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

Sviluppo e analisi sperimentale di una interfaccia innovativa per il controllo di sistemi multirobot per applicazioni di ricerca e soccorso urbani

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

Relatore: Prof. Francesco Amigoni

Tesi di Laurea di:
Alain Caltieri, matricola 765689

Anno Accademico 2011-2012

Sommario

La ricerca scientifica dedica grandi sforzi agli aspetti avanzati della robotica autonoma. Ciononostante, l'utilizzo di tecnologie robotiche in ambiti critici, quali la ricerca e il soccorso urbani, è limitata a robot esclusivamente tele-operati, nonostante i vantaggi che sistemi multirobot autonomi o semi-autonomi possono potenzialmente offrire. Una delle cause di questa tendenza è la mancanza di fiducia degli umani responsabili delle missioni, che richiedono un maggior grado di governo sui sistemi autonomi e devono dunque essere integrati al meglio nel ciclo di controllo.
Lo scopo di questa tesi è la realizzazione di una interfaccia che migliori l'integrazione della componente umana nel ciclo di controllo e permetta una naturale ed efficace gestione del sistema multirobot.
Per raggiungere tale obiettivo abbiamo implementato una GUI allo stato dell'arte e l'abbiamo migliorata con un sistema di gestione delle notifiche dei robot e la possibilità di utilizzare comandi di alto livello che semplifichino le dinamiche d'interazione fra operatore umano e sistema robotico. I risultati dei test in laboratorio e la vittoria nella competizione RoboCup Rescue Simulation League 2012 dimostrano come queste migliorie portino dei vantaggi nella gestione del sistema e permettano l'utilizzo di gruppi numerosi di robot.

Ringraziamenti

Ringrazio il professor F. Amigoni per avermi seguito con pazienza e perizia durante questo lavoro.
Ringrazio i ragazzi del team PoAReT per il loro eccellente lavoro e soprattutto per l'indimenticabile esperienza, il cameratismo e l'amicizia che abbiamo condiviso.
Ringrazio tutti coloro che mi sono stati vicini durante la mia crescita. Non menziono nessuno in particolare solo perché non ho abbastanza spazio né abbastanza talento per mettere per iscritto un ringraziamento adeguato a quello che avete fatto per me. Grazie.

Indice

Capitolo 1

Introduzione

Notevoli sono i progressi compiuti nell'area della robotica autonoma, e molti saranno quelli a venire. Lo scenario auspicabile nel lungo termine consiste di sistemi robotici avanzati in grado di eseguire in piena autonomia compiti complessi che per gli esseri umani sono particolarmente faticosi o rischiosi. In quest'ottica si colloca la ricerca scientifica sui sistemi robotici mobili per il soccorso urbano, che dovrebbero salvare la vita a numerosi soccorritori umani e saper gestire le diverse e complesse situazioni che si affrontano in quest'area. L'ottimismo dell'ambiente scientifico convoglia dunque i maggiori sforzi alla ricerca di soluzioni per sistemi robotici pienamente autonomi. Ne sono un esempio le numerosissime manifestazioni e competizioni orientate alla robotica autonoma per applicazioni critiche. Ma qual è, oggi, l'utilizzo di questi sistemi nella pratica?
Sistemi robotici vengono già impiegati in diverse aree critiche, fra cui la ricerca e soccorso urbani (*Urban Search And Rescue, USAR*), ma nessun sistema autonomo viene regolarmente utilizzato sul campo durante missioni critiche. Ciò non è dovuto solo alle prestazioni dei sistemi autonomi, che anzi sono spesso sorprendentemente buone, ma è piuttosto collegato alla fiducia che i responsabili delle missioni critiche hanno verso tali sistemi. Il non dover intervenire nel sistema esclude dalla missione i soccorritori umani, che non possono né contribuire pienamente con la loro esperienza né controllare il sistema.
Se a ciò aggiungiamo i diversi risultati sperimentali che dimostrano che le prestazioni di un sistema in cui la componente umana viene integrata nel ciclo di controllo sono superiori sia a quelle dei sistemi puramente autonomi sia a quelle dei sistemi puramente manuali, possiamo affermare che, per contribuire nel breve termine a missioni di soccorso urbano con sistemi robotici, bisogna necessariamente compiere dei passi avanti nelle tecniche d'integra-

zione della componente umana all'interno del ciclo si controllo.
Lo scopo di questa tesi è duplice:

- Realizzare una interfaccia grafica utente che includa gli elementi allo stato dell'arte nel controllo di sistemi multirobot per la ricerca e il soccorso urbano;
- Introdurre innovazioni rilevanti per permettere l'efficace integrazione della componente umana nel sistema di controllo.

Dopo una prima fase di analisi dello stato dell'arte per ricavare i principi sui quali costruire l'interfaccia grafica utente, abbiamo esaminato i flussi informativi tra l'operatore e il sistema robotico: una gestione ottimale di tali flussi permette all'operatore di interfacciarsi col sistema in maniera naturale ed efficace e dare così il massimo contributo nelle missioni.
Abbiamo progettato due innovazioni principali:

Comandi di alto livello che permettono all'operatore di intervenire sul comportamento del sistema multirobot in maniera naturale, interagendovi come farebbe con una squadra di soccorritori umani.

Sistema di filtraggio delle notifiche che si occupano di aggregare, filtrare e ordinare le notifiche provenienti dai robot a seconda del loro contenuto, dello stato della missione, delle condizioni di lavoro dell'operatore e dell'affidabilità dell'agente che invia la notifica.

In questo lavoro di tesi abbiamo implementato per intero l'interfaccia grafica utente, alla quale abbiamo aggiunto le innovazioni progettate. Il sistema così realizzato è stato analizzato sperimentalmente per valutare gli effetti di tali innovazioni. L'interfaccia creata controlla un sistema multirobot chiamato *PoAReT* (*Politecnico di Milano Autonomous Robotic Rescue Team*). Il sistema PoAReT, con l'interfaccia e le innovazioni sviluppate in questa tesi, ha partecipato e vinto la competizione internazionale RoboCup Rescue Simulation League 2012 tenutasi a Città del Messico. Inoltre i test sperimentali hanno dimostrato come le innovazioni introdotte favoriscano l'interazione dell'utente col sistema, garantendo da una parte un carico di lavoro minore e dall'altra una miglior consapevolezza della situazione. Una più estesa attività di validazione del sistema, unita a prove sul campo sono necessari in futuro per valutare a pieno la bontà della soluzione adottata ed individuare ulteriori miglioramenti.

Lo USAR è una delle applicazioni in cui l'utilizzo di sistemi multirobot presenta le potenzialità maggiori, visti i rischi che corrono i soccorritori umani durante tali missioni. Tuttavia, le poche applicazioni sul campo di sistemi robotici hanno riguardato l'utilizzo di singoli, complessi e pesanti robot tele-operati e rappresentano il più delle volte dei fallimenti [1]. L'utilizzo

di sistemi robotici autonomi necessita di trovare maggior fiducia presso gli operatori umani. A tale scopo, le funzioni autonome dei robot non devono escludere la componente umana dal controllo della situazione. Tali funzionalità autonome devono, invece, essere a supporto dell'attività dell'operatore, che deve poterne disporre secondo quelle che ritiene siano le scelte più idonee per il successo della missione [2].
In [3] gli autori dimostrano come sistemi per lo USAR pienamente autonomi, in cui l'essere umano non deve e non può intervenire, hanno generalmente prestazioni inferiori rispetto a sistemi mixed initiative, in cui il livello di autonomia viene regolato in collaborazione dal sistema e dall'operatore umano. Gli stessi risultati si riscontrano in molti altri test sperimentali e non sono confinati all'ambito USAR, come sottolineato in [4].
Nell'ambito della *Human-Robot Interaction* (*HRI*) sono stati riconosciuti diversi ruoli all'essere umano incaricato di controllare il sistema. La ricerca accademica pone particolare enfasi sull'autonomia del sistema robotico e sul cambiamento del ruolo della componente umana da operatore in grado di intervenire e collaborare col sistema a supervisore della corretta esecuzione dei compiti [5]. Data la diversa natura dei due ruoli, ognuna delle due opzioni richiede un'apposita interfaccia di controllo, progettata specificamente per i compiti riservati all'essere umano [6].
Date queste premesse, l'obiettivo di questa tesi è la realizzazione di una interfaccia in grado di includere l'operatore nel ciclo di controllo, in modo da ottenere prestazioni adeguate alla criticità della missione e permettere all'operatore un pieno controllo del sistema. Intendiamo, tramite questo lavoro, migliorare la fiducia dell'operatore sul sistema ed aumentarne in maniera rilevante sia il controllo effettivo sia quello percepito, prerequisiti importanti per una futura effettiva applicazione dei sistemi robotici autonomi nel campo delle missioni critiche.
Questo lavoro di tesi inizia da una fase di analisi della letteratura disponibile sui principi di realizzazione delle interfacce utente per sistemi multirobot, in particolare per lo USAR, e di studio dei sistemi implementati e innovativi presentati negli ultimi anni col nostro stesso obiettivo.
Abbiamo individuato così le caratteristiche fondamentali che l'interfaccia d'interazione di un sistema multirobot per lo USAR deve avere:

- Interfaccia mappa-centrica, ossia in cui la mappa globale dell'ambiente è il principale oggetto d'interazione e controllo;
- Informazioni aggregate relative ai singoli robot, che permettano un'intelligente utilizzo dello spazio su schermo;
- Flussi video di qualità regolabile, per limitare l'occupazione di banda in comunicazione e permettere il controllo efficace dei robot quando

necessario;

- Robustezza, sia ai guasti nella rete che nelle macchine del sistema, con la facilità di recuperare il controllo una volta risolti;
- Livello di controllo sui robot regolabile durante le missioni.

Questi principi si sono affermati durante anni di ricerca e si sono dimostrati efficaci nella realizzazione delle interfacce d'interazione e relative interfacce grafiche utente. A riprova di ciò tutti i sistemi analizzati sono basati sugli stessi principi riassunti sopra [2] [5] [7] [8].
Partendo da questi principi abbiamo realizzato una interfaccia grafica utente che li rispettasse tutti e fosse dunque all'avanguardia. Una ulteriore analisi della letteratura e l'esperienza acquisita durante questa prima fase hanno permesso di identificare i due principali problemi che la ricerca scientifica sta affrontando: il carico di lavoro cui è sottoposto l'operatore durante le missioni (*workload*) e la consapevolezza della situazione che riesce a raggiungere per compiere scelte idonee (*situation awareness*).
Analizzando le problematiche che affliggono le interfacce di controllo abbiamo notato come queste dipendano fortemente dai flussi informativi che intercorrono fra sistema ed operatore, composti dalla gestione delle notifiche dei robot (informazioni da sistema verso operatore) e dalla generazione e comunicazione dei comandi (informazioni da operatore a sistema). Riducendo l'intensità di questi flussi informativi e contemporaneamente aumentando la qualità delle singole informazioni trasmesse possiamo da un lato ridurre il workload e dall'altro aumentare la situation awareness.
Abbiamo introdotto nell'interfaccia utente due rilevanti innovazioni che permettessero di migliorare i flussi informativi.
Miglioriamo le informazioni trasmesse dall'operatore al sistema tramite i comandi di alto livello. Questi sono comandi espressi dall'operatore in maniera naturale e dall'ovvio significato, quali l'esplorare una certa area o lungo una data direzione. Il sistema interpreta tali comandi e li traduce in direttive relative alla politica di esplorazione autonoma dei singoli robot. Questi ultimi si coordinano tra loro e decidono le proprie destinazioni realizzando le azioni di esplorazione vere e proprie e tenendo in considerazione le indicazioni dell'operatore.
Il flusso di informazioni da parte dei robot e dirette all'operatore viene migliorato tramite un sistema di filtraggio delle notifiche. Il sistema da noi implementato permette l'aggregazione delle notifiche da parte dello stesso robot e relative a problematiche simili nonché la classificazione delle notifiche (ed eventualmente il filtraggio delle stesse) sulla base della loro priorità stimata. La stima della priorità tiene conto di fattori quali la missione attuale, l'affidabilità dei moduli del singolo robot (stimata a partire dalle dinamiche

d'interazione dell'operatore), il workload attuale e le preferenze specificate dall'operatore. Questo approccio rende il processo di classificazione e filtraggio delle notifiche dinamico: esso viene adattato automaticamente dal sistema a seconda dello stato attuale e viene anche aggiustato dall'operatore a seconda delle sue preferenze.
Una volta implementate le due innovazioni descritte abbiamo svolto una fase di valutazione del sistema. Per avere un'indicazione quanto più affidabile della bontà della soluzione proposta abbiamo adottato due impegnativi banchi di prova dalle caratteristiche complementari.
Innanzitutto abbiamo eseguito una serie di test sperimentali su 14 partecipanti, ognuno sottoposto a 9 prove. Si è in questo modo testato l'impatto delle innovazioni sulle prestazioni del sistema, valutate sulla base di misure comuni in letteratura [2] [9]. Le diverse prove sono state eseguite con un numero crescente di robot (2, 5 e 8 rispettivamente) e con diverse configurazioni del sistema per valutare l'impatto delle nuove funzionalità anche al crescere delle dimensioni del gruppo di robot da controllare. I risultati ottenuti confermano che i comandi di alto livello sono preferiti nella quasi totalità dei casi alle politiche di esplorazione autonoma libera e garantiscono un maggior livello di controllo e fiducia sul sistema. Tale effetto positivo si affianca alla drastica riduzione dello *idle time*, ossia il tempo in cui i robot non eseguono azioni utili. Inoltre si è verificato che il sistema di filtraggio delle notifiche garantisce una netta riduzione del workload e una migliore gestione delle eccezioni. Dei 14 partecipanti ai test effettuati, 7 sono operatori esperti e 7 inesperti. L'analisi comparata dei risultati ci ha permesso di notare come le prestazioni dei due gruppi migliorino in maniera simile grazie alle innovazioni introdotte, che dunque sono efficaci indipendentemente dall'esperienza dell'operatore.
Per garantire l'obiettività dei risultati e soprattutto per permettere un confronto diretto con gruppi di ricerca provenienti da tutto il mondo che lavorano sulle stesse problematiche, abbiamo messo alla prova il sistema partecipando alla competizione internazionale RoboCup Rescue Simulation League. Grazie anche alle innovazioni introdotte in questa tesi ed alla robustezza dell'interfaccia di controllo da noi implementata la squadra PoAReT, alla sua prima partecipazione, ha vinto la competizione battendo squadre dalla ben più lunga esperienza.
I risultati ottenuti hanno mostrato i miglioramenti nella gestione del sistema multirobot introdotti dalle innovazioni proposte. Tuttavia prove più approfondite, con un maggior numero di tester, con l'utilizzo di misure più raffinate ed eventualmente con test con robot reali potranno eventualmente confermare questi risultati ed evidenziare con più certezza pregi e difetti

della nostra soluzione, identificando ulteriori innovazioni utili. Inoltre varie migliorie tecniche sono inseribili nell'interfaccia grafica realizzata, e potrebbero aumentarne l'efficacia. A titolo d'esempio si potrebbe utilizzare una mappa 3D dell'ambiente, eventualmente arricchita con i flussi video, migliorie proposte rispettivamente in [10] e [11], assieme alle nostre innovazioni per verificarne la compatibilità e le prestazioni complessive.
Il nostro lavoro non pretende di risolvere completamente le problematiche del workload e della situation awareness, ma aspira ad essere un piccolo passo verso soluzioni che integrino la componente umana in un sistema multirobot nella maniera più efficace possibile e permettano sia di raggiungere prestazioni adeguate alla criticità del compito sia di ottenere la massima fiducia degli utilizzatori del sistema.

Strutturiamo questa tesi come segue.
Nel Capitolo 2 si analizza lo stato dell'arte della HRI per sistemi multirobot in ambito USAR, espandendo ulteriormente le motivazioni che spingono verso una migliore integrazione della componente umana all'interno del ciclo di controllo. Analizziamo l'attuale situazione delle missioni USAR in cui i soccorritori mettono a rischio le loro vite ed i vantaggi che si potrebbero ottenere dall'utilizzo di sistemi multirobot. Analizziamo le difficoltà che si incontrano nella ricerca di nuove soluzioni e come l'utilizzo di simulatori permetta di ridurre tali problematiche. Analizziamo anche le più famose competizioni il cui scopo è dare impulso alla ricerca sui sistemi robotici per lo USAR e, dopo aver studiato nel dettaglio il ruolo dell'essere umano in questi sistemi, descriviamo alcuni dei più innovativi sistemi attualmente implementati con i quali confrontare il nostro lavoro.
Nel Capitolo 3 si imposta il problema di ricerca, giungendo alla definizione dei nostri obiettivi e all'identificazione delle principali problematiche che affrontiamo. Si procede con l'analisi dei flussi di informazioni fra operatore e sistema multirobot e si definiscono le innovazioni che andiamo a introdurre, giustificandone l'ideazione. Inoltre si identificano le misure con le quali valutare le prestazioni del sistema e i principi fondamentali per realizzare una interfaccia grafica utente che sia allo stato dell'arte.
Nel Capitolo 4 si mostra l'architettura del sistema implementato. Si descrivono le funzionalità dei robot controllati e soprattutto ci si focalizza sul risultato di questo lavoro di tesi: l'interfaccia grafica utente e la stazione base di controllo del sistema. Si mostrano i principali moduli e la loro organizzazione nonché il sistema effettivamente implementato così come utilizzato nelle verifiche sperimentali. Infine, si presentano le dinamiche d'interazione col sistema per i principali compiti che deve svolgere l'operatore.

Nel Capitolo 5 si riportano e discutono i risultati delle prove eseguite, sia i test in laboratorio, sia i risultati ottenuti nella competizione. Si propone anche un confronto con alcuni sistemi concorrenti.
Infine, nel Capitolo 6 traiamo le conclusioni sul lavoro svolto e i risultati ottenuti e analizziamo alcuni possibili sviluppi futuri.

Capitolo 2

Stato dell'arte

In questo capitolo descriviamo lo stato dell'arte nell'ambito delle procedure di soccorso urbano, con particolare enfasi sul ruolo attuale e le aspettative future della robotica autonoma. Analizziamo quindi le arene usate per il test di sistemi robotici per il soccorso urbano ed il ruolo che l'umano assume in tali sistemi. Concludiamo il capitolo con la descrizione di alcuni sistemi considerati di riferimento dello stato dell'arte dal punto di vista delle dinamiche di interazione della componente umana col sistema semi-automatico.

2.1 Urban Search And Rescue

Lo *USAR* (*Urban Search And Rescue*, ricerca e soccorso urbani) consiste di tutte le procedure volte al soccorso di vittime umane in ambienti pericolosi, sia interni che esterni ad edifici.

