_INFO` contiene informazioni aggiuntive sullo stato del joypad che non sono necessarie per il controllo della carrozzina. Le coppie nome valore sono le seguenti:

- *scalefactor* indica a quanto è impostato il valore massimo del fondo scala per il joypad. Come introdotto nella Sezione 4.2.6, è possibile limitare il valore del fondo scala del comando del joypad utilizzando i pulsanti numerati da 1 a 4, con corrispondenti valori di fondo scala 0.4, 0.6, 0.8 e 1. Si noti che questa informazione non è necessaria per controllare la carrozzina. Infatti, qualora si stia utilizzando un valore di fondo scala diverso da 1 è il membro stesso che provvede a effettuare la proporzione e inviare nel messaggio `MSG_FROM_JOYPAD_BRIEF` i valori

corretti. Ad esempio, considerando valore di fondo scala 0.4, se la leva del joypad è in posizione "avanti tutta", con corrispondente valore pari a 1, *joypadfwrw* assume valore 0.4; se la leva fosse esattamente a metà tra la posizione di riposo e la posizione "avanti tutta", con valore teorico 0.5, il valore di *joypadfwrw* è 0.2.

- *joychannel* indica quale dei tre controlli disponibili sul joypad si sta utilizzando per comandare la carrozzina: 0 indica la leva analogica di sinistra, 1 quella di destra e 2 il controllo digitale. Premendo il pulsante associato a una leva analogica si imposta il comando con quella leva, oppure, qualora fosse già attiva quella leva per il comando, si passa il controllo al pannello digitale.

- *joyreqstop* assume valore 0 o 1. Quando assume valore 1 indica la richiesta da parte dell'utente di interrompere l'esecuzione del programma di controllo della carrozzina, espressa mediante la pressione del pulsante 10 presente sul joypad. Questa funzionalità è utile in fase di test e debug dell'applicazione, ma sarà rimossa quando si otterrà un applicazione definitiva.

L'analogia dei dati contenuti nel messaggio `MSG_FROM_JOYPAD_BRIEF` con quelli di `MSG_FROM_MOTION_BRIEF` è evidente e permette di estendere il controllore fuzzy al fine di supportare la guida combinata con i due dispositivi finora analizzati. Il sistema di controllo risulta dunque esteso, comprendendo ora oltre a MotorExpert e BrianExpert anche il modulo JoypadExpert, come visibile in Figura 6.6.

Per supportare contemporaneamente il comando da parte di joystick e joypad è necessario aggiungere le seguenti funzionalità al controllore Mr.Brian:

- Fuzzyficazione dei dati contenuti nei messaggi `MSG_FROM_JOYPAD_BRIEF` e introduzione dei predicati necessari.

- Specifica del comportamento `FollowJoypadMotion`.

- Attuazione mutuamente esclusiva di uno solo tra `FollowJoypadMotion` e `FollowJoystickMotion`.

La fuzzyficazione dei dati contenuti nei messaggi di tipo `MSG_FROM_-JOYPAD_BRIEF` avviene con l'insieme di insiemi fuzzy `FWRWSPEEDREAD` e `RX-LXSPEEDREAD` precedentemente introdotti per la guida con il joystick. Anche i predicati definiti sono molto simili a quelli introdotti per il joystick; a titolo

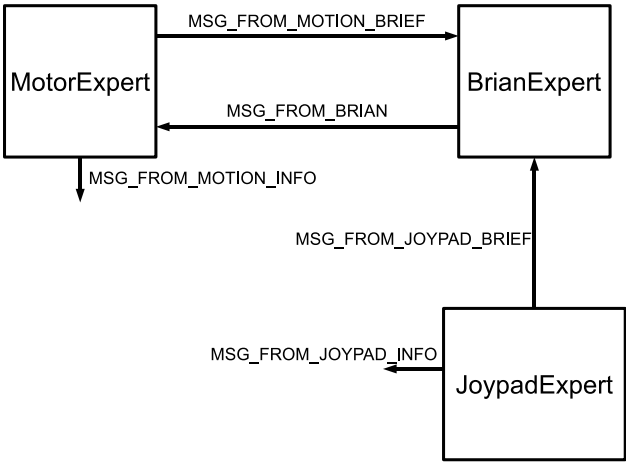


Figura 6.6: Schema a blocchi del sistema con i membri BrianExpert, MotorExpert e JoypadExpert

di esempio se ne riportano solo alcuni:

```
FwRwSpeedJoySteady =   (D  joyfwrw   STEADY);
FwSpeedJoyVerySlow =   (D  joyfwrw  VSLOWFW);
                    ...
    LxSpeedJoyFast =   (D  joyrxlx   FASTLX);
LxSpeedJoyVeryFast =   (D  joyrxlx  VFASTLX);
```

Il comportamento `FollowJoypadMotion` è analogo a `FollowJoystick-Motion` e permette di "copiare" sulle uscite del controllore i movimenti imposti con il joypad. Ad esempio, alcune regole sono:

```
(FwRwSpeedJoySteady) =>  (fwrwcmd   STEADY);
(FwSpeedJoyVerySlow) =>  (fwrwcmd  VSLOWFW);
                      ...
    (LxSpeedJoyFast) =>  (rxlxcmd   FASTLX);
(LxSpeedJoyVeryFast) =>  (rxlxcmd  VFASTLX);
```

La mutua esclusione tra i dispositivi di comando è necessaria per garantire all'utente che un solo dispositivo alla volta può guidare la carrozzina, ma allo stesso tempo è importante che il meccanismo di scelta del dispositivo sia automatico, ovvero che l'utente non debba specificare quale dispositivo vuole utilizzare. Si è realizzato un meccanismo di mutua esclusione che da

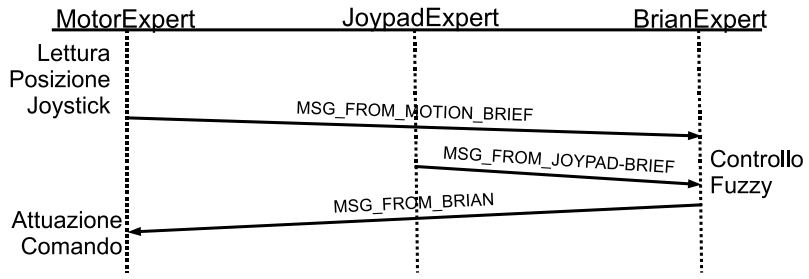


Figura 6.7: Sequenza di esecuzione con i membri BrianExpert, MotorExpert e JoypadExpert e i messaggi scambiati

priorità ai comandi della leva del joystick: se il joystick è in posizione di riposo, ovvero con la leva in posizione centrale, allora il joypad può comandare la carrozzina, ovvero il comportamento `FollowJoypadMotion` è attivo e `FollowJoystickMotion` non lo è; quando il joystick non è in posizione di riposo, il comando imposto dal joypad viene ignorato, ovvero il comportamento attivo è `FollowJoystickMotion`. Sfruttando le condizioni CANDO, introdotte nella Sezione 3.6.1, che governano l'attivazione dei comportamenti di Mr.Brian è possibile gestire la mutua esclusione con le condizioni appena descritte.

Un esempio di sequenza di esecuzione del sistema con i tre membri finora introdotti è mostrata in Figura 6.7. BrianExpert riceve i messaggi di MotorExpert e di JoypadExpert e, dopo aver svolto il ciclo di controllo, invia il messaggio per l'attuazione a MotorExpert.

6.2.3 Riproduzione di file audio

Una funzionalità che è stata aggiunta alla carrozzina, ma che non è assolutamente legata al controllo del movimento della carrozzina stessa, è la riproduzione di file audio. Grazie alla creazione di un membro chiamato AudioExpert è possibile richiedere l'esecuzione di contenuti multimediali alla carrozzina. Il membro AudioExpert ha un database di file audio identificati da un codice numerico e attiva la riproduzione di un file audio nel momento in cui riceve un messaggio di richiesta con un codice numerico valido.

Lo sviluppo di questo membro si è reso necessario per permettere di utilizzare la carrozzina durante il Festival della Scienza di Genova 2007 [2] come "robot guida". Durante l'esposizione la carrozzina era teleguidata con il joypad e, quando era necessario, l'utente che la guidava comandava

[2]Genova, 25 ottobre - 6 novembre 2007, http://festivalscienza.it

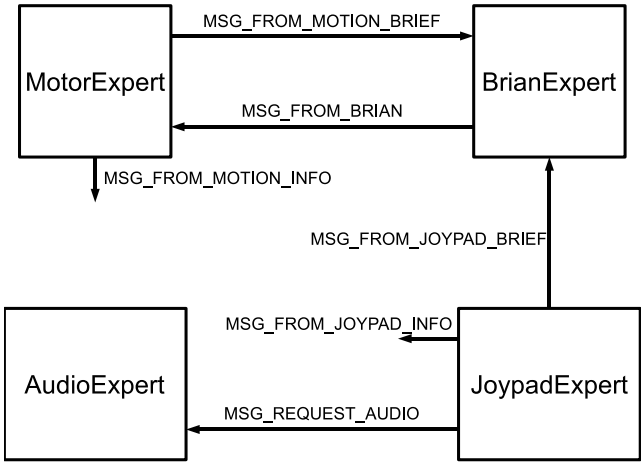


Figura 6.8: Schema a blocchi del sistema con i membri BrianExpert, MotorExpert, JoypadExpert e AudioExpert

l'esecuzione di brani di spiegazione dei luoghi raggiunti sempre utilizzando il joypad.

La funzionalità di riproduzione di contenuti multimediali può sembrare al momento superflua, ma potrà essere sviluppata in futuro per segnalare all'utente eventuali condizioni di errore o di pericolo, o anche per integrarla con un sistema di comando vocale per la carrozzina.

A ogni invocazione AudioExpert attende un messaggio di tipo `MSG_REQUEST_AUDIO` che contiene una sola coppia nome valore di nome *reqaudio* che specifica il codice numerico del file audio da eseguire. L'attesa del messaggio è bloccante, ovvero, a differenza dei membri presentati fino a ora, se non vengono inviati messaggi di tipo `MSG_REQUEST_AUDIO` il membro non termina la sua esecuzione. Come già detto, l'unico membro che attualmente si occupa di produrre tali messaggi è JoypadExpert, che è stato esteso associando alla pressione dei bottoni numerati 5, 6, 7 e 8 l'invio di messaggi di richiesta dei file audio con indice rispettivamente pari a 0,1,2 e 3.

Lo schema complessivo del sistema, con i membri MotorExpert, BrianExpert, JoypadExpert e AudioExpert è mostrato in Figura 6.8.

6.2.4 Evitamento ostacoli

Uno dei compiti a cui deve assolvere il software di controllo è quello di garantire la sicurezza dell'utente durante la guida, provvedendo a integrare le informazioni sensoriali per proporre comportamenti di evitamento collisioni ed

evitamento ostacoli. I sensori che permettono di rilevare gli ostacoli presenti nell'ambiente sono i due scanner laser Hokuyo URG-04LX, montati in posizione simmetrica uno a destra e uno a sinistra della carrozzina. Il membro che si occupa della gestione di un singolo scanner laser è HokuyoExpert.

HokuyoExpert acquisisce periodicamente le informazioni da un sensore laser Hokuyo URG-04LX e permette di trattarle opportunamente per produrre informazioni sintetiche. Ogni membro si occupa di un solo sensore laser, quindi saranno attivi contemporaneamente due membri di questo tipo, uno per il sensore laser di sinistra e uno per quello di destra. Ogni membro è a conoscenza del sensore che rappresenta, ovvero all'atto della creazione viene associato al membro una stringa che identifica in modo univoco il sensore (e.g., *left* e *right*). Quando un membro di tipo HukuyoExpert viene inizializzato, esso invia allo scanner laser il comando di scansione a ciclo continuo, in modo che il laser provveda a inviare periodicamente i dati della scansione senza bisogno di alcun altro comando. A ogni attivazione il membro legge i dati dal sensore laser a esso associato e, qualora la lettura dei dati non vada a buon fine, provvede a inviare un comando di reset al sensore laser, inviando prima un messaggio d'errore (di tipo `MSG_FROM_HOKUYO_ERR`) che indica il malfunzionamento temporaneo del sistema. Il contenuto di questo messaggio è costituito da una sola coppia nome valore. Il nome indica il sensore laser che non funziona correttamente, mentre il valore è fissato a 1. La gestione del messaggio d'errore verrà descritta nella Sezione 6.2.9, quando si introdurrà il modulo che gestisce gli errori del sistema.

Per perseguire l'obiettivo di evitare gli ostacoli è necessario che le informazioni sulla distanza degli ostacoli rilevate con i sensori laser siano disponibili al controllore del movimento, rappresentato in questo caso da Mr.Brian. Dato che il numero di dati fornito dai sensori laser a ogni scansione è elevato (682 rilevazioni, una ogni 0.36 ° circa su un range di 240 °), è impossibile trattare tali informazioni una a una nel controllore fuzzy, ma è necessario rappresentarle in maniera più sintetica. Il membro HokuyoExpert permette di rappresentare sinteticamente i dati definendo un certo numero di zone angolari sui dati delle scansioni all'interno delle quali viene scelto come valore riassuntivo la distanza inferiore riscontrata. Considerando, ad esempio, il range di scansione di 240 ° con le 682 misure di distanza rilevate e una suddivisione uniforme in 4 zone da 60 ° ciascuna, il sistema seleziona le 4 distanze minime rilevate nelle quattro zone. Si noti che il numero di zone e i limiti angolari non sono fissati a priori ma configurabili.

I dati sintetici devono essere inviati al membro BrianExpert per essere utilizzati nel controllore fuzzy. Il messaggio che si occupa di descrivere le informazioni sulla distanza degli ostacoli è `MSG_FROM_HOKUYO_MINDIST`. A ogni

zona identificata corrisponde una coppia nome valore, quindi il contenuto del messaggio varia in base all'ampiezza delle zone che si desidera controllare. Il nome associato a ogni zona è composto dal prefisso *mindist_* seguito dal nome del sensore laser che ha effettuato la rilevazione (e.g. *left* o *right*), dal suffisso *_zone_* e da un numero che indica la zona, nella seguente forma *mindist_*⟨nome⟩*_zone_*⟨n⟩ . Il numero della zona è progressivo procedendo in senso antiorario, partendo da 0. Ad esempio, considerando il membro che si occupa del sensore laser di sinistra e definendo quattro zone con limiti $[-120 \quad -60 \quad 0 \quad 60 \quad 120]$, alla zona $[-120 \quad -60]$ corrisponde il nome *mindist_left_zone_0*, alla zona $[-60 \quad 0]$ il nome *mindist_left_zone_1* e così via. Per uno studio più dettagliato della suddivisione in zone effettivamente utilizzata si rimanda alla Sezione 7.4.

Il controllore fuzzy deve essere esteso per ragionare anche sui dati sintetici prodotti dai sensori laser. A ogni variabile *mindist_*⟨nome⟩*_zone_*⟨n⟩ deve essere associato un tipo di dato fuzzy per la fuzzyficazione dei dati e dei predicati per trattarli. Supponendo di definire gli insiemi `VICINO` `MEDIO`, `LONTANO` e `MOLTOLONTANO` e di associarli a un insieme `TDISTANZA`, è possibile creare dei predicati che descrivono la distanza dall'ostacolo rilevato, il sensore a cui si riferiscono e la zona considerata. A questo livello non è opportuno utilizzare come nome della zona il numero progressivo che la identifica, ma è più intuitivo descrivere la zona in base al suo orientamento relativo alla carrozzina (e.g., Nord, NordEst, NordOvest, etc.). Supponendo che la zona numero 3 dello scanner laser di sinistra sia rivolta a Nord relativamente alla carrozzina, si possono definire i seguenti predicati:

```
LeftVicinoNord =    (D  mindist_left_zone_3        VICINO);
 LeftMedioNord =    (D  mindist_left_zone_3         MEDIO);
LeftLontanoNord =   (D  mindist_left_zone_3       LONTANO);
LeftMoltoLontanoNord = (D  mindist_left_zone_3  MOLTOLONTANO);
```

Per intervenire sul controllo del moto della carrozzina in modo da evitare gli ostacoli è necessario creare il comportamento `ObstacleAvoidMinDist`. Per garantire che questo comportamento sia indipendente dal dispositivo di comando in uso e possa agire sulle azioni proposte dall'utente che comanda la carrozzina, è necessario inserirlo in un livello superiore della gerarchia di controllo. I comportamenti finora introdotti per Mr.Brian erano infatti tutti di livello 1 e proponevano azioni tramite le variabili *fwrwcmd* e *rxlxcmd*. Collocando al secondo livello della gerarchia di controllo il comportamento per l'evitamento ostacoli è possibile riutilizzare le variabili di uscita come ingressi del sistema e agire, considerando i predicati introdotti

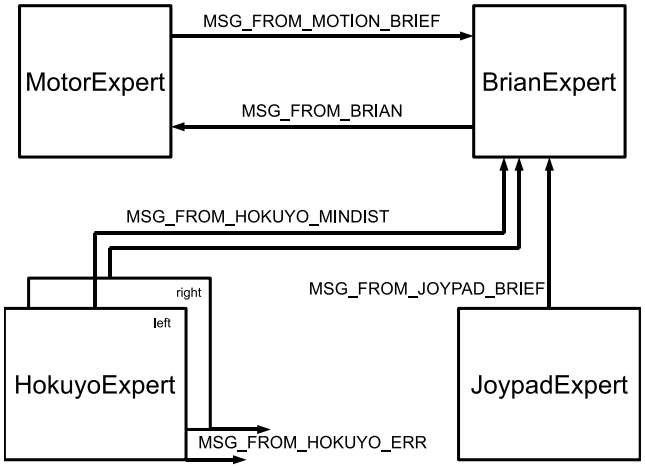


Figura 6.9: Schema a blocchi del sistema con i membri BrianExpert, MotorExpert, JoypadExpert e HokuyoExpert. AudioExpert è stato tralasciato per semplicità

che descrivono gli ostacoli, in modo da evitare collisioni. Ad esempio, un predicato molto semplice che può essere implementato per evitare che la carrozzina si muova in avanti quando ha un ostacolo vicino frontalmente è il seguente:

```
(AND (OstacoloVicinoFrontale)
  (DirAvanti)
)
=>
(&DEL.fwrwcmd ANY)
(fwrwcmd STEADY);
```

dove `OstacoloVicinoFrontale` è un predicato che assume valore vero quando in almeno una delle zone che si trovano di fronte alla carrozzina è stata rilevata una distanza definita vicina nella fuzzyficazione dei dati, mentre `DirAvanti` indica che il comando proposto dal livello inferiore del controllo (ovvero dai comportamenti che comandano la carrozzina con il joystick o con il joypad) è in direzione frontale.

Lo schema del sistema di controllo esteso con i due membri HokuyoExpert e i messaggi da loro prodotti è mostrato in Figura 6.9. Si noti che per semplicità il membro AudioExpert non è più rappresentato. Lo schema del controllore fuzzy a 2 livelli è mostrato in Figura 6.10.

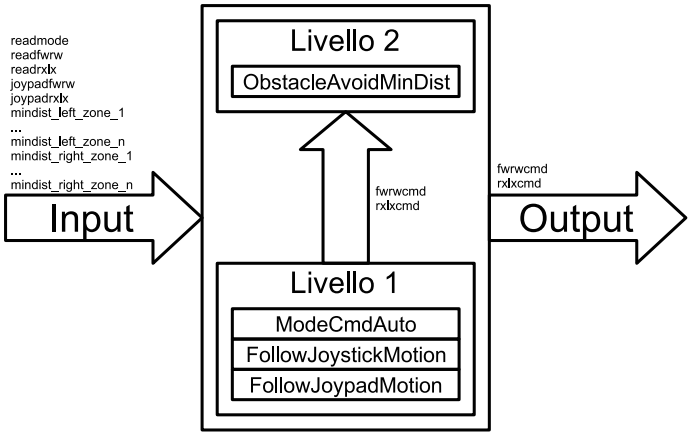


Figura 6.10: Controllore fuzzy a 2 livelli per la gestione di joystick, joypad e evitamento ostacoli grazie all'uso dei sensori laser

6.2.5 Informazioni sul moto della carrozzina

Per poter sviluppare un sistema di evitamento ostacoli efficiente è necessario, oltre a uno studio approfondito delle zone da utilizzare per la rilevazione degli ostacoli e delle regole da specificare nel comportamento `ObstacleAvoidMinDist`, conoscere le informazioni che riguardano il moto della carrozzina. Queste informazioni, rese disponibili dal sensore XSens Mti presentato nella Sezione 3.2, possono essere utilizzate per migliorare il sistema di evitamento ostacoli e per permettere una linea di sviluppo del sistema in varie direzioni.

Il membro introdotto nel sistema per la gestione dei dati provenienti dal sensore XSens MTi è chiamato MTiExpert. La funzione svolta da questo membro è di leggere periodicamente le informazioni dal sensore MTi e inviarle in un messaggio di tipo `MSG_FROM_IMU`. Le coppie nome valore contenute in questo messaggio sono:

- *imu_period* specifica il periodo trascorso tra una lettura e l'altra. Tipicamente questo valore è fisso a 20ms, ma qualora esso dovesse cambiare, sarà possibile comunicarlo agli altri membri, rendendo il sistema flessibile.

- *imu_acc_x*, *imu_acc_y* e *imu_acc_z* specificano le accelerazioni registrate sui tre assi in m/s^2.

- *imu_gyr_x*, *imu_gyr_y* e *imu_gyr_z* specificano le velocità di rotazione angolare in rad/s rilevate intorno ai tre assi.

- *imu_mag_x*, *imu_mag_y* e *imu_mag_z* specificano l'intensità del campo magnetico terrestre rilevato lungo i tre assi.

Si noti che nella Sezione 4.2.2 si è ipotizzato che non tutti i dati forniti dal sensore sono necessari allo sviluppo del sistema di controllo, ma si è preferito realizzare il messaggio completo per permettere l'estensione del sistema in modo semplice: ad esempio, qualora si voglia estendere il sistema in modo da utilizzare le informazioni sull'intensità del campo magnetico, non sarà necessario modificare questo membro, ma crearne uno che lavora sui dati contenuti nel messaggio `MSG_FROM_IMU`.

I dati sensoriali provenienti dal sensore XSens MTi e comunicati tramite il messaggio `MSG_FROM_IMU` potrebbero essere utilizzati direttamente dal controllore fuzzy. Si potrebbero infatti creare delle regole che coinvolgono anche i dati prodotti da questo sensore nei comportamenti. Per garantire una maggiore estendibilità del sistema si è però introdotto un nuovo membro, chiamato PoseExpert. Analizzato a scatola chiusa, 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. La parte relativa alla posizione della carrozzina verrà trattata nella Sezione 6.2.6, dove si introducono i membri che gestiscono il sistema di localizzazione. Per quanto riguarda le informazioni sulla velocità, PoseExpert opera da semplice ripetitore e integratore delle informazioni inviate da MtiExpert. Ad ogni ciclo infatti controlla se è disponibile un messaggio di tipo `MSG_FROM_IMU` e ne preleva i dati trasmessi a cui è interessato. Come specificato nella Sezione 4.2.2, i dati utili sono la velocità di rotazione intorno all'asse z e il modulo della velocità lungo gli assi x e y. La velocità di rotazione intorno all'asse z è disponibile in *imu_gyr_z* ed è espressa in radianti al secondo. PoseExpert si occupa di effettuare banalmente la conversione in gradi al secondo di questo valore. Per quanto riguarda la velocità sugli assi x, y PoseExpert si occupa di integrare i valori di *imu_acc_x* e di *imu_acc_y* nel tempo, implementando il metodo d'integrazione numerico dei trapezi: dette $a_x(\mathrm{t})$ e $a_y(\mathrm{t})$ le accelerazioni attuali, $a_x(\mathrm{t}-1)$ e $a_y(\mathrm{t}-1)$ le accelerazioni rilevate al ciclo precedente, Δt il periodo tra una rilevazione e l'altra dei dati delle accelerazioni è possibile calcolare la velocità lungo l'asse x e y con le seguenti formule:

$$\begin{aligned} v_x(\mathrm{t}) = &\quad v_x(\mathrm{t}-1) + (a_x(\mathrm{t}-1) + a_x(\mathrm{t})) * \Delta t/2 \\ v_y(\mathrm{t}) = &\quad v_y(\mathrm{t}-1) + (a_y(\mathrm{t}-1) + a_y(\mathrm{t})) * \Delta t/2 \end{aligned}$$

da cui si ricava facilmente il modulo della velocità:

$$v(\mathrm{t}) = \sqrt{v_x(\mathrm{t})^2 + v_y(\mathrm{t})^2} \tag{6.1}$$

Il messaggio `MSG_FROM_POSE_ACTUAL_SPEED` prodotto da PoseExpert contiene tre coppie nome valore:

- *speed_module* contiene il valore del modulo della velocità della carrozzina, calcolato con l'Equazione 6.1.
- *speed_angle* contiene la velocità rotazionale intorno all'asse z rilevata con il sensore XSens Mti e trasformata in gradi al secondo.
- *know_speed* assume valore sempre pari a 1, indica che la velocità è nota.