2.1.1 Sistemi attualmente usati

Viste le caratteristiche del problema, con rischi elevati, necessità di gestire numerose eccezioni, elaborare numerose informazioni e soprattutto la necessità di avere buona mobilità all'interno di ambienti incontrollabili, impervi e, spesso, in veloce evoluzione, oggi le squadre di soccorso nello USAR sono quasi esclusivamente composte da umani e canidi. Se da una parte tale composizione permette di ottenere prestazioni ottime in termini di reattività e mobilità in ambienti difficili, dall'altra comporta un ingente dispendio di vite: secondo l'*USFA* (*United States Fire Administration*) il numero di vittime umane è di circa 100 individui ogni anno, con decine di migliaia di feriti nei soli Stati Uniti d'America [12] [13], nazione che si caratterizza per l'eccellenza nell'addestramento e nell'attrezzatura dei suoi vigili del fuoco.

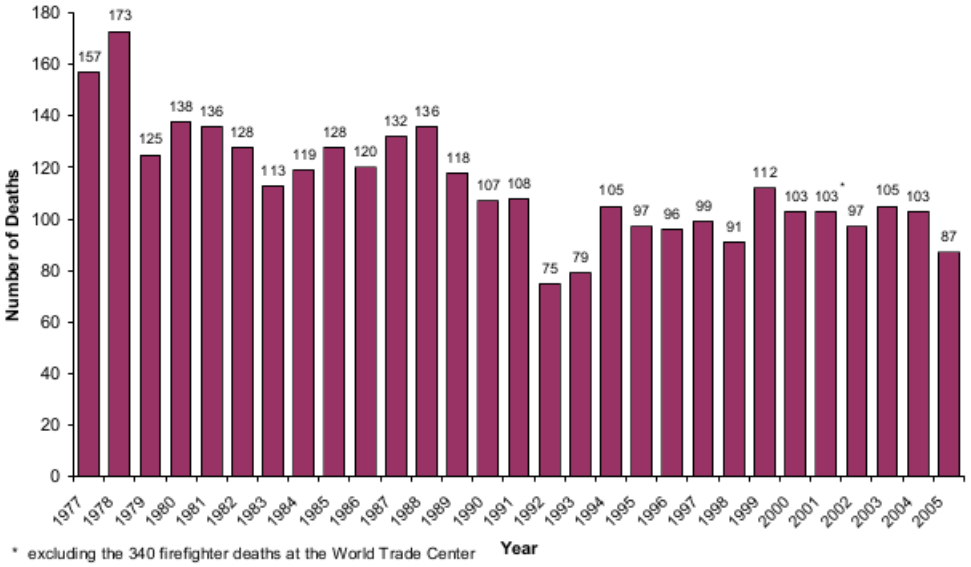

Figura 2.1: Decessi in azione nel corpo dei vigili del fuoco degli USA. Fonte: [12].

In Figura 2.1 sono riportati i decessi durante operazioni di soccorso dei vigili del fuoco americani dal 1977 al 2005, a titolo di esempio.

L'uso di robot nello USAR

Date le premesse risulta chiaro come sia importante migliorare le tecniche di soccorso urbano sia per la riduzione delle perdite tra i soccorritori sia per una maggiore efficienza nel soccorso stesso con un maggior numero di vittime soccorse. Una delle prospettive più interessanti vede l'utilizzo di piccoli robot per la ricerca di vittime in ambienti particolarmente pericolosi. Lo scenario più comune, tuttavia, prevede l'utilizzo di robot complessi, costosi e pesanti. Questa soluzione è stata già applicata in diverse occasioni: nel 2001 al World Trade Center, nel 2005 per gli uragani Katrina, Rita and Wilma e diversi altri [1] ed è ancora oggetto di sperimentazione in numerosi dipartimenti di vigili del fuoco mondiali [14].

Limiti dei sistemi robotici per lo USAR

I principali problemi che limitano l'uso di sistemi robotici per azioni di soccorso sono in gran parte legati alle caratteristiche degli ambienti, molto ostici, che ostacolano la mobilità dei robot riducendone ampiamente l'applicabilità. In [1] si citano diversi casi di fallimento di applicazione di sistemi robotici avanzati nello USAR, giustificati dal fatto che si trattava di sistemi

pesanti, ingombranti e costosi. Un nuovo paradigma si rende dunque necessario, basato su squadre di robot piccoli e mobili, poco costosi e sacrificabili, che permettano una esplorazione delle aree a maggior rischio evitando di mettere in pericolo la vita dei soccorritori umani. Tale approccio sposta anche parte delle problematiche da affrontare dall'ambito della robotica a quello della *HRI (Human Robot Interaction)*: le squadre di robot dovranno essere gestibili da uno o comunque pochi operatori umani per permettere un'azione di soccorso efficiente. Diventa necessario passare dall'idea di robot completamente tele-operati a sistemi multirobot semiautonomi gestiti da un solo operatore, messo in condizione di interagire col sistema in maniera efficace.

USARSim per la progettazione di nuovi sistemi

Le difficoltà nella mobilità dei robot, unito al loro costo di realizzazione, rischia di limitare la ricerca scientifica su aspetti che astraggono da tali problematiche, come la realizzazione di interfacce grafiche multirobot, lo studio di procedure di HRI, la realizzazione di politiche di esplorazione autonoma, ecc. Per tale motivo sono stati realizzati numerosi simulatori che permettono di astrarre dalle problematiche di realizzazione dei robot fisici. Il simulatore da noi utilizzato, uno dei più completi ed affidabili [15], è USARSim, basato sul motore grafico e fisico Unreal Engine 3. Garantisce un buon realismo, una buona varietà di robot implementati e tempi di setup molto brevi.

2.2 Arene di test

Come per molte altre problematiche scientifiche, anche nell'ambito dello USAR sono state stabilite delle arene per misurare le prestazioni delle diverse soluzioni proposte. Introduciamo alcune di queste arene di test, spesso incluse in competizioni internazionali di rilievo, che ovviamente non sostituiscono i test in laboratorio ma che riescono a dare spinta alla ricerca e favoriscono enormemente gli scambi di conoscenze fra diverse realtà accademiche.

2.2.1 RoboCup Rescue Robot League

Inclusa nella più ampia competizione RoboCup [16], la RoboCup Rescue Robot League si occupa di problematiche di USAR. Durante la competizione viene realizzata un'arena di prova dove far competere robot realizzati appositamente per una mobilità ottimale nella ricerca di vittime o oggetti [17]. La competizione si concentra maggiormente su problematiche di mobilità, nonché di cattura ed interpretazione delle informazioni dei sensori, con un

approccio tradizionale al controllo del sistema, basato sulla tele-operazione di ogni singolo robot.

2.2.2 RoboCup Rescue Simulation League

Versione simulata della RoboCup Rescue, con focus molto maggiore su sistemi multirobot comandati da un solo operatore. Particolare enfasi sulle problematiche legate all'esplorazione autonoma dei robot e sulle dinamiche di controllo [18]. Il sistema trattato in questa tesi è stato presentato all'edizione 2012 della RoboCup Rescue Simulation League tenutasi in Città del Messico, vincendo il primo premio con il team *PoAReT*(*Politecnico di Milano Autonomous Robotic Rescue Team*).

2.2.3 MAGIC - Multi Autonomous Ground-robotic International Challenge

La competizione MAGIC mira alla realizzazione di sistemi multirobot quanto più possibile autonomi [19]. Forte enfasi viene posta sia sull'HRI sia sulle politiche di esplorazione e mappatura. Durante la competizione si affrontano situazioni sia di USAR sia di identificazione di minacce in area urbana, quali ad esempio attentatori o ordigni esplosivi. L'uso di sistemi robotici in tali ambiti è molto simile a quello nelle applicazioni di USAR, soprattutto nella fase di ricerca delle minacce, esplorazione dell'area e mappatura della stessa.

2.2.4 ELROB - European Land-Robot Trial

Si tratta di uno dei principali eventi dimostrativi dei risultati della robotica in Europa [20]. Lo scopo è illustrare i risultati ottenuti dalla ricerca e usarli per risolvere le problematiche reali in un periodo relativamente breve. Non ha un'area principale di applicazione, ma fra le tante problematiche affrontate lo USAR ha un ruolo di rilievo.

2.2.5 Controversie

L'approccio utilizzato dalle competizioni di robotica, sia quelle qui brevemente descritte sia le numerose altre non citate, permette di mettere in contatto istituti di ricerca diversi, che altrimenti non terrebbero conto dei reciproci risultati. È ovvio quali benefici si possa trarre da tale approccio. D'altro canto, proprio la natura competitiva di questi eventi, ha portato alla ricerca sempre più spinta del risultato, a scapito della qualità dei lavori scientifici qualora la competizione non sia in grado di orientare gli sforzi

dei partecipanti nella giusta direzione [21]. Per questo motivo i risultati in una competizione non sono da considerarsi sufficienti ed è sempre necessario verificare i risultati ottenuti con altri test sperimentali ripetibili ed affidabili.

2.3 Il ruolo dell'umano

Uno degli approcci di maggior interesse nell'applicazione della robotica alle problematiche dello USAR è, come già anticipato, quello che prevede l'utilizzo di squadre di piccoli robot comandati da un solo operatore. Le principali difficlotà sono relative all'integrazione della componente umana nel ciclo di controllo del sistema.
Analizziamo prima la discrepanza fra la ricerca di una maggiore autonomia in sistemi robotici, in particolare per lo USAR, e gli attuali utilizzi della tecnologia sul campo, con robot perlopiù tele-operati. Quindi facciamo una breve analisi delle capacità dei sistemi autonomi e dell'operatore umano, mostrandone la naturale complementarietà. Presentiamo poi i ruoli che tradizionalmente sono coperti da operatori umani in sistemi multirobot, sottolineandone pregi e difetti. Infine chiudiamo la sezione focalizzandoci sull'approccio da noi ritenuto più congeniale, col supporto di diversi risultati empirici ottenuti sia in ambito USAR che non. In questa sezione astraiamo in gran parte dalle problematiche tipiche della HCI (Human-Computer Interaction) per quanto riguarda la realizzazione delle interfacce d'interazione dell'umano col sistema, ma ci focalizziamo solo sui compiti da assolvere. Tali problematiche saranno affrontate nel Capitolo 3 e, più nel dettaglio, nel Capitolo 4.

2.3.1 Uomo e macchina

Numerosissimi eventi legati alla robotica e all'intelligenza artificiale hanno come obiettivo di lungo termine la realizzazione di sistemi completamente autonomi, che non abbiano in alcun modo bisogno dell'intervento umano. Ottimo esempio sono le competizioni della DARPA (Defense Advanced Research Projects Agency), promosse con un esplicito obiettivo: avere veicoli militari terrestri completamente autonomi entro il 2015 [22] [23]. A valle di tali propositi, oggi numerosi robot sono utilizzati dalle forze armate americane, ma la quasi totalità di essi è tele-operata [24]. Secondo [2] uno dei principali motivi per cui sistemi completamente autonomi, anche con ottime prestazioni nei test, non vengono poi effettivamente utilizzati sul campo è l'esclusione dell'umano dal ciclo di controllo. Numerosi altri autori hanno sottolineato questo aspetto, sia argomentandolo da un punto di vista tecni-

co [2] [5], sia da uno etico (citiamo in particolare [23] e [25]). Da un punto di vista tecnico, facendo una breve analisi di alto livello, è chiaro come umani e robot siano molto diversi, idonei a specifici compiti complementari. Da una parte la componente robotica è in grado di svolgere egregiamente compiti ben definiti che comportano un carico di lavoro cognitivo elevato, quali coordinamento di base, algoritmi di esplorazione ottimi o localizzazione e mappatura. D'altro canto la stessa componente robotica ha scarse prestazioni (in genere) nella gestione delle eccezioni, nella mobilità ed attuazione, nell'elaborazione delle informazioni a priori e nella creazione ed esecuzione di una strategia creativa di alto livello. In queste aree non esistono sistemi autonomi in grado di competere con un umano sufficientemente addestrato. È chiaro come l'argomentazione non si applichi soltanto all'ambito militare, tratto ad esempio per l'enorme mole di dati e controversie che si hanno a disposizione, ma è valida in generale, e lo USAR non fa eccezione. Come abbiamo già enfatizzato si tratta di un compito critico, in ambiente ostile e dinamico dove sia la gestione delle eccezioni e delle informazioni a priori, sia le scelte strategiche guidate dall'esperienza, da segnali difficilmente elaborabili automaticamente o anche dall'intuito, assumono una grande importanza. La complementarietà di umano e sistema robotico in quest'area è ragionevolmente la soluzione ottima nel breve e medio termine ed è stata esplorata approfonditamente in diversi studi, raggiungendo una base teorica che classifica i possibili ruoli associati all'umano nella gestione di sistemi multirobot.

2.3.2 Ruolo della componente umana

Diverse tassonomie tipiche dell'HRI sono state definite in [26]. In Figura 2.2 è presentato uno schema riassuntivo delle tassonomie proposte in letteratura, tratto da [26]. Le tassonomie contemplate sono:

A. Interazione uno a uno, tipicamente utilizzata per la tele-operazione di un singolo robot da parte di un singolo operatore umano;
B. Un solo operatore umano interagisce con un sistema multi-robot. L'interazione non è col singolo robot ma con il sistema multirobot nel suo complesso, che si occupa del coordinamento dei robot e fornisce i comandi effettivi ai singoli robot. I robot devono comunicare fra loro per potersi coordinare e scambiare i dati rilevanti;
C. L'operatore controlla direttamente i singoli robot, che non hanno interazioni fra loro;
D. Un insieme coordinato di operatori controlla un singolo robot. Gli operatori fanno parte di un sistema che coniuga il lavoro dei due ope-

ratori e funziona da interfaccia d'interazione con il robot, a cui viene mandato un singolo comando per volta;

E. Più umani mandano indipendentemente comandi ad un singolo robot, che si occupa di risolvere i conflitti, ordinarli per priorità ed eseguirli;

F. Un insieme di operatori si coordina per comandare un sistema multirobot coordinato. Due sistemi, uno per la coordinazione e comunicazione dei robot ed uno per la coordinazione e comunicazione fra gli operatori, sono necessari;

G. Un insieme coordinato di operatori controlla un gruppo di robot fra loro non interagenti e non coordinati. Un possibile esempio consiste in un gruppo di soccorritori, ognuno incaricato di tele-operare un robot, che si coordinano per la ricerca di feriti in uno scenario di USAR;

H. Un gruppo di umani interagisce indipendentemente con un sistema multirobot in cui i robot cooperano e si coordinano per risolvere i conflitti, definire le priorità dei compiti ed infine eseguirli in maniera autonoma ed efficace.

Nel sistema presentato in questa tesi teniamo come riferimento una configurazione intermedia fra quella in figura segnata come B e quella C, con l'operatore umano che può comunicare coi singoli robot che collaborano a loro volta tra loro per eseguire in maniera ottima i compiti loro assegnati. È chiaro che, nonostante in Figura 2.2 si prendano ad esempio due soli robot, l'approccio possa essere facilmente esteso ad un numero N arbitrario di robot. La tassonomia di riferimento è dunque presentata in Figura 2.3, con una simbologia similare.

In [27] sono descritti i 5 principali ruoli che un umano può assumere in un sistema di controllo multirobot, ognuno caratterizzato da diversi livelli di *workload* (carico di lavoro) e *situation awareness* (consapevolezza della situazione, abbreviata SA). Di conseguenza ogni modello è adatto a diverse applicazioni, come vedremo nel seguito.

Supervisor interaction. Un supervisore svolge ruoli di alto livello, monitorando lo stato del sistema e dell'ambiente e compiendo solo, eventualmente, decisioni strategiche. Tipico nel caso in cui l'autonomia sia molto affidabile e si occupi di quasi tutti i compiti, ma sia comunque necessario mantenere un controllo umano sul sistema. Molto usato in ambito industriale e recentemente applicato anche in ambito USAR [5]. Per un più approfondito studio delle interfacce di supervisione multirobot si veda [6].

Operator interaction. L'operatore deve svolgere azioni di controllo diretto dei robot, monitorare l'esecuzione delle azioni ed eventualmente

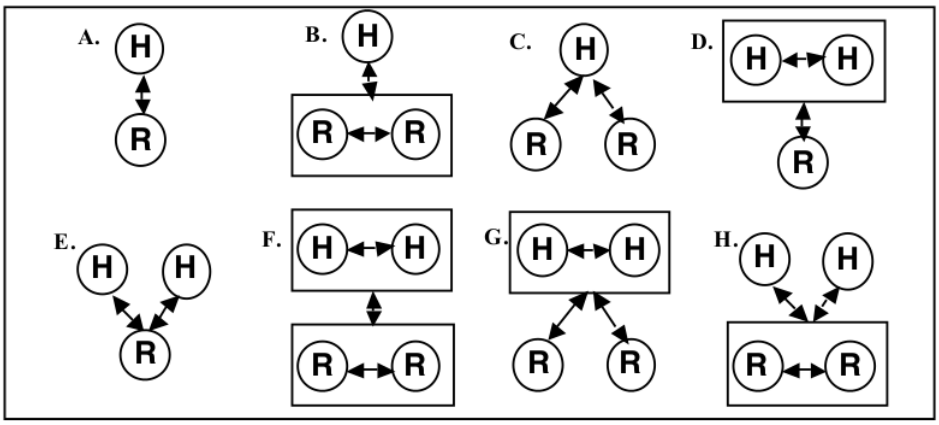

Figura 2.2: Tassonomie di HRI. Figura tratta da [26].

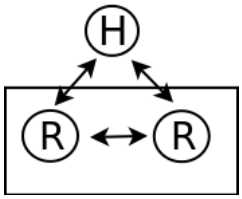

Figura 2.3: Tassonomia di riferimento per l'interazione dell'operatore umano col sistema multirobot.

correggerle, anche prendendo direttamente il controllo ed escludendo l'autonomia (tele-operazione del robot). Mentre la supervisione prevede azioni di relativamente lungo termine e di alto livello, il ruolo di operatore si concentra sull'effettiva attuazione delle azioni da eseguire. Questo modello d'interazione è largamente utilizzato in ambito USAR, ed in generale ovunque si svolgano principalmente compiti di esplorazione. A seconda delle caratteristiche del sistema può avere sfumature molto diverse, passando da un controllo solo tele-operato dei robot ad uno *mixed initiative* (approfondito nel seguito) molto più vicino al ruolo del supervisore.

Mechanic interaction. L'essere umano ha il compito di intervenire direttamente sull'hardware dei robot. Poco interessante in ambito USAR.

Peer interaction. Robot e umani compongono i team di lavoro ed eseguono congiuntamente i compiti della missione. Ne è stato proposto e teorizzato l'utilizzo in diversi ambiti [28].

Bystander role. Si tratta delle persone che devono avere coscienza del lavoro dei robot per potervi interagire a diverso titolo. In ambito USAR ricoprono questo ruolo le vittime, che devono in qualche modo interagire coi robot o quantomeno non devono esserne messe in pericolo.