Con la creazione di MTiExpert e PoseExpert è necessario estendere nuovamente il controllore fuzzy realizzato. Infatti, i messaggi di tipo `MSG_FROM_POSE_ACTUAL_SPEED` possono essere utilizzati per creare dei sistemi di evitamento ostacoli più complessi e allo stesso tempo più efficienti: la conoscenza della velocità di movimento della carrozzina rende infatti possibile prendere decisioni più precise sulle manovre da compiere per evitare gli ostacoli. Ad esempio, qualora si riscontrasse la presenza di un ostacolo in posizione frontale a una distanza di un metro dalla carrozzina, conoscendo la velocità della carrozzina stessa è possibile creare diverse regole di comportamento: se la velocità è elevata è necessario frenare la carrozzina, ovvero comandare un "indietro tutta", se invece la velocità è bassa è sufficiente inibire le azioni che porterebbero la carrozzina ad accelerare in direzione frontale. Questo tipo di ragionamento ben si adatta alla struttura del controllore fuzzy realizzata, infatti è sufficiente far ricevere al membro BrianExpert i messaggi di tipo `MSG_FROM_POSE_ACTUAL_SPEED`. I valori *speed_angle* e *speed_module* vengono fuzzyficati con opportuni insiemi fuzzy (e.g., `ROTVELOCEORARIA`, `ROTVELOCEANTIORARIA`,... `ROTVELOCEORARIA` e `VELOCE`,`MEDIO`,`LENTO`,`FERMO`) e, una volta definiti i predicati associati, è possibile realizzare comportamenti più complessi. Ad esempio, considerando l'esempio utilizzato nella Sezione 6.2.4, si potranno aggiungere le seguenti condizioni:

```
(AND (OstacoloVicinoFrontale)
  (AND (DirAvanti)
    (VelAttualeElevata)
  )
)
```

```
=>
(&DEL.fwrwcmd ANY)
(fwrwcmd VFASTRW);

(AND (OstacoloVicinoFrontale)
  (AND (DirAvanti)
    (VelAttualeBassa)
  )
)
=>
(&DEL.fwrwcmd ANY)
(fwrwcmd STEADY);
```

I predicati `VelAttualeBassa` e `VelAttualeElevata` si riferiscono al valore della velocità tangenziale calcolato.

Un'altra estensione introdotta, che riguarda il solo membro BrianExpert, consiste nel considerare la velocità sia attuale che precedente. Il confronto tra le due velocità, rotazionali o tangenziali, permette di stimare come varia nel tempo la velocità e regolare di conseguenza l'azione di controllo. Ad esempio, definendo i predicati `VelPrecedenteBassa` e `VelPrecedenteElevata` è possibile creare ulteriori regole di controllo per migliorare il comportamento dinamico della carrozzina. Allo stesso modo si possono utilizzare predicati relativi alla velocità rotazionale. La struttura del controllore fuzzy risulta ulteriormente modificata ed è mostrata in Figura 6.11, mentre l'architettura del sistema software a cui sono stati aggiunti i membri MtiExpert e PoseExpert è visibile in Figura 6.12.

Importanza di PoseExpert

La struttura di PoseExpert è molto semplice e la presenza di un membro di questo genere può sembrare superflua, in quanto, per quanto visto finora, questo membro svolge solo semplici calcoli sui dati sensoriali e propone i risultati in uscita. L'importanza di questo mebro è però evidente quando si considerano possibili evoluzioni del sistema. Considerando, ad esempio, di dotare la carrozzina di un sistema di odometria, la presenza di PoseExpert rende agevole l'introduzione di un nuovo membro. Supponendo di chiamare il nuovo membro OdometryExpert e di voler mantenere attivo anche il membro MtiExpert, è possibile modificare il solo membro PoseExpert in modo che esso utilizzi le informazioni sia di OdometryExpert che di MTiExpert. Il controllore fuzzy non dovrà essere esteso per gestire i nuovi dati, in quando esso riceverà solo i messaggi prodotti da PoseExpert. PoseExpert potrebbe

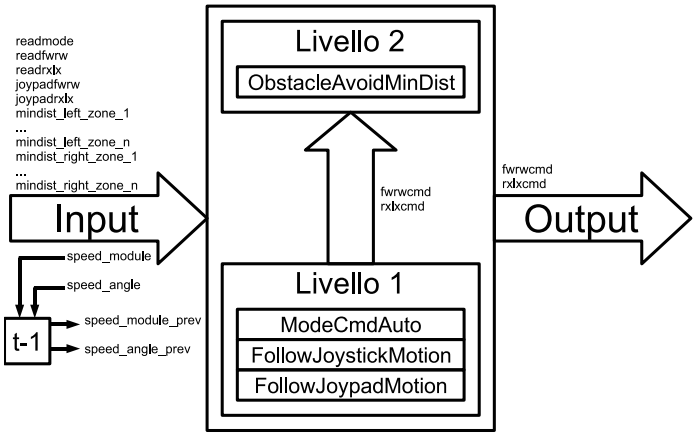


Figura 6.11: Controllore fuzzy a 2 livelli per la gestione di joystick, joypad, evitamento ostacoli grazie all'uso dei sensori laser e delle informazioni sulla velocità della carrozzina attuale e precedente

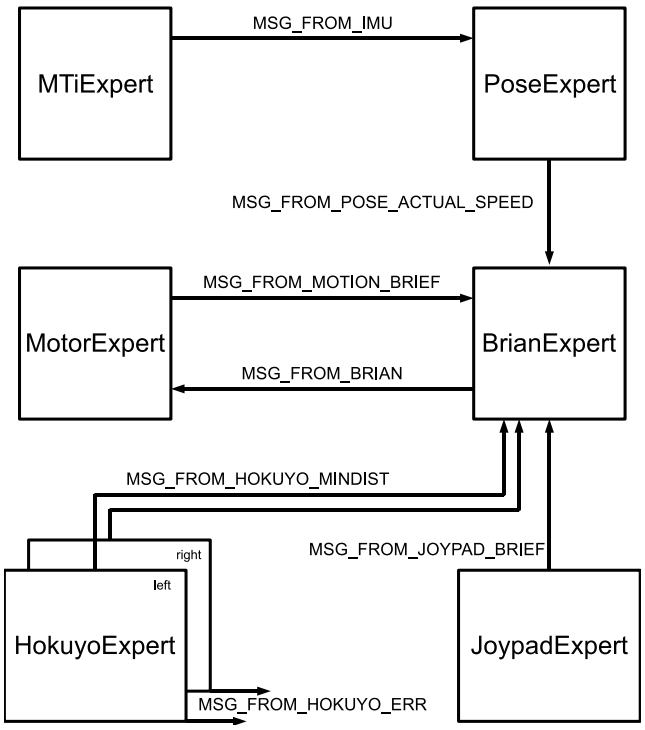


Figura 6.12: Schema a blocchi del sistema con i membri BrianExpert, MotorExpert, JoypadExpert, HokuyoExpert e PoseExpert

utilizzare le informazioni dei membri a esso collegati in modo differente: se, ad esempio OdometryExpert fosse molto preciso sulle basse velocità ma poco sulle velocità elevate, mentre MTiExpert sviluppasse un comportamento opposto, sarebbe possibile introdurre un criterio di scelta che compila i messaggi informativi sulla velocità della carrozzina in base alla condizione di esercizio.

Per facilitare l'estensione del membro PoseExpert sono stati previsti un certo numero di messaggi finora inutilizzati. Uno di essi è `MSG_FROM_POSE_UNKNOW_ACTUAL_SPEED`, che comunica l'impossibilità di conoscere la velocità della carrozzina. Questo tipo di messaggio risulta molto utile se i sistemi utilizzati per rilevare la velocità della carrozzina non garantiscono di conoscere l'entità delle velocità in ogni momento. Il contenuto del messaggio è *know_speed* che assume sempre valore 0. Nella Sezione 6.2.5 si era introdotto il messaggio `MSG_FROM_POSE_ACTUAL_SPEED` che conteneva anch'esso *know_speed*, con valore fissato a 1. La variabile *know_speed* è molto importante per sviluppare condizioni di attivazione dei comportamenti (CANDO) nel controllore fuzzy. È stato possibile creare due comportamenti di evitamento ostacoli, uno più complesso ed efficiente attivo quando la velocità della carrozzina è nota, e uno più semplice, ma meno efficiente, attivo quando la velocità non è nota.

Una coppia di messaggi analoga a quella appena presentata è costituita da `MSG_FROM_POSE_ACTUAL_POS` e `MSG_FROM_POSE_UNKNOW_ACTUAL_POS`, che comunicano la posizione della carrozzina. Il metodo con cui viene calcolata la posizione della carrozzina è presentato nella Sezione 6.2.6, in questo momento è sufficiente notare come anche in questo caso la presenza del membro PoseExpert renda flessibile anche il calcolo della posizione della carrozzina. Supponendo infatti di disporre di un sistema di rilevamento assoluto della posizione che operi a una frequenza lenta e di un sistema di stima della velocità della carrozzina che opera a frequenza più elevata è possibile calcolare in modo preciso la posizione della carrozzina integrando le informazioni odometriche e utilizzando periodicamente il sistema assoluto per azzerare gli errori.

Il membro PoseExpert con i messaggi che produce è mostrato in Figura 6.13.

6.2.6 Rilevamento posizione in ambienti indoor

Nel Capitolo 5 si è introdotto un sistema di localizzazione che permette di conoscere la posizione di un robot mobile equipaggiato con una telecamera rivolta verso il soffitto su cui sono applicati dei fiducial marker. Il mem-

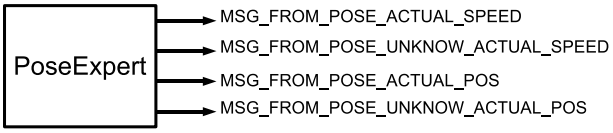


Figura 6.13: Struttura completa del membro PoseExpert

bro che 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. Il compito di questo membro è di acquisire periodicamente un fotogramma dalla telecamera e calcolare, quando possibile, la posizione della carrozzina rispetto al sistema di riferimento assoluto. Quando la posizione della carrozzina è disponibile, il membro VisionExpert invia 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. PoseExpert, come illustrato nella Sezione 6.2.5, è utilizzato per ora solo come "ripetitore" dei messaggi, ovvero ricevendo i messaggi da VisionExpert si occupa di comporre il messaggio `MSG_FROM_POSE_ACTUAL_POS`. Qualora PoseExpert rilevi che VisionExpert non ha inviato messaggi per un certo periodo di tempo, invia il messaggio `MSG_FROM_POSE_UNKNOW_ACTUAL_POS`.

Il messaggio `MSG_FROM_POSE_ACTUAL_POS` è costituito dai campi *position_x*, *position_y* e *position_a*, che hanno lo stesso significato dei campi X, Y e A del messaggio `MSG_FROM_VISION_POS`. Il messaggio `MSG_FROM_POSE_UNKNOW_ACTUAL_POS` non contiene invece alcun valore. Si noti che in questo caso non è necessario inserire un campo *know_pos*, analogo a *know_speed* del messaggio `MSG_FROM_POSE_ACTUAL_SPEED`, che assuma valore 1 quando la posizione è nota e 0 quando non lo è. L'utilità del flag *know_speed* è legata al suo utilizzo nelle condizioni CANDO del controllore fuzzy che utilizza direttamente le

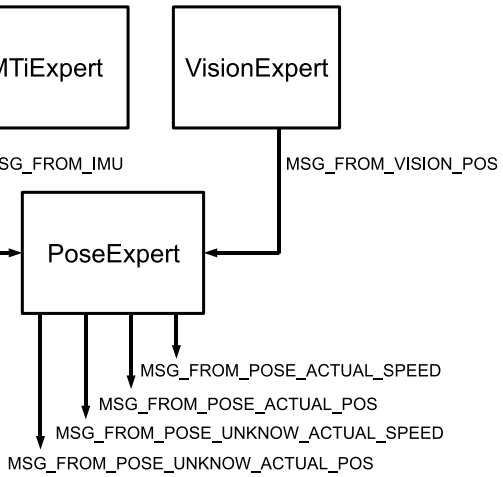


Figura 6.14: Sottosistema composto da VisionExpert, MTiExpert e PoseExpert

informazioni della velocità. Le informazioni sulla posizione della carrozzina non possono essere utilizzate direttamente dal controllore fuzzy, ma saranno utilizzate per pianificare ed eseguire percorsi (si veda la Sezione 6.2.7) e quindi è inutile utilizzare un flag che indichi la conoscenza della posizione della carrozzina.

L'architettura composta dai membri VisionExpert, MTiExpert e PoseExpert e i messaggi scambiati sono visibili in Figura 6.14.

Il membro PoseExpert potrebbe essere ulteriormente esteso per calcolare la velocità di movimento della carrozzina a partire dalle posizioni rilevate, ma si è scelto di non implementare questa funzionalità. Infatti la stima della velocità basata sulla differenza delle posizione nel tempo sarebbe disponibile solo quando il sistema si muove in ambienti noti, mentre il sistema di evitamento ostacoli è sempre attivo. Dato che le informazioni sulla velocità di movimento sono disponibili continuamente grazia alle informazioni provenienti da MTiExpert, si è scelto di non complicare ulteriormente il membro PoseExpert con l'introduzione di un nuovo sistema di stima della velocità che sarebbe disponibile saltuariamente.

6.2.7 Pianificazione ed esecuzione di percorsi

La pianificazione di percorsi è un'attività che coinvolge l'utente, il sistema di localizzazione e un modulo di pianificazione. L'utente deve specificare il punto di destinazione da raggiungere, il sistema di localizzazione è necessario per specificare il punto di partenza della pianificazione e per gestire

l'esecuzione del percorso, mentre il pianificatore è il modulo che effettivamente svolge il compito di cercare un percorso su una mappa da un punto di partenza a un punto di arrivo. Il membro che si occupa della pianificazione di un percorso è SpikeExpert; SequencerExpert gestisce la sequenza di operazioni che portano dalla richiesta di pianificazione alll'esecuzione del percorso e GuiExpert si occupa dell'interazione con l'utente.

Per prima cosa è importante definire il funzionamento del membro SpikeExpert. Come il nome del membro facilmente ricorda, il pianificatore in uso è Spike, presentato nella Sezione 6.1.2. SpikeExpert è un membro molto semplice, che attende l'arrivo di un messaggio di richiesta di pianificazione e risponde con un messaggio che descrive il piano prodotto oppure con un messaggio che comunica l'impossibilità di costruire un piano. Più in dettaglio, il messaggio atteso da SpikeExpert è `MSG_FROM_SEQUENCER_PLAN_REQUEST`, mentre i due messaggi che possono essere prodotti sono `MSG_FROM_SPIKE_VIA_POINT_LIST` e `MSG_FROM_SPIKE_GOAL_UNRECHABLE`.

La struttura del messaggio `MSG_FROM_SEQUENCER_PLAN_REQUEST` è composta da:

- *plan_x_start* e *plan_y_start* che specificano il punto di partenza del piano richiesto.
- *plan_x_goal* e *plan_y_goal* che specificano il punto di arrivo del piano richiesto.
- *line_p1_x*, *line_p1_y*, *line_p2_x*, *line_p2_y* specificano le coordinate di una linea che deve essere aggiunta alla mappa come ostacolo dinamico. È possibile ripetere più volte questa sequenza di quattro coppie nome valore per specificare più linee che rappresentano gli ostacoli dinamici.

SpikeExpert si pone in attesa di messaggi del suddetto tipo e, appena ne riceve uno, aggiunge gli ostacoli dinamici alla mappa e crea un percorso dal punto di partenza al punto di arrivo. Se non è possibile creare un percorso che collega il punto di partenza e il punto di arrivo viene inviato il messaggio `MSG_FROM_SPIKE_GOAL_UNRECHABLE` che non contiene alcuna informazione. Altrimenti viene generato un messaggio `MSG_FROM_SPIKE_VIA_POINT_LIST` che contiene la lista dei punti di via espressi con le coppie nome valore *point_x* e *point_y*. Si noti che gli ostacoli dinamici sono validi solo per la singola pianificazione, ovvero a una successiva pianificazione di un percorso gli ostacoli dinamici precedentemente utilizzati non saranno più presenti nella mappa.

SequencerExpert si occupa di gestire la pianificazione e l'esecuzione del percorso, infatti è il membro che produce il messaggio di richiesta di piani-

ficazione di un percorso atteso da SpikeExpert. La sequenza di operazioni che deve svolgere per creare un piano è la seguente:

1. Attendere la comunicazione del punto di destinazione.
2. Se è nota la posizione in cui si trova la carrozzina richiedere al membro che si occupa dei sensori laser il profilo degli ostacoli rilevati, in modo da creare gli ostacoli dinamici; altrimenti attendere che la posizione sia nota per effettuare questa operazione.
3. Richiedere la pianificazione a SpikeExpert dal punto in cui si trova la carrozzina al punto di goal specificato.
4. Attendere la lista dei via point del piano. Se non è possibile generare un piano, ritornare al punto 1, altrimenti iniziare la navigazione verso il primo via point. Quando si raggiunge l'ultimo via point terminare l'esecuzione del piano.

Inoltre si deve considerare che il piano può essere interrotto in qualsiasi momento, riportando il sistema al punto iniziale.

Il membro SequencerExpert è costantemente in attesa di messaggi di tipo `MSG_FROM_POSE_ACTUAL_POS` e `MSG_FROM_POSE_UNKNOW_ACTUAL_POS`, grazie ai quali mantiene aggiornate delle variabili di stato che indicano la posizione attuale della carrozzina o la non disponibiltà della posizione.

Per conoscere il punto di destinazione del piano, SequncerExpert attende un messaggio di tipo `MSG_GOAL_XY` che specifica la coordinata x e y della destinazione nei campi *goal_x* e *goal_y*. Se il punto da raggiungere e la posizione attuale sono noti SequencerExpert invia un messaggio di tipo `MSG_FROM_SEQUENCER_ASK_FOR_SEGMENTS`, con contenuto vuoto, che richiede ai sensori laser di inviare il profilo degli ostacoli rilevati. Per come finora è stato presentato il membro HokuyoExpert, questa funzionalità non è stata introdotta; l'estensione che permette di supportarla è presentata in seguito, per ora è sufficiente considerare che ognuno dei membri di gestione dei sensori laser risponde con un messaggio di tipo `MSG_FROM_HOKUYO_POINT_LIST`. È necessario attendere che entrambi i membri che si occupano della gestione delle informazioni degli scanner laser inviino una sequenza di punti che descrive il profilo degli ostacoli in coordinate cartesiane riferite al sensore laser stesso.

Per integrare nella mappa le informazioni sugli ostacoli dinamici rilevati è necessario trasformare i punti che li descrivono in modo da riferirli al sistema di riferimento assoluto. Sarà necessario quindi introdurre una rototraslazione che porta dal sistema di riferimento centrato sul sensore laser al

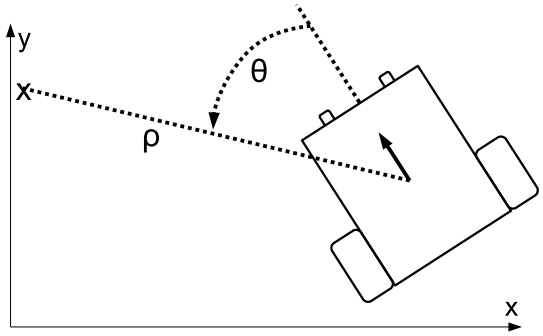


Figura 6.15: Sistema di rappresentazione delle informazioni che guidano il raggiungimento di un via point, dove con ρ si indica la distanza dal via point e con θ l'angolo tra la direzione attuale e la traiettoria

sistema di riferimento solidale con la carrozzina e successivamente, nota la posizione della carrozzina e il suo orientamento nel sistema di riferimento assoluto, calcolare la posizione dei punti nella mappa. Più precisamente, detta T_R^W la matrice che descrive la posizione della carrozzina nel sistema di riferimento assoluto, $T_{L_{left}}^R$ la trasformazione che porta dal sistema di riferimento solidale con la carrozzina al sistema di riferimento solidale con il sensore laser di sinistra, $P_i^{L_{left}}$ i punti rilevati dal sensore laser espressi nel sistema di riferimento solidale con il sensore laser stesso, è possibile per ogni punto rilevato calcolare la posizione nel sistema di riferimento assoluto con

$$P_i^W = T_R^W \cdot T_{L_{left}}^R \cdot P_i^{L_{left}} \tag{6.2}$$

Analogamente si procederà per i punti rilevati dal sensore laser di destra, utilizzando la matrice $T_R^{L_{right}}$.

Noti i punti che descrivono gli ostacoli dinamici rilevati al momento in cui si è scelto di pianificare un percorso, la posizione della carrozzina che rappresenta il punto di partenza e il punto di destinazione è possibile comporre e inviare il messaggio `MSG_FROM_SEQUENCER_PLAN_REQUEST` che richiede la pianificazione del percorso. Se la risposta a tale messaggio è `MSG_FROM_SPIKE_GOAL_UNRECHABLE`, l'esecuzione di SequencerExpert riprende dall'inizio, ovvero dall'attesa di un nuovo messaggio di specifica del punto di destinazione. Se la risposta è `MSG_FROM_SPIKE_VIA_POINT_LIST` allora SequencerExpert memorizza tutti i via point specificati dal pianificatore e comincia l'esecuzione del percorso.

L'esecuzione del percorso si basa sul raggiungimento sequenziale dei via point e il meccanismo con il quale la carrozzina viene guidata è basato su un comportamento fuzzy di nome `FollowViaPoint` aggiunto a Mr.Brian al

primo livello. Per permettere il funzionamento di tale comportamento è necessario specificare la distanza che separa la carrozzina dal punto da raggiungere e l'angolo che sussiste tra la direzione in cui è orientata la carrozzina e il segmento che congiunge il punto in cui si trova la carrozzina con il punto da raggiungere, come mostrato in Figura 6.15. Questi due dati, denotati con *rho_to_via_point* e *angle_to_via_point* e trasmessi nel messaggio `MSG_FROM_SEQUENCER_VIAPOINT_INFO`, permettono, una volta fuzzyficati e associati a predicati che li valutano, di scrivere regole che permettano il movimento autonomo della carrozzina, come ad esempio:

```
(AND (ViaPointNear
  (ViaPointN)
)
=>
(fwrwcmd VSLOWFW)
(rxlxcmd STEADY);
```

Dove con `ViaPointNear` si indica che la distanza dal punto da raggiungere non è elevata e con `ViaPointN` si indica che la carrozzina è orientata nella direzione della retta che la congiunge al punto da raggiungere.

Il messaggio `MSG_FROM_SEQUENCER_VIAPOINT_INFO` appena descritto è sempre accompagnato dal messaggio `MSG_FROM_SEQUENCER_EXECUTING_PATH` che contiene un solo campo, di nome *executing_path* con valore 1. Qualora durante l'esecuzione di un percorso la posizione della carrozzina diventa ignota, ad esempio a causa di un movimento che porta in un'area non coperta da tag, l'esecuzione automatica del percorso deve essere interrotta, in quanto non è più possibile produrre i valori di *rho_to_via_point* e *angle_to_via_point*. Per fare ciò SequencerExpert imposta *executing_path* a 0, inviando dunque il solo messaggio `MSG_FROM_SEQUENCER_EXECUTING_PATH` appena descritto. Quando la posizione della carrozzina ritorna disponibile viene valutata la distanza tra l'ultima posizione nota e quella attuale. Se questa distanza è inferiore a un certo valore di soglia si continua l'esecuzione del percorso, altrimenti si richiede automaticamente una nuova pianificazione del percorso dal punto attuale al punto di goal precedentemente specificato.