Il ruolo cui siamo maggiormente interessati in questa tesi e il più usato nello USAR, è quello dell'operator interaction. Come abbiamo detto, i compiti dell'operatore ed il modo in cui li esegue sono fortemente influenzati dal livello di autonomia del sistema.

2.3.3 L'autonomia - breve tassonomia

L'introduzione di procedure automatiche all'interno di una sistema, nello USAR come in molte altre aree, non elimina il contributo umano, ma lo modifica e auspicabilmente lo ottimizza. Gli autori di [29] indicano quattro macro-aree di applicazione dell'autonomia in un sistema complesso:

- Acquisizione delle informazioni;
- Analisi delle informazioni;
- Presa delle decisioni e selezione delle azioni da eseguire;
- Esecuzione delle azioni.

I sistemi più complessi, per permettere all'operatore di averne un buon controllo (il che si traduce nell'avere opportuni livelli di workload e SA) devono intervenire con un opportuno livello di autonomia in ognuna delle quattro aree. Bisogna fare attenzione al fatto che un livello di autonomia più elevato non implica prestazioni migliori, come illustrato nella Sezione 2.3.4, col

Low	(1)	human does the whole job up to the point of turning it over to the computer to implement
	(2)	computer helps by determining the options
	(3)	computer helps to determine options and suggests one, which human need not follow
	(4)	computer selects action and human may or may not do it
	(5)	computer selects action and implements it if human approves
	(6)	computer selects action, informs human in plenty of time to stop it
	(7)	computer does whole job and necessarily tells human what it did
	(8)	computer does whole job and tells human what it did only if human explicitly asks
	(9)	computer does whole job and decides what the human should be told
High	(10)	computer does the whole job if it decides it should be done, and if so, tells human, if it decides that the human should be told

Tabella 2.1: Livelli da autonomia proposti in [30].

supporto di numerose prove sperimentali. All'interno delle suddette aree si può dunque adottare un livello di autonomia opportuno, scelto da un continuo di possibilità. Ad esempio [30], esteso in [31], propone una scala dei possibili livelli di autonomia, suddividendoli nei dieci livelli principali mostrati in Tabella 2.1. Partendo dal più basso livello d'autonomia, in cui l'essere umano compie tutto il lavoro decisionale per poi delegare all'elaboratore (o nel nostro caso ai robot) la sua esecuzione, si arriva al livello in cui è l'elaboratore ad eseguire tutto il lavoro decisionale e decide anche se l'essere umano deve essere informato o meno sulle azioni effettuate. I livelli intermedi identificano le situazioni in cui le decisioni vengono prese in parte via via maggiore dall'elaboratore, limitando sempre di più l'apporto della componente umana ed il suo livello di controllo sul sistema.
Altri modelli sono stati presentati, con focus su aspetti diversi [32], ma riteniamo i dieci livelli presentati una rappresentazione teorica chiara ed idonea agli scopi della nostra tesi, in quanto sono incentrati sulle dinamiche decisionali e astraggono completamente da quelle di esecuzione dei compiti, che diamo per scontato vengano eseguiti in toto dai robot senza alcun intervento umano.

2.3.4 USAR con approccio Mixed initiative

Quando si crea un sistema complesso multirobot e lo si vuole dotare di autonomia per favorire la gestione del sistema da parte dell'operatore umano, bisogna tenere in forte considerazione il livello di autonomia ottimo che permette di ottenere le prestazioni migliori. Individuare tale livello di autonomia a priori può essere difficoltoso, quindi spesso si adottano soluzioni tecnologicamente più raffinate, con livelli di autonomia variabili secondo tre metodologie:

Adjustable autonomy. il livello di autonomia è configurabile dall'utente durante l'utilizzo del sistema. Ad esempio può essere possibile intervenire sulle soglie di rilevanza degli eventi che vengono notificati all'operatore, o sulle soglie di tolleranza nell'esecuzione delle azioni prima di richiedere un intervento umano.

Adaptive autonomy. il livello di autonomia viene automaticamente individuato dal sistema. Ad esempio un sistema adattativo può automaticamente filtrare le notifiche all'operatore, evitando di sovraccaricarlo, e selezionare il numero di notifiche da mostrare in base al livello misurato di workload.

Mixed initiative. combina entrambi gli approcci precedenti. Il livello di autonomia viene deciso congiuntamente dal sistema e dall'operatore umano, con priorità a quest'ultimo. Gli autori di [3] dimostrano come questo approccio ottenga in genere le migliori performance in assoluto in sistemi USAR. Le prove eseguite da [3] si riferiscono ad un ambiente simulato con un sistema multirobot strutturalmente simile a quello presentato in questo lavoro di tesi.

Durante le missioni con il sistema multirobot, l'operatore è sottoposto ad un certo workload che ne influenza le prestazioni e che chiameremo *workload totale*, ad indicare che è quello complessivo dell'intero sistema e include i workload dovuti ai singoli robot, alla loro coordinazione, a fattori esogeni o altro. Un secondo tipo di workload che ci interessa analizzare è il *workload specifico*, ossia il workload dovuto ad ogni singolo robot e che si ripercuote sull'operatore. Esso è dovuto alle dinamiche di controllo dei robot, sia durante il monitoraggio delle azioni del robot e nell'analisi dei dati da esso raccolti, sia nella decisione delle azioni da compiere e nella comunicazione delle stesse al robot. Per fare un esempio, il workload totale di un sistema con N robot è dato dalla somma del workload specifico di ognuno dei robot, cui va aggiunto il workload dovuto alle dinamiche di coordinazione, monitoraggio del sistema complessivo, creazione di strategie di alto livello e dovuto anche a fattori esogeni (elementi di distrazione nell'ambiente circo-

stante l'operatore, malfunzionamenti nell'interfaccia di controllo, ecc). Un aumento del livello di autonomia dei robot, in linea di massima, permette la riduzione del workload specifico, permettendo quindi una gestione migliore del sistema, aspetto che in ambito USAR si traduce in aree esplorate più vaste, nella capacità di gestire più robot, e di conseguenza nella possibilità di trovare più vittime in minor tempo. Tuttavia, l'aumento dell'autonomia comporta una riduzione della SA ed un aumento del workload totale, legato al maggior utilizzo delle capacità del sistema [33]. Nel Capitolo 3 descriviamo il nostro approccio alla risoluzione del problema. Abbiamo deciso di adottare per il nostro sistema un approccio mixed initiative, visti i risultati sperimentali di [3], [8] e [34].
Per un'analisi completa degli effetti dell'autonomia e di altri fattori sulle dinamiche di controllo di sistemi multirobot da parte dell'operatore, col supporto di numerose prove sperimentali, si veda [4].

2.4 Sistemi paragone

Descriviamo in questo paragrafo, a conclusione del capitolo, quattro sistemi utilizzati per operazioni di USAR (ma anche di esplorazione e mappatura in genere) che rappresentano il vero e proprio stato dell'arte col quale confrontare le nostre soluzioni ed il nostro sistema.
Tutti e quattro i sistemi sono stati testati sia in prove ripetibili che in competizioni internazionali e contengono notevoli innovazioni dal punto di vista della HRI, innovazioni sulle quali andiamo maggiormente a focalizzare la nostra attenzione.

2.4.1 Steel - Carnegie Mellon University, Pittsburgh, USA

Il progetto Steel, della Carnegie Mellon University, ha partecipato con successo a diverse edizioni della RoboCup Rescue Simulation League [11] [35]. Steel è basato su un sistema modulare chiamato MrCS, i cui singoli moduli sono in genere implementazioni di algoritmi noti e testati che si inseriscono nel sistema tramite proxies che fanno da interfacce di comunicazione fra i moduli stessi. L'architettura è presentata in Figura 2.4. I robot, simulati nell'ambiente USARSim, vengono comandati tramite interfacce dette Driver che sono collegate al sistema tramite un proxy chiamato Machinetta, creato appositamente per il coordinamento di agenti robotici che si trovino ad operare in ambienti dinamici e pericolosi. I robot comunicano fra di loro e con l'operatore tramite il communication server. L'operatore interagisce col sistema tramite un'interfaccia utente connessa anch'essa tramite proxy

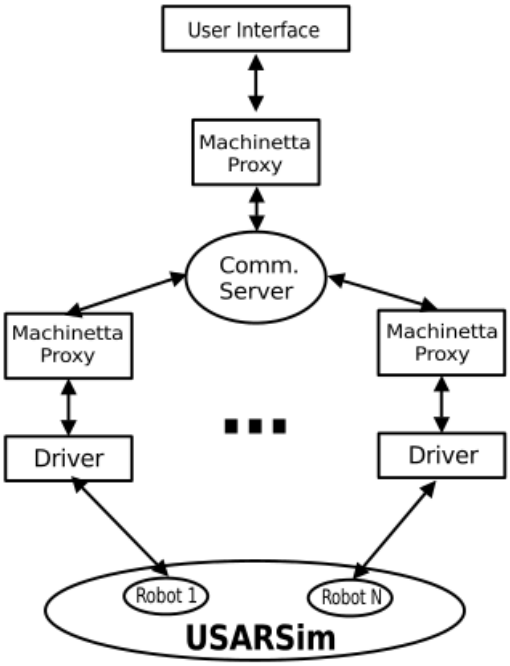

Figura 2.4: Architettura del sistema Steel - Carnegie Mellon University. Fonte: [11].

Machinetta al resto dell'architettura.
Le principali funzionalità fornite dai moduli di cui sono dotati i singoli robot includono *SLAM* (*Simultaneous Localization And Mapping* per la mappatura dell'area e la localizzazione dei robot nella mappa), esplorazione autonoma, coordinamento fra robot e rilevamento vittime. Dal punto di vista dell'interazione con l'utente, MrCS permette diversi livelli di interazione:

- Piena autonomia con selezione di *aree di interesse* che il sistema cerca di esplorare a priorità maggiore. I moduli di coordinamento ed esplorazione autonoma si occupano di allocare i robot ai vari compiti e di calcolare i percorsi da seguire;
- Impostazione manuale del percorso da percorrere per il singolo robot, tramite una serie di *waypoint*, ossia punti da raggiungere situati nelle immediate vicinanze del robot o di un waypoint precedente. I waypoint sono caratterizzati dal non coinvolgimento di funzioni automatiche se non per l'esecuzione di semplici azioni di movimento, quindi devono essere raggiungibili tramite una rotazione ed una traslazione senza alcun ostacolo nel percorso. Nel sistema Steel è possibile specificare una serie di waypoint per formare un percorso complesso, purché da ogni waypoint sia raggiungibile direttamente il successivo. Nella Figura 2.5

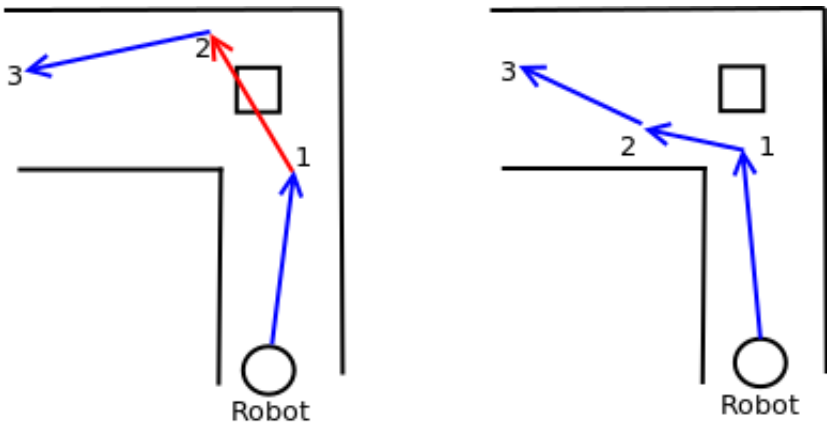

Figura 2.5: Esempio di utilizzo di comandi tramite waypoint. Nella figura a sinistra un percorso che comporta un incidente poiché un ostacolo si frappone fra il waypoint 1 e 2. A destra un percorso correttamente specificato per superare l'ostacolo.

si mostra un esempio di utilizzo dei waypoint;

- Tele-operazione del robot. L'operatore assume pieno controllo del robot e ne attua direttamente le azioni.

Per quanto riguarda la *GUI* (*Graphical User Interface*, interfaccia grafica utente), componente fondamentale per definire le dinamiche di interazione dell'operatore col sistema, si tratta di una classica GUI mappa-centrica. Lo scopo principale della GUI di Steel è permettere all'operatore di gestire diversi robot contemporaneamente, impartendo loro dei comandi diretti o lasciando che sia il sistema stesso a prendere le decisioni. Lo strumento principale con cui interagisce l'operatore è dunque la mappa, scelta che si è dimostrata molto efficace in compiti di USAR [36] e oggi è utilizzata da numerosi sistemi. In Figura 2.6 si mostra la GUI del sistema Steel.

2.4.2 Hector - Darmstadt University, Darmstadt, Germania

Partecipante dal 2009 alla RoboCup Rescue Robot League, il sistema Hector della Darmstadt University, presenta un sistema multirobot fisico con un approccio molto interessante riguardo le interazioni dell'operatore col sistema [37] [38]. L'obiettivo del team di ricerca è quello di portare l'operatore ad assumere un ruolo di puro supervisore. La comunicazione fra essere umano e robot è guidata da eventi e non è continua come in altri sistemi, dove viene concessa all'operatore la possibilità di intervenire sui robot a piacere. Il supervisore controlla le diverse informazioni in modo asincrono rispetto al momento della loro generazione, in linea col concetto di supervisore già illustrato nella Sezione 2.3.2. L'architettura generale di

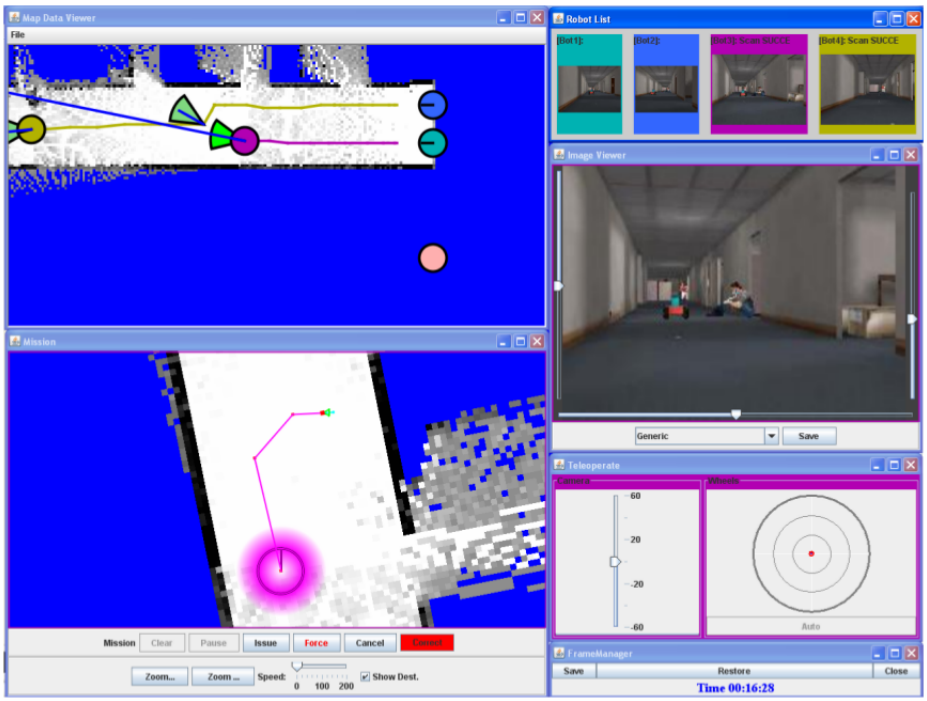

Figura 2.6: GUI del sistema Steel - Carnegie Mellon University. Fonte: [11]

interazione è rappresentata in Figura 2.7. Tale sistema si basa sulla teoria CEP (Complex Event Processing), usata nei sistemi di comunicazione [39] e fondata sull'utilizzo di algrebre per il confronto fra eventi, la loro aggregazione, classificazione e filtraggio. Il supervisore è in grado di definire delle politiche di comportamento del robot, interpretate ed eseguite dal componente chiamato policy system, presente in ciascuno dei robot. Le politiche vengono tradotte in obiettivi richiesti al robot in termini di eventi obiettivo e, tramite CEP, vengono definite le priorità e le sequenze di azioni effettivamente da intraprendere. Queste azioni portano a degli eventi effettivi, che vengono classificati e tradotti in notificazioni o richieste per l'operatore. Il policy system filtra tali informazioni e richieste provenienti dai robot per ridurle ad un sotto-insieme privo di ridondanze o informazioni superficiali e le propone all'operatore, chiudendo il ciclo di controllo.
Sulla base della classificazione degli eventi si decide se e come presentarli al supervisore. Hector prevede tre tipi di notifiche [5], in linea coi possibili livelli d'autonomia di [30]:

Decisione autonoma con veto. la decisione viene presa autonomamente dal sistema. Il supervisore umano può eventualmente annullarla e sovrascriverla entro un certo tempo, oltre il quale viene avviata l'ese-

cuzione delle azioni legate alla decisione. Questo comportamento appartiene al livello di autonomia indicato col numero (6) nella Tabella 2.1.

Decisione autonoma con conferma. il sistema prende la decisione ma attende la conferma del supervisore prima di eseguire le azioni relative. Equivale al livello (5) nella Tabella 2.1.

Decisione del supervisore. Il sistema notifica al supervisore proponendo una serie di decisioni possibili. Sarà il supervisore a scegliere la decisione da attuare. Nella classificazione in Tabella 2.1, questo comportamento equivale al livello (2).

Dati i possibili eventi e le possibili notifiche verso il supervisore, il sistema sceglie autonomamente come classificare gli eventi e quali notificare sulla base di politiche dinamiche che variano a seconda della missione corrente, della fiducia del supervisore nel sistema e delle impostazioni selezionate dal supervisore stesso.

Riteniamo che l'approccio di Hector possa ridurre notevolmente il carico di lavoro cui è sottoposto l'operatore (supervisore) e migliorarne la SA, ma non garantisce prestazioni ottime in ambito USAR, per le motivazioni esaustivamente argomentate nei paragrafi precedenti. In particolare, la riduzione del controllo sul sistema da parte della componente umana e le prestazioni autonome dei robot, inferiori a quelle raggiungibili in un sistema mixed initiative, sono ingiustificabili in ambito USAR, dove sono in gioco vite umane e l'ambiente è pericoloso e in rapida evoluzione durante le missioni.