Un via point si considera raggiunto quando la distanza tra esso e la posizione della carrozzina è inferiore a una soglia stabilita. Se esiste un via point successivo, SequencerExpert continua l'esecuzione del percorso utilizzando le coordinate del nuovo via point per generare i valori di *rho_to_via_point* e *angle_to_via_point*. Si noti che il cambio di via point da raggiungere non è notificato al controllore fuzzy, in quanto esso continua a ricevere messaggi di tipo `MSG_FROM_SEQUENCER_VIAPOINT_INFO` ed eseguire il ciclo di controllo. Se

il via point raggiunto è l'ultimo, ovvero il punto di destinazione del percorso pianificato, verrà inviato un messaggio `MSG_FROM_SEQUENCER_EXECUTING_PATH` con valore di *executing_path* 0 e il membro SequencerExpert si riporterà nella condizione iniziale, in cui attende nuove richieste di pianificazione.

In qualsiasi momento dell'esecuzione del membro SequenceExpert, se viene ricevuto un messaggio di tipo `MSG_STOP_PLAN_EXECUTION` il piano corrente viene abbandonato e l'esecuzione di SequenceExpert viene riportata alla condizione iniziale. Se invece viene ricevuto un messaggio di tipo `MSG_GOAL_XY`, ovvero viene richiesta una nuova pianificazione con le nuove specifiche del punto di destinazione, SequencerExpert interrompe la sua esecuzione per servire la nuova richiesta, che ha l'effetto di cancellare quella precedente. È previsto un altro messaggio di richiesta di pianificazione ed esecuzione del percorso che SequenceExpert è in grado di analizzare: il messaggio è `MSG_GO_TO_GOAL_COMMAND` e richiede di effettuare una nuova pianificazione mantenendo il punto di destinazione precedentemente impostato.

I messaggi `MSG_STOP_PLAN_EXECUTION`, `MSG_GO_TO_GOAL_COMMAND` e `MSG_GOAL_XY` sono generati dal membro che si occupa di gestire l'interazione con l'utente tramite l'interfaccia grafica, che sarà presentata nella Sezione 6.2.8.

Guida automatica con regole fuzzy

Il comportamento `FollowViaPoint` presentato è estendibile utilizzando le informazioni già note al controllore fuzzy, come, ad esempio, la velocità con cui si muove la carrozzina. In questo modo è possibile creare regole di comportamento più complesse che permettono di reagire meglio nelle situazioni in cui la carrozzina opera. Ad esempio, qualora il punto da raggiungere sia frontale rispetto alla carrozzina e vicino, ma la carrozzina stia ruotando velocemente in senso orario, è necessario imporre una rotazione antioraria per evitare di allontanarsi dalla traiettoria che porta a raggiungere il punto di destinazione:

```
(AND (ViaPointNear)
  (AND (ViaPointN)
    (RotVeloceOrario)
  )
)
=>
(rxlxcmd VFASTLX);
```

Il comportamento `FollowViaPoint` può essere considerato allo stesso modo di un dispositivo di comando della carrozzina e può essere posto allo stesso livello dei comportamenti che gestiscono le proposte del joystick e del

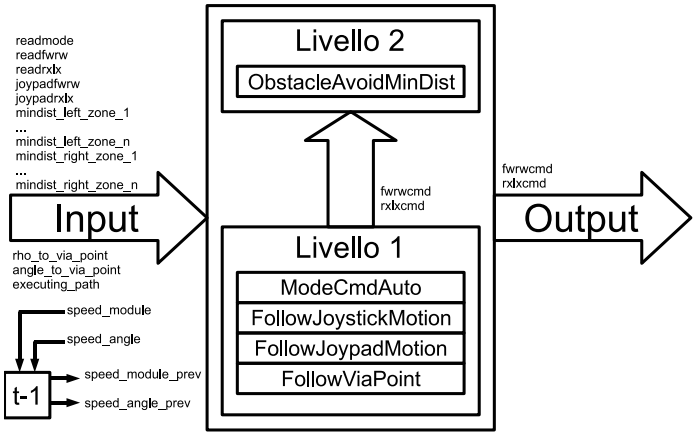


Figura 6.16: Controllore fuzzy a 2 livelli per la gestione di joystick, joypad, guida autonoma ed evitamento ostacoli

joypad, ovvero `FollowJoystickMotion` e `FollowJoypadMotion`. I comportamenti di comando al primo livello del controllore fuzzy sono ora tre ed è necessario garantire la mutua esclusione tra le proposte di ognuno di essi. In particolare è opportuno inserire delle regole di attivazione dei comportamenti CANDO che permettono la guida con il joystick in qualsiasi momento, la guida con il joypad solo quando il joystick è in posizione di riposo e la guida automatica solo quando entrambi i dispositivi di comando sono inutilizzati e il valore di *executing_path* comunicato da `MSG_FROM_SEQUENCER_EXECUTING_PATH` è pari a 1. Con la catena di priorità appena descritta si garantisce che qualora fosse attiva l'esecuzione automatica di percorsi, l'utente può comunque prendere immediatamente il controllo della carrozzina muovendo uno dei due dispositivi di comando. L'esecuzione del percorso riprende appena i dispositivi di comando a priorità maggiore vengono riportati in posizione di riposo.

Si noti che il comportamento di secondo livello che effettua l'evitamento ostacoli è sempre attivo, qualsiasi sia il dispositivo che comanda il movimento della carrozzina.

Lo schema del controllore fuzzy a 2 livelli con i nuovi comportamenti introdotti è mostrato in Figura 6.16, mentre lo schema dell'architettura composta dai membri che cooperano nella pianificazione ed esecuzione di un percorso è mostrato in Figura 6.17.

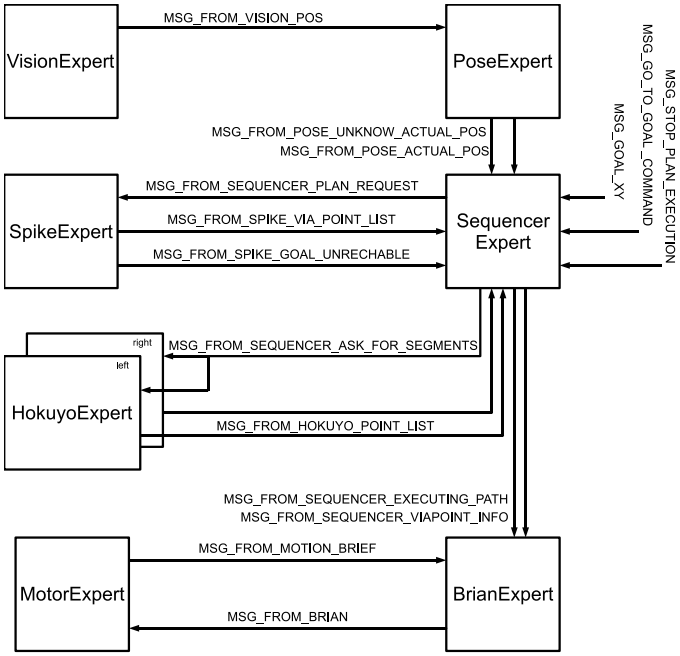


Figura 6.17: Architettura del sistema che governa la pianificazione e l'esecuzione di percorsi automatici

Estrazione del profilo degli ostacoli

Nella descrizione del funzionamento del membro SequencerExpert si è tralasciata l'estensione dei membri HokuyoExpert che permette di conoscere il profilo degli ostacoli. Innanzitutto è necessario permettere a HokuyoExpert di ricevere i messaggi di tipo `MSG_FROM_SEQUENCER_ASK_FOR_SEGMENTS` con corpo vuoto. HokuyoExpert prima di leggere i dati dal sensore laser controlla se è stato inviato un messaggio `MSG_FROM_SEQUENCER_ASK_FOR_SEGMENTS` e, in caso affermativo, dopo la scansione risponde con il messaggio `MSG_FROM_HOKUYO_POINT_LIST`. Il contenuto di quest'ultimo messaggio è costituito da una lista di punti le cui coordinate sono specificate nei campi *hokuyo_point_x_*$\langle nome \rangle$ e *hokuyo_point_y_*$\langle nome \rangle$, dove $\langle nome \rangle$ indica se il sensore laser considerato è il destro o il sinistro. La lista di punti rappresenta una spezzata che indica il profilo degli ostacoli rilevato.

Per creare il messaggio appena descritto è necessario convertire i dati della scansione dei sensori laser, espressi in coordinate polari rispetto al sensore stesso, in coordinate cartesiane. Considerando che ogni sensore laser produce più di 600 rilevazioni a ogni scansione, si otterrebbero più di 600 segmenti che descrivono il profilo degli ostacoli rilevati. È evidente che

un numero di segmenti così elevato potrebbe comportare problemi sia per quanto riguarda il trasporto all'interno di un messaggio, che diventerebbe eccessivamente lungo, sia per la ricerca di un percorso da parte del pianificatore. È preferibile creare un messaggio più semplice, che descriva con un numero inferiore di segmenti il profilo degli ostacoli. Per fare ciò si è utilizzato un metodo di segmentazione ricorsivo. Considerando i punti $p_1 \dots p_n$ che rappresentano un profilo segmentato, è possibile tracciare un segmento tra il punto p_1 e p_n e individuare il punto p_m, con $2 \leq m \leq n-1$ tale che la distanza di p_m dal segmento $\overline{p_1 p_n}$ è massima. Se tale distanza è superiore a una soglia fissata si aggiunge il punto p_m alla lista dei punti che comporranno il profilo finale degli elementi. Ricorsivamente si riapplica lo stesso algoritmo ai punti $p_1 \dots p_m$ e $p_m \dots p_n$, ovvero cercando il punto a distanza massima da $\overline{p_1 p_m}$ e da $\overline{p_m p_n}$. In questo modo si ottiene un profilo segmentato che garantisce di mantenere inalterate le caratteristiche salienti del profilo originale, diminuendo il numero di punti utilizzati.

L'algoritmo ricorsivo, descritto in pseudo codice è il seguente:

Algoritmo 1 SemplificaProfilo(array p,n,s)

```
L=[1]
SemplificaProfiloRicorsivo(p,1,n,s,L)
L=[L; n]
return L
```

Algoritmo 2 SemplificaProfiloRec(p,i,j,s,L)

```
if i < j then
  m = indice del punto a massima distanza da p_i p_j, i < m < j
  d = distanza tra p_m e p_i p_j
  if d > s then
    SemplificaProfiloRicorsivo(p,i,m,s,L)
    L=[L; m]
    SemplificaProfiloRicorsivo(p,m,j,s,L)
  end if
end if
```

La complessità computazionale di questo algoritmo è nel caso medio di $O(n \log(n))$ e nel caso pessimo di $O(n^2)$. Il numero di punti generati non può essere stabilito a priori, ma dipende dal profilo considerato e dalla soglia utilizzata. È abbastanza logico considerare che nel caso specifico in cui si utilizza l'algoritmo, ovvero per semplificare i profili generati da uno scanner

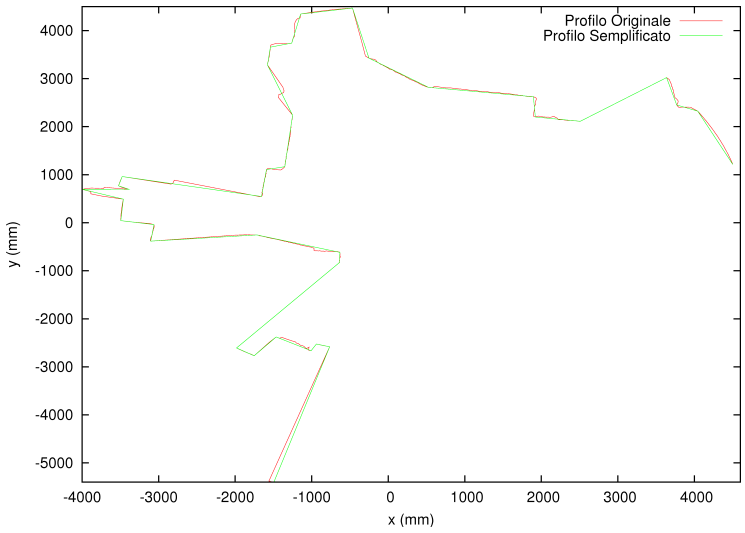


Figura 6.18: Profilo rilevato da uno scanner laser Hokuyo URG-04LX (in rosso) e semplificazione ottenuta con una soglia di $100mm$ *(in verde)*

laser, il rumore che affligge le misure rende possibile ridurre nettamente il numero di punti generati rispetto al numero di punti di partenza.

Un esempio del profilo rilevato con un sensore laser e la sua semplificazione ottenuta con una soglia impostata a un valore di 100mm è mostrato in Figura 6.18. I punti originali sono 682, mentre quelli del profilo semplificato sono 36.

Esempi di esecuzione

Per maggiore chiarezza sono presentati alcuni esempi di funzionamento del sistema che si occupa di pianificare ed eseguire il percorso. In Figura 6.19 è mostrata una interazione tipica tra i membri. Innanzitutto, come ipotesi semplificativa supponiamo che sia sempre disponibile la posizione della carrozzina, ovvero che periodicamente sia ricevuto un messaggio di tipo `MSG_FROM_POSE_ACTUAL_POS` e non arrivi mai un messaggio di tipo `MSG_FROM_POSE_UNKNOW_ACTUAL_POS`.

La sequenza è la seguente:

1. Alla ricezione di un messaggio di tipo `MSG_GOAL_XY`, dato che è disponibile la posizione della carrozzina, il membro SequencerExepert richiede ai due membri HokuyoExpert il profilo degli ostacoli rilevati, con un messaggio di tipo `MSG_FROM_SEQUENCER_ASK_FOR_SEGMENTS`.

2. SequencerExpert riceve entrambi i messaggi `MSG_FROM_HOKUYO_POINT_LIST` contenenti i punti che descrivono il profilo semplificato degli ostacoli e trasforma i punti in coordinate assolute. Conoscendo la posizione attuale, il punto di destinazione e la descrizione degli ostacoli dinamici richiede al membro SpikeExpert di pianificare un percorso, utilizzando il messaggio `MSG_FROM_SEQUENCER_PLAN_REQUEST`.

3. Spike comunica a SequencerExpert il risultato della pianificazione, in questo caso utilizzando il messaggio `MSG_FROM_SPIKE_VIA_POINT_LIST`.

4. SequencerExpert può ora iniziare a comandare il controllore fuzzy in modo da raggiungere successivamente i via point. I messaggi inviati sono `MSG_FROM_SEQUENCER_VIAPOINT_INFO` e `MSG_FROM_SEQUENCER_EXECUTING_PATH`. Quest'ultimo messaggio specifica che la guida automatica è in esecuzione, ed è indicato in figura con <1>.

5. Quando tutti i via point sono stati raggiunti, compreso il punto di destinazione del percorso, SequencerExpert invia il messaggio `MSG_FROM_SEQUENCER_EXECUTING_PATH` per comunicare che la guida automatica non è più attiva.

In Figura 6.20 è mostrato il funzionamento del sistema che governa la guida automatica nel caso in cui, durante l'esecuzione di un percorso, la posizione della carrozzina diventi ignota e successivamente torni nota. Supponendo che sia già in esecuzione un percorso, ovvero che SequenceExpert comunichi periodicamente a BrianExpert le informazioni riguardo al via point da raggiungere, si considera il caso in cui la posizione della carrozzina risulta non più nota, comunicato tramite il messaggio `MSG_FROM_POSE_UNKNOW_ACTUAL_POS`. SequencerExpert reagisce a questo evento inviando a BrianExpert il comando che interrompe l'esecuzione del percorso. Quando la posizione della carrozzina torna a essere nota, essa viene confrontata con l'ultima posizione nota. Supponendo che la distanza tra le due rilevazioni rientra in una certa tolleranza, l'esecuzione automatica del percorso riprende da dove era stata interrotta.

In Figura 6.21 è mostrato un caso simile al precedente, in cui però al momento della ripresa dell'esecuzione del percorso, la distanza tra la posizione precedente e quella attuale risulta troppo elevata. In questo caso è necessario ripianificare il percorso, senza modificare il punto di destinazione.

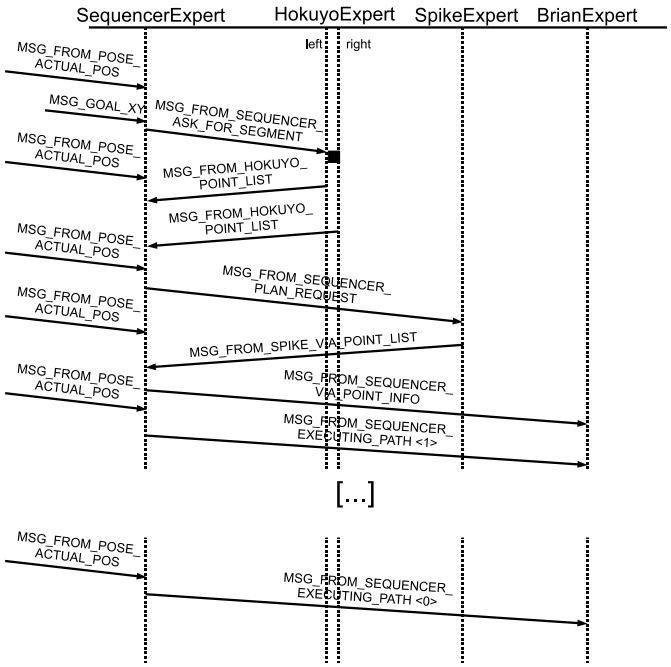


Figura 6.19: Flusso dei messaggi tra i membri coinvolti nella pianificazione ed esecuzione dei percorsi

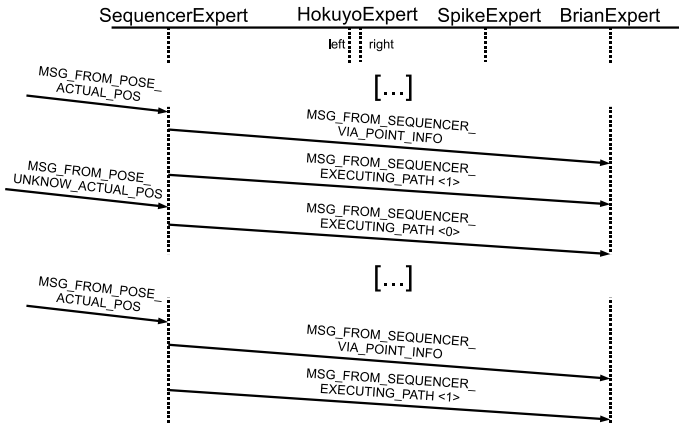


Figura 6.20: Flusso dei messaggi nel caso in cui l'esecuzione dei percorsi viene interrotta e successivamente ripresa

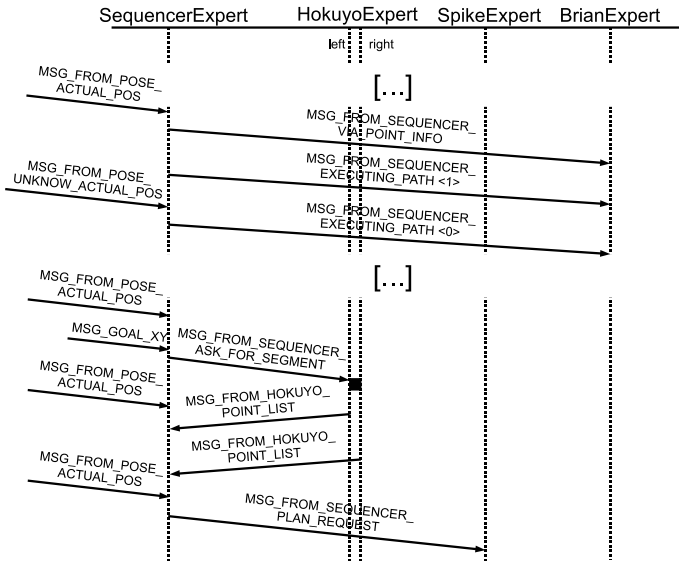


Figura 6.21: Flusso dei messaggi nel caso in cui l'esecuzione dei percorsi viene interrotta ed è necessario ripianificare il percorso

6.2.8 Interfaccia grafica

L'interfaccia grafica è il componente meno critico dell'applicazione finora presentata. La sua scarsa importanza è dovuta al fatto che l'obiettivo di questo lavoro non è sviluppare un applicazione completa in tutte le sue parti, ma solo di creare la base software di controllo della carrozzina elettrica. Uno studio approfondito dell'interfaccia grafica comporta la conoscenza precisa del tipo di utente che deve utilizzarla, in quanto deve garantire accessibiltià anche a coloro che hanno più difficoltà di movimento e coordinazione. In questo momento l'interfaccia grafica risulta utile come strumento di debug e controllo on-line dell'esecuzione del software di controllo.

La realizzazione dell'interfaccia grafica è affidata al membro GuiExpert. Come funzioni fondamentali, necessarie al funzionamento del sistema proposto, essa deve essere in grado di produrre i messaggi di comando per l'esecuzione automatica dei percorsi, ovvero deve produrre i messaggi di tipo `MSG_GOAL_XY` e `MSG_STOP_PLAN_EXECUTION`. Per produrre i messaggi di specifica del goal si è previsto di utilizzare due campi testuali in cui specificare le coordinate del punto da raggiungere. L'invio del messaggio è effettuato alla pressione di un bottone. Tramite un'altro bottone è possibile invece inviare il messaggio di interruzione dell'esecuzione del piano. Non è previsto attualmente alcun modo per inviare il messaggio `MSG_GO_TO_GOAL_COMMAND`.

Dal punto di vista delle informazioni visualizzate si è ritenuto significativo indicare in modo preciso e in tempo reale il comando prodotto dal controllore fuzzy, accompagnato dalle informazioni sulla posizione e velocità della carrozzina e dalle informazioni relative all'esecuzione automatica di percorsi. Per realizzare questo scopo, GuiExpert riceve i seguenti messaggi:

- `MSG_FROM_BRIAN`, che permette di conoscere l'azione di controllo intrapresa.
- `MSG_FROM_POSE_ACTUAL_POS` e `MSG_FROM_POSE_UNKNOW_ACTUAL_POS` permettono di visualizzare la posizione della carrozzina quando questa è disponibile.
- `MSG_FROM_POSE_UNKNOW_ACTUAL_SPEED` e `MSG_FROM_POSE_ACTUAL_SPEED` permettono di visualizzare la velocità della carrozzina quando questa è disponibile.
- `MSG_FROM_SEQUENCER_EXECUTING_PATH` permette di indicare se è in corso l'esecuzione di un percorso automatico o no.
- `MSG_FROM_SEQUENCER_VIAPOINT_INFO` indica le informazioni relative al via point che si sta cercando di raggiungere.

In Figura 6.22 è visualizzata l'interfaccia grafica realizzata. In alto è presente una zona che indica con l'uso di progress bar il valore del controllo prodotto. Nella parte centrale sono indicate le informazioni su posizione e velocità della carrozzina. Nella parte inferiore sono presenti due campi testuali che permettono di specificare la posizione da raggiungere e due pulsanti per comandare l'esecuzione o l'interruzione del percorso. Infine sono mostrate le informazioni sul via point che si sta inseguendo.

6.2.9 Gestione errori

L'ultimo membro che non è ancora stato descritto è ErrorExpert. Questo membro si occupa di rilevare e gestire i malfunzionamenti del sistema monitorando i messaggi inviati dagli altri membri del sistema. Attualmente questo membro monitora solo due tipi di messaggi: `MSG_FROM_HOKUYO_ERR` e `MSG_FROM_HOKUYO_MINDIST` per controllare il corretto funzionamento degli scanner laser. Quando questo membro riceve un messaggio che indica il malfunzionamento di uno scanner laser, produce un messaggio di tipo `MSG_FROM_ERROR_MANAGER`, che contiene il campo *general_error* con valore 1. Quando l'errore viene ripristinato, ovvero viene ricevuto un messaggio di

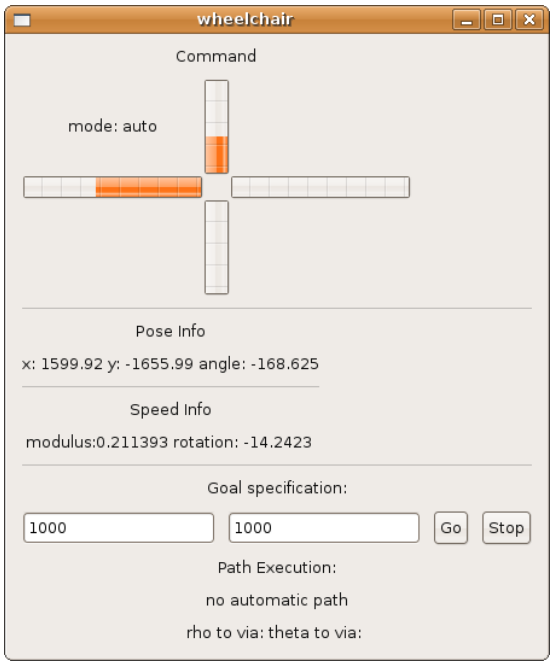


Figura 6.22: Interfaccia grafica del software della carrozzina

tipo `MSG_FROM_HOKUYO_MINDIST` che indica il corretto funzionamento dello scanner laser, allora viene emesso un nuovo messaggio di tipo `MSG_FROM_ERROR_MANAGER` con valore 0 del campo *general_error*. Si noti che i messaggi emessi dai due sensori laser sono dello stesso tipo, ma la provenienza è distinguibile in base al contenuto. Grazie a questa caratteristica è possibile capire quale sensore laser ha subito malfunzionamenti e attendere un suo messaggio che indichi la ripresa del normale funzionamento del sensore laser. Qualora entrambi i sensori laser dovessero causare errori, la permanenza dello stato d'errore generale si protrarrebbe fino a quando entrambi i sensori laser ripristinano il loro normale funzionamento.

La gestione dell'errore è effettuata dal controllore fuzzy, che è stato esteso con un nuovo comportamento, posto al livello più alto, ovvero il terzo, di nome `GeneralError`. L'attivazione di questo comportamento è associata al valore del campo *general_error*. Il compito di questo comportamento è impedire qualsiasi movimento della carrozzina. Il comportamento fuzzy è costituito da un'unica regola, che è la seguente:

```
(Always)
=>
(&DEL.rxlxcmd ANY)
```

```
(&DEL.fwrwcmd ANY)
(fwrwcmd STEADY)
(rxlxcmd STEADY);
```

Le funzionalità di questo membro possono essere notevolmente sviluppate. Si potrebbero ad esempio implementare controlli sulla periodicità delle letture di MotorExpert (monitorando ad esempio il messaggio `MSG_FROM_MOTION_INFO`), in modo da assicurare che il circuito che comunica con la carrozzina funzioni correttamente. Allo stato attuale questo controllo è stato ritenuto superfluo, in quanto la carrozzina viene utilizzata in ambienti controllati e il circuito di interfaccia della carrozzina con il PC è risultato molto stabile. In una versione definitiva sarebbe opportuno inserire controlli più precisi per garantire maggiore sicurezza dell'utente a bordo.

6.2.10 Commento sull'architettura

La struttura generale dell'architettura, con tutti i membri fino a ora presentati e i messaggi prodotti, è mostrata in Figura 6.23. Alcuni messaggi risultano inutilizzati, questo a dimostrazione del fatto che l'architettura è stata progettata e realizzata in funzione di sviluppi futuri. L'estensione è facilitata dalla modularità dell'architettura software che permette di cambiare o aggiungere facilmente membri senza modificare tutto il software di controllo. Inoltre, anche il controllore fuzzy è facilmente estendibile, in quando anch'esso è modulare e i comportamenti sono progettati indipendentemente uno dall'altro.

I membri che potrebbero essere più sviluppati sono GuiExpert, PoseExpert, oltre a ErrorExpert le cui possibili estensioni sono già state presentate nella sezione precedente. GuiExpert è il membro che si occupa dell'interfaccia utente. Grazie all'architettura proposta è possibile cambiare l'interfaccia utente del sistema in base alle esigenze dell'utente. Sarebbe possibile anche spostare l'interfaccia utente su un'altra Agorà del sistema DCDT implementata in un altro programma, in modo da rendere completamente indipendente il sistema di controllo dall'interfaccia grafica. La realizzazione di interfacce grafiche personalizzate risulterebbe ulteriormente facilitata, in quanto non sarebbe più necessario nemmeno ricompilare il software di controllo, ma solo quello di interfaccia.

Il membro PoseExpert può essere esteso in modo da integrare informazioni differenti per stimare meglio il moto della carrozzina. Attualmente il sistema che misura la velocità della carrozzina è assolutamente indipendente dal sistema che ne stima la posizione, nonostante le informazioni dei due sistemi siano entrambe convogliate all'interno di PoseExpert. Si potrebbe-

ro implementare metodi di stima del moto e della posizione che combinano tutte le informazioni disponibili, oppure si potrebbero facilmente aggiungere altri sensori e metodi di stima.

Come esempio dimostrativo per mostrare la semplicità di estensione dell'architettura proposta, supponiamo di disporre di un sistema di misura della velocità della carrozzina basato su una telecamera che inquadra il pavimento e di voler integrare le informazioni da esso prodotte nel membro PoseExpert. È necessario implementare un nuovo membro, che chiameremo VisualOdometryExpert che rende disponibile le informazioni sulla velocità rilevata tramite un messaggio, che chiameremo `MSG_FROM_VISUAL_ODOMETRY`. È facile notare che le uniche modifiche necessarie al sistema per integrare il nuovo membro riguardano il membro PoseExpert, che dovrà tenere conto dei messaggi provenienti da VisualOdometryExpert, da MTiExpert e da VisionExpert e integrare con un criterio opportuno le informazioni ricevute. La modifica introdotta risulta completamente trasparente al sistema complessivo e al controllore fuzzy.

6.2.11 Temporizzazione

Presentando l'architettura del sistema si è volutamente tralasciata la descrizione della temporizzazione dei membri introdotti. Per scegliere la durata del periodo di esecuzione di un membro è necessario considerare il tempo di calcolo necessario al membro stesso, che non deve risultare superiore al periodo considerato.

La periodicità di alcuni membri è strettamente legata alla funzione svolta. Ad esempio, i due membri di tipo HokuyoExpert hanno una periodicità strettamente legata alla possibilità di acquisire nuovi dati dai sensori, mentre il tempo di esecuzione del membro MotorExpert è legato alla frequenza con cui il circuito di interfaccia tra la carrozzina e il PC comunica sulla porta seriale.

Le periodicità scelte per i vari membri sono elecate e motivate di seguito:

MotorExpert viene rieseguito ogni 20ms, in quanto il circuito di interfaccia tra il PC e la carrozzina produce nuovi dati e accetta nuovi comandi a questa frequenza.

BrianExpert viene rieseguito ogni 20ms. Esso lavora in stretto legame con MotorExpert. La temporizzazione proposta permette di produrre il controllo alla stessa frequenza con cui è posibile comandare il moto della carrozzina.

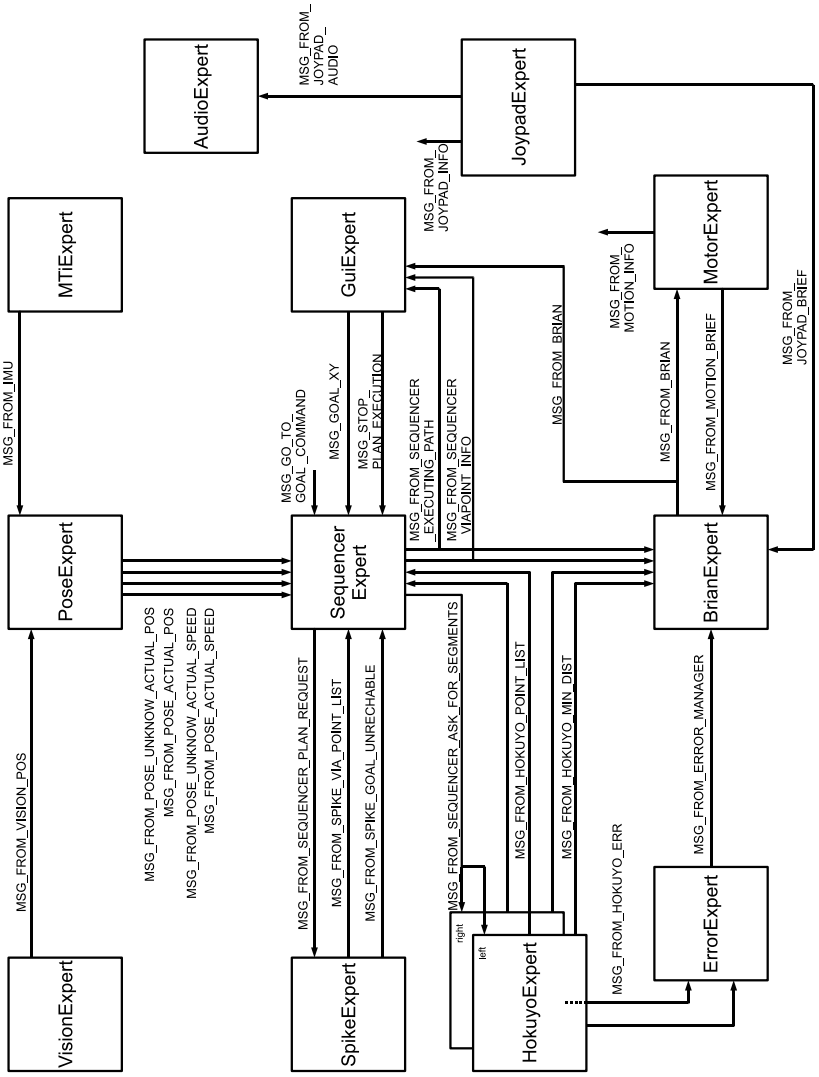


Figura 6.23: Architettura completa del software di controllo della carrozzina

JoypadExpert viene rieseguito ogni 20ms, per permettere di comandare la carrozzina con la stessa frequenza con cui MotorExpert invia i comandi.

HokuyoExpert viene rieseguito ogni 100ms, infatti gli scanner laser Hokuyo URG-04LX sono in grado di produrre 10 scansioni al secondo.

VisionExpert viene rieseguito ogni 66ms, in quanto la telecamera è impostata in modo da acquisire 15 immagini al secondo, che vengono analizzate una a una. Sarebbe possibile impostare la telecamera in modo da acquisire fino a 30 immagini al secondo, ma il tempo necessario ad analizzare il singolo frame, per quanto sia variabile, risulta da prove pratiche spesso superiore al tempo disponibile, pari a circa 33ms. Si è quindi preferito mantenere un frame-rate più basso per garantire una maggiore costanza delle rilevazioni. Qualora infatti l'analisi dell'immagine richiedesse un tempo superiore al tempo proposto come periodicità, un certo numero di frame sarebbero persi. L'aumento della frequenza di acquisizione delle immagini non farebbe altro che aumentare il numero di frame persi nelle situazioni critiche, senza grandi benefici nelle situazioni comuni.

MTiExpert viene rieseguito ogni 20ms per poter leggere correttamente tutti i dati provenienti dal sensore XSens MTi, che è impostato per produrre dati a una frequenza di 50Hz.

PoseExpert viene rieseguito ogni 20ms, in quanto è necessario che esso ascolti sia i messaggi di VisionExpert che di MtiExpert, che lavorano rispettivamente con periodo di 66ms e 20ms. Per non perdere messaggi di MTiExpert è necessario lavorare alla stessa frequenza di quest'ultimo membro.

SequencerExpert viene rieseguito ogni 66ms, in quanto il suo funzionamento è strettamente legato a quello di PoseExpert, anche se solo per quanto riguarda la conoscenza della posizione della carrozzina, che essendo prodotta dal sistema di visione, lavora a una frequenza di 66ms.

SpikeExpert viene rieseguito ogni 500ms. Il modulo SpikeExpert si occupa di eseguire le pianificazioni dei percorsi, quindi il tempo di riesecuzione non è critico ed è stato scelto un valore piuttosto elevato, in quanto è ragionevole supporre che, una volta richiesta una pianificazione, non ne venga richiesta un'altra in un tempo di pochi decimi di secondo.

GuiExpert ha una periodicità di 100ms, per garantire una visualizzazione sufficientemente fluida delle informazioni dell'interfaccia grafica, senza appesantire troppo l'esecuzione dell'applicazione.

AudioExpert ha una periodocità di 1 secondo, in quanto la funzionalità non è critica e, qualora viene richiesta la riproduzione di un file audio, è ragionevole considerare che non passerà meno di un secondo prima dell'esecuzione di un nuovo brano.

ErrorExpert viene rieseguito ogni 100ms in quanto controlla solo i messaggi di HokuyoExpert che è anch'esso eseguito ogni 100ms.

6.2.12 Produzione di log

Una funzionalità prevista per il software non ancora illustrata è la produzione di log dettagliati e facilmente analizzabili. I file di log sono utili sia per il debug dell'applicazione, sia per l'analisi del comportamento del sistema al fine di studiare in che punti mìgliorarne il comportamento.

Le soluzioni individuate per produrre log dettagliati sono tre:

1. Associare a ogni membro un file di testo che viene compilato durante l'esecuzione del membro stesso.

2. Creare un membro che si occupa di ricevere tutti i messaggi inviati dagli altri membri e di farne il log su un'unico file.

3. Creare un oggetto accessibile da tutti i membri residenti sulla stessa Agorà che rende mutuamente esclusivo l'accesso a un unico file di log.

La prima soluzione permette di creare log molto dettagliati sul funzionamento del singolo membro, in quanto la "storia" dell'esecuzione risulta descritta in modo molto preciso e chiaro. Purtroppo però è difficile risalire al comportamento complessivo del sistema, in quanto sarebbe necessario unire i dati dei singoli file seguendo una sequenza temporale.

La creazione di un membro che si occupa di ricevere tutti i messaggi spediti dagli altri membri del sistema permette di realizzare un sistema di log che manterrebbe traccia del funzionamento del sistema nel suo complesso. Questo metodo presenta però due problemi abbastanza evidenti. Innanzitutto il sistema complessivo risulterebbe appesanto dalla presenza di un membro che si occupa solo del log dei messaggi. Inoltre il log potrebbe risultare scarno e povero di dati salienti: infatti le informazioni che descrivono il funzionamento del sistema non sono solo quelle contenute nei messaggi, in quanto in alcuni casi è necessario inserire nel log dati che non sono

presenti nei messaggi inviati, ma sono noti solo al membro stesso durante l'esecuzione. Ad esempio è utile inserire il time stamp dell'istante di inizio dell'esecuzione di alcuni membri per controllare il periodo di attivazione per controllare la loro periodicità. A tale informazione non si può risalire con i messaggi scambiati tra i membri, ma solo con una rilevazione effettuata internamente al membro stesso.

La terza soluzione risulta la più vantaggiosa e permette di costruire un log globale del sistema, mantenendo la coerenza temporale degli eventi e posticipando la scelta degli eventi di cui effettuare il log all'atto dell'implementazione di ogni singolo membro. È inoltre possibile scrivere sul log informazioni "nascoste" anche al membro stesso, in quanto l'oggetto risulta accessibile anche da tutti i moduli software realizzati. Si pensi ad esempio al caso in cui il modulo di gestione dei sensori laser rilevi un errore durante la comunicazione con il laser stesso. Il membro che gestisce il singolo sensore laser è a conoscenza solo dell'evento accaduto, mentre all'interno della libreria è possibile prevedere un log dettagliato del tipo di errore rilevato.

L'oggetto che permette il logging è realizzato tramite una classe statica che espone solo tre metodi. Sono previsti un metodo di inizializzazione che permette di aprile il file su cui viene effettuato il log e un metodo di chiusura che chiude il file. L'ultimo metodo previsto, che si occupa dell'effettiva scrittura delle informazioni di log, deve essere gestito in modo da garantire la mutua esclusione, ovvero prima di servire una nuova richiesta di scrittura deve avere terminato la scrittura della precedente. Il metodo di scrittura accetta come dato di input una stringa di testo, a cui viene anteposto il timestamp dell'istante in cui la scrittura viene effettuata e concatenato un ritorno a capo. Per convenzione si è stabilito che ogni stringa scritta sul file di log non deve contenere ritorni a capo, in modo che il file di log sia costituito da una sola riga per ogni evento. Uno svantaggio di questo metodo di effettuare logging è dato dal fatto che, qualora si debba dividere l'esecuzione del software di controllo su più macchine, risulterebbe impossibile effettuare un unico logging. Supponendo però di utilizzare un sistema di sincronizzazione degli orologi di sistema dei computer su cui le applicazioni sono in esecuzione, sarebbe possibile effettuare separatamente il log e successivamente unire le diverse parti basandosi sui timestamp registrati all'interno dei singoli log.

Con un file di log di questo tipo è molto semplice estrarre gli istanti di tempo in cui il membro BrianExpert ha iniziato la sua attività e creare un grafico che monitora il periodo di tempo di esecuzione del membro stesso, ottenuto come sottrazione tra un timestamp e il precedente. Ad esempio, il grafico dell'attivazione del membro BrianExpert è mostrato in

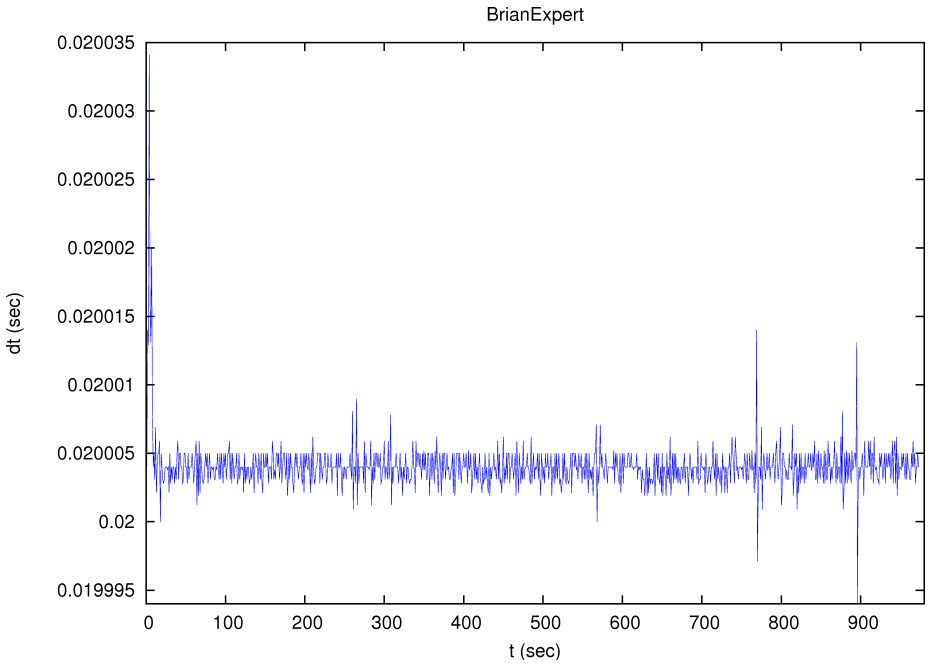


Figura 6.24: Tempo di esecuzione del membro BrianExpert estratto dal file di log del sistema

Figura 6.24. Allo stesso modo, conoscendo le informazioni prodotte dagli altri membri è possibile analizzare dati come posizione della carrozzina nell'ambiente, andamento della velocità nel tempo, percorso pianificato e percorso effettivamente seguito, etc.

L'analisi dei file di log può avvenire in modo "manuale", ovvero tramite l'analisi della sequenza delle informazioni registrate, oppure in modo automatico. L'analisi manuale è un metodo valido soprattutto per capire se il sistema si comporta nel modo previsto o per individuare quali condizioni hanno creato problemi all'esecuzione del controllo. L'analisi automatica è utile per valutare il funzionamento del sistema quando l'esecuzione è stabile. Conoscendo infatti quali sono le informazioni che i singoli membri registrano durante la loro esecuzione è possibile estrarre tramite l'uso di script solo le informazioni salienti che descrivono porzioni del funzionamento del sistema. Ad esempio, supponendo che il membro BrianExpert richieda al sistema di logging di scrivere la stringa `Brian Expert run duty` a ogni sua attivazione, una riga del file di log corrispondente risulta

```
1204797883.582073 Brian Expert run duty
```

Capitolo 7

Risultati sperimentali

In questo capitolo sono analizzati i risultati ottenuti, sia dal punto di vista della costruzione della struttura che dal punto di vista del comportamento del controllore nello svolgere i compiti di assistenza alla guida e di guida autonoma.

7.1 Struttura di supporto

La struttura di supporto è stata realizzata come si era ipotizzato in fase di progetto e si è rivelata adatta agli scopi prefissati. Infatti è risultato semplice e agevole montare tutti i componenti previsti. In particolare si è scelto di posizionare il computer sulla base della struttura in profilati d'alluminio e di utilizzare una delle mensole posizionate a metà altezza del parallelepipedo per alloggiare il sensore XSens MTi. Tutti gli altri componenti sono stati montati nelle posizioni previste nella progettazione, come, ad esempio, i laser in posizione frontale all'altezza delle ginocchia dell'utente e la telecamera rivolta verso il soffitto posta nel punto più alto della struttura. Le scanalature dei profilati hanno agevolato il posizionamento dei cavi necessari, offrendo una sede naturale in cui farli scorrere.

Anche le procedure che permettono di montare e smontare la struttura sulla carrozzina sono molto semplici, infatti i punti di fissaggio della struttura sono solo quattro, due davanti al sedile e due sulla base posizionata sulla parte posteriore del sedile.

In Figura 7.1 è mostrata la carrozzina con la struttura di supporto realizzata e tutti i sensori e dispositivi montati.

Figura 7.1: La carrozzina elettrica Rabbit con la struttura di supporto e tutti i componenti aggiunti

7.2 Guida della carrozzina

Non considerando l'esecuzione di percorsi automatici, la guida della carrozzina è effettuabile sia tramite l'uso del joystick originale che tramite l'uso del joypad. Considerando solo queste due semplici funzionalità è possibile cominciare a commentare i risultati ottenuti, estendendo successivamente il panorama d'analisi a tutte le funzionalità previste.

La carrozzina elettrica che è stata realizzata è stata utilizzata durante il Festival della Scienza di Genova[1] come veicolo teleguidato con il joypad. In questo contesto la carrozzina svolgeva il ruolo di "guida turistica" per i bambini che partecipavano all'evento, riproducendo anche file audio di spiegazione richiamati sempre con l'uso del joypad. L'esperienza presso il festival della scienza ha permesso di testare approfonditamente il circuito e il software di controllo, che non hanno evidenziato alcun problema, rimanendo in attività anche per più di 8 ore al giorno. In questo contesto la carrozzina era sprovvista degli altri sensori, ma dotata solo del computer di controllo, del monitor touch screen e del joypad senza fili. Anche l'architettura software era notevolmente semplificata, in quanto un numero consistente di membri non era necessario per realizzare le funzionalità richieste.

La carrozzina è stata esposta durante il convegno *Disability and advanced research*[2] e in questa occasione è stato possibile testare il software di controllo limitato alle funzioni di guida con joystick e joypad, utilizzando anche la carrozzina in condizioni normali di traffico cittadino e su marcipiedi e strade per effettuare gli spostamenti tra i differenti edifici del Politecnico di Milano. La buona reattività del sistema di guida ha permesso di controllare agevolmente il moto della carrozzina per tratti anche piuttosto lunghi e su superfici sconnesse che necessitano di repentine correzioni da parte del guidatore. Alcune fotografie scattate durante il trasferimento della carrozzina sono visibili in Figura 7.2.

Per valutare le prestazioni offerte dal software di controllo limitato alla sola parte relativa alla guida si è proposto ad alcune persone di guidare la carrozzina senza il software di controllo e con il software di controllo attivo, per valutare le eventuali differenze nella risposta della carrozzina introdotte dal software di controllo. All'unanimità si è concluso che non sono riscontrabili differenze apprezzabili e che la guida della carrozzina risulta fluida e precisa anche con il software di controllo attivo.