Il sistema Hector si basa su *ROS* (*Robot Operating System*) come framework per lo sviluppo, ed utilizza un'interfaccia utente composta da moduli disponibili su ROS: *rosgui* per la visualizzazione delle informazioni mandate dai robot e *rviz* per le mappe e le interazioni con le stesse. L'interfaccia utente è dunque standard, con innovazioni soprattutto nella parte di gestione delle notifiche, col chiaro intento di avvicinare l'operatore ad assumere un ruolo da supervisore puro. In Figura 2.8 mostriamo l'interfaccia grafica del sistema Hector così come presentata alla RoboCup Rescue Robot League 2012.

2.4.3 Jacobs University, Brema, Germania

La Jacobs University si è presentata diverse volte sia alla manifestazione ELROB (European Land-Robot Trial) sia alla competizione RoboCup Rescue Simulation League, presentando in entrambe le sedi un sistema molto simile dal punto di vista dell'HRI [8]. L'implementazione della Jacobs University ha struttura modulare, con singoli moduli che assolvono alle funzioni di

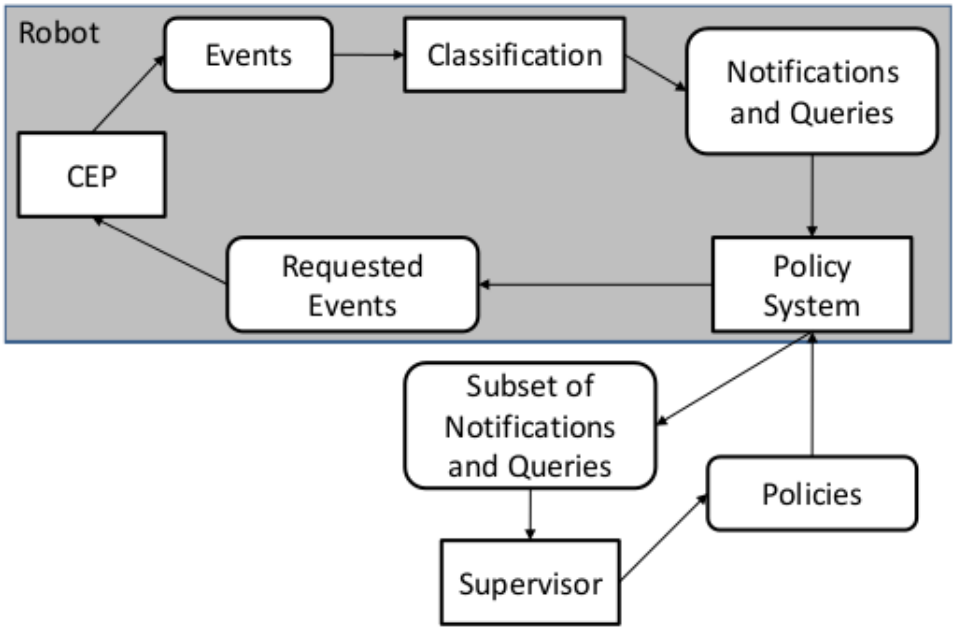

Figura 2.7: Architettura dell'interazione fra operatore e sistema del Team Hector. Fonte: [5].

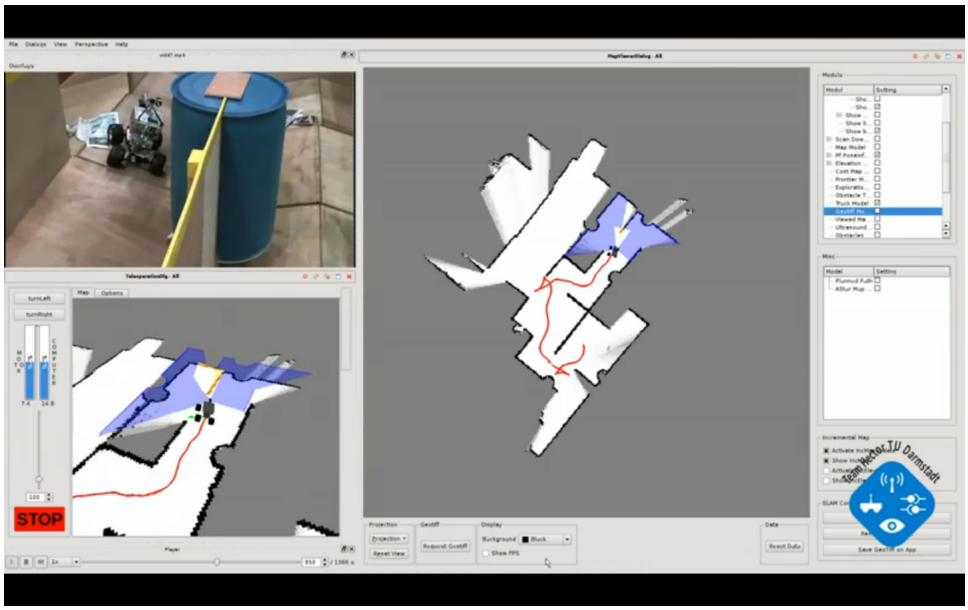

Figura 2.8: Interfaccia grafica del sistema Hector presentata alla RoboCup Rescue Robot League 2012.

SLAM, *obstacle avoidance* (per evitare gli ostacoli), coordinamento, esplorazione autonoma e calcolo del percorso automatico. L'architettura generale è illustrata in Figura 2.9. Il sistema si compone di tre parti principali: i componenti multirobot, che implementano le funzioni necessarie per la comunicazione fra robot e con l'operatore e contiene il modulo di coordinamento dei robot; l'autonomia del singolo robot, che determina le singole azioni da compiere e assolve alle funzioni di pianificazione dei movimenti evitando gli ostacoli e mappatura dell'ambiente; infine, una parte di implementazione di basso livello delle azioni da eseguire, che attua direttamente i meccanismi del robot nell'ambiente simulato.
Dal punto di vista dell'interazione con l'operatore umano, il sistema permette all'operatore di intervenire a diversi livelli d'astrazione [8] (Figura 2.10):

- Piena autonomia con coordinamento automatico;
- Specifica di waypoint, con capacità di evitare gli ostacoli;
- Tele-operazione.

La struttura dell'interfaccia grafica è anch'essa mappa-centrica, a riprova dell'efficacia di tale approccio. In Figura 2.11 mostriamo l'interfaccia grafica utilizzata nella competizione RoboCup Rescue Simulation League nell'edizione 2009.
Non sono incluse funzionalità di filtraggio e aggregazione delle notifiche all'utente, aspetto che probabilmente complica non poco il lavoro dell'operatore. Di questo sistema abbiamo a disposizione dati quantitativi sulle prestazioni. Con tali dati confronteremo le prestazioni del nostro sistema nel Capitolo 5.

2.4.4 Team Michigan - University of Michigan, Ann Arbor, USA

Il Team Michigan ha partecipato e vinto l'edizione 2010 della competizione MAGIC, che si focalizza sull'esplorazione, la mappatura e il disinnesco di minacce di diverso tipo. Il sistema consiste di 14 robot [2], gestiti da due operatori, uno (*sensor operator*) addetto alla validazione e integrazione dei dati ricevuti dai robot (correzione mappe, verifica dati dei sensori), l'altro (*task operator*) addetto all'allocazione dei compiti ai singoli robot. A supporto degli operatori ci sono 4 interfacce, due contenenti un insieme degli stati dei robot ed una mappa 3D complessiva, le altre due tramite cui gli operatori interagiscono. Mostriamo le interfacce grafiche del Team Michigan nella Figura 2.12.

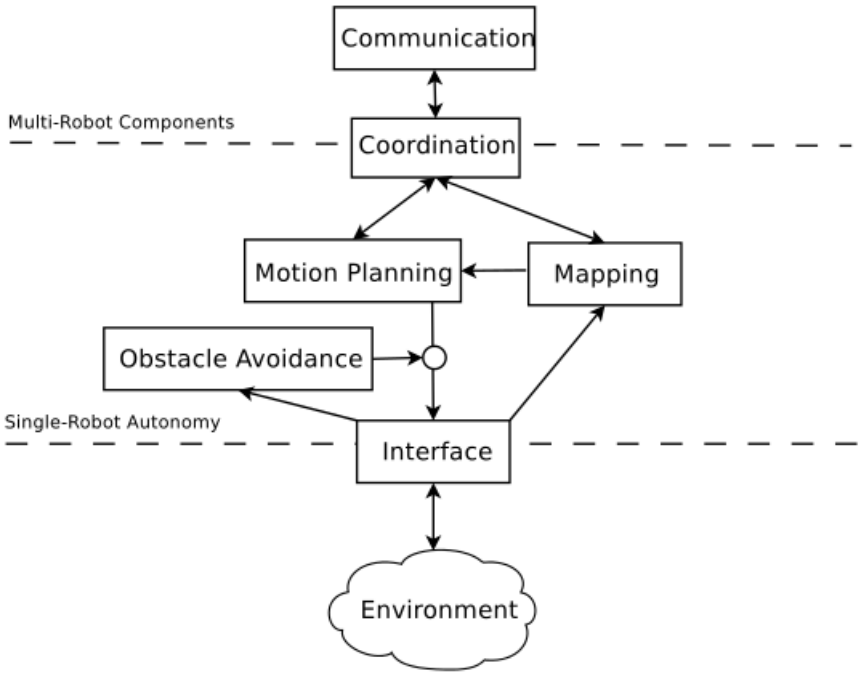

Figura 2.9: Architettura del sistema proposto dalla Jacobs University. Fonte: [8].

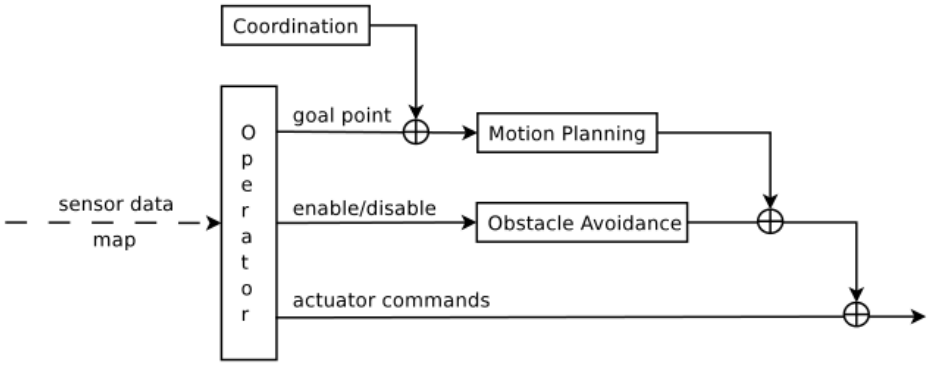

Figura 2.10: Dinamiche d'interazione fra operatore e sistema multirobot nel sistema proposto dalla Jacobs University. Fonte: [8].

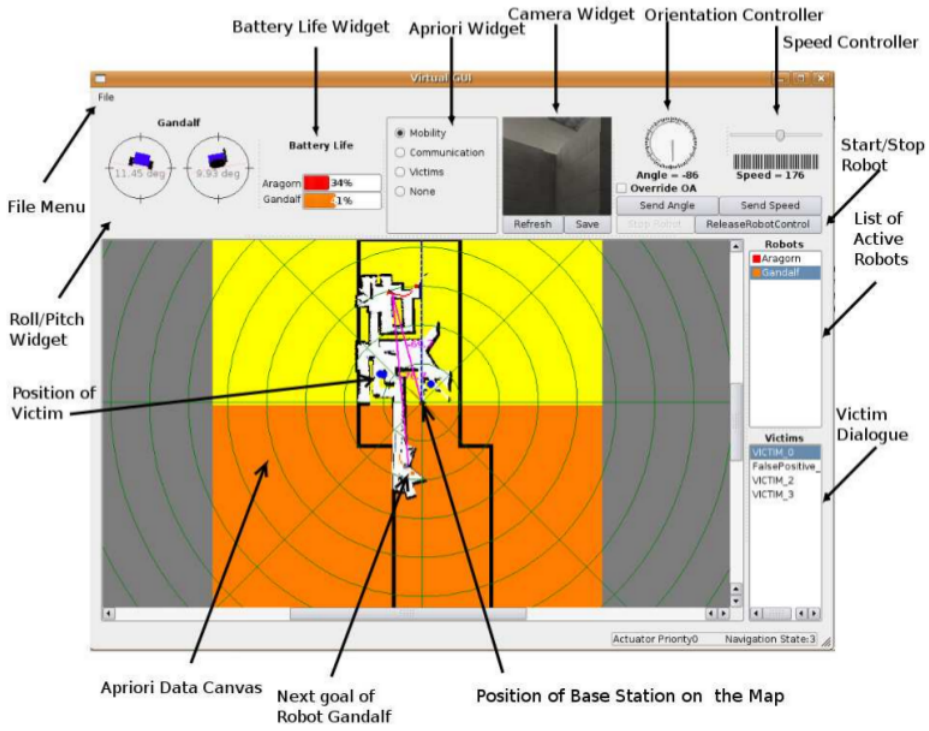

Figura 2.11: Interfaccia grafica del sistema della Jacobs University nella RoboCup Rescue Simulation League del 2009. Fonte: [8].

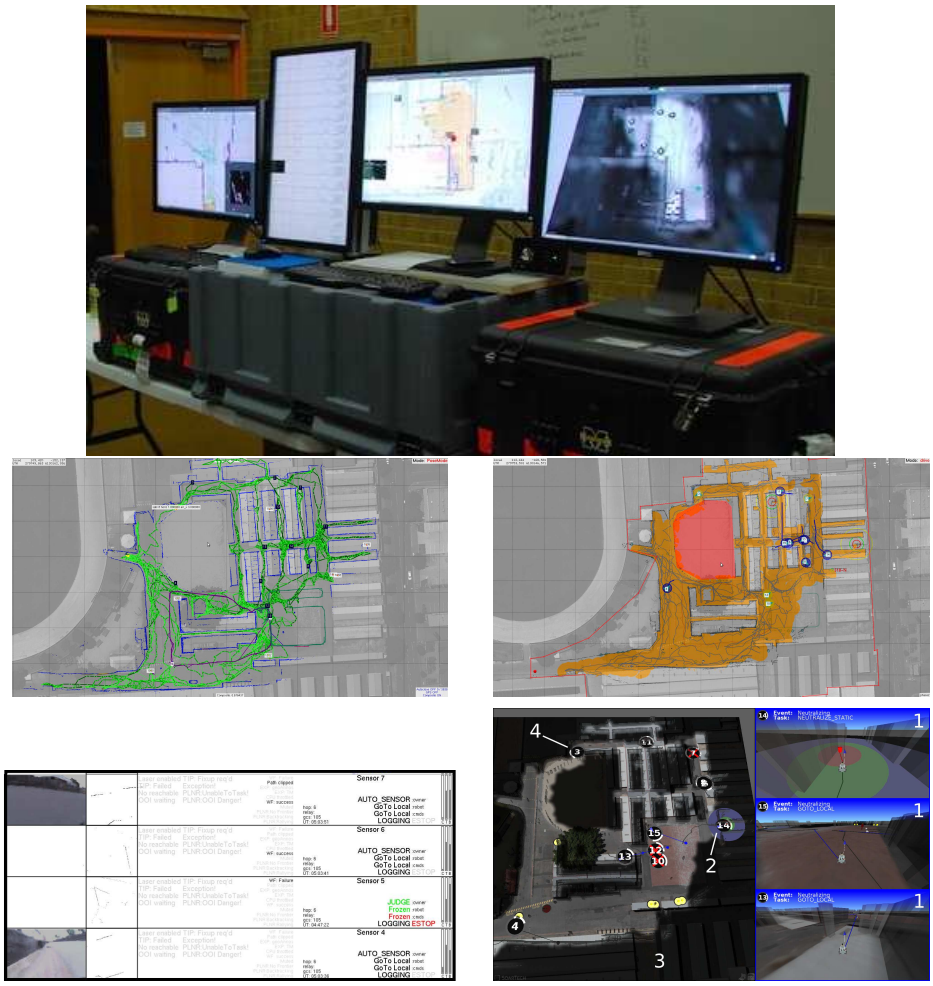

Figura 2.12: Le quattro interfacce grafiche utilizzate dal Team Michigan. Dall'alto: le quattro interfacce come utilizzate durante la competizione MAGIC 2010, l'interfaccia d'interazione per il sensor operator (a sinistra) e per il task operator (a destra), le interfacce di consultazione delle informazioni aggregate (a sinistra) e la mappa 3D chiamata SAGE (a destra). Fonte: [2]

Focalizziamo l'attenzione sulle modalità di interazione a disposizione del task operator:

Piena autonomia. i robot si allocano i compiti e lavorano in piena autonomia, con la sola supervisione dell'operatore;

Semi-autonomo. l'operatore indica al robot dove dovrà recarsi ed il sistema si occupa della navigazione fino a tale punto. Da notare che si tratta di una navigazione diversa da quella basata su waypoint, visto che il percorso non viene specificato dall'umano ma generato da un *path planner*;

Manuale. tele-operazione classica;

Stop. ferma il robot selezionato fino a nuovo ordine.

Una delle interfacce di monitoraggio, chiamata SAGE [10], contiene una mappa 3D ed un sistema di classificazione della priorità dei messaggi, aggregazione e filtraggio. Tuttavia il sistema non è adattativo automaticamente e non tiene conto delle condizioni di lavoro dell'operatore.

Anche del Team Michigan disponiamo di una serie di misure quantitative, con cui, nel Capitolo 5, confronteremo le prestazioni del nostro sistema.

2.5 Discussione

L'autonomia, come abbiamo visto, non può risolvere, almeno nel breve termine, molte delle problematiche che possono essere invece affrontate da sistemi robotici col supporto di una componente umana. La componente umana da sola, tuttavia, ha delle prestazioni scarse qualora sia necessario affrontare carichi di lavoro alti e controllare sistemi composti da molti robot. Abbiamo visto come umani e sistemi decisionali autonomi abbiano caratteristiche e capacità complementari e numerosi studi confermano che la loro inclusione congiunta nel ciclo di controllo dei sistemi porta a prestazioni molto più elevate sia rispetto ai sistemi manuali (con robot tele-operati) sia rispetto a quelli esclusivamente autonomi.

Lo USAR si presenta come una delle aree dove maggiore è il potenziale di applicazione di sistemi robotici. Tuttavia, date le sue caratteristiche, rappresenta una sfida insormontabile per sistemi che non includono la componente umana nel ciclo di controllo.

Dati questi presupposti, affrontiamo nel seguito il problema di costruire un sistema di interazione fra operatore e sistema multirobot in grado di garantire l'efficace controllo di numerosi robot, mantenendo accettabile sia il workload che la situation awareness.

Capitolo 3

Impostazione del problema di ricerca

Nel presente capitolo analizziamo nel dettaglio il problema di ricerca affrontato. Forniamo quindi le definizioni teoriche fondamentali per l'analisi e la comprensione del problema ed in particolare ci focalizziamo sulle problematiche aperte che il nostro lavoro tenta di affrontare in modo innovativo.