[1] Genova, 25 ottobre - 6 novembre 2007, http://festivalscienza.it
[2] Politecnico di Milano, 11 ottobre 2007

Figura 7.2: Guida della carrozzina con Joypad durante il trasferimento effettuato per partecipare al convegno Disability and advanced research

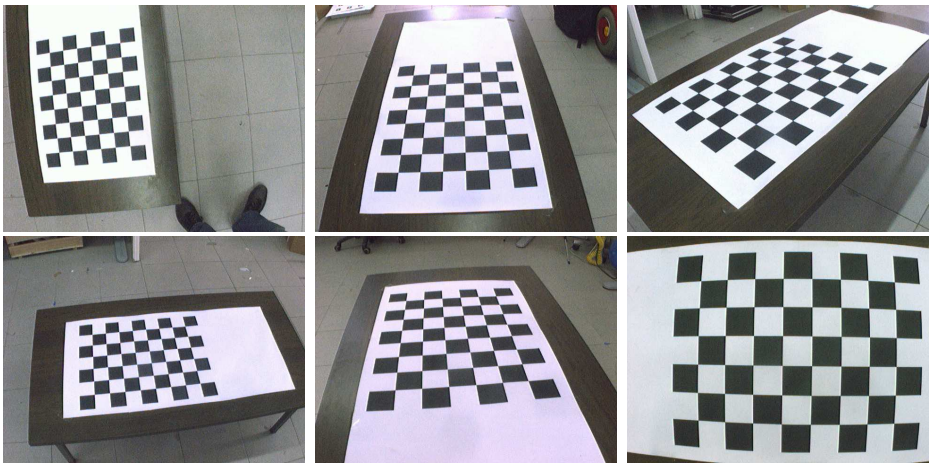


Figura 7.3: Alcune immagini utilizzate per la calibrazione della telecamera

7.3 Localizzazione basata su marker

Il sistema di localizzazione realizzato si è rivelato efficiente e adatto agli scopi per cui è stato realizzato. Per valutare i risultati ottenuti è importante considerare sia il funzionamento del sistema di localizzazione che la procedura di calibrazione e di setup prevista.

Come descritto nella Sezione 5.5.3, è necessario prima di tutto calibrare la telecamera con un tool di calibrazione standard. Questa operazione va effettuata una sola volta, in quanto una volta bloccata la messa a fuoco dell'ottica della telecamera, i parametri di calibrazione restano costanti. Per calibrare la telecamera con il tool Camera Calibration Toolbox for Matlab[3]

[3]http://www.vision.caltech.edu/bouguetj/calib_doc

Figura 7.4: Alcune immagini del sistema di calibrazione registrate

i	$\hat{z}_i$ (mm)	$\hat{\beta}_i$ (rad)	$\hat{\gamma}_i$ (rad)
0	2234.8	-0.0166	-3.0841
1	2271.3	0.0175	-3.1078
2	2266.2	-0.0069	-3.1200
3	2282.5	0.0044	3.1365
4	2252.4	-0.2847	-3.0774
5	2303.6	0.0022	3.1407
6	2238.7	-0.0529	-3.0620
7	2263.7	0.0575	-3.1096
8	2246.4	0.0655	-3.0987

Tabella 7.1: Parametri costanti rilevati per i nove marker utilizzati

è stato necessario registrare alcune immagini riprese da diverse posizioni e angolazioni di una scacchiera di cui è nota la lunghezza del lato della singola cella. Sono state utilizzate 36 immagini per la calibrazione, sei delle quali sono mostrate in Figura 7.3. La matrice dei parametri intrinseci ottenuta è la seguente:

$$K = \begin{pmatrix} f_1 & \alpha \cdot f_1 & x_c \\ 0 & f_2 & y_c \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 658.0 & 0 & 321.6 \\ 0 & 662.5 & 258.2 \\ 0 & 0 & 1 \end{pmatrix}$$

L'uso di questo tool di calibrazione ha inoltre permesso di stimare i parametri di compensazione della distorsione radiale dovuta all'ottica della carrozzina.

La telecamera calibrata è stata montata sulla carrozzina e, dopo aver posizionato nove marker sul soffitto, sono state registrate 3937 immagini, di cui alcune sono riportate in Figura 7.4. Con queste immagini è stato possibile stimare i parametri costanti delle matrici che descrivono la posizione di ogni tag rispetto alla telecamera, ovvero i valori $\hat{z}_i$, $\hat{\beta}_i$ e $\hat{\gamma}_i$. I risultati ottenuti sono riportati in Tabella 7.1.

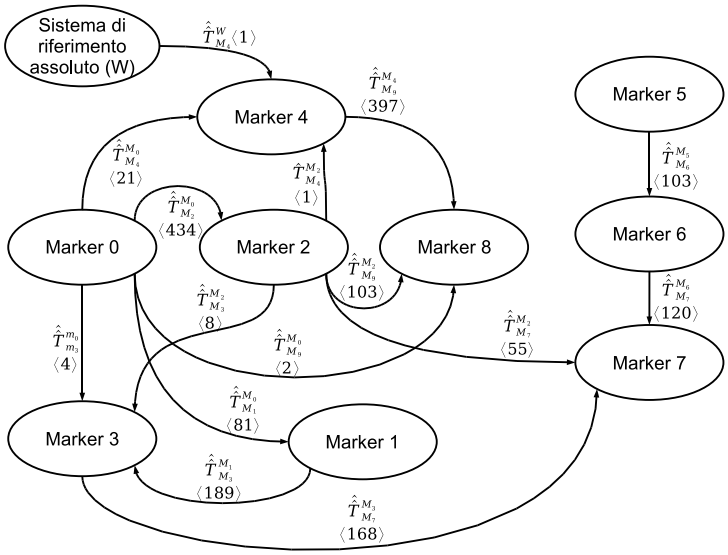


Figura 7.5: Grafo delle relazioni tra i marker

Una successiva analisi delle immagini ha permesso di calcolare le posizioni e gli orientamenti relativi dei marker. Il grafo delle relazioni costruito è mostrato in Figura 7.5. Si è scelto come marker base, ovvero come marker a cui il sistema di riferimento assoluto è riferito, il marker identificato dal codice 4. La matrice di rototraslazione che descrive la posizione e l'orientamento del marker 4 rispetto al sistema assoluto è stata scelta come una semplice rotazione di 180 ° intorno all'asse x, che permette di ottenere l'asse z del sistema di riferimento assoluto opposto a quello del sistema di riferimento solidale con il marker numero 4, ovvero rivolto dal pavimento verso il soffitto.

Dal grafo è stato estratto l'albero di copertura di peso massimo, che permette di definire un cammino unico tra il sistema di riferimento assoluto e ogni nodo: il risultato è riportato in Figura 7.6. Partendo dal sistema di riferimento assoluto è possibile risalire alla posizione e orientamento di ogni marker. La posizione dei maker rilevata, considerando le sole coordinate x e y, è mostrata in Figura 7.7. Si può notare che l'albero ottenuto è molto sbilanciato, in quanto è costituito da un percorso obbligato senza alcuna diramazione. Alberi di questo tipo potrebbero influenzare in modo negativo la precisione del sistema di localizzazione, in quanto l'effetto dell'errore si accumula con l'allungarsi dei cammini. È quindi necessario verificare che il sistema di localizzazione sia in grado di dare risultati buoni anche in questa particolare condizione che si è verificata. Tale verifica, i cui risultati sono

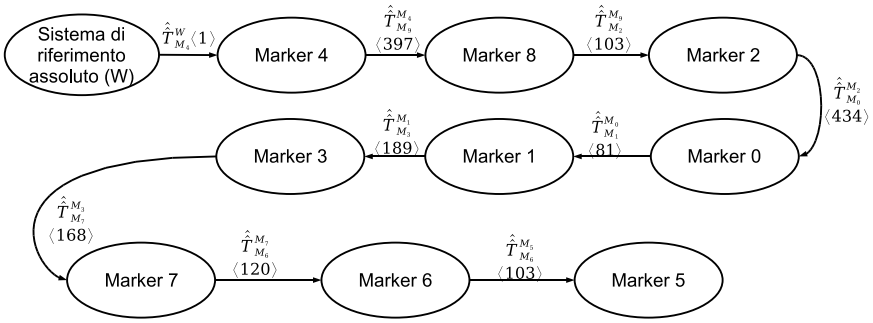


Figura 7.6: Albero di copertura di peso massimo estratto delle relazioni tra i marker

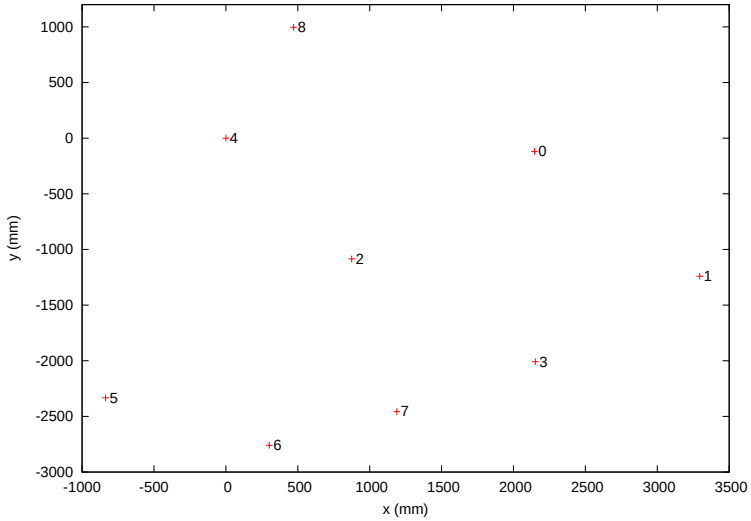


Figura 7.7: Posizione assoluta dei marker

riportati più avanti in questa sezione, ha mostrato che la configurazione ottenuta per le relazioni tra i marker è comunque efficiente. Se la verifica avesse mostrato problemi evidenti, sarebbe stato necessario rieseguire la procedura di calibrazione del sistema, registrando altre immagini da aggiungere a quelle analizzate, al fine di ottenere una diversa configurazione del grafo di partenza.

Alla fine del processo di calibrazione del sistema di localizzazione si è provveduto a specificare manualmente la mappa dell'ambiente in cui il sistema è stato installato. Le misure sono state effettuate con delle semplici rulline metriche e si è ritenuto significativo inserire, oltre ai muri perimetrali, solo gli oggetti statici che meno probabilmente cambieranno posizione nel-

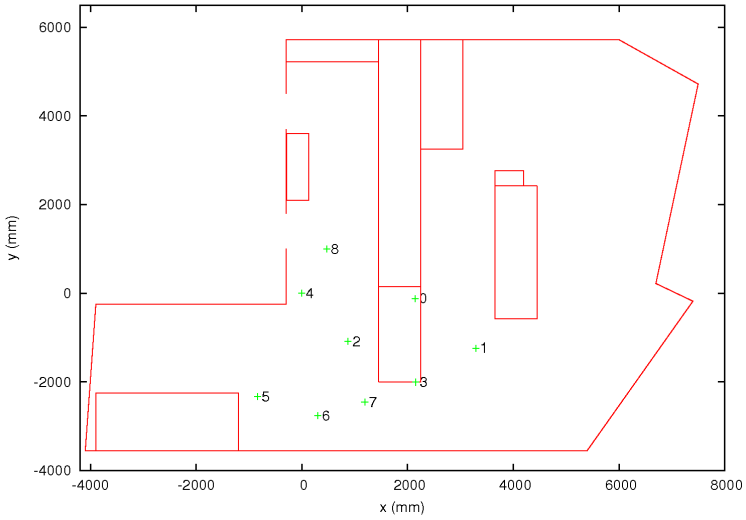


Figura 7.8: Posizione dei marker nella mappa

l'ambiente, come tavoli, scaffali e armadi. In Figura 7.8 è mostrata la mappa dei marker sovrapposta a quella dell'ambiente. È subito evidente che il sistema di localizzazione installato non copre tutto l'ambiente disponibile, ma solo una zona di esso, ritenuta sufficientemente spaziosa per i test da condurre. La copertura di una zona maggiore sarebbe inoltre stata di ostacolo alla normale attività del laboratorio in cui i test sono stati svolti. Si noti che qualora si volesse estendere la zona coperta sarebbe necessario posizionare altri marker sul soffitto e ripetere il processo di calibrazione aggiungendo alle immagini precedentemente rilevate quelle registrate con i nuovi tag.

Per valutare la bontà della calibrazione ottenuta si sono effettuati alcuni percorsi guidando manualmente la carrozzina. In Figura 7.9 è visualizzato un percorso piuttosto complesso che è stato realizzato. Il percorso risulta ben distinguibile e anche la coerenza con la mappa è evidente: non ci sono casi di false rilevazioni o errori sistematici tali da portare la posizione della carrozzina fuori dallo spazio considerato. Si notano però alcuni errori di posizionamento locali, ovvero del rumore evidente nel percorso sviluppato che però non hanno comportato problemi nell'uso del sistema di localizzazione per la guida automatica, come sarà mostrato nelle sezioni seguenti. Inoltre si può notare che in alcuni punti la traccia del percorso rilevato non è continua, ovvero il sistema di localizzazione non è stato in grado di localizzare la carrozzina. Questo è dovuto al movimento eccessivamente rapido della carrozzina che ha prodotto immagini mosse in cui non erano rilevabili i

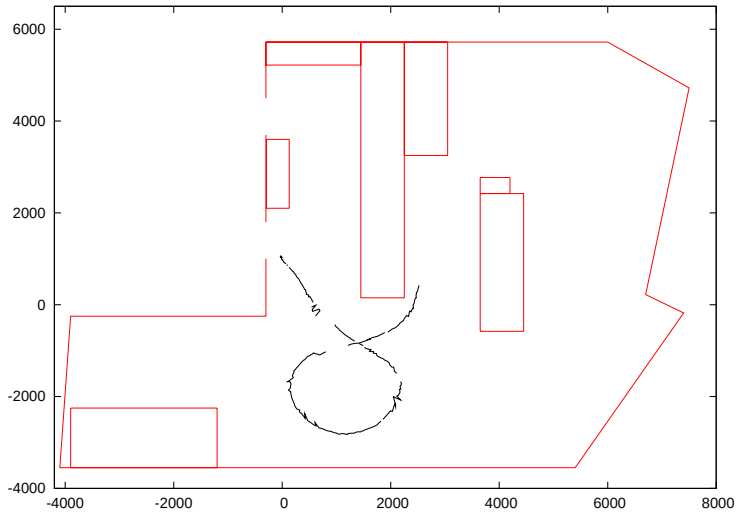


Figura 7.9: Percorso registrato con il sistema di localizzazione

marker presenti sul soffitto. Questo problema non è stato considerato critico in quanto il sistema di localizzazione è di supporto alla guida autonoma che sviluppa velocità di movimento basse, soprattutto in fase di sviluppo del sistema.

7.4 Guida semiautonoma

La guida semiautonoma della carrozzina permette di far muovere la carrozzina seguendo fedelmente il comando imposto dall'utente quando non si è in presenza di ostacoli, intervenendo solo in situazioni di potenziale pericolo. L'obiettivo di questo lavoro è di sviluppare semplici comportamenti di evitamento ostacoli, allo scopo di dimostrare che la strada intrapresa per sviluppare una carrozzina dalle funzionalità estese ha delle potenzialità da sfruttare. Non si è dunque posta particolare attenzione nel considerare situazioni complesse in cui la carrozzina è chiamata a operare, ma solo esempi semplici effettuati in ambienti controllati.

Il sistema di evitamento ostacoli sviluppato prevede che la zona scansionata dai sensori laser sia suddivisa in "spicchi", all'interno dei quali viene selezionata, per ogni scansione, la distanza minima rilevata. Il valore di distanza minima rilevata in ogni zona viene fuzzyficato per essere utilizzato nelle decisioni del controllore fuzzy. A differenza di quanto supposto nella Sezione 6.2.4, si è notato in fase di test che l'uso di un solo tipo di dato fuzzy

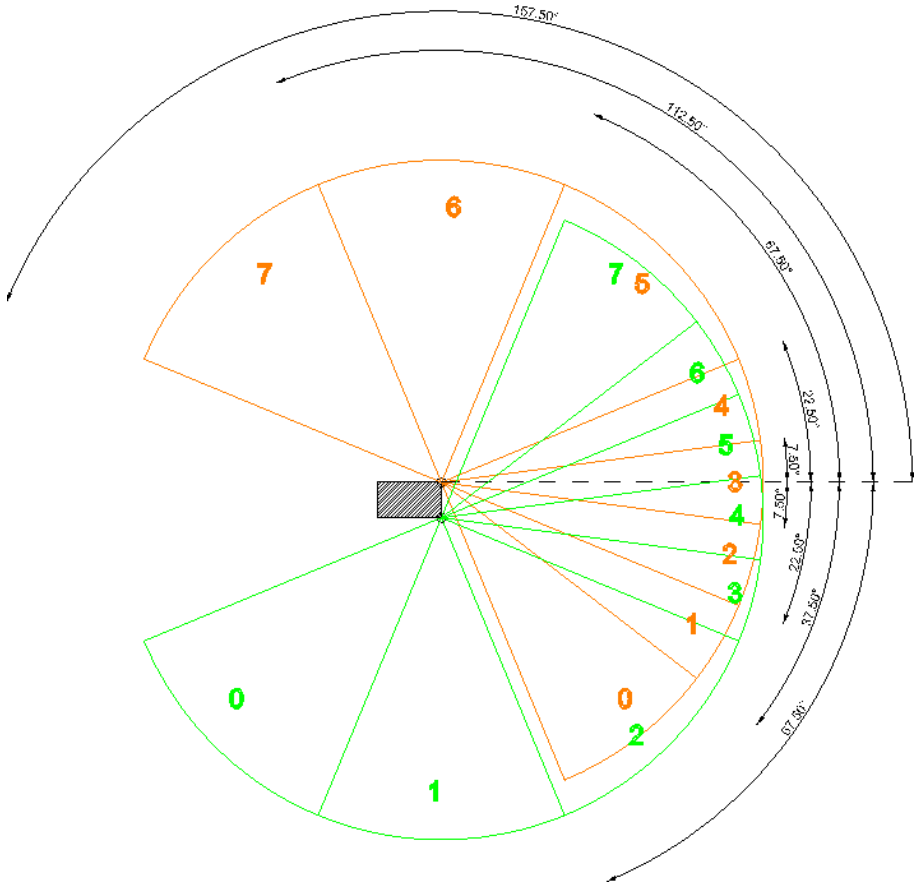


Figura 7.10: Zone angolari identificate per i due scanner laser

per descrivere i valori della distanza non è sufficiente. Infatti è immediato notare che gli ostacoli che possono interessare il moto della carrozzina sulle zone laterali non sono a distanze superiori al metro, mentre ostacoli frontali anche a distanza di due metri sono da tenere in considerazione, in quanto è necessario iniziare delle fasi di rallentamento per evitare la collisione. Si è scelto di mantenere la suddivisione di ogni tipo di dati al massimo in quattro insiemi fuzzy, ovvero `VICINO`, `MEDIO`, `LONTANO`, `MOLTOLONTANO`, dove i limiti e la forma degli insiemi fuzzy possono essere stabiliti differentemente per ogni tipo di dato. In alcune zone si è ritenuto superfluo utilizzare quattro insiemi per descrivere il dato di distanza e ci si è quindi limitati a utilizzarne solo alcuni tra quelli proposti.

Le zone in cui ogni scansione è suddivisa sono 8 e sono numerate in senso antiorario a partire da 0, come mostrato in Figura 7.10. Considerando che i sensori sono posizionati uno a destra e uno a sinistra della carrozzina, è

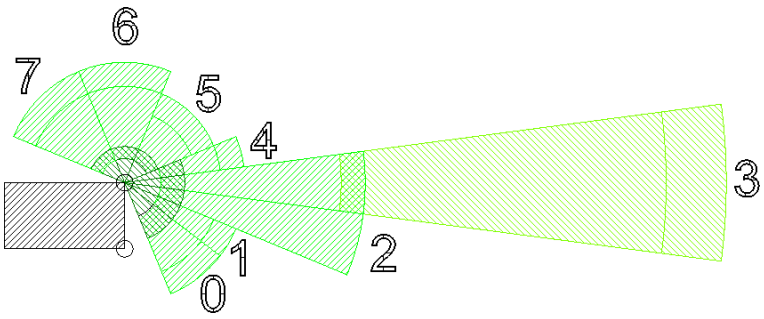


Figura 7.11: Zone utilizzate nella specifica dei tipi di dati fuzzy relativi al laser sinistro

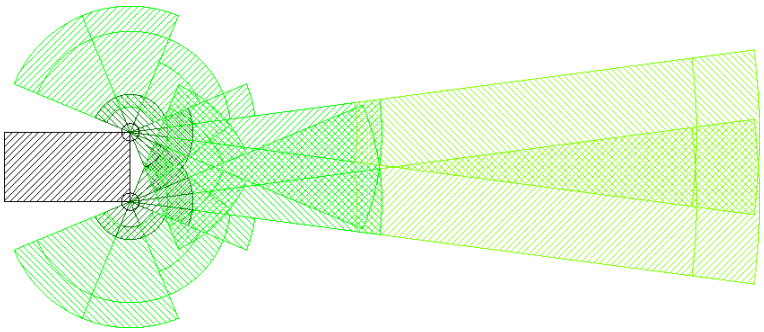


Figura 7.12: Zone utilizzate nella specifica dei tipi di dati fuzzy relativi a entrambi gli scanner laser

facile notare che la zona 0 del sensore sinistro è simmetrica alla zona 7 del sensore destro, la zona 1 del sensore sinistro è simmetrica alla zona 6 del sensore destro, etc. Grazie a questa simmetria si è associato un diverso tipo di dato fuzzy a ogni coppia di zone simmetriche e non a ogni singola zona. In Tabella 7.2 sono riportati i limiti angolari di ogni zona e il tipo di dato che si è scelto di associare, mentre la forma e i limiti degli insiemi fuzzy associati a ogni tipo di dato identificato è riportata in Tabella 7.3. In Figura 7.11 sono mostrate le zone coperte dagli insiemi fuzzy specificati per il sensore di sinistra, mentre in Figura 7.12 sono sovrapposte le zone coperte da entrambi i sensori laser. Le tre tonalità di colori differenti utilizzate indicano gli insiemi fuzzy associati alle distanze: il verde più scuro indica `VICINO` e le tonalità più chiare rappresentano in ordine `MEDIO` e `LONTANO`, mentre l'insieme `MOLTOLONTANO` non è mostrato per non complicare ulteriormente la figura.

Per poter sviluppare un sistema di evitamento ostacoli efficiente è necessario conoscere la velocità di movimento della carrozzina. Senza questo dato

Zona		Inizio	Fine	Tipo di dato
Left	0	-67.5	-37.5	MINDISTOST_LEFT0_RIGHT7
	1	-37.5	-22.5	MINDISTOST_LEFT1_RIGHT6
	2	-22.5	-7.5	MINDISTOST_LEFT2_RIGHT5
	3	-7.5	7.5	MINDISTOST_LEFT3_RIGHT4
	4	7.5	22.5	MINDISTOST_LEFT4_RIGHT3
	5	22.5	67.5	MINDISTOST_LEFT5_RIGHT2
	6	67.5	112.5	MINDISTOST_LEFT6_RIGHT1
	7	112.5	157.5	MINDISTOST_LEFT7_RIGHT0
Right	0	-157.5	-112.5	MINDISTOST_LEFT7_RIGHT0
	1	-112.5	-67.5	MINDISTOST_LEFT6_RIGHT1
	2	-67.5	-37.5	MINDISTOST_LEFT5_RIGHT2
	3	-37.5	-22.5	MINDISTOST_LEFT4_RIGHT3
	4	-22.5	-7.5	MINDISTOST_LEFT3_RIGHT4
	5	-7.5	7.5	MINDISTOST_LEFT2_RIGHT5
	6	7.5	22.5	MINDISTOST_LEFT1_RIGHT6
	7	22.5	67.5	MINDISTOST_LEFT0_RIGHT7

Tabella 7.2: Angoli per la suddivisione in zone della scansione effettuata con i sensori laser

infatti il controllore sarebbe ad anello aperto, in quanto esso governerebbe il moto della carrozzina senza conoscere e poter gestire gli effetti prodotti. La stima della velocità tangenziale è effettuata, come presentato nella Sezione 6.2.5, integrando nel tempo i dati dell'accelerazione lineare rilevati dal sensore XSens MTi, mentre la velocità rotazionale è fornita direttamente dal sensore stesso. L'integrazione nel tempo di dati rumorosi può divergere, ovvero produrre valori inesatti che crescono anche indefinitamente. Per valutare se fosse possibile utilizzare la velocità stimata per effettuare il controllo, si sono registrati e successivamente analizzati i dati di velocità in un tempo di circa 20 minuti. Il risultato ottenuto è mostrato in Figura 7.13 ed è evidente che la velocità stimata diverge rapidamente. Infatti, considerando che all'inizio e alla fine della registrazione la carrozzina era ferma e che in numerosi altri momenti è stata fermata, si dovrebbero notare dei tratti in cui la velocità scende verso zero, mentre questo non accade mai. In Figura 7.14 è riportata la velocità rilevata nei primi 10 secondi della registrazione precedente, in cui la carrozzina era ferma: è evidente il trend di crescita della velocità.

Le cause della deriva sono da ricercarsi nei dati di partenza, ovvero nelle accelerazioni rilevate dal sensore XSens MTi. In Figura 7.15 sono riportate

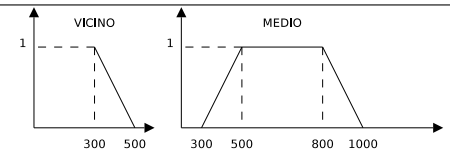
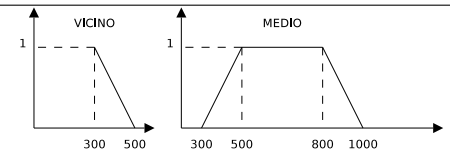
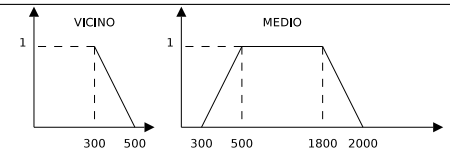
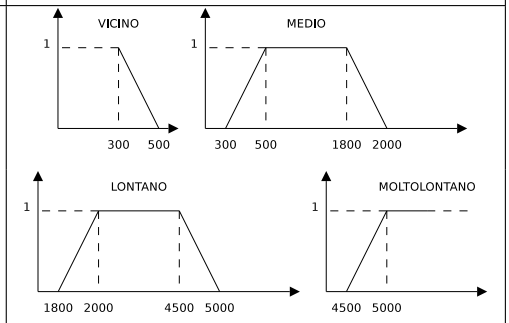
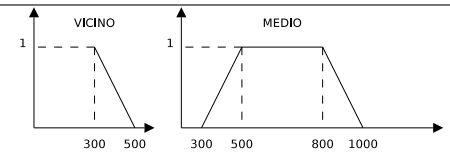
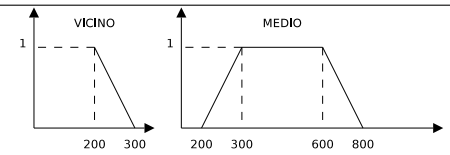
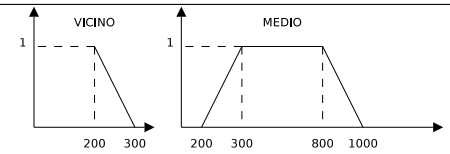
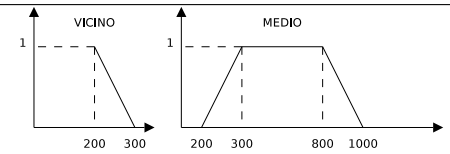
Tipo di dato	Insiemi fuzzy
MINDISTOST_LEFT0_RIGHT7	
MINDISTOST_LEFT1_RIGHT6	
MINDISTOST_LEFT2_RIGHT5	
MINDISTOST_LEFT3_RIGHT4	
MINDISTOST_LEFT4_RIGHT3	
MINDISTOST_LEFT5_RIGHT2	
MINDISTOST_LEFT6_RIGHT1	
MINDISTOST_LEFT7_RIGHT0	

Tabella 7.3: Insiemi fuzzy utilizzati per fuzzyficare le distanze rilevate con i sensori laser

le accelerazioni registrate sugli assi x e y nei primi 10 secondi della registrazione. È evidente che la media dei dati non è nulla, ovvero è presente una componente che durante l'integrazione viene sempre sommata e porta a divergere il valore dell'integrale stesso. Questo può essere dovuto al posizionamento non perfetto del sensore. Ad esempio, se l'asse x del sensore fosse leggermente inclinato verso il basso e non perfettamente parallelo al pavimento, nella misura dell'accelerazione rientrerebbe una componente della gravità terrestre, che quindi porterebbe la misura ad avere una media non nulla. Questo accade in quanto il sensore non misura direttamente le accelerazioni, ma le forze che agiscono su di esso. Estrando i dati di accelerazione dalle forze, nota la massa mobile, si ottiene che la forza di gravità produce un output. Per ovviare a questo problema si è pensato di rimuovere la media dell'accelerazione rilevata in un periodo di tempo in cui la carrozzina era ferma e quindi sottrarre tale valore dai dati di accelerazione utilizzati per calcolare la velocità. Purtroppo però analizzando meglio i dati dell'accelerazione in momenti diversi in cui la carrozzina era ferma si è evidenziato che le medie dell'accelerazione erano molto differenti tra loro, come mostrato in Figura 7.16. Questo è probabilmente dovuto al fatto che la carrozzina si può fermare con il sensore inclinato in modo leggermente diverso ogni volta, ad esempio, per colpa delle ruote tassellate o della conformazione del terreno. L'impossibilità di identificare un valore medio che rappresenta l'accelerazione rilevata a veicolo fermo, impedisce di pensare di correggere il metodo di stima della velocità semplicemente sottraendo l'accelerazione media a ogni rilevazione. Uno studio più approfondito del sensore e un'analisi metodica dei dati da esso prodotti potrebbe portare a creare un metodo di stima della velocità basato sull'integrazione nel tempo dei dati dell'accelerazione più preciso ed efficiente.

Si è deciso di non utilizzare l'informazione relativa alla velocità di movimento della carrozzina, ma solo quella relativa alla velocità rotazionale. Questo impedisce di sviluppare un controllo preciso della carrozzina, in quanto non è possibile conoscere a che velocità essa si sta muovendo, ma solo a che velocità sta ruotando. Studiando in modo empirico il comportamento della carrozzina si è però concluso che i movimenti più problematici da eseguire per la carrozzina sono le rotazioni sul posto o con bassa velocità, in quanto l'orientamento delle due ruote libere anteriori può impedire o favorire la rotazione della carrozzina. Il movimento in direzione frontale è invece molto più predicibile, ovvero la carrozzina risponde in modo più preciso al comando imposto con la leva.

Il dato relativo alla velocità rotazionale è stato fuzzyficato con il tipo di dato `SPEEDANGLE`, i cui insiemi fuzzy sono mostrati in Tabella 7.4.

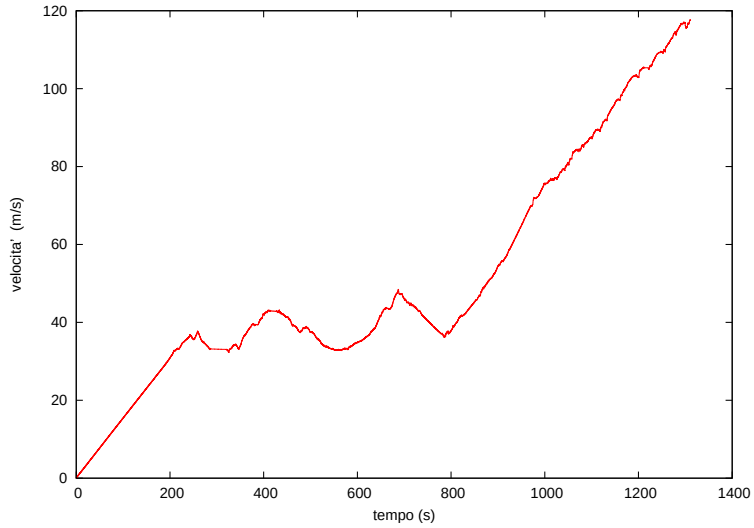


Figura 7.13: Velocità calcolata come integrale dell'accelerazione in un tempo di 20 minuti

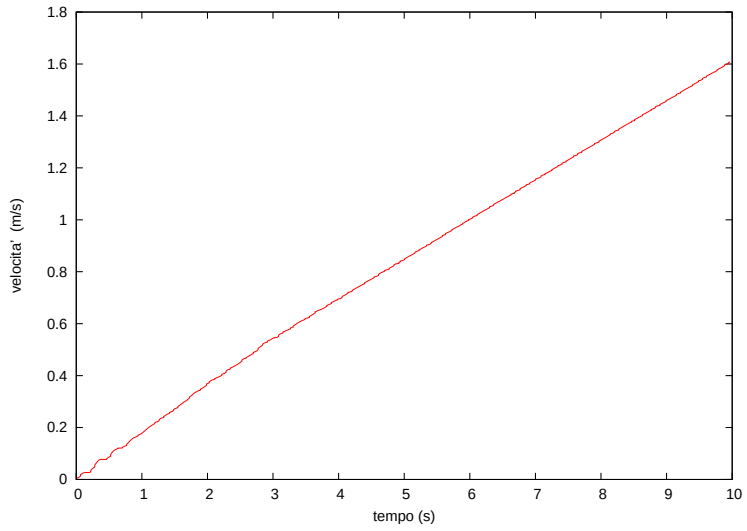


Figura 7.14: Velocità calcolata come integrale dell'accelerazione in un tempo di 10 secondi

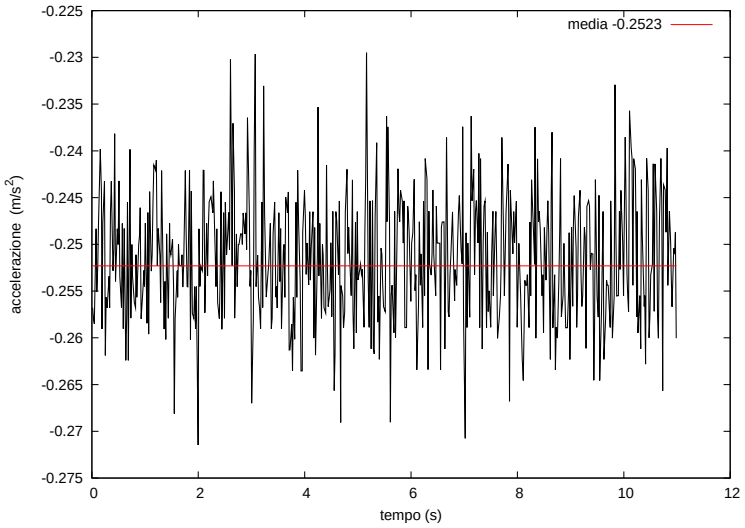


Figura 7.15: Accelerazione rilevata sull'asse x in un tempo di 10 secondi con la carrozzina ferma

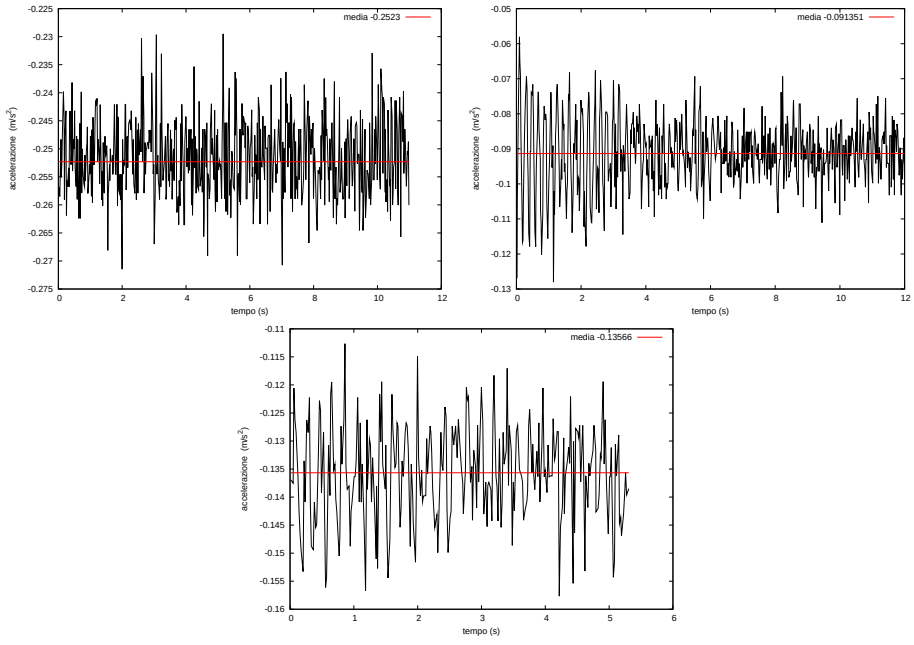


Figura 7.16: Confronto tra i dati di accelerazione rilevati sull'asse x con la carrozzina ferma

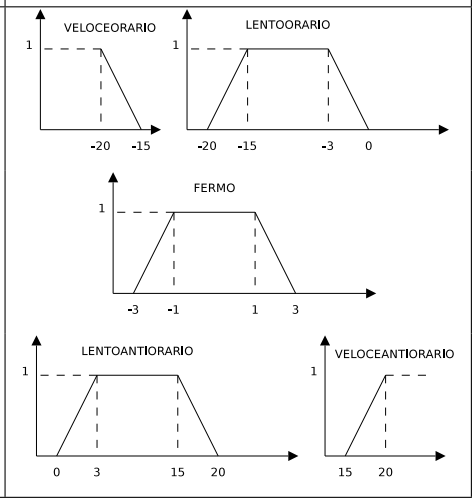
Tipo di dato	Insiemi fuzzy
SPEEDANGLE	

Tabella 7.4: Insiemi fuzzy utilizzati per i dati relativi alla velocità rotazionale

Utilizzando gli insiemi fuzzy introdotti per descrivere gli ostacoli e la velocità rotazionale è stato possibile sviluppare il comportamento `Obstacle-AvoidMinDist` che governa il movimento della carrozzina.

I comportamenti emergenti che la carrozzina evidenzia durante la guida semiautonoma sono i seguenti:

1. Arresto in presenza di ostacoli frontali.
2. Mantenimento della distanza di sicurezza dai muri.
3. Evitamento di ostacoli frontali decentrati rispetto alla carrozzina.
4. Impedimento della rotazione in presenza di ostacoli laterali.

L'arresto in presenza di ostacoli frontali è una delle funzionalità più semplici e allo stesso tempo più utili che il sistema di evitamento ostacoli rende disponibile. Muovendo la carrozzina in direzione perpendicolare verso un muro, essa rallenta mano a mano la sua corsa, fermandosi in sicurezza quando si è a una distanza di circa 20 centimetri dal muro. È inoltre garantito che in presenza di ostacoli frontali vicini la carrozzina non può né muoversi in avanti né ruotare, in quanto questi movimenti possono portare alla collisione con l'ostacolo.

Il mantenimento della distanza di sicurezza dai muri garantisce di poter percorrere lunghi corridoi seguendo uno dei muri che li delimita. Quando si muove la carrozzina in direzione incidente a un muro, la sua traiettoria viene deviata e resa parallela al profilo del muro, come schematizzato in

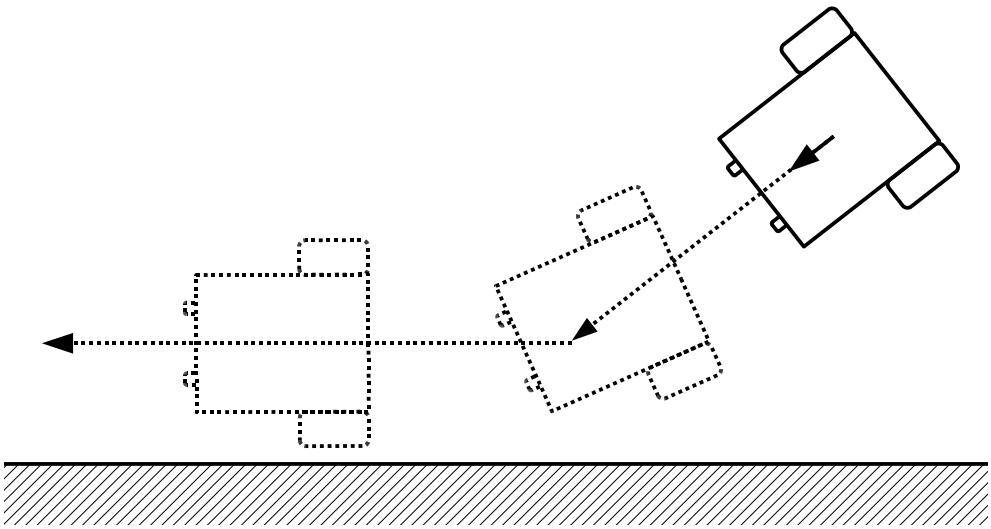


Figura 7.17: Mantenimento della distanza costante da un muro della carrozzina

Figura 7.17. In questo modo un utente che non fosse sufficientemente preciso nel comandare la carrozzina deve semplicemente muovere la carrozzina verso il muro che vuole seguire e lasciare che il sistema di assistenza alla guida posizioni la carrozzina in direzione parallela al profilo del muro. Il sistema di assistenza alla guida impedisce che l'utente imponga comandi di rotazione che lo possano portare a impattare con il muro stesso, aiutandolo quindi a percorrere tratti rettilinei delimitati da pareti. Si noti che qualora l'approccio al muro fosse effettuato in direzione perpendicolare la carrozzina si arresterebbe senza iniziare il moto di inseguimento del profilo del muro.

Quando si impone il comando "avanti" e sono presenti ostacoli frontali decentrati rispetto alla carrozzina ed è disponibile uno spazio di manovra sufficientemente ampio, la carrozzina modifica la sua traiettoria in modo da evitare l'ostacolo, ovvero schiva l'ostacolo e continua la sua marcia in una direzione differente da quella scelta dall'utente. Il percorso seguito dalla carrozzina per evitare con una svolta a sinistra un ostacolo di forma rettangolare è mostrato in Figura 7.18. Il comando imposto era quello di "avanti", quindi una volta evitato l'ostacolo la carrozzina ha proseguito nella direzione utilizzata per evitare l'ostacolo. Un eventuale aggiramento completo dell'ostacolo è lasciato alla volontà e al comando dell'utente, che verrebbe assistito in questo caso solo nella prima fase della manovra. La posizione della carrozzina durante il moto è stata rilevata con il sistema di localizzazione, ma è importante ricordare che il sistema di evitamento ostacoli è

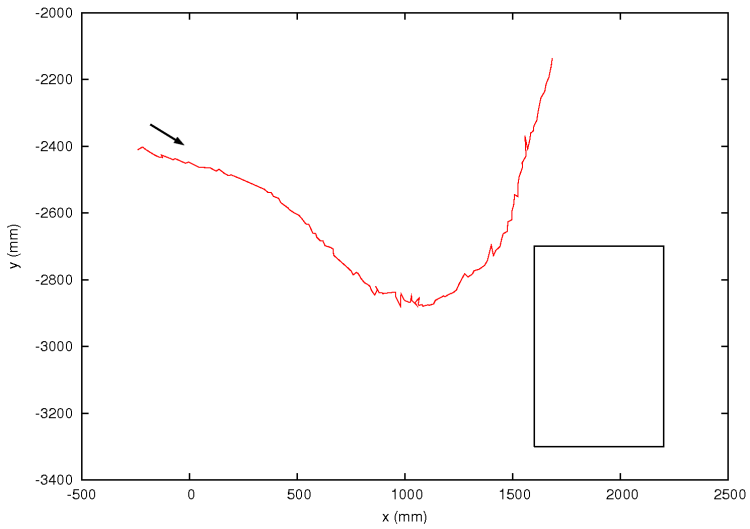


Figura 7.18: Percorso seguito dalla carrozzina per evitare un ostacolo. La freccia indica la direzione del movimento.

indipendente dalla localizzazione ed è quindi disponibile anche in ambienti ignoti.

Qualora l'utente comandi di ruotare in una direzione tale da portare a un impatto con un ostacolo, il comando viene ignorato, come illustrato schematicamente in Figura 7.19. Questa azione è combinata con tutte le precedenti: quindi qualora fossere presenti, ad esempio, un ostacolo frontale e uno laterale sinistro, l'unico movimento possibile sarebbe una rotazione verso destra, in quanto sia la rotazione verso sinistra che il movimento frontale sarebbero impediti al fine di evitare collisioni. Qualora l'ostacolo frontale fosse molto vicino anche la rotazione verso destra sarebbe impedita, in quanto porterebbe la parte anteriore sinistra della carrozzina a collidere. In questo caso l'unico movimento permesso sarebbe quello all'indietro. Si noti che il sistema di evitamento ostacoli non può intervenire sui movimenti all'indietro, dato che la zona posteriore della carrozzina non è coperta da alcun sensore di rilevamento ostacoli.

Il sistema proposto non è da considerarsi completo e non è in grado di evitare correttamente tutti gli ostacoli che si possono presentare durante l'esecuzione di un percorso e di impedire le collisioni in ogni caso. Innanzitutto bisogna considerare che per come sono posizionati i laser, ovvero paralleli al pavimento, essi non sono in grado di rilevare ostacoli di altezza inferiore al loro piano di scansione. Inoltre non possono in alcun modo rilevare la presenza di rampe di scale in discesa o di altri avvallamenti del terreno.

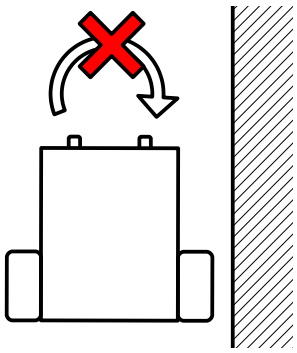


Figura 7.19: Schema che illustra il funzionamento del sistema nel caso in cui si richieda una rotazione perocolosa

Anche le scale in salita non sono rilevate correttamente, in quanto il laser misura la distanza dello scalino che è alla sua stessa altezza dal pavimento, e non quella del primo scalino. I tavoli non sono rilevati in modo corretto, in quanto risultano visibili solo le gambe di sostegno ma non il piano; allo stesso modo alcuni tipi di sedie e di altri oggetti di arredamento non sono correttamente riconosciuti. Più in generale, gli ostacoli che non presentano una parete verticale vengono rilevati in modo non corretto e dunque non sempre evitati. I sensori laser stessi hanno dei limiti e non riconoscono come ostacoli o riconoscono in modo errato i vetri, in quanto il segnale emesso li attraversa o viene riflesso in modo anomalo.

Considerando gli ostacoli riconosciuti, possiamo identificare alcune situazioni tipiche in cui il sistema non si comporta in modo adeguato. In primo luogo il sistema si comporta in modo inaspettato nei passaggi stretti, come, ad esempio, le porte. Cercando, ad esempio, di attraversare una porta aperta il sistema reagisce in maniera differente in base al posizionamento della carrozzina: se essa è ben centrata nell'ingresso il sistema rallenta il moto della carrozzina e ne permette l'ingresso, qualora invece la carrozzina è leggermente decentrata, la carrozzina viene deviata in modo da evitare l'ostacolo rilevato, impedendone l'ingresso nella porta. Nelle prove effettuate si è notato che risulta difficile far passare la carrozzina in porte larghe 80cm, come quelle abitualmente installate nelle normali abitazioni, mentre è più agevole passare in passaggi di larghezza almeno pari a 90cm. Dal momento che il passaggio attraverso porte è una delle attività fondamentali per il movimento in interni, è evidente che questo comportamento andrà raffinato prima di poter considerare il sistema effettivamente utilizzabile in contesti reali.

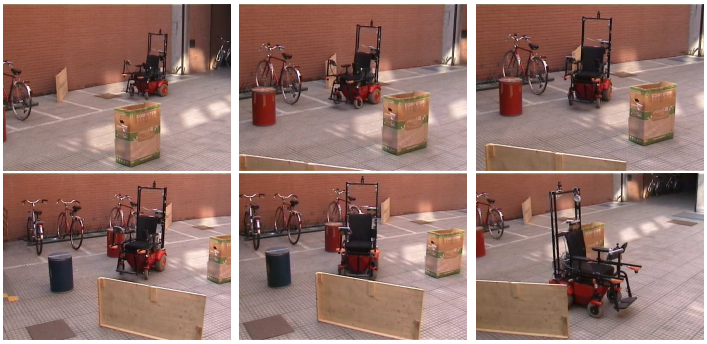


Figura 7.20: Guida semiautonoma in un percorso con ostacoli

Dal punto di vista dei comportamenti, il sistema realizzato è fortemente espandibile, sia tramite l'aggiunta di nuove regole fuzzy, sia tramite l'inserimento di nuovi comportamenti per gestire situazioni particolari, come, ad esempio, i passaggi stretti. Il comportamento di evitamento ostacoli realizzato ha dimostrato che è possibile sviluppare un sistema intelligente che assista l'utente nella guida della carrozzina. Per quanto riguarda gli ostacoli che non possono essere rilevati, è necessario invece aggiungere altri sensori che si dedichino ai casi non coperti dai laser: ad esempio, uno scanner laser inclinato verso il basso permetterebbe di rilevare sia le rampe di scale in discesa che quelle in salita.