3.1 Obiettivo della tesi

L'obiettivo del nostro lavoro è quello di apportare dei miglioramenti alle procedure d'interazione dell'operatore umano con un sistema multirobot per lo USAR, con lo scopo di migliorare le prestazioni del sistema complessivo. In particolare miriamo a realizzare un sistema in grado di permettere all'operatore umano di supportare un elevato numero di robot in missioni complesse e in ambienti ostili. La realizzazione di tale obiettivo di alto livello passa necessariamente dalla realizzazione di un'interfaccia grafica utente (*GUI, Graphical User Interface*) innovativa, alla cui progettazione, implementazione e validazione è dedicata questa tesi. L'interfaccia è stata progettata ed implementata totalmente durante questo lavoro di tesi ed è stata presentata alla RoboCup Rescue Simulation League 2012 in Città del Messico come interfaccia d'interazione del sistema PoAReT del Politecnico di Milano, vincitore della competizione.
In questo capitolo ci riferiamo col termine *interfaccia* alle modalità d'interazione fra due agenti, in particolare operatore e sistema multirobot, e non ad un'interfaccia grafica utente, intesa come strumento che permette

l'implementazione dell'interfaccia d'interazione, alla quale ci riferiamo con il termine GUI.

3.1.1 Introduzione generale

Come già visto nel Capitolo 2, l'integrazione della componente umana all'interno di un sistema robotico permette di ottenere prestazioni migliori sia rispetto a sistemi completamente manuali sia rispetto a quelli completamente automatici [3]. Diventa, dunque, indispensabile integrare la componente umana nella maniera più proficua possibile nel ciclo di controllo. Interfacce poco funzionali dal punto di vista della HRI possono infatti portare ad una riduzione drastica delle prestazioni; difficoltà nel controllo e nell'interazione col sistema possono portare a risultati persino peggiori rispetto a quelli di sistemi esclusivamente manuali o esclusivamente automatici [4]. Numerosi sono gli aspetti fondamentali nella realizzazione di una GUI efficace per la gestione di un sistema multirobot, in particolar modo per applicazioni critiche quali lo USAR. Bisogna, infatti:

- Fornire all'operatore tutte le informazioni che gli sono necessarie, senza però sovraccaricarlo. Abbiamo concentrato in particolar modo i nostri sforzi su questo punto introducendo un *sistema di filtraggio dei messaggi* (*NFS*, Notification Filtering System) approfondito nella Sezione 3.5;
- Fornire all'operatore gli strumenti adeguati per l'interazione con il sistema, garantendogli sempre la possibilità d'intervento e un livello di controllo adeguato alla situazione. Anche su questo punto si è concentrato il nostro lavoro, giungendo ai risultati presentati in questo capitolo con il nome di *comandi di alto livello* (o *HLC, High Level Commands*), per la cui descrizione approfondita si veda la Sezione 3.4;
- Fornire ai robot i mezzi per comunicare in maniera efficace le informazioni all'operatore. Aggregare opportunamente tali informazioni per renderle analizzabili velocemente ed occupare spazio minimo sullo schermo;
- Garantire la robustezza del sistema ai guasti ed agli errori;
- Garantire prestazioni adeguate alle condizioni di lavoro, riducendo al minimo i tempi di elaborazione e mitigando, per quanto possibile, i ritardi di rete nella comunicazione coi robot;
- Garantire la corretta risposta del sistema a fronte dei comandi espressi dall'operatore;

- Garantire la sicurezza delle persone che dovranno interagire a vario titolo col sistema (operatore, soccorritori e vittime).

Buona parte di queste problematiche hanno una soluzione ingegneristica ben collaudata e solida, testata nel corso di anni di ricerca; altre sono invece sorte man mano che le GUI si sono raffinate e sono diventate sempre più usabili, fornendo funzionalità maggiori. Analizziamo brevemente nella prossima sezione gli approcci classici, ossia tutte quelle soluzioni affermate da anni di ricerca che dobbiamo tenere in considerazione nello sviluppo della nostra soluzione.

3.1.2 Approcci collaudati

In questa sezione analizziamo le soluzioni proposte dalla letteratura per alcuni problemi di ricerca classici legati alla HRI, particolarmente adatte a sistemi multirobot per lo USAR. Vengono esposte le possibili scelte e argomentate senza approfondire quelle da noi adottate durante la realizzazione della GUI che è oggetto di questa tesi, per le quali si rimanda alla Sezione 3.6.

Focus della GUI: video-centrico o mappa-centrico

Il primo problema da affrontare è quello di scegliere il componente principale della GUI. Tradizionalmente per l'ambito USAR si sono imposti due approcci concorrenti: l'approccio *mappa-centrico*, in cui la mappa globale dell'ambiente esplorato dai robot è il componente centrale e più rilevante della GUI, e l'approccio *video-centrico*, in cui il ruolo centrale è affidato ai flussi video provenienti dalla telecamera del robot controllato direttamente. Numerosi studi hanno confrontato i due approcci per sistemi multirobot [36], dimostrando come l'approccio video-centrico sia particolarmente adatto all'utilizzo in sistemi che devono gestire uno o pochi robot, preferibilmente tele-operati [40] [41]. L'approccio mappa-centrico si è invece dimostrato più idoneo per la gestione di un numero elevato di robot. Un'estensione dell'approccio mappa-centrico può prevedere l'utilizzo di mappe 3D [10] o mappe con l'integrazione di flussi video.
Una soluzione mappa-centrica è stata adottata da buona parte dei sistemi considerati nel Capitolo 2 [5] [8] [11]. In Figura 3.1 mostriamo un confronto tra una GUI mappa-centrica ed una video-centrica.

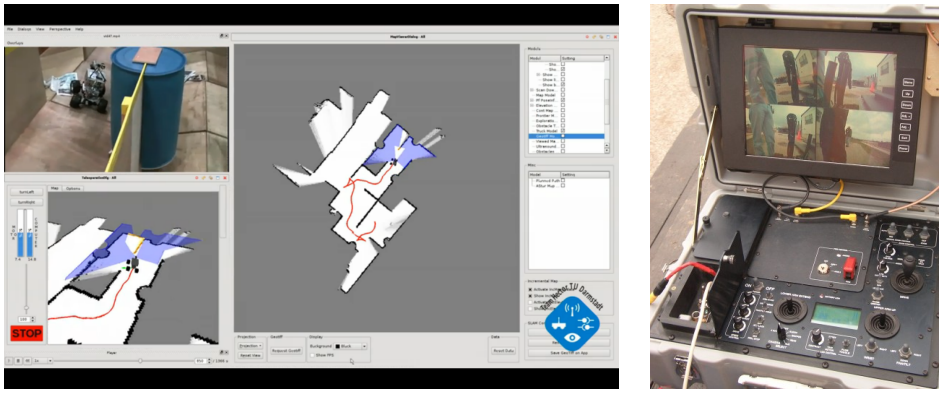

Figura 3.1: Sulla sinistra la GUI del team Hector [38], come presentata alla RoboCup Rescue Robot League 2012, ottimo esempio d'interfaccia mappa-centrica. Sulla destra l'interfaccia per la tele-operazione di un robot Talon, esempio di applicazione sul campo di un'interfaccia video-centrica.

Livello d'interazione e controllo

Abbiamo già evidenziato nel Capitolo 2 come gran parte della ricerca accademica sui sistemi robotici si focalizzi su sistemi completamente autonomi, sicuramente i più innovativi e stimolanti da un punto di vista scientifico, mentre l'applicazione di tali sistemi sul campo sia limitata a sistemi esclusivamente tele-operati.
È evidente come un controllo esclusivamente tele-operato non sia adatto al controllo di un gruppo anche piccolo di robot. Tuttavia, tale modalità d'interazione è la più efficace per la risoluzione delle eccezioni legate alla navigazione del robot. Per tale motivo viene spesso usata anche nei sistemi di controllo multirobot [3] [8].
Assieme al controllo tele-operato possono essere previste diverse modalità di controllo di più alto livello, fra le quali il più comune è il controllo tramite *waypoint*: l'operatore specifica dei punti da raggiungere ai robot che, autonomamente, attuano le azioni necessarie a raggiungerlo. I waypoint sono comandi semplici, che i robot eseguono senza alcun processo di calcolo del percorso e sono, quindi, utilizzati solo per spostamenti relativamente brevi e senza ostacoli, come già spiegato nel Capitolo 2.
Altro controllo molto usato è quello cosiddetto a *punto di destinazione*, molto simile dal punto di vista della HRI al controllo tramite waypoint. La differenza fondamentale è che nel controllo tramite waypoint il sistema si occupa semplicemente di attuare delle azioni di base per far giungere il robot alla destinazione, con un semplice controllo ad anello chiuso del moto. Nel controllo a punto di destinazione, invece, il sistema si occupa anche del calcolo

del percorso tramite un modulo di path planning, operazione che altrimenti dovrebbe essere compiuta dall'operatore umano.
Infine, l'ultimo comune livello di controllo è la piena autonomia: i robot esplorano liberamente l'area alla ricerca di vittime e l'operatore ne monitora il corretto funzionamento ed interviene dove opportuno, assumendo un ruolo da supervisore. Il lato negativo di questa modalità è lo scarso controllo dell'operatore sui robot.
L'adozione di una di queste modalità d'interazione e controllo non è certo esclusiva: le diverse modalità possono essere combinate in un unico sistema, come fatto in [8] e [11], e come da noi implementato.

Aggregazione delle informazioni

L'operatore, dovendo gestire più robot in un ambiente sconosciuto e pericoloso, e non essendo presente fisicamente sulla scena, deve elaborare una notevole mole di dati provenienti dai numerosi sensori di ciascun robot. Un'opportuna aggregazione e visualizzazione dei dati permette all'operatore di elaborare rapidamente ed in modo naturale, quindi meno faticoso a livello cognitivo, tali informazioni e mantenere così una migliore situation awareness.
Tecniche di aggregazione di base adottate comunemente in sistemi di controllo multirobot sono:

Camera thumbnails. Ovvero delle piccole porzioni di schermo dedicate alla visualizzazione dei flussi video di tutti i robot, ridotti sia in termini di risoluzione che di framerate per essere meno onerosi da un punto di vista computazionale e di traffico di rete.

Informazioni aggregate dei singoli robot. Si tratta di una porzione di GUI riservata a dati aggregati riguardanti il singolo robot e contenente: la missione attuale, la posizione, il livello di batteria, di connettività con altri robot o la stazione base, ecc. Nella Figura 3.2 è mostrata un'interfaccia grafica per un sistema multirobot per lo USAR che riserva buona parte dello spazio proprio a informazioni aggregate sui singoli robot.

Mappa arricchita. Si arricchisce la mappa con delle indicazioni utili all'operatore, come ad esempio la posizione di vittime o di pericoli. Una problematica di questo approccio è il rischio di confondere l'operatore qualora tali indicazioni vengano inserite in maniera autonoma, mentre si rischia di sovraccaricarlo qualora gli si chieda conferma per ognuna di esse [42]. Possibili soluzioni sono una forte integrazione con altri elementi dell'interfaccia per rendere più fluido l'inserimento di tali in-

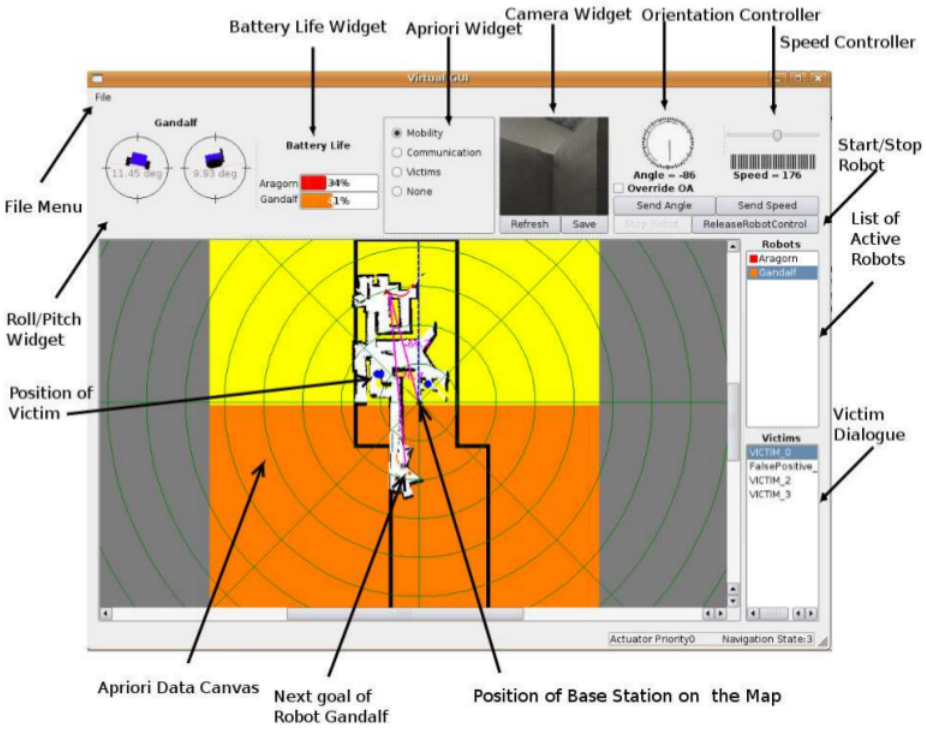

Figura 3.2: GUI del team della Jacobs University, partecipante alla RoboCup Rescue Simulation League nel 2009. Fonte: [8]

dicazioni sulla mappa, oppure l'utilizzo di un'interfaccia separata per la mappa arricchita, come fatto da [10]. In Figura 3.3 mostriamo una delle interfacce grafiche del Team Michigan, descritto nel Capitolo 2: si tratta di una mappa tridimensionale arricchita di numerose indicazioni in maniera autonoma. Per non confondere l'operatore durante la missione, tale interfaccia, chiamata SAGE, è separata da quelle usate per il controllo dei robot, e viene consultata solo occasionalmente dall'operatore per avere una miglior situation awareness [2].

Gestione dei fattori esogeni

Durante un'operazione di USAR bisogna garantire che le prestazioni del sistema nella gestione di ritardi causati dalla rete di comunicazione e ritardi causati dal carico computazionale, siano le migliori possibili. Piccoli ritardi nei flussi video e nel controllo sono tollerabili, purché siano quanto più possibile costanti [6]. Quindi, qualora le prestazioni della rete o del computer su cui deve lavorare il sistema non siano sufficienti a garantire controllo e feed-

Figura 3.3: SAGE, interfaccia grafica del Team Michigan basata su una rappresentazione tri-dimensionale della mappa arricchita da numerose indicazioni. Fonte: [10]

back in tempo reale, il sistema deve essere in grado di limitare la quantità e qualità di informazioni trasmesse ed elaborate per uniformare gli eventuali ritardi.

Robustezza del sistema

Nelle applicazioni critiche, USAR incluso, è fondamentale garantire la robustezza del sistema nella sua interezza, sia per quanto riguarda il funzionamento dei singoli robot sia quello della base di controllo e relativa interfaccia dell'operatore. Si deve pertanto essere in grado di prevedere, in fase d'implementazione, i diversi problemi che possono presentarsi e che devono essere affrontati durante le missioni di soccorso:

Perdita di comunicatività coi robot. Bisogna garantire la ripresa del controllo sul robot non appena questo torna connesso alla stazione di controllo. Protocolli di routing sono indispensabili per garantire una più affidabile connettività ed un maggior raggio di controllo da parte dell'operatore: tramite un protocollo di routing in grado di calcolare i percorsi in maniera dinamica, infatti, i robot possono comunicare con l'operatore o con altri robot anche se non connessi direttamente, purché vi sia un robot, o un insieme di robot, in grado di creare un ponte di comunicazione fra i due nodi che vogliono comunicare.

Malfunzionamenti dell'interfaccia utente. È necessario separare logicamente e fisicamente il funzionamento dell'interfaccia di controllo da

quello dei robot per far sì che gli uni siano robusti di fronte a problemi degli altri. Bisogna anche rendere possibile la ripresa normale della missione in maniera trasparente ai singoli robot una volta risolti i problemi.

3.1.3 Misura delle prestazioni

Analizziamo ora gli aspetti che vengono influenzati positivamente dalla realizzazione ottimale di una GUI per un sistema multirobot in ambito USAR. Introduciamo anche possibili misure di prestazione che in letteratura vengono utilizzate per valutare i sistemi multirobot [2] [9], e in particolare le interfacce d'interazione con l'operatore, e ne estraiamo quelle che sono adatte anche a missioni di USAR. Abbiamo utilizzato tali misure per valutare le performance del nostro sistema e ne illustriamo i risultati nel Capitolo 5. Per ognuno degli aspetti elencati nella Sezione 3.1.2, gli effetti che si hanno sull'intero sistema e sulle dinamiche d'interazione con l'operatore sono molto rilevanti. Diversi studi raccolgono l'effetto di ogni modifica al design della GUI e di ogni variazione delle caratteristiche del sistema sulle prestazioni. Per approfondimenti sugli effetti in ambito USAR vedere [4], [9] e [43].
Possiamo raggruppare i possibili effetti rispetto alle principali famiglie di attività che il sistema deve compiere.

Navigazione dell'ambiente

Questa famiglia include tutti i compiti legati alla navigazione degli ambienti. Modifiche alle dinamiche d'interazione o alle caratteristiche del sistema possono avere effetti sulla quantità di area esplorata piuttosto che sull'affidabilità con cui i robot si muovono. Le più comuni misure prestazionali legate a compiti di navigazione sono:

Percentuale compiti di navigazione completati con successo. Ove è possibile distinguere i singoli compiti di navigazione, una buona misura delle prestazioni del sistema è misurare la percentuale di tali compiti completata con successo. Un compito di navigazione può essere, ad esempio, navigare da un punto A ad un punto B indicato dall'operatore, oppure una sessione limitata nel tempo di tele-operazione, e si può considerarlo completato con successo qualora non ci siano incidenti. Un incidente può essere lo scontrarsi contro un ostacolo o un muro o il provocare danno ad una vittima. La misura viene effettivamente utilizzata in ambito USAR, ed è particolarmente utile per provare l'efficacia di un'interfaccia il cui focus è la tele-operazione dei robot. Risulta meno adatta quando si vuole controllare un insieme di robot

dotati di autonomia: il paradigma d'interazione dell'operatore è diverso, di conseguenza non è più possibile distinguere con chiarezza i singoli compiti di navigazione. Per questo motivo questa misura non viene da noi utilizzata.

Area coperta (mappata). Una misura tipica delle prestazioni di un sistema robotico cui uno dei principali obiettivi è l'esplorazione è proprio l'area mappata. Risulta anche di facile misurazione. L'operatore umano può dare un buon contributo alle politiche di esplorazione, sia con strategie di alto livello che tengano in considerazione delle informazioni disponibili a priori o altre informazioni di cui dispone l'operatore, sia grazie a interventi di basso livello che aiutino i robot a districarsi anche in situazioni complicate dove l'autonomia potrebbe andare in crisi. L'area esplorata, quindi, è una misura sia della qualità complessiva del sistema, sia dell'efficacia dell'integrazione della componente umana all'interno del ciclo di controllo, e per questo motivo è una delle principali misure da noi utilizzate per misurare le prestazioni della nostra soluzione.

Devianza dal percorso pianificato. L'operatore o il modulo di path planning indicano un percorso ad alto livello che il robot deve seguire. Più il robot si discosta da tale percorso, più il sistema è impreciso. Data la natura del problema USAR, questa misura viene utilizzata massivamente per la validazione degli automatismi di basso livello del robot nell'ambiente fisico. Avendo realizzando il prototipo della soluzione in un ambiente simulato, la misura perde di significato e non è particolarmente utile a misurare l'efficacia nel sistema d'interazione dell'operatore col sistema, pertanto non viene utilizzata.

Ostacoli evitati con successo. Quando un robot urta un ostacolo possono avvenire diversi eventi negativi, dal danneggiamento dell'hardware del robot stesso allo sfasamento dei dati rilevati nel momento dell'urto, per finire con una inevitabile perdita di tempo utile per lo svolgimento della missione. Questa misura è strettamente legata al concetto di successo per un compito di navigazione, come accennato sopra, e non viene utilizzata per lo stesso motivo: scarsa significatività in sistemi multirobot, ancor di più in sistemi con robot simulati quale il nostro.

Ostacoli non evitati, ma comunque superati. Come sopra, si tratta di una misura poco significativa in un sistema multirobot, in particolar modo se i robot sono simulati.

Tempo per completare un compito. Il tempo che un robot impiega per completare un compito di navigazione è una buona misura dell'efficienza del singolo robot nello svolgere tali compiti. A causa della difficoltà

d'identificare i singoli compiti di navigazione, questa misura non viene usata per valutare il nostro sistema.

Tempo speso nelle diverse modalità di controllo della navigazione. Analizzare la distribuzione del tempo speso nelle varie modalità di navigazione permette di avere un'idea dell'utilizzo del sistema da parte dell'operatore. Ad esempio, un operatore che dedica molto tempo alla tele-operazione dei robot, probabilmente, sta affrontando seri problemi nella navigazione dell'ambiente, problemi che rendono difficoltoso l'utilizzo di procedure di esplorazione autonoma. Dall'analisi delle modalità di funzionamento si evince anche il livello di controllo che l'operatore tenta di ottenere: un operatore potrebbe affidarsi poco alle funzionalità automatiche poiché non le ritiene affidabili o sufficientemente efficaci e potrebbe dunque imporre le destinazioni ai robot tramite dei waypoint. Abbiamo utilizzato anche questa misura.

Tempo che l'operatore dedica al compito. Il tempo che l'operatore dedica al singolo compito di navigazione aumenta con le difficoltà nell'attuazione dei meccanismi di movimento del robot e con le difficoltà cognitive del compito. L'interfaccia utente può lenire le problematiche legate al carico cognitivo, da un lato riducendo il workload dell'operatore per permettergli una maggior concentrazione sul compito, e dall'altra garantendogli una situation awareness migliore in modo che abbia una chiara idea di cosa fare. Utilizziamo questa misura, con l'ipotesi che l'operatore sia sempre coinvolto in compiti di navigazione o identificazione. Questa ipotesi è verificata a meno dei periodi di tempo che l'operatore non dedica affatto alla missione, periodi che in ambito USAR hanno rilevanza scarsa, vista la natura critica della missione.

Rapporto tempo operatore su tempo robot. Definito come il rapporto fra il tempo totale speso dall'operatore nella risoluzione di compiti di navigazione diviso il tempo totale speso da tutti i robot nella navigazione. È un buon indice di quanto si riescono a sfruttare i robot per l'esplorazione dell'ambiente. Questa misura è fortemente influenzata dalle dinamiche d'interazione fra operatore e sistema, e dunque la utilizziamo per valutare le prestazioni della nostra soluzione.

Percezione

Le misure di percezione sono relative alla capacità d'informare l'operatore in maniera efficace sull'ambiente attorno ai robot, nonostante l'operatore si trovi fisicamente distante. La percezione della situazione è influenzata dalle procedure d'interazione, dalle procedure di notifica all'operatore e dalla

rappresentazione delle informazioni. Ad esempio, una rappresentazione più naturale delle minacce, possibilmente integrandole sulla mappa utilizzata durante i compiti di navigazione, permette all'operatore di essere più cosciente della situazione, lo aiuta ad orientarsi meglio e favorisce la creazione di una più efficiente strategia con un minor sforzo cognitivo. Alcune comuni misure di percezione sono:

Detection ratio. Rapporto fra gli elementi d'interesse individuati e il totale di quelli presenti nell'ambiente e quindi potenzialmente individuabili. Indica quanto l'operatore sia riuscito a individuare gli elementi chiave della missione. Questa misura dipende da diversi aspetti legati alla GUI ed in genere dalle procedure di HRI: un sistema di riconoscimento autonomo degli elementi d'interesse (nello USAR possono essere le vittime, ad esempio) integrato opportunamente nell'interfaccia, un workload ottimale e quanto più omogeneo possibile, una situation awareness sufficiente e un sistema di notifica opportuno sono tutti elementi che permettono una più alta detection ratio. Tale misura è usata per valutare il nostro sistema in relazione alle vittime da trovare.

Recognition ratio. Definita come rapporto fra gli elementi d'interesse riconosciuti e quelli visti. Come si può notare, la misura è diversa dalla detection ratio, che considera non solo gli elementi visti ma tutti quelli presenti. Vista la difficoltà d'indicare quali elementi siano stati visualizzati a schermo e quali no, la misura non è stata utilizzata per la valutazione del nostro sistema.

Efficienza e fatica nelle procedure d'identificazione attiva. Misura che considera l'efficienza nei processi d'identificazione. Legata molto alle caratteristiche del flusso video, quindi di scarso interesse per i propositi della tesi.

Gestione del sistema

La facilità di gestione del sistema e la sua usabilità influenzano fortemente le prestazioni in missione. Un sistema facile da gestire favorisce le azioni dell'operatore che ha in ogni momento il controllo sugli aspetti più importanti, e contemporaneamente ne evita il sovraccarico con aspetti che hanno, nel contesto specifico, poca rilevanza o possono essere gestiti affidabilmente in maniera automatica. Alcune misure di prestazione in quest'area sono:

Fan out. Si definisce fan out il numero di robot che un operatore è in grado di gestire *efficacemente*. Gestire efficacemente un robot significa, in missioni di USAR, esplorare una sufficiente area mantenendo delle capacità percettive opportune per l'individuazione di vittime o elementi

d'interesse. Non avendo la possibilità di raccogliere una misura quantitativa adeguata sulle capacità percettive, a causa dell'aleatorietà nel trovare o meno oggetti d'interesse, abbiamo definito efficiente la gestione di un robot in grado di esplorare almeno l'area che esplora un operatore esperto gestendo un solo robot esclusivamente tele-operato. A seconda delle caratteristiche della mappa tale misura può cambiare. Calcoliamo il fan out come:

$$fan\ out = \frac{area\ esplorata\ totale}{area\ esplorata\ da\ un\ robot\ tele\text{-}operato}$$

Integriamo tale misura con un fan out basato sulla distanza percorsa, ottenuto come:

$$fan\ out = \frac{distanza\ percorsa\ totale}{distanza\ percorsa\ da\ un\ robot\ tele\text{-}operato}$$

Tempo d'intervento. Tempo che l'operatore impiega, mediamente, per rispondere ad una richiesta d'intervento da parte di un robot. Dipende dal workload cui è sottoposto l'operatore e dal sistema di notifica dei messaggi, quindi è interessante valutarlo per il nostro sistema.

Discrepanze nel livello d'autonomia. Misura la capacità dell'operatore di selezionare l'opportuno livello di autonomia in ogni situazione. Risulta difficile la sua valutazione pratica e quindi non abbiamo usato tale misura.

Manipolazione

Robot manipolatori vengono abitualmente usati in diversi ambiti quali quello militare (per disinnescare ordigni o addirittura per l'utilizzo di armi [24]) e aerospaziale. In ambito USAR l'utilizzo di robot con capacità di manipolazione è ancora limitato, ma è possibile una futura applicazione massiccia. Data la natura della nostra ricerca, focalizzata su dinamiche d'interazione in compiti prevalentemente di navigazione ed identificazione, non siamo interessati a valutare in alcun modo le prestazioni relative alla manipolazione, attività che non vengono eseguite durante le missioni di prova. A titolo puramente informativo citiamo due misure appartenenti a questa famiglia: il **livello di computazione cognitiva**, cui è sottoposto l'operatore durante i compiti di manipolazione, e gli **errori di contatto**, ossia il numero di errori compiuti durante la manipolazione di oggetti.

Missione USAR complessiva

Le misure che valutano le prestazioni della missione complessiva permettono di valutare la bontà del sistema nella sua interezza. Pecca principale di que-

ste misure, oltre al fatto che vengono influenzate da molti fattori diversi, è la loro aleatorietà: è infatti possibile che un sistema ottimo abbia scarse prestazioni a livello globale solo a causa di una serie di sfortunate coincidenze. Per questo motivo, queste misure vengono utilizzate affiancandole a numerose altre in grado di ridurre tale incertezza. Le misure da noi considerate sono:

Efficacia globale del sistema. Numerose metriche sono proposte per avere una misura di efficacia globale del sistema. Nelle varie competizioni dedicate a missioni di USAR vengono usate metriche composte da valutazioni su area esplorata, tempo impiegato, numero di vittime trovate, qualità della mappa fornita, ecc... Per semplicità, e in linea con la scelta della RoboCup Rescue Simulation League, consideriamo come misura di prestazione globale il numero di vittime trovate durante la prova.

Sforzo dell'operatore durante l'interazione. Lo scopo è valutare quanto l'operatore ha faticato nello svolgere la prova. Si tratta di una misura soggettiva focalizzata sulle dinamiche d'interazione e sulle relative scelte progettuali. Nel nostro caso abbiamo raccolto questa misura tramite un questionario sottoposto ai tester dopo le prove effettuate.

Prestazioni proprie dell'operatore

Le misure più importanti da raccogliere per validare il nostro lavoro sono, ovviamente, quelle relative alle prestazioni dell'operatore durante le missioni. È fondamentale, infatti, capire se le innovazioni introdotte nella HRI possono migliorare le prestazioni della componente umana nel ciclo di controllo, e di conseguenza le prestazioni globali del sistema. Le misure più usate e rilevanti sono due:

Situation Awareness (SA). La consapevolezza della situazione influenza la capacità dell'operatore di compiere scelte ottimali durante una missione di USAR. Una SA scarsa porta, infatti, a scelte sub-ottime quando non del tutto sbagliate, ad esempio comandare ad un robot di esplorare un'area già esplorata oppure provocare un incidente critico di navigazione compromettendo l'utilizzo del robot per il resto della missione. La misurazione della SA è un processo critico e vulnerabile a grossolani errori.

Esistono diverse procedure per la misurazione della SA in diversi ambiti, inclusi i sistemi multirobot per missioni di USAR. Ne citiamo due in particolare: *SAGAT* (*Situation Awareness Global Assessment Technique* [44] [45], tecnica per la valutazione globale della situation

awareness) e *SART* (*Situation Awareness Rating Technique* [46], tecnica di valutazione della situation awareness). SAGAT ottiene la misura interrompendo i test (che devono dunque essere necessariamente simulati) e ponendo all'operatore dei questionari per valutarne la SA. SART è invece una tecnica soggettiva, basata sull'autovalutazione della SA dell'operatore e postuma al test. Un confronto fra SAGAT e SART viene effettuato da [47] ed evidenzia come le due misure non siano fra loro correlate, aprendo la strada ad ulteriore ricerca sulle misure opportune per la misurazione della SA [41] e a diverse controversie. Un'analisi delle numerose misure esistenti per valutare la SA è proposta da [48] e [49], che evidenziano come non esista ad oggi una misura universalmente riconosciuta. Per questa ragione, e per i costi sia temporali che monetari di adozione di SAGAT, la misurazione della SA per il nostro sistema viene fatta in maniera soggettiva e qualitativa.

Workload. Carico di lavoro. È stato verificato che all'aumentare del numero di robot il principale problema sia proprio l'aumento del workload, che non permette una gestione ottimale del sistema a causa della saturazione delle capacità cognitive dell'operatore [7]. Diverse metriche sono state proposte in letteratura per valutare il workload, incluse misure psico-fisiologiche [50], misure soggettive [51] e misure indirette (numero di click, numero di cambiamenti di selezione di robot, numero di notifiche visualizzate [2]). Utilizziamo, per semplicità di misurazione e per affidabilità, una combinazione delle misure indirette: consideriamo il workload complessivo come la somma di tutti gli *eventi cognitivi* cui è sottoposto l'operatore. Un evento cognitivo è qualsiasi evento che comporti uno sforzo cognitivo all'operatore. A titolo di esempio, sono eventi cognitivi:

- La selezione di un robot;
- Una sessione di tele-operazione;
- Un comando waypoint o HLC;
- Un'interazione con la mappa;
- L'esibizione di una notifica all'operatore;
- L'interazione dell'operatore con una notifica;
- Il cambiamento delle configurazioni della GUI.

Il workload totale è dato dalla somma del numero di eventi cognitivi. Per semplicità e per ridurre al minimo l'aleatorietà della misura non consideriamo nel computo del workload tutti gli eventi cognitivi esogeni, quali l'interazione con il resto della squadra di soccorso umana o eventi legati all'ambiente in cui si trova l'operatore. L'utilizzo di un

simulatore dell'ambiente e dei robot ci permette di escludere tali elementi dall'analisi e di concentrarci sul workload dovuto esclusivamente alla gestione del sistema multirobot.

Altre

Altre misure possono essere introdotte per valutare aspetti interessanti nelle prestazioni del sistema o nelle modalità di utilizzo di alcuni suoi componenti. Abbiamo introdotto le seguenti:

Idle time dei robot. L'idle time è il tempo in cui i robot non effettuano alcun compito utile e sono fermi. Questo tempo è ovviamente sfruttabile in maniera più proficua durante le missioni. Un idle time totale basso, dunque, è sintomo di un uso estensivo ed efficace dei robot.

Percentuale messaggi utili. La percentuale di messaggi classificati come utili dall'operatore è un indice del modus operandi dello stesso: un messaggio viene classificato come utile quando l'operatore vi interagisce direttamente per essere focalizzato sul robot che ha sollevato la richiesta. Questo si traduce in un paradigma di utilizzo diverso dell'applicazione, non più guidata dalla strategia di alto livello dell'essere umano, la cui qualità decade all'aumentare del workload e al diminuire della SA, ma dagli eventi sollevati dai robot. Per un numero di robot elevato ci aspettiamo che tale paradigma di controllo sia più efficace.

Tabella riassuntiva

Presentiamo nella Tabella 3.1 le possibili misure fin qui descritte, con l'indicazione di quali fra queste sono state effettivamente usate per la valutazione delle prestazioni del nostro sistema.

3.2 Problematiche affrontate nella tesi

In questa seconda parte del capitolo sono analizzate le problematiche principali che affrontiamo. Tali problematiche sono caratteristiche di qualsiasi sistema robotico controllato da un essere umano. Bisogna tenerne particolarmente conto durante la progettazione delle dinamiche d'interazione di sistemi multirobot, e ancor più enfasi hanno nello USAR, date le caratteristiche delle missioni e la loro criticità.
Queste problematiche limitano fortemente il numero di robot manovrabili da un solo operatore, rendendo le dinamiche d'interazione complesse ed ingestibili.

Famiglia	Misura	Utilizzata
Navigazione	Percentuale di compiti di navigazione completati con successo	No
	Area coperta (mappata)	Si
	Devianza dal percorso pianificato	No
	Ostacoli evitati con successo	No
	Ostacoli non evitati, ma comunque superati	No
	Tempo per completare un compito	No
	Tempo speso nelle diverse modalità di navigazione	Si
	Tempo che l'operatore dedica al compito	Si
	Rapporto tempo operatore su tempo robot	Si
Percezione	Detection ratio	Si
	Recognition ratio	No
	Efficienza e fatica nelle procedure d'identificazione attiva	No
Gestione	Fan out	Si
	Tempo d'intervento	Si
	Discrepanze nel livello d'autonomia	No
Manipolazione	Livello di computazione cognitiva	No
	Errori di contatto	No
Missione	Efficacia globale del sistema	Si
	Sforzo dell'operatore durante l'interazione	Si
Operatore	Situation Awareness	Si
	Workload	Si
Altre	Idle time dei robot	Si
	Percentuale messaggi utili	Si

Tabella 3.1: Misure comunemente utilizzate nella HRI per misurare le prestazioni del sistema, organizzate in famiglie di compiti come in [9].

Fra le diverse problematiche esistenti, due meritano particolare attenzione, sia a causa del loro effetto sulle prestazioni complessive del sistema, sia per l'interesse che riscuotono presso la comunità scientifica.

3.2.1 Workload

Come già accennato in questo capitolo, il workload è un aspetto fondamentale delle dinamiche d'interazione fra operatore umano e sistema multirobot. Le capacità cognitive di un operatore umano sono limitate, per cui la gestione di un sistema complesso multirobot può facilmente saturarle, anche con pochi robot da gestire. La realizzazione di una buona GUI deve passare necessariamente da un'analisi delle dinamiche d'interazione e di presentazione delle informazioni all'operatore, con lo scopo di ridurre al minimo il workload specifico e permettere una miglior scalabilità col numero di robot [7]. Questo aspetto viene approfondito nella Sezione 3.4.

3.2.2 Situation awareness

La SA influenza pesantemente le scelte dell'operatore: con una bassa SA, infatti, può compiere gravi errori di valutazione che portano a perdite di tempo, danni ai robot o addirittura a danni alle vittime da soccorrere.
La SA si riduce drasticamente con l'aumentare del numero di robot, in parte a causa dell'intrinseco aumento di workload cognitivo, in parte a causa della difficoltà di mantenere ed elaborare la posizione ed i compiti di ogni singolo robot.
Nella Sezione 3.3 analizziamo nel dettaglio i flussi di informazioni che viaggiano attraverso l'interfaccia di controllo. Da tale analisi partiamo a definire le innovazioni introdotte per affrontare le due problematiche (scarsa SA e workload elevato) appena descritte. Infine nella Sezione 3.6 descriviamo la GUI da noi implementata e contenente, assieme allo stato dell'arte descritto nella Sezione 3.1.2, le innovazioni del NFS e degli HLC.