Alcune foto scattate durante un percorso eseguito in modo semiautonomo dalla carrozzina, in cui l'unico comando imposto era la direzione avanti, mentre tutti i movimenti rotazionali erano imposti dal sistema di evitamento ostacoli, sono mostrate in Figura 7.20. Alcune immagini riprese in soggettiva dalla carrozzina durante l'esecuzione dello stesso percorso con le stesse modalità sono mostrate in Figura 7.21.

7.5 Guida autonoma

Il sistema di guida autonoma permette di pianificare percorsi, grazie all'uso del pianificatore Spike, e di eseguirli seguendo successivamente i via point identificati. L'obiettivo di questo lavoro è, come nel caso della guida semiautonoma, mostrare che il sistema sviluppato costituisce una buona base per lo sviluppo di comportamenti complessi che servano a dotare di autonomia la carrozzina elettrica.

Come illustrato nella Sezione 6.2.7, il sistema di guida autonoma è basato sull'azione congiunta di più membri del sistema:

Figura 7.21: Guida semiautonoma in un percorso con ostacoli ripresa dal sedile della carrozzina

- Il sistema di localizzazione è necessario per conoscere la posizione della carrozzina nel momento in cui si richiede la pianificazione di un percorso e durante tutta l'esecuzione del percorso per calcolare la distanza e l'angolo tra la carrozzina e il via point.

- Il pianificatore è necessario per calcolare il percorso teorico che porta dal punto di partenza al punto di arrivo.

- I membri che gestiscono i sensori laser sono necessari sia per identificare gli ostacoli dinamici da aggiungere alla mappa nel momento in cui si pianifica un percorso, sia per fornire i dati relativi alle distanze dagli ostacoli durante l'esecuzione del percorso.

- Il controllore fuzzy governa il moto della carrozzina per raggiungere i via point mantenendo attivo il comportamento di evitamento ostacoli.

Innanzitutto è necessario descrivere i due tipi di dati fuzzy utilizzati per la gestione del movimento della carrozzina, ovvero `VIAPOINTDISTANCE` che descrive la distanza dal via point che si sta cercando di raggiungere e `VIAPOINTANGLE`, che descrive la rotazione da compiere per orientare la carrozzina verso il via point. Gli insiemi fuzzy utilizzati sono mostrati in Tabella 7.5 e le regole create per il movimento autonomo utilizzano come dati per le decisioni la distanza dal punto di via e la velocità rotazionale rilevata al tempo attuale e al ciclo precedente.

Per valutare il comportamento del sistema si sono effettuati test che hanno permesso di conoscere le possibilità offerte dal sistema e di scoprirne i limiti. Il primo test effettuato è costituito dalla richiesta di pianificazione di un movimento rettilineo non ostruito da ostacoli. Il percorso pianificato,

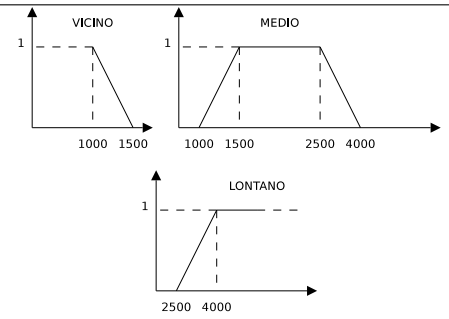
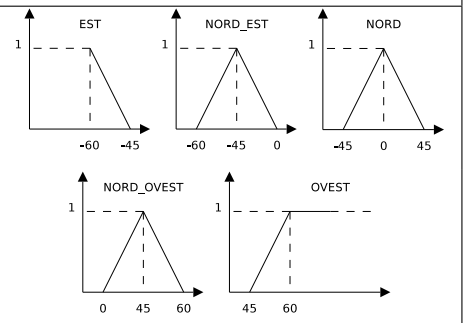
Tipo di dato	Insiemi fuzzy
VIAPOINTDISTANCE	
VIAPOINTANGLE	

Tabella 7.5: Insiemi fuzzy utilizzati per i dati relativi alla distanza e all'orientamento rispetto ai via point da raggiungere

gli ostacoli dinamici rilevati al momento della pianificazione e il percorso seguito dalla carrozzina è visualizzato in Figura 7.22. La carrozzina si muove agevolmente nel percorso raggiungendo facilmente la destinazione. Osservando il grafico si potrebbe pensare che, qualora il punto di goal fosse stato spostato a una quota x inferiore a -2000mm, la presenza del profilo degli ostacoli dinamici rilevato con i sensori laser avrebbe impedito la pianificazione del percorso. È facile però notare che la forma del profilo rilevato in quel tratto è l'approssimazione di un arco di circonferenza, che non rappresenta un vero ostacolo, ma il valore massimo di distanza misurato dallo scanner laser e quindi queste informazioni non sono significative ai fini della pianificazione. Per estromettere queste informazioni dalla pianificazione, al fine di non impedire la creazione di percorsi complessi, il pianificatore utilizzato permette di definire una"zona di competenza" di forma quadrata all'esterno della quale i dati relativi agli ostacoli dinamici non sono considerati. In questo caso, considerando che la distanza massima rilevata dai sensori laser è di circa 5 metri, si è utilizzata una zona di 2 metri, in modo da considerare solo gli ostacoli rilevati localmente al momento della pianificazione e di non compromettere la possibilità di creare percorsi complessi.

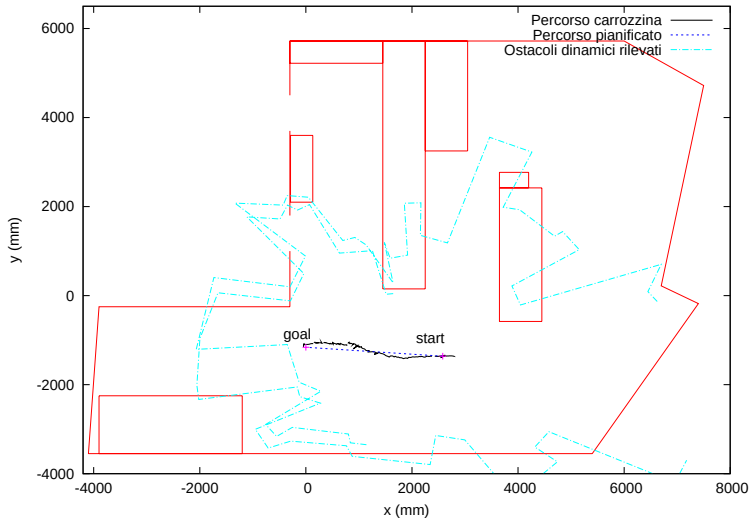


Figura 7.22: Esecuzione automatica di un semplice percorso rettilineo

Un caso più complesso considerato è quello di un movimento a "L", ovvero in cui è necessario effettuare una svolta netta in una direzione. Sono stati eseguiti più percorsi di questo tipo, e i risultati sono mostrati nelle Figure 7.23, 7.24 e 7.25. È evidente che l'introduzione di una curva nel percorso rende più difficile l'esecuzione del percorso da parte della carrozzina. Nonostante questo il percorso viene portato a termine e il punto di goal raggiunto, anche se il percorso seguito non ricalca fedelmente quello proposto dal pianificatore.

Non è stato possibile approfondire ulteriormente l'analisi del comportamento della carrozzina nel caso di percorsi più complessi. Infatti la zona coperta dal sistema di localizzazione, ovvero quella in cui sono visibili i marker sul soffitto, non è molto ampia. Per questo motivo è stato difficile realizzare percorsi complessi senza che la carrozzina si ritrovasse in condizioni di dover interrompere l'esecuzione automatica di percorsi per indisponibilità del dato di posizione. Quando l'esecuzione dei percorsi veniva interrotta per mancanza di riferimenti assoluti di posizione, spesso era sufficiente guidare manualmente la carrozzina fino a rendere nuovamente nota la posizione della carrozzina, permettendo così di ripianificare autonomamente il percorso da compiere e riprendere l'esecuzione.

La presenza di ostacoli imprevisti durante l'esecuzione del percorso pianificato comporta quasi sempre l'arresto della carrozzina, mentre si verifica poche volte il caso in cui l'ostacolo viene aggirato e il percorso proseguito.

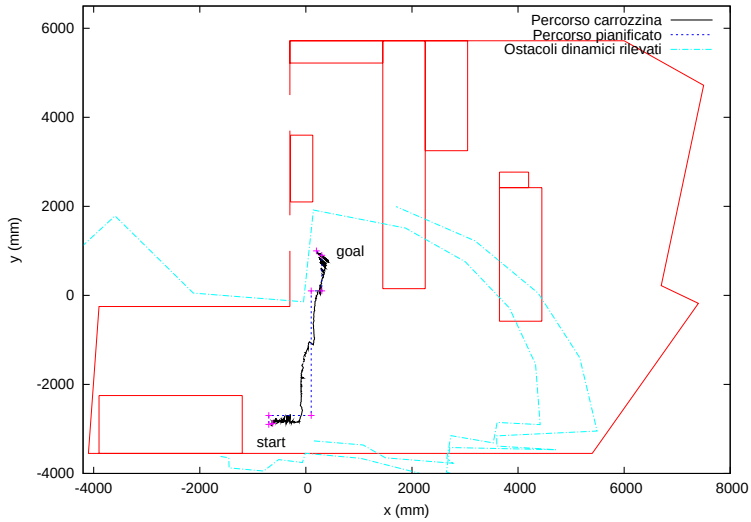


Figura 7.23: Esecuzione automatica di un percorso a "L" (1)

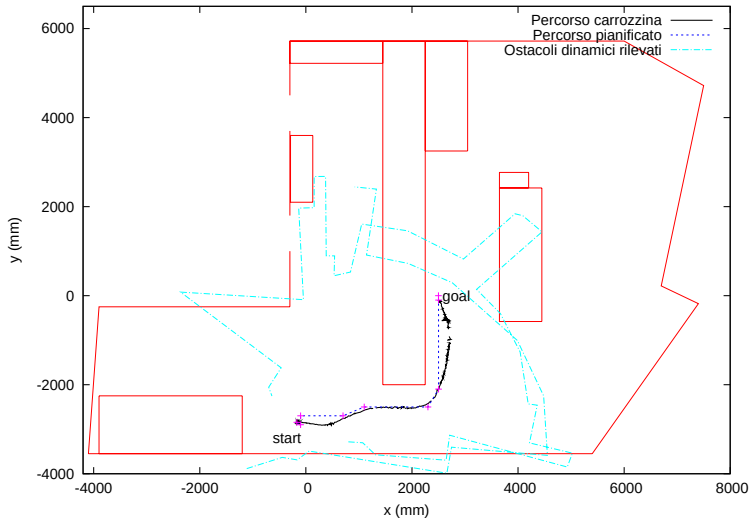


Figura 7.24: Esecuzione automatica di un percorso a "L" (2)

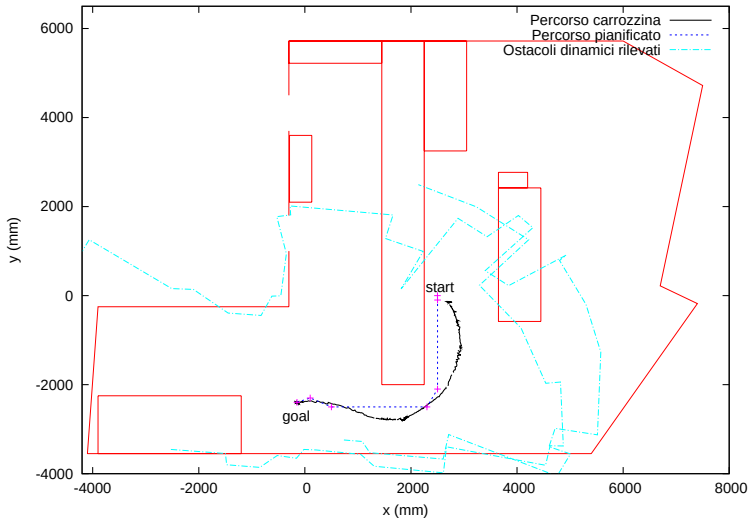


Figura 7.25: Esecuzione automatica di un percorso a "L" (3)

Ci si aspetterebbe che la sovrapposizione degli effetti del comportamento di evitamento ostacoli e di guida autonoma porti ad aggirare completamente l'ostacolo e proseguire il percorso in direzione del via point, ma questo non succede. Infatti la presenza di un ostacolo dovrebbe provocare la produzione di movimenti di controllo che fanno deviare il percorso seguito rispetto alla traiettoria pianificata, annullando tutti i movimenti che il sistema di guida automatica propone per riavvicinarsi alla traiettoria pianificata. Una volta superato l'ostacolo i comandi impartiti dal sistema di guida autonoma non sarebbero più annullati e permetterebbero di riportare la carrozzina sulla traiettoria pianificata, completando quindi l'aggiramento dell'ostacolo. In Figura 7.26 è schematizzato il funzionamento del comportamento previsto.

Le cause della non corretta sovrapposizione dei comportamenti sono legate al fatto che il comportamento di evitamento ostacoli è stato sviluppato basandosi sui movimenti effettuati durante la guida manuale. Per quanto i comportamenti siano tra di loro indipendenti e la guida automatica si comporti, dal punto di vista del controllore, come un dispositivo di input, i movimenti prodotti dal sistema di guida automatica non risultano sufficientemente compatibili con quelli prodotti da un utente e non vengono quindi correttamente corretti dal sistema di evitamento ostacoli. Le linee di sviluppo che si possono seguire per migliorare questo comportamento sono due: adattare il comportamento di guida automatica e rendere il controllo prodotto più simile a quello eseguito da un utente oppure migliorare il

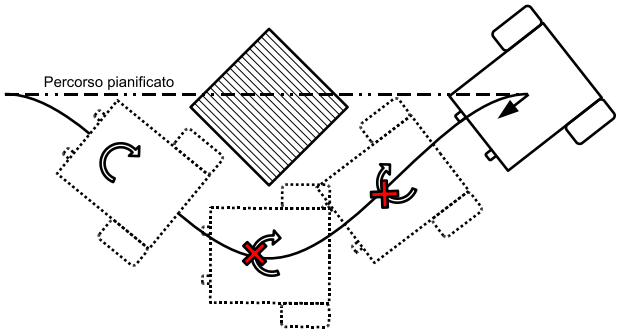


Figura 7.26: Comportamento previsto per l'aggiramento di un ostacolo con azione congiunta del sistema di guida autonoma e di evitamento ostacoli

comportamento di evitamento ostacoli in modo da garantire una corretta reazione del sistema in entrambi i casi di guida citati.

7.6 Tempi di esecuzione

Una volta realizzato il software di controllo si è evidenziata la necessità di capire se la potenza di calcolo messa a disposizione dal computer di bordo alloggiato sulla carrozzina fosse sufficiente. Per valutare questo dato si sono analizzati i periodi di esecuzione dei membri più significativi per il controllo della carrozzina, ovvero il tempo che intercorre tra una attivazione del membro e la successiva. I membri considerati sono:

1. MotorExpert.
2. BrianExpert.
3. JoypadExpert.

Gli altri membri presentano periodi di attivazione più lunghi e sono per questo stati considerati come meno critici.

I tempi di attivazione rilevati in un periodo di 10 secondi, corrispondenti a 500 attivazioni del membro BrianExpert, sono riportati in Figura 7.27. Il tempo teorico di attivazione è fissato a 20ms e la media delle attivazioni ottenute è di circa 22ms, con una deviazione standard associata di 2,6ms. Dal grafico è facile notare che si ottengono punte di ritardo nell'esecuzione anche superiori ai 10ms. Questo evidenzia che la potenza di calcolo disponibile non è del tutto adeguata e può comportare ritardi anche consistenti nella produzione dell'azione di controllo. Le stesse conclusioni si possono

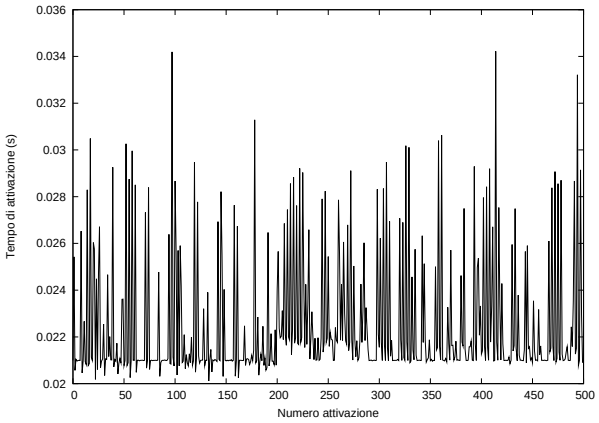


Figura 7.27: Tempi di esecuzione del membro BrianExpert

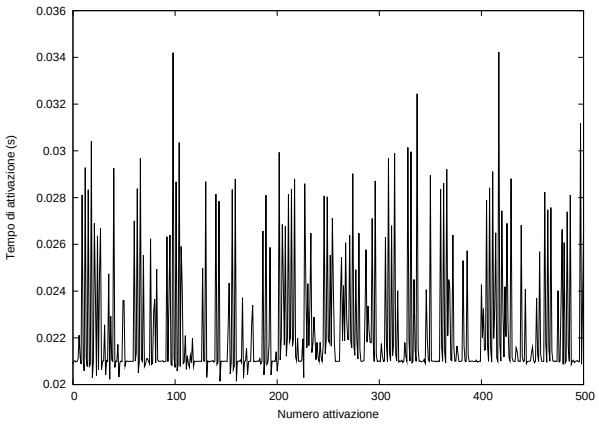


Figura 7.28: Tempi di esecuzione del membro MotorExpert

trarre analizzando i tempi di attivazioni rilevati per i membri MotorExpert e JoypadExpert, i cui grafici sono riportati rispettivamente in Figura 7.28 e 7.29.

Nonostante i ritardi evidenziati nell'esecuzione periodica del ciclo di controllo, il moto della carrozzina risulta comunque fluido e ben controllabile. È però evidente che la configurazione hardware scelta è al limite delle sue capacità ed è auspicabile che per sviluppare ulteriormente il sistema di controllo della carrozzina sia introdotto un secondo computer su cui spostare alcuni membri. In particolare si è cercato di analizzare quale fosse il membro che più richiede risorse di calcolo per la sua esecuzione. Considerando le funzioni svolte è facile supporre che il membro VisionExpert, a cui è affidata

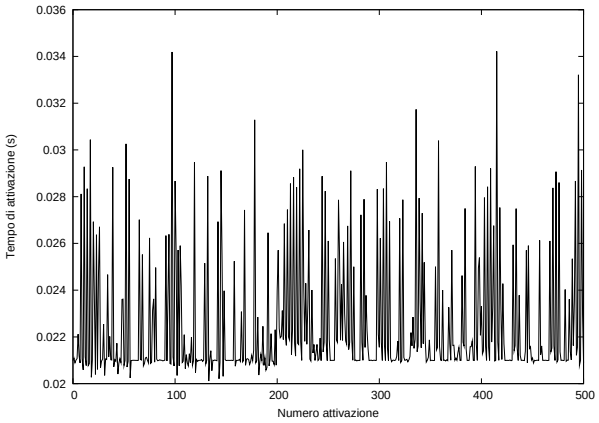


Figura 7.29: Tempi di esecuzione del membro JoypadExpert

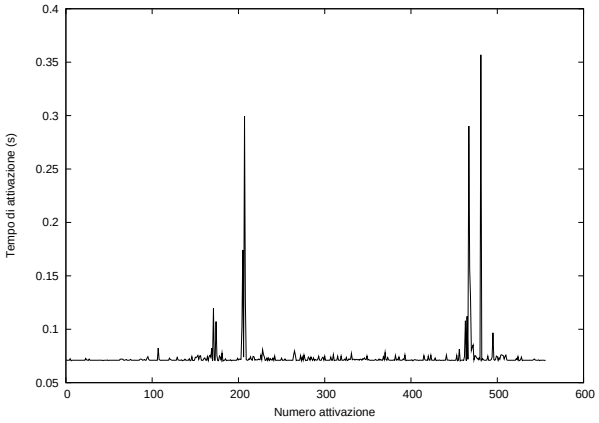


Figura 7.30: Tempi di esecuzione del membro VisionExpert

la gestione del sistema di localizzazione, sia il più pesante a livello di tempi di calcolo. In Figura 7.30 sono riportati i tempi di esecuzione del membro VisionExpert. Il tempo teorico di esecuzione è di 66ms, corrispondenti all'analisi di 15 fotogrammi al secondo, ma è subito evidente che in molti casi l'analisi richiede tempi nettamente superiori, a volte anche di alcuni decimi di secondo. Su 557 attivazioni considerate, il tempo di esecuzione medio è risultato pari a 0.074ms, con una deviazione standard associata di 20ms. VisionExpert è dunque il candidato principale a essere spostato su un altra unità di calcolo, dove potrebbe sfruttare molte più risorse e non influenzerebbe l'esecuzione dei membri dedicati al controllo della carrozzina.

Capitolo 8

Conclusioni e sviluppi futuri

In questo capitolo si svolge l'analisi dei punti di forza e si presentano possibili miglioramenti del sistema realizzato, basandosi sui risultati ottenuti (illustrati nel Capitolo 7). Durante l'analisi sono proposte alcune soluzioni che possono rafforzare i punti critici. L'ultima sezione presenza un possibile sviluppo futuro per il sistema di controllo della carrozzina.

8.1 Conclusioni

Grazie al contributo del presente lavoro si è trasformata una normale carrozzina elettrica commerciale, per il cui comando era disponibile un solo dispositivo, in una carrozzina dalle funzionalità estese, dotata di logica di controllo, funzionalità di assistenza alla guida e di guida autonoma.

Le fasi che si sono susseguite nello sviluppo della carrozzina sono le seguenti:

- Scelta dei sensori e dei componenti hardware adatti alla robotizzazione della carrozzina.
- Creazione di una struttura di supporto che, una volta montata sulla carrozzina, permette di alloggiare i sensori e i dispositivi da utilizzare.
- Progetto e implementazione di un'architettura di controllo modulare che permette di realizzare funzionalità di guida assistita e di guida autonoma e che supporta l'uso di più dispositivi di comando.
- Test del sistema così realizzato in ambienti controllati.

In particolare si è introdotta la possibilità di guidare la carrozzina con diversi dispositivi di comando quali, ad esempio, il joypad senza fili. Inoltre sono state sviluppate funzionalità di assistenza alla guida, che evitano

che l'utente possa incorrere in situazioni pericolose (come le collisioni con ostacoli) e allo stesso tempo aiutano la guida, assistendo l'utente in manovre critiche quali l'evitamento ostacoli. Un'ultima funzionalità sviluppata riguarda il movimento autonomo della carrozzina, coadiuvato dalla presenza di un sistema di localizzazione e da un pianificatore di percorsi.

Lo sviluppo di una carrozzina autonoma d'ausilio ai disabili motori non può considerarsi concluso con il presente lavoro. Infatti il sistema presenta notevoli prospettive di sviluppo, sia come nuove funzionalità da aggiungere, sia come miglioramento di ciò che con questo lavoro è stato realizzato.

8.2 Sensori utilizzati

Analizzando il comportamento dei sensori utilizzati ci si è resi conto di alcune caratteristiche che possono essere ricercate per migliorare il comportamento della carrozzina e, se possibile, ridurre ulteriormente i costi di realizzazione.