3.3 Progetto logico della soluzione del problema

I flussi di informazioni da e verso l'operatore rappresentano i mezzi di interazione che la componente umana ha per interfacciarsi col sistema. Parte di queste informazioni permettono all'operatore di impartire ordini ai robot, e rappresentano quindi i flussi di operazioni da Uomo a Sistema. Il flusso inverso permette di ricevere dati dai robot sulle variabili d'interesse per la missione e rappresentano il flusso di informazioni da Sistema a Uomo. In

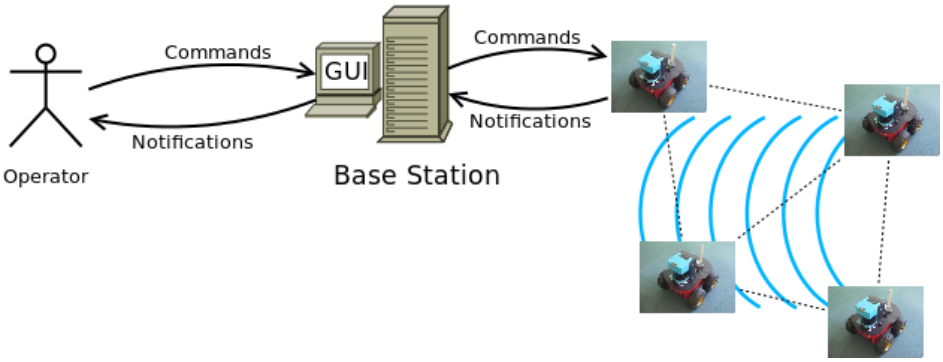

Figura 3.4: Flussi di informazioni tra operatore e robot nel nostro sistema PoAReT. Ogni robot riceve comandi dall'operatore e invia notifiche attraverso la stazione base.

Figura 3.4 è mostrato lo schema dei flussi informativi per l'interazione dell'operatore coi singoli robot.
La realizzazione di un'interfaccia d'interazione passa necessariamente dall'analisi di questi flussi, analisi che nel nostro caso ha portato alle due innovazioni da noi introdotte: i comandi di alto livello (HLC) e il *sistema di filtraggio delle notifiche* (o *Notification Filtering System*, *NFS*).
Il progetto logico della nostra soluzione parte dallo studio dei suddetti flussi di informazioni.

3.4 Flusso da Uomo a Sistema

L'operatore interagisce col sistema mandando ai singoli robot i comandi che vuole eseguiti. Come abbiamo visto nella Sezione 3.1.2, tali comandi possono essere espressi a diversi livelli d'astrazione, e si traducono in un diverso livello di controllo. Questo aspetto viene analizzato da [4], [30] e [31], e l'abbiamo affrontato più estesamente nel Capitolo 2.
Controllare il sistema tramite comandi di basso livello provoca un carico cognitivo maggiore per l'operatore, pertanto, tale approccio è bene limitarlo alle situazioni eccezionali dove l'intervento diretto dell'operatore è reso indispensabile. D'altro canto, la piena autonomia non permette di esercitare alcun tipo di controllo sul sistema e in particolare sulle politiche di esplorazione, rendendo impossibile sfruttare il contributo della componente umana, relegata ad un ruolo da supervisore. Il principale difetto dei comandi tramite waypoint o punto di destinazione è che sfruttano l'autonomia soltanto per il controllo del moto ed eventualmente per la pianificazione del percorso. È quindi impossibile integrare nel sistema funzionalità di esplorazione autonoma e coordinamento fra robot, poiché le operazioni di alto livello, quali

appunto coordinamento e strategie d'esplorazione, sono svolte dall'operatore umano. Il principale aspetto negativo di questo approccio è l'aumento del workload cui è sottoposto l'operatore e la conseguente difficoltà di controllare un numero elevato di robot [7] [52]. L'utilizzo della piena autonomia allevia il problema del controllo di gruppi numerosi di robot, ma, relegando l'umano al ruolo di supervisore, ha prestazioni peggiori rispetto al caso d'inclusione dell'umano nel ciclo di controllo [3] [4] [33] [34].
Comandi semi-autonomi, ossia comandi eseguiti in maniera autonoma dal sistema ma seguendo delle specifiche direttive da parte dell'operatore, possono permettere il contributo della componente umana nelle politiche d'esplorazione, sfruttando così le informazioni a priori, l'intuito e i segnali elaborati dall'operatore e non sfruttabili altrimenti dal sistema.
Un sistema multirobot tradizionale, come quelli presentati nel Capitolo 2, permette solo l'utilizzo della piena autonomia, dei waypoint e della teleoperazione [2] [8] [11] [38]. Per manovrare un numero elevato di robot in maniera efficiente, l'operatore dovrà sfruttare quanto più possibile l'autonomia. Supponiamo ora che questo sistema tradizionale debba essere usato per esplorare un albergo in cui vi sia stato un'incidente alla ricerca di vittime. L'autonomia dei singoli robot, assieme alle procedure di coordinamento fra essi, creeranno una politica di esplorazione ottimale per coprire la maggior area possibile in breve tempo. Se, però, abbiamo dell'informazione a priori (ad esempio sappiamo che, essendo le 13:00 è molto probabile che la maggior parte delle persone coinvolte si trovino nel ristorante dell'albergo) oppure abbiamo dell'informazione non processabile dal sistema (ad esempio dalle finestre di un'area ci giungono delle richieste di aiuto o delle persone che sono sfuggite al pericolo ci informano che si stava tenendo una festa nella sala conferenze) non siamo in grado di fornire queste informazioni al sistema e siamo quindi costretti o ad attenerci alle politiche di esplorazione autonoma, che diventano in questo caso sub-ottime, oppure a controllare i robot ad un più basso livello d'astrazione.
Con la disponibilità di strumenti per specificare direttive d'esplorazione di alto livello e quindi asservire l'autonomia dei robot alle decisioni dell'operatore è possibile sfruttare tutta l'esperienza e le informazioni a disposizione dell'umano, riuscendo ad affrontare in maniera più efficace le diverse situazioni che, in particolare in ambito USAR, possono presentarsi. Con riferimento all'esempio dell'albergo appena presentato, l'operatore può indirizzare i robot verso la direzione da cui vede le persone chiedere aiuto, oppure, avendo a disposizione la planimetria dell'edificio, può mandare i robot a esplorare con urgenza la zona della sala conferenze dove suppone ci potrebbero essere più persone, contribuendo alla specifica delle politiche di esplorazione.

Altro aspetto positivo dell'introduzione di comandi semi-autonomi è l'aumento del livello di controllo sui singoli robot funzionanti in modalità autonoma: specificando le indicazioni per l'esplorazione, l'operatore può avere una chiara idea di quali azioni i robot compiono, e riesce ad intervenire a un livello d'astrazione molto elevato sul comportamento dei robot. Ciò si traduce in carichi di lavoro relativi ai singoli robot più bassi e quindi in una migliore scalabilità del sistema col numero di robot. L'operatore non sarà, infatti, costretto ad attuare manualmente degli interventi per specificare strategie di esplorazione, compiendo necessariamente grande sforzo cognitivo per decidere le destinazioni ottime e coordinare i robot [53]. Potrà, invece, specificare delle politiche di alto livello, lasciando compiti onerosi a livello cognitivo, quali la determinazione delle destinazioni dei singoli robot e il loro coordinamento, al sistema automatico e potendo concentrarsi sulla gestione globale del sistema e sulla gestione delle eccezioni.
Infine, questo approccio permette all'operatore di comportarsi con i robot in maniera simile a come si comporterebbe con un team di ricerca umano, ossia specificando delle direttive e lasciando una certa autonomia nell'esecuzione delle stesse [54] [55] [56]. L'operatore potrà, ad esempio, comandare di esplorare una certa area, oppure di dirigersi lungo una certa direzione. Il sistema di navigazione e di esplorazione farà il resto, permettendo anche l'utilizzo di politiche di esplorazione più sofisticate da quelle usate dalle squadre di soccorritori umane, mappando l'area per migliorare ulteriormente la SA dell'operatore e sfruttando sensori avanzati per individuare vittime o elementi d'interesse che gli umani potrebbero faticare a individuare. Ciò permette, secondo le nostre ipotesi, di raggiungere una SA maggiore rispetto al caso di piena autonomia.
Questa innovazione e il paradigma di controllo risultante rendono più intuitivo l'utilizzo del sistema agli operatori esperti di missioni di USAR.
Abbiamo realizzato tale sistema di comandi con il nome di comandi di alto livello (HLC). Essi consistono di indicazioni ai singoli robot sulle politiche di esplorazione, permettendo di specificare comandi quali esplorare una certa area o esplorare lungo una certa direzione. I comandi non impongono vincoli all'autonomia dei robot, che quindi possono combinare le indicazioni dell'operatore con altri aspetti, quali la distanza di un'area sconosciuta, la sua ampiezza prevista, le posizioni e obiettivi dei robot vicini, e altre indicazioni semantiche e previsioni. Si lascia al sistema automatico l'onere di elaborare tutte le informazioni e scegliere la destinazione ottima per ogni robot, liberando l'operatore dal compiere tale costosa scelta e garantendogli nel contempo un buon livello di controllo sul sistema. Questo approccio permette anche di ridurre i tempi morti (chiamato anche *idle time*), in quanto

una volta concluso lo HLC indicato dall'operatore il robot non rimarrà inattivo, bensì proseguirà con l'esplorazione in maniera autonoma.
Da un punto di vista tecnico, gli HLC sono stati implementati integrando le preferenze dell'operatore all'interno dell'algoritmo di valutazione delle destinazioni possibili dei robot. Tale algoritmo, basato sul Multiple-Criteria Decision-Making (MCDM), valuta congiuntamente diversi criteri di scelta e prendere la decisione ottima [57]. Gli algoritmi di coordinamento fanno sì che i robot scelgano destinazioni diverse e sia massimizzata l'area esplorata. Gli HLC vengono impartiti dall'operatore interagendo con la mappa dell'ambiente, in modo che sia semplice e naturale indicare le decisioni strategiche.
Come abbiamo sottolineato nel Capitolo 2, umani e agenti automatici hanno competenze e capacità complementari. Lasciando i compiti di coordinamento e di aggregazione dei criteri di esplorazione all'autonomia nei robot, l'operatore umano può concentrarsi sui compiti che gli sono più congeniali, quali le scelte strategiche di alto livello (che tramite gli HLC può compiere con naturalezza e minimo sforzo) e la gestione delle eccezioni, sempre possibile grazie a comandi di basso livello come waypoint o tele-operazione.
In Figura 3.5 è rappresentato il flusso di operazioni della nostra soluzione, con l'inclusione degli HLC e ne viene fatto un confronto con il flusso di operazioni del sistema multirobot della Jacobs University. In Figura 3.6 sono, invece, rappresentati i principali moduli coinvolti durante l'esecuzione dei comandi all'interno di ogni singolo robot.
Per aiutare l'operatore nell'altro suo compito congeniale, la gestione delle eccezioni, dobbiamo analizzare i processi di notifica dei robot e i relativi flussi d'informazione.

3.5 Flusso da Sistema a Uomo

Durante le operazioni di USAR, i singoli robot dovranno eseguire i comandi impartiti ed allo stesso tempo aiutare l'operatore a focalizzare l'attenzione dove richiesto notificando gli eventi rilevanti. Bisogna aggregare i messaggi simili, eliminare quelli superflui e classificare i rimanenti secondo il livello di priorità con cui devono essere gestiti. Ad esempio, [5] prende come riferimento i cinque livelli di priorità dei messaggi tipico degli ambienti di sviluppo software: debug, information, warning, error, e fatal. Da questi estrae tre livelli (information, warning ed error) che utilizza per classificare i messaggi in arrivo e presentarli al supervisore.
Nel nostro caso i robot devono:

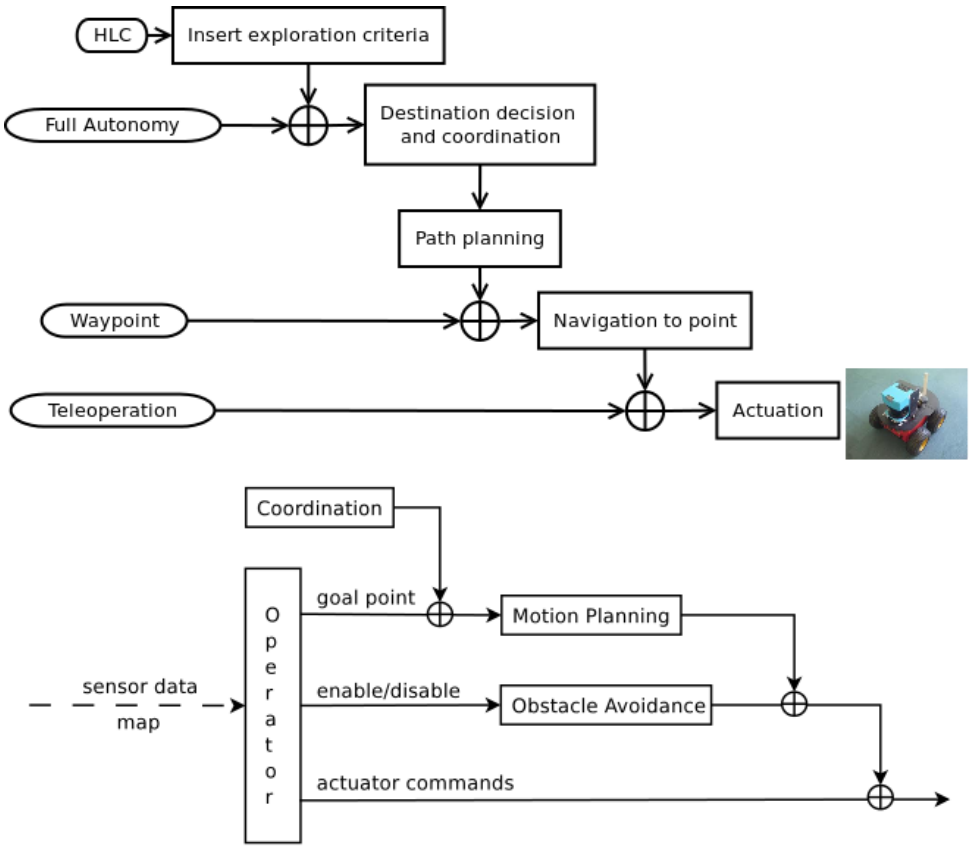

Figura 3.5: Interazione con i singoli robot nel nostro sistema (sopra) e nel sistema della Jacobs University (sotto, fonte: [8]). Da notare come gli HLC influenzino solo le politiche di esplorazione autonoma, mantenendo il resto della soluzione simile allo stato dell'arte ben consolidato.

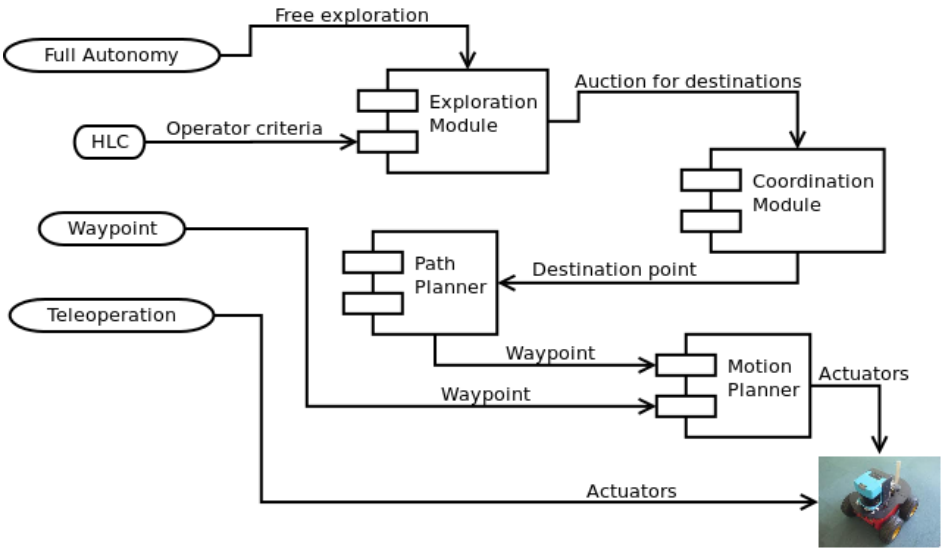

Figura 3.6: Moduli coinvolti nella esecuzione delle diverse tipologie di comando, da quelli più ad alto livello (HLC e piena autonomia) che sfruttano tutto il sistema, a quelli a più basso livello per la risoluzione delle eccezioni ed il controllo manuale.

- Notificare la presenza di vittime rilevate tramite un modulo di *victim detection* in grado di esaminare le immagini della telecamera e assegnare una confidenza alla presenza o meno di una vittima;
- Identificare i guasti e gli errori e segnalarli all'operatore. Tra questi possiamo individuare i fallimenti del path planner nel calcolo di un percorso o uno scontro con un ostacolo e la conseguente impossibilità di riprendere l'esplorazione;
- Segnalare la conclusione di un'attività, come ad esempio il raggiungimento di un waypoint impartito dall'utente, per indicare che il robot è disponibile per ulteriori comandi.