Gli scanner laser utilizzati per il rilevamento degli ostacoli offrono misure molto precise delle distanze degli oggetti dell'ambiente che li circonda e sono in grado di operare senza vistosi problemi in svariate condizioni. Ad esempio, anche le superfici che assorbono maggiormente la luce, ovvero quelle di colore nero, non creano particolari problemi di misura. Infatti le distanze rilevate possono risultare sovrastimate o sottostimate a causa delle delle caratteristiche della superficie incontrata, ma è molto difficile che il sensore laser non rilevi un ostacolo presente. Questo garantisce che il sistema di evitamento ostacoli possa basarsi su informazioni con un alto grado di affidabilità. Allo stesso tempo però bisogna considerare che gli scanner laser effettuano scansioni solo su un piano, che nel nostro caso è stato posto parallelo al terreno. Questo non permette di rilevare oggetti come i tavoli, il cui profilo rilevato dagli scanner laser è relativo solo ai sostegni che poggiano sul terreno, mentre quasi tutta la superficie di ingombro viene ignorata. Anche tutti gli oggetti che sono più in basso rispetto al piano di scansione non sono rilevabili, così come rampe di scale in discesa, che non risultano assolutamente visibili, o in salita, di cui viene erroneamente stimata la distanza, in quanto viene misurata la distanza dello scalino che si trova ad altezza da terra pari a quella dello scanner laser. Inoltre la zona posteriore e parte della zona laterale della carrozzina non sono coperte dalla scansione dei due scanner laser utilizzati.

Per migliorare il sistema di rilevamento ostacoli è possibile aggiungere nuovi sensori per coprire le zone non considerate. Per questo si potrebbero utilizzare scanner laser inclinati verso il terreno, che permetterebbero di rilevare ostacoli bassi e rampe di scale in discesa (rilevabili come "assenza del

piano del pavimento"). Altrimenti è possibile scegliere di aggiungere sensori più semplici ed economici, come i sonar, per coprire le zone mancanti. Un'altra soluzione possibile, che ridurrebbe ulteriormente i costi di realizzazione, è quella di rinunciare completamente agli scanner laser e utilizzare solo sensori sonar. Ricordando però che il profilo rilevato dagli scanner laser viene utilizzato anche per rappresentare gli ostacoli dinamici all'atto della pianificazione, si conlcude facilmente che la rinuncia all'uso dei sensori laser renderebbe impossibile sviluppare questa funzionalità, in quanto i sensori sonar non sono sufficientemente precisi per costruire una rappresentazione metrica dell'ambiente.

I dati forniti dal sensore inerziale e giroscopico utilizzato sono stati sfruttati in modo molto limitato, infatti di tutti i dati disponibili, ovvero le accelerazioni lungo i tre assi, le velocità di rotazione intorno ai tre assi e le intensità del campo magnetico, l'unico che effettivamente è stato utilizzato nel controllo è il dato relativo alla rotazione della carrozzina. Si è rinunciato a utilizzare i dati di accelerazione per stimare la velocità raggiunta, in quanto con il metodo utilizzato il valore della velocità divergeva molto rapidamente, rendendo tale dato inutilizzabile per il controllo.

Come possibile sviluppo è da prendere in considerazione la possibilità di cercare sul mercato e acquistare un sensore meno complesso, e quindi più economico, che fornisca solo i dati effettivamente necessari. Alternativamente è possibile studiare in modo approfondito il comportamento del sensore al fine di sviluppare un metodo efficace di stima della velocità, che coinvolga nel calcolo un numero maggiore di dati, come ad esempio i dati relativi al campo magnetico, che permetterebbero di compensare l'inclinazione del sensore rispetto al terreno. Un'altra possibilità è quella di abbandonare l'uso del sensore inerziale e giroscopico ed equipaggiare la carrozzina con un sistema di odometria. Tale sistema potrebbe essere sviluppato con tecnologie differenti, come, ad esempio, tramite l'uso di di encoder posti sulle ruote oppure di telecamere che inquadrano il terreno su cui si sviluppa il moto e stimano la velocità del movimento basandosi sull'analisi delle immagini rilevate.

8.3 Sistema di localizzazione

Il sistema di localizzazione realizzato permette di rilevare in tempo reale e in modo semplice la posizione della carrozzina in un ambiente noto opportunamente attrezzato con marker posizionati sul soffitto. I pregi principali di questo sistema sono:

- Costo estremamente contenuto, in quanto i marker sono stampati su semplici fogli di carta e fissati sul soffitto con nastro adesivo.
- Facilità di posizionamento dei marker, in quanto i marker sono posizionati arbitrariamente e la loro posizione relativa viene rilevata in fase di calibrazione del sistema.
- Scalabilità su vaste zone, grazie al numero elevato di marker disponibili.

Il problema principale riscontrato con il sistema di localizzazione è costituito dal rumore che affligge la misure della posizione e dell'orientamento dei marker e che ha reso necessario lo sviluppo di un sistema di correzione dei dati rilevati. Complessivamente il sistema di localizzazione corretto con le modalità proposte si comporta in modo adeguato alle richieste. Infatti rende disponibile in tempo reale la posizione della carrozzina, con una precisione che ha permesso di utilizzarlo come informazione base per lo sviluppo del sistema di guida autonoma.

Un possibile sviluppo del sistema di localizzazione può essere indirizzato al supporto di più mappe di tag per supportare la localizzazione in ambienti differenti. Si pensi, ad esempio, al caso in cui la carrozzina debba essere utilizzata dall'utente sia nella sua abitazione che sul luogo di lavoro. È opportuno che il sistema di localizzazione sia in grado di riconoscere, in base ai marker rilevati, in che luogo si trova, così da riferire la posizione della carrozzina rispetto a un sistema di riferimento assoluto diverso per ogni ambiente considerato e comunicare al pianificatore quale mappa utilizzare.

Un'altro sviluppo consiste in un'analisi approfondita degli errori commessi nella stima della posizione e dell'orientamento del marker, in modo da valutare se esistono strade migliori per correggere gli errori del sistema. Sarebbe anche possibile implementare ex-novo un sistema di localizzazione basato su fiducial marker, cercando se esistono condizioni più favorevoli che permettono di sviluppare un sistema più preciso rispetto a quello utilizzato e ad altri sistemi attualmente disponibili. Ad esempio sarebbe possibile studiare se forme differenti dei marker possono portare a risultati migliori nella stima della posizione e della rotazione del marker.

8.4 Software di controllo

Il software di controllo sviluppato è adeguato ai compiti che deve svolgere e ha come punti di forza la strutturazione modulare gestita dal framework DCDT e il controllore fuzzy Mr.Brian. L'elevato grado di separazione e

indipendenza tra i moduli software realizzati, garantisce che il software di controllo sia facilmente modificabile ed estendibile. Allo stesso modo, anche il controllore fuzzy è basato su un'architettura gerarchica che permette di comporre in modo semplice comportamenti indipendenti, garantendo quindi la semplicità di estensione e di modifica.

Una possibile modifica del software, proposta già nella Sezione 7.6, è la divisione del software su più unità di calcolo. In particolare si è ipotizzato di dividere la parte relativa al sistema di localizzazione, che si occupa di analizzare le immagini rilevate dalla telecamera, dal resto del sistema di controllo. Questa modifica si renderà necessaria qualora si decidesse di sviluppare ulteriormente il software e di introdurre nuovi membri, ad esempio dedicati allo sfruttamento dei dati di sensori aggiuntivi, in quanto nella Sezione 7.6 si è mostrato che la potenza di calcolo disponibile è appena sufficiente per gestire le funzioni attuali.

Una caratteristica del software che è stata trascurata in questo lavoro è l'interfaccia grafica, che è stata realizzata solo per fornire informazioni sul funzionamento del software utili al debugging. Uno sviluppo possibile è la realizzazione di un'interfaccia grafica adatta alle esigenze di un disabile. In quest'ottica è necessario studiare se sia possibile utilizzare una semplice interfaccia grafica o se sia utile integrare lo sviluppo con altri sistemi di input, come dispositivi di comando vocale o di tracciamento del moto oculare. Sarebbe inoltre auspicabile studiare interfacce grafiche differenti rivolte a specifiche categoria di utenti.

8.5 Funzionalità estese

Le funzionalità con cui la carrozzina è stata estesa sono:

- Guida con diversi dispositivi di comando.
- Guida assistita.
- Guida autonoma.

I dispositivi di comando utilizzati per comandare la carrozzina sono il joystick originale e il joypad. La gestione dei dispositivi è affidata ad alcuni moduli software dedicati e il calcolo del comando da inviare alla carrozzina è effettuato dal controllore fuzzy. La gestione dei dispositivi risulta efficiente e precisa, in quanto essi permettono di controllare la carrozzina in modo fluido. Grazie alla modularità del sistema software e del controllore, è molto semplice inserire nella struttura di controllo della carrozzina altri dispositivi di comando. Un possibile sviluppo di questo lavoro riguarda quindi

l'integrazione di altri dispositivi di comando, come, ad esempio, un sistema basato sull'analisi del movimento oculare o della testa, o altri dispositivi che possono risultare più comodi e semplici da utilizzare a determinate categorie di utenti.

La funzionalità di guida assistita impedisce l'esecuzione di comandi impartiti dall'utente che potrebbero portare a collidere con ostacoli. Questa funzionalità è stata testata in ambienti controllati e ha evidenziato risultati interessanti, mostrati nella Sezione 7.4. Le funzionalità di assistenza alla guida più rappresentative sono il mantenimento della distanza di sicurezza dai muri e l'evitamento di ostacoli frontali decentrati rispetto alla carrozzina. Con il mantenimento della distanza costante dai muri è possibile facilitare il compito dell'utente che desidera attraversare corridoi o stanze. L'evitamento ostacoli aiuta invece l'utente nelle manovre, evitando la collisione con gli ostacoli e impostando la manovra che permette di schivare l'ostacolo stesso.

Un punto di forza del sistema sviluppato è dato dal fatto che la scelta dell'azione da intraprendere è eseguita autonomamente dal controllore, senza che l'utente debba selezionare il comportamento desiderato. Ad esempio, il sistema sceglierà automaticamente se seguire un muro o se arrestare la carrozzina basandosi sul movimento con cui la carrozzina si avvicina al muro: se la direzione di approccio è perpendicolare la carrozzina si arresta, altrimenti prosegue il moto affiancandosi al muro stesso. In alcune delle carrozzine funzionalità estese presentate nel Capitolo 2 la scelta della funzionalità è lasciata all'utente, ma è facile immaginare come questa operazione possa essere difficoltosa per alcune categorie di disabili motori. L'uso di un controllore fuzzy reattivo ha reso possibile realizzare un sistema flessibile che reagisce in modo appropriato alle caratteristiche dell'ambiente e ai comandi impartiti dall'utente, senza necessità di selezionare qual è il comportamento desiderato.

Il funzionamento del sistema di assistenza alla guida può essere ulteriormente sviluppato e perfezionato al fine di portarlo a operare correttamente in tutte le condizioni che possono presentarsi durante l'uso pratico di una carrozzina elettrica. Questo risultato è da perseguire soprattutto tramite il perfezionamento delle regole fuzzy che governano il sistema di evitamento ostacoli, senza però trascurare che la disponibilità di ulteriori informazioni, come la velocità tangenziale con cui si muove la carrozzina, permetterebbe di raffinare ulteriormente il comportamento.

La funzionalità di guida autonoma è stata testata su percorsi semplici e si è evidenziato che il sistema permette di pianificare correttamente percorsi, ma l'esecuzione corretta e precisa di tali percorsi non è garantita. Infatti, come mostrato nella Sezione 7.5, la carrozzina non segue in modo molto

preciso il percorso pianificato e questo può portare in alcuni casi a non completare l'esecuzione del percorso. Con il presente lavoro si è mostrato che il sistema di localizzazione proposto, il software di controllo sviluppato e i sensori utilizzati sono in grado di pianificare ed eseguire semplici percorsi, ma per arrivare a sviluppare comportamenti di controllo del moto autonomo che possano essere effettivamente utilizzati da utenti disabili è necessario un lavoro di raffinamento consistente del sistema e delle regole che governano il movimento autonomo, al fine di ottenere un esecuzione del percorso più fluida e precisa. Ad esempio, sarebbe necessario studiare il comportamento del sistema di guida autonomo nel momento in cui uno dei via point previsti nel piano non è raggiungibile a causa di ostacoli imprevisti. In questo caso sarebbe necessario studiare un metodo automatico che comandi la ripianificazione al fine di trovare un percorso alternativo.

8.6 Controllo della carrozzina tramite BCI

Un'ulteriore ipotesi di sviluppo della carrozzina prevede la possibilità di introdurre un modulo BCI (*Brain-Computer Interface*) per comandare il moto della carrozzina. Un modulo BCI rappresenta un'interfaccia diretta tra le attività celebrali e un computer. Le tecniche di monitoraggio delle attività celebrali sono numerose, tra cui la più semplice, meno invasiva ed economica è basata sull'uso di un elettroencefalografo e piccoli elettrodi posizionati sullo cuoio capelluto. Numerosi studi in questo ambito sono al vaglio della comunità scientifica [14] e permettono di identificare con diverse tecniche i comandi impartiti dall'utente. Innanzitutto è necessario distinguere tra tecniche basate sulla rilevazione di attività volontarie del cervello o su attività involontarie, chiamate anche *potenziali evocati*. Le attività volontarie del cervello possono essere sfruttate per realizzare un'interfaccia BCI in quanto esse sono modulabili dalla persona dopo un periodo di addestramento. Sarebbe dunque possibile comandare la carrozzina con un effetto proporzionale alla "volontà" dell'utente, ovvero comandando la carrozzina attraverso una sorta di "joystick immaginario". Utilizzando invece tecniche basate sul riconoscimento dei potenziali evocati sarebbe possibile, ad esempio, proporre all'utente un'interfaccia grafica su cui sono visualizzate quattro frecce direzionali che vengono illuminate in ordine casuale. Quando l'utente si concentra su una delle frecce, la sua attività celebrale involontaria viene stimolata dal lampeggio della freccia considerata. Tale attività può essere rilevata con l'uso dell'elettroencefalografo e portata come informazione di comando al computer che gestisce il moto della carrozzina. Con lo stesso principio di funzionamento sarebbe possibile proporre all'utente la scelta tra

varie destinazioni predefinite e, una volta selezionata una destinazione, attivare la funzionalità di guida autonoma della carrozzina per raggiungere la destinazione.

È facile immaginare che la frequenza con cui possono essere impartiti i comandi tramite l'uso di interfacce BCI è molto più bassa rispetto a un sistema di guida tradizionale. Per questo motivo assumerebbe notevole importanza il sistema di evitamento ostacoli, che permetterebbe di eseguire in sicurezza comandi di durata temporale medio-lunga.

Presso il Politecnico di Milano sono allo studio alcune tecniche di rilevazione dell'attività celebrale con lo scopo di progettare e realizzare un modulo BCI per la guida della carrozzina elettrica sviluppata in questo lavoro. Le funzionalità della carrozzina necessitano di essere ulteriormente sviluppate prima di essere pronte per un'integrazione con un sistema di comando di questo tipo, ma anche l'evoluzione del sistema BCI è in corso, quindi è ragionevole ipotizzare che i due progetti possano percorre strade di sviluppo parallele, per essere integrati quanto entrambi saranno considerati sufficientemente maturi e affidabili.

Bibliografia

[1] R. C. Arkin. Motor schema based navigation for a mobile robot. In *Proocedings of the IEEE International Conference on Robotics and Automation*, 1987.

[2] M. Billinghurst and H. Kato. Collaborative augmented reality. *Communications of the ACM*, 45:64–70, 2002.

[3] A. Bonarini, M. Matteucci, and M. Restelli. A novel model to rule behavior interaction. In *Proceedings of the 8th Conference on Intelligent Autonomous Systems (IAS-8)*, page 199 206, Amsterdam, 2004. IOS Press.

[4] Andrea Bonarini, Giovanni Invernizzi, Thomas Halva Labella, and Matteo Matteucci. an architecture to coordinate fuzzy behaviors to control an autonomous robot. *Fuzzy Sets and Systems*, 134, 2003.

[5] J. Borenstein, H. R. Everett, L. Feng, S. W. Lee, and R. H. Byrne. *Where am I? Sensors and Methods for Mobile Robot Positioning*. 1996.

[6] U. Borgolte, H. Hoyer, C. Bühler, H. Heck, and R. Hoelper. Architectural Concepts of a Semi-autonomous Wheelchair. *Journal of Intelligent and Robotic Systems*, 22:233–253, 1998.

[7] G. Bourhis, K. Moumen, P. Pino, S. Rohmer, and A. Pruski. Assisted navigation for a powered wheelchair. In *International conference on systems, man and cybernetics*, 1993.

[8] R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2:14–23, 1986.

[9] Simone Ceriani and Marco Dalli. Guida all'installazione dell'interfaccia per pc. Laboratorio di Intelligenza Artificiale e Robotica A.A. 2006-2007.

[10] Simone Ceriani and Marco Dalli. Robotizzazione carrozzina elettrica per disabili. Laboratorio di Intelligenza Artificiale e Robotica A.A. 2006-2007.

[11] J. Connell and P. Viola. Cooperative control of a semi-autonomous mobile robot. In *Proceedings of the IEEE International Conference on Robotics And Automation*, 1990.

[12] Guillermo Del Castilloa, Steven Skaara, Antonio Cardenasb, and Linda Fehr. A sonar approach to obstacle detection for a vision-based autonomous wheelchair. *Robotics and Autonomous Systems*, 54:967–981, 2006.

[13] D. Ding and R. A. Cooper. Electric powered wheelchairs. *IEEE Control Systems Magazine*, 25:22–34, 2005.

[14] Guido Dornhege. *Increasing Information Transfer Rates for Brain-Computer Interfacing.* phdthesis, University of Potsdam, 2006.

[15] Mark Fiala. Vision guided control of multiple robots. In *First Canadian Conference on Computer and Robot Vision*, 2004.

[16] Mark Fiala. Comparing ARTag and ARToolkit Plus Fiducial Marker Systems. In *IEEE International Workshop on Haptic Audio Visual Environments and their Applications*, pages 148–153, 2005.

[17] Christian Geiger, Joerg Stoecklein, Florian Klompmaker, and Robin Fritze. Development of an Augmented Reality Game by Extending a 3D Authoring System. In *Proceedings of the international conference on Advances in computer entertainment technology*, 2007.

[18] M. Habib. Can plannig and reactive systems realize an autonomus navigation. *International Symposium on Robotics (ISR99)*, 11(2), 1999.

[19] HaGi Sonic co., 535, Yongsan-dong, Yuseong-gu, Daejeon, Corea. *User's Guide Localization system StarGazer for Intelligent Robots*, 2007.

[20] L. Iezzoni, E. McCarthy, R. Davis, and H. Siebens. Mobility difficulties are not only a problem of old age. *Journal of General Internal Medicine*, 16(4):235–243, 2001.

[21] N.I. Katevas, N.M. Sgouros, S.G. Tzafestas, G. Papakonstantinou, P. Beattie, J.M. Bishop, P. Tsanakas, and D. Koutsouris. The au-

tonomous mobile robot SENARIO: a sensor aided intelligent navigation system for powered wheelchairs. *IEEE Robotics & Automation Magazine*, 4:60–70, 1997.

[22] H. Kato and M. Billinghurst. marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd International Workshop on Augmented Reality*, 1999.

[23] Y. Kuno, N. Shimada, and Y. Shirai. Look where you're going [robotic wheelchair]. *IEEE Robotics & Automation Magazine*, 10:26–34, 2003.

[24] A. Lankenau, O. Meyer, and B. Krieg-Bruckner. Safety in robotics: the Bremen Autonomous Wheelchair. *Advanced Motion Control*, pages 524–529, 1998.

[25] A. Lankenau and T. Röfer. A versatile and safe mobility assistant. *IEEE Robotics & Automation Magazine*, 8:29–37, 2001.

[26] A. Lankenau, T. Röfer, and B. Krieg-Brückner. Self-Localization in Large-Scale Environments for the Bremen Autonomous Wheelchair. In Habel C. Wender K.F. Freksa C., Brauer W., editor, *Spatial Cognition III: Routes and Navigation, Human Memory and Learning, Spatial Representation and Spatial Learning*, volume 2685 of *Lecture Notes in Computer Science*, pages 34–61. Springer, 2003.

[27] Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[28] S.P. Levine, D.A. Bell, L.A. Jaros, R.C. Simpson, Y. Koren, and J. Borenstein. The NavChair Assistive Wheelchair Navigation System. *IEEE Transactions on Rehabilitation Engineering*, 7:443–451, 1999.

[29] R. Madarasz, L. Heiny, R. Cromp, and N. Mazur. The Design of an Autonomous Vehicle for the Disabled. *IEEE Journal of Robotics and Automation*, RA-2:117–126, 1986.

[30] P. Maes. Situated agents can have goals. In *Designing autonomous agents*, 1990.

[31] C. Mandel, K. Huebner, and T. Vierhuff. Towards an autonomous wheelchair: Cognitive aspects in service robotics. In *Proceedings of Towards Autonomous Robotic Systems*, 2005.

[32] M. Mazo. An integral system for assisted mobility. *IEEE Robotics & Automation Magazine*, 8:46–56, 2001.

[33] Paolo Meriggi. *Processing and Communication Systems for Device Communities*. PhD thesis, Università degli studi di Brescia, 2005.

[34] D. P. Miller and M. G. Slack. Design and testing of a low-cost robotic wheelchair prototype. *Autonomous Robots*, 2:77–88, 1995.

[35] Salvatore Nicosia and Francesco Martinelli. Dispense del corso di robotica industriale (pianificazione del moto dei robot). Technical report, Università degli Studio di Roma Tor Vergata, 2000.

[36] E. Prassler, J. Scholz, and P. Fiorini. A robotics wheelchair for crowded public environment. *IEEE Robotics & Automation Magazine*, 8:38–45, 2001.

[37] A. Pruski and G. Bourhis. The VAHM project: a cooperation between an autonomous mobileplatform and a disabled person. In *IEEE International Conference on Robotics and Automation*, 1992.

[38] Sven Rönnbäck. *On Methods for Assistive Mobile Robots*. PhD thesis, Luleå University of Technology, 2006.

[39] Shigeru Saito, Atsushi Hiyama, Tomohiro Tanikawa, and Michitaka Hirose. Indoor marker-based localization using coded seamless pattern for interior decoration. In *IEEE Virtual Reality Conference*, pages 67–74, 2007.

[40] H. Seki, S. Kobayashi, Y. Kamiya, M. Hikizu, and H. Nomura. Autonomous/semi-autonomous navigation system of a wheelchair byactive ultrasonic beacons. In *IEEE International Conference on Robotics and Automation, 2000. Proceedings. ICRA '00.*, 2000.

[41] SICK AG, Erwin-Sick-Str. 1 D-79183 Waldkirch, Germania. *SICK Automatic Identification Catalog*, 2006.

[42] R. Simpson. Smart wheelchairs: A literature review. *Journal of Rehabilitation Research & Development*, 42(4):423–438, 2005.

[43] R. Simpson, E. Lopresti, S. Hayashi, I. Nourbakhsh, and D. Miller. The smart wheelchair component system. *Journal Rehabil Res Dev*, 41:429–42, 2004.

[44] R. Simpson, D. Poirot, and F. Baxter. The Hephaestus Smart Wheelchair system. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 10:118–122, 2002.

[45] Fabio Sora. FollowMe: composizione "informata" di reattività e pianificazione in un robot guida. Master's thesis, Politecnico di Milano, 2005.

[46] D. Tefft, P. Guerette, and J. Furumasu. Cognitive predictors of young childrens readiness for powered mobility. *Developmental Medicine & Child Neurology*, 41(10):665–670, 1999.

[47] S. Thrun, M. Beetz, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Hänel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. Probabilistic Algorithms and the Interactive Museum Tour-Guide Robot Minerva. *The International Journal of Robotics Research*, 19:972–999, 2000.

[48] Ubisense Limited, St Andrew's Road, Chesterton,Cambridge, Regno Unito. *Ubisense Brochure*.

[49] Ubisense Limited, St Andrew's Road, Chesterton,Cambridge, Regno Unito. *Ubisense Datasheet*, 2006.

[50] Ubisense Limited, St Andrew's Road, Chesterton,Cambridge, Regno Unito. *Ubisense System Overview*, 2007.

[51] Daniel Wagner and Dieter Schmalstieg. ARToolKitPlus for Pose Tracking on Mobile Devices. In *Computer Vision Winter Workshop*, 2007.

[52] Eric Woods, Paul Mason, and Mark Billinghurst. MagicMouse: an inexpensive 6-degree-of-freedom mouse. In *Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, 2003.

[53] XSens Technologies, Enschede, The Netherlands. *MTi Miniature Attitude and Heading Reference System*.

[54] H. Yanco. Integrating Robotic Research: A Survey of Robotic Wheelchair Development, 1998.

[55] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.