Senza questi processi di notifica, l'operatore deve adottare una procedura di monitoraggio manuale, come ad esempio una scansione dei robot ciclica o guidata dalla posizione sulla mappa, quando non addirittura casuale. Tale approccio è sub-ottimo in quanto possono essere trascurati dei robot che hanno bisogno d'intervento immediato e viene invece dedicato tempo a quelli che stanno già operando a pieno regime. Un paradigma d'intervento guidato dagli eventi notificati dai robot può migliorare, secondo le nostre ipotesi, il controllo del sistema e ridurre i tempi di attesa portando a migliori prestazioni complessive.
Al crescere del numero di robot, e delle loro funzionalità, cresce anche il numero di notifiche inviate all'operatore, che ben presto può trovarsi sommerso

dalle richieste e non avere il tempo per gestirle. Il risultato è un ritorno ad una procedura di monitoraggio dei robot manuale, spesso trascurando del tutto i robot che operano in zone periferiche, con conseguente decadimento delle prestazioni. Inoltre il workload imposto dalla molteplicità dei messaggi risulta troppo elevato per una gestione opportuna del sistema, spiazzando l'operatore.
Anche nell'ipotesi di pochi robot da gestire, l'operatore potrebbe essere intento a risolvere delle situazioni critiche che richiedono la sua piena attenzione e potrebbe trascurare le notifiche ricevute. Finita l'azione, tale notifiche rischiano di non essere più aggiornate e dunque essere di scarsa utilità, quando non fuorvianti. Le notifiche hanno, infatti, importanza decrescente col tempo, fino a diventare trascurabili oltre una certa soglia.
Quando l'operatore riceve troppe notifiche tende a trascurarle o a gestire solo le prime. Questo comportamento è difficilmente modificabile. Sarebbe invece auspicabile che l'operatore seguisse dapprima le richieste più importanti e solo successivamente le altre. Tuttavia, per valutare l'importanza di una notifica, l'operatore deve compiere dei notevoli sforzi cognitivi, per capire ad esempio quale robot l'ha mandata e ricollegarlo alla missione che gli aveva assegnato. Tali sforzi e tempo speso nella classificazione delle notifiche non è giustificabile e potrebbe essere meglio usato nella risoluzione di eccezioni. Uno strumento di classificazione delle notifiche che tenga conto delle condizioni operative dell'operatore (ad esempio il workload a cui è sottoposto) permette di ordinare le notifiche secondo degli specifici criteri d'importanza e quindi favorisce la corretta gestione delle eccezioni.
Durante una missione di USAR possono esserci dei guasti che non sono rilevati, ma che provocano un comportamento scorretto di qualche modulo del singolo robot. Ad esempio, un problema all'attuazione delle ruote dei robot potrebbe trarre in inganno il sistema e fargli rilevare ostacoli inesistenti, oppure dei problemi alla telecamera potrebbero confondere il modulo di victim detection portandolo a classificare ogni immagine come contenente una vittima. In questi casi si solleverebbe un numero di notifiche molto elevato e tutte inutili per l'operatore, bloccando il corretto funzionamento del sistema di notifica. Bisogna, dunque, analizzare il numero di notifiche mandate da ogni singolo modulo e robot nell'unità di tempo, per rilevare questi malfunzionamenti o comunque comportamenti non idonei. Bisogna anche permettere all'operatore di specificare quando un messaggio è un falso positivo o un vero positivo, in modo da adattare automaticamente i filtraggi applicati alle notifiche per tenere in considerazione l'affidabilità dei moduli.
Ogni operatore può voler gestire in modo diverso le notifiche: un operatore potrebbe ad esempio basare il controllo del sistema sulla mappa, trascuran-

do in parte i flussi video dei robot e quindi rischiando di non vedere delle vittime trovate. In questo caso l'operatore vorrebbe che le notifiche relative a possibili vittime trovate vengano mantenute per più tempo ed abbiano maggior priorità, proprio per sopperire agli aspetti negativi del suo modus operandi. Al contrario, un operatore che preferisce lavorare focalizzandosi sui flussi video potrebbe mancare di notare dei robot bloccati da qualche ostacolo, o robot che hanno completato una missione, e dunque vorrebbe che i messaggi relativi a errori di navigazione o completamento delle missioni abbiano maggior priorità. L'operatore, dunque, deve essere in grado di aggiustare, anche dinamicamente durante le missioni, le procedure di gestione delle notifiche.
L'operatore affianca alla gestione delle eccezioni il controllo dei robot, basandosi su delle strategie d'esplorazione di alto livello. Durante la specifica delle destinazioni o degli HLC ai robot, potrebbe dedicare l'attenzione ad agenti che hanno richiesto un intervento senza passare dalla gestione esplicita della notifica. In questi casi la notifica perde d'importanza e deve essere archiviata per non confondere l'operatore in caso la gestisse a postumi.
È evidente come sia necessario intervenire nella gestione delle notifiche dei robot. Ricapitolando, bisogna tenere conto di:

- workload dell'operatore;
- Affidabilità dei moduli;
- Preferenze dell'utente;
- Possibili malfunzionamenti dei moduli;
- Importanza dei messaggi;
- Tempo trascorso dalla generazione del messaggio;
- Interazioni con i robot che hanno richiesto un intervento senza passare dalla gestione della notifica.

Notevole rilevanza ha assunto la gestione delle notifiche nei più recenti sistemi multirobot [2], in particolare in quelli in cui l'essere umano è principalmente supervisore [5].
In questa tesi affrontiamo questo aspetto introducendo il NFS, implementato in un modulo chiamato *notification manager*, in grado di gestire le notifiche in maniera efficace, filtrarle, classificarle secondo la loro importanza e confidenza, eliminarle quando non più rilevanti o quando già gestite e in grado di elaborare gli input dell'utente per modificare le politiche di gestione delle notifiche.
Tramite il notification manager le richieste vengono aggregate, filtrate e classificate all'atto della ricezione, secondo la procedura illustrata dal diagramma di flusso in Figura 3.7.
I principali passi per la gestione delle nuove notifiche sono:

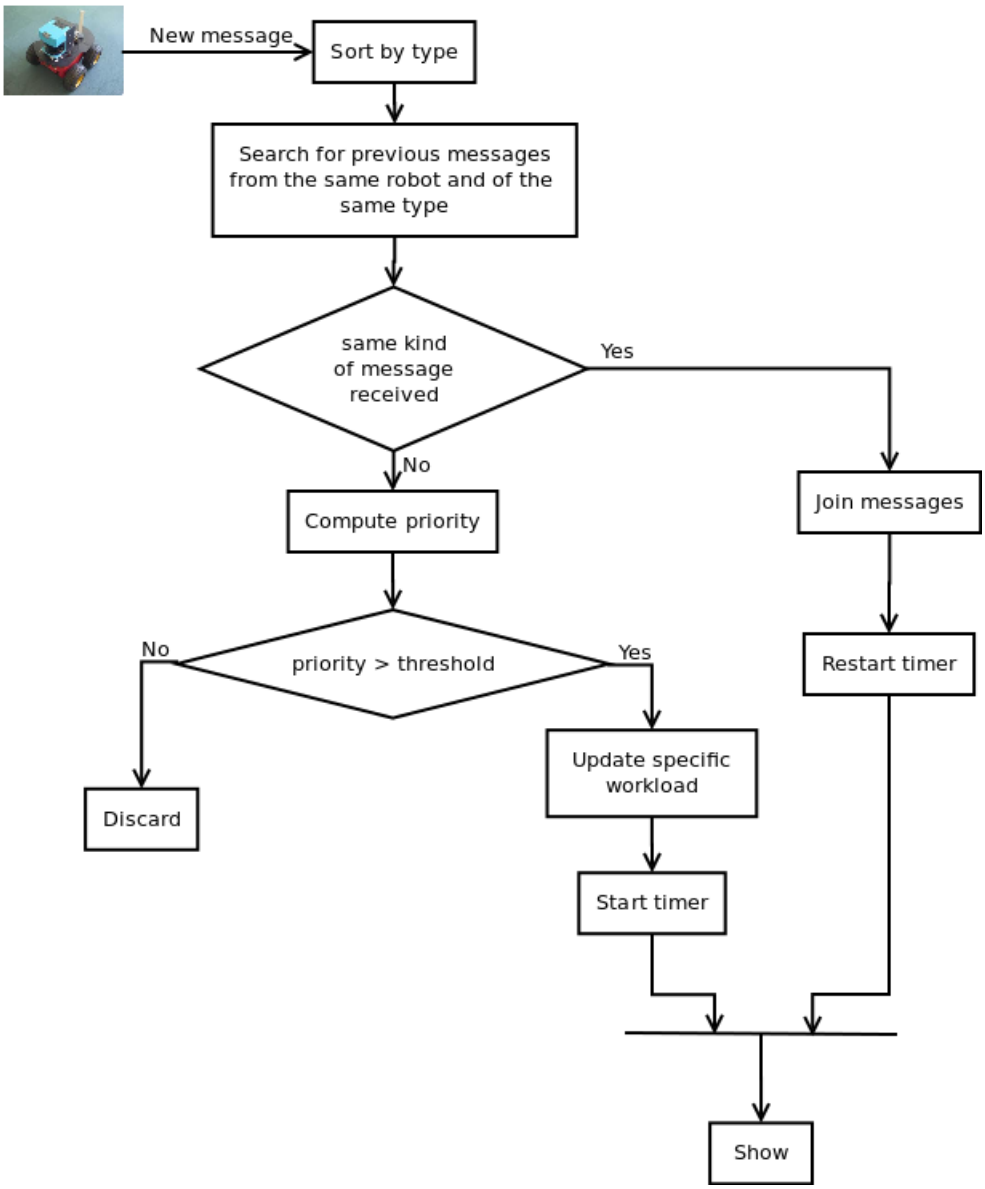

Figura 3.7: Procedura di gestione dell'arrivo delle notifiche.

1. Controlliamo che lo stesso robot non abbia già una richiesta pendente dello stesso tipo. In caso positivo si aggregano i messaggi e si riavvia il timer per l'archiviazione automatica del messaggio.
2. Se non ci sono messaggi da aggregare, viene calcolata la priorità della notifica ricevuta in funzione del workload, delle preferenze dell'operatore, dell'affidabilità del modulo e della confidenza della notifica. In particolare si considerano i seguenti contributi:

$$\text{workload}^{-1} = 100 - \frac{100}{1 + Ke^{-x}}$$

Dove x è il numero di messaggi che sono correntemente visualizzati all'operatore. Il parametro K è stato imposto pari a 100 per ottenere il grafico in Figura 3.8, una funzione sigmoidale che decade velocemente superati i quattro messaggi visualizzati contemporaneamente. La scelta è dettata dal fatto che già un numero di messaggi pari a quattro appesantiscono la gestione delle notifiche per l'operatore. Come si può notare il valore della variabile è tanto più elevato quanto più il workload dovuto alle notifiche è basso.

$$\text{affidabilità} = \frac{100}{1 + e^{-\frac{z}{H}}}$$

Dove $H = 4$, per ottenere la curva in Figura 3.9. La variabile z è il *bilancio di affidabilità* del modulo: come si può vedere in Figura 3.10, l'operatore può archiviare una notifica positivamente, incrementando di una unità il bilancio di affidabilità del modulo oppure archiviarlo negativamente, decrementandolo dello stesso valore. All'inizio della missione il bilancio di affidabilità viene posto a zero per ogni modulo, per essere influenzato da subito dalle interazioni dell'operatore. L'affidabilità viene calcolata per ogni tipo di messaggio diverso, fra cui messaggi di rilevamento vittime, di errore, di feedback o di estrazione di informazione semantica.
Il valore complessivo è dato infine da:

$$\text{priorità} = \text{confidenza}\frac{\text{affidabilità} + \text{workload}^{-1} + \text{rilevanza}}{300}$$

Il parametro *rilevanza* viene impostato dall'operatore dinamicamente durante la missione ed è relativo al singolo tipo di messaggio, proprio come l'affidabilità. I valori iniziali sono di 100 per le indicazioni sulle vittime, 80 per gli errori in genere, 50 per i feedback e 10 per i messaggi sulla semantica dell'ambiente. Ovviamente questi valori devono essere

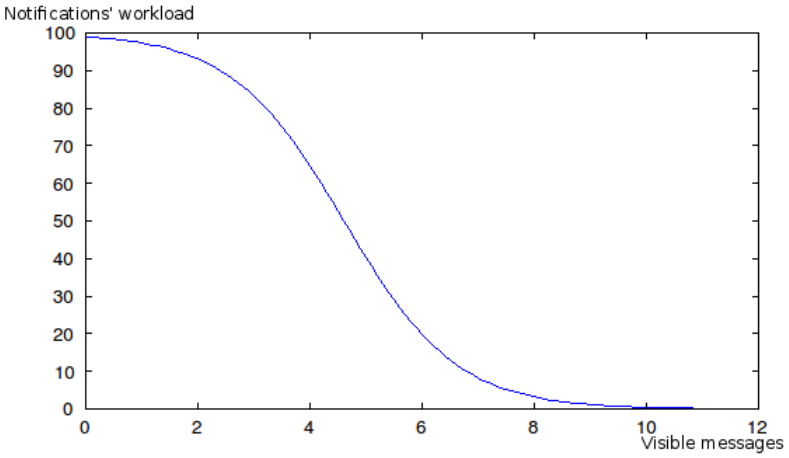

Figura 3.8: Funzione per il calcolo dello stato del workload relativo al sistema di notifica ed usato nel computo della priorità dei messaggi. Sulle ascisse il numero di messaggi contemporaneamente visualizzati sulla GUI, sulle ordinate il valore risultante da usare nel calcolo della priorità.

adattati alle caratteristiche del sistema multirobot controllato ed al tipo di missione e ambiente affrontati. Infine il valore di confidenza della notifica viene calcolato dal modulo che la manda all'interno del singolo robot ed è un valore compreso tra 0 e 100, così come il valore di priorità finale ottenuto.

3. Si confronta la priorità della notifica con una soglia minima. Se la priorità non supera tale soglia, la notifica viene scartata. Il valore di default per la soglia di priorità è dieci, in modo da eliminare inizialmente solo i messaggi palesemente irrilevanti. L'operatore può modificare il valore di tale soglia durante le missioni per ottimizzare le prestazioni del filtraggio e nel contempo evitare di perdere notifiche utili.
4. Si aggiorna l'attuale livello di workload specifico e si avvia il timer per l'archiviazione automatica.

Quando le notifiche non sono più utili all'azione di controllo dell'operatore devono essere archiviate. Nella Figura 3.10 vengono rappresentate le diverse possibili situazioni in cui viene archiviato un messaggio:

- Alla scadenza del tempo per la gestione: come abbiamo visto, la rilevanza di una notifica decade al passare del tempo. Notifiche passate sono di scarso interesse e possono anche confondere l'operatore nella gestione del sistema. Per tale motivo il notification manager archivia automaticamente le notifiche dopo un certo intervallo di tempo (25 secondi). L'archiviazione non ha effetti sulla determinazione dell'af-

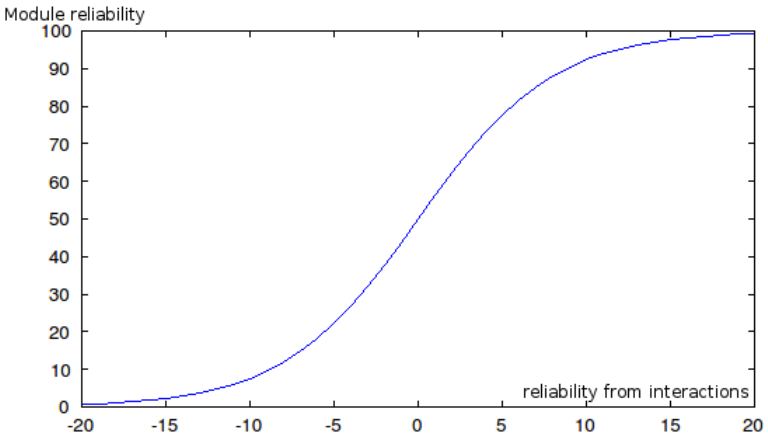

Figura 3.9: Funzione per il calcolo del parametro relativo all'affidabilità del modulo che invia la notifica, usato nel computo della priorità dei messaggi. Sulle ascisse il punteggio di affidabilità assegnato al modulo durante l'interazione dell'operatore col sistema. Sulle ordinate il valore calcolato di affidabilità del modulo in esame.

fidabilità del modulo che ha sollevato la notifica, per questo motivo diciamo che si tratta di una archiviazione *neutra*. I messaggi relativi alle vittime non vengono mai archiviati per eventi temporali e devono essere gestiti dall'operatore;

- Quando l'operatore seleziona un robot, quindi si focalizza sulla sua gestione, tutte le notifiche da esso sollevate vengono archiviate, fatta eccezione per quelle relative alle vittime trovate. Anche in questo caso l'archiviazione è neutra;
- L'operatore può selezionare un messaggio per focalizzare il sistema sul suo contenuto: la mappa viene centrata sul robot che ha sollevato la notifica ed esso viene selezionato. Nel caso di una possibile vittima individuata il sistema mostra una serie di immagini relative alla vittima, che l'operatore può esaminare prima di confermarne la presenza. Questa operazione non implica l'archiviazione del messaggio;
- L'operatore archivia esplicitamente il messaggio come *positivo*. Il sistema si focalizza sul robot che ha sollevato la notifica e l'affidabilità del modulo viene incrementata;
- L'operatore archivia il messaggio come *negativo*. Il sistema non si focalizza sul robot. Il messaggio viene cancellato e l'affidabilità del modulo che l'ha sollevato viene ridotta.

Grazie ad una migliore gestione delle notifiche, adattabile automaticamente e aggiustabile dall'operatore dinamicamente, abbiamo reso possibile un'inte-

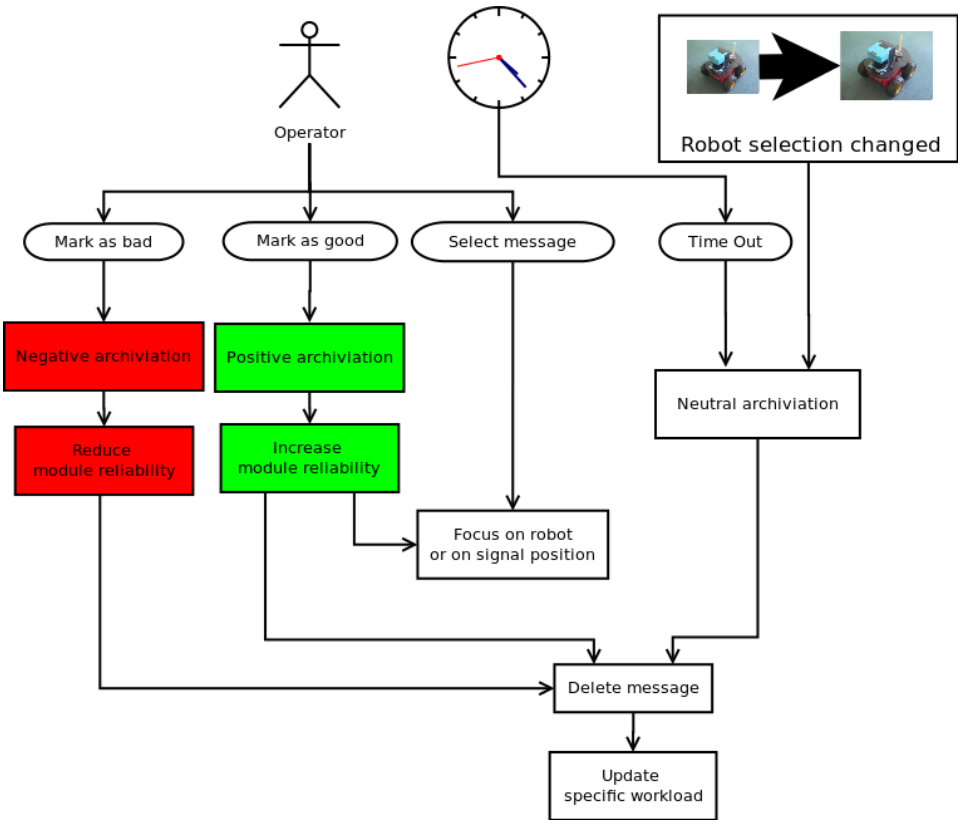

Figura 3.10: Procedure di archiviazione delle notifiche. Possono essere avviate dall'operatore, da eventi temporali o da selezioni di robot durante le normali operazioni di gestione del sistema.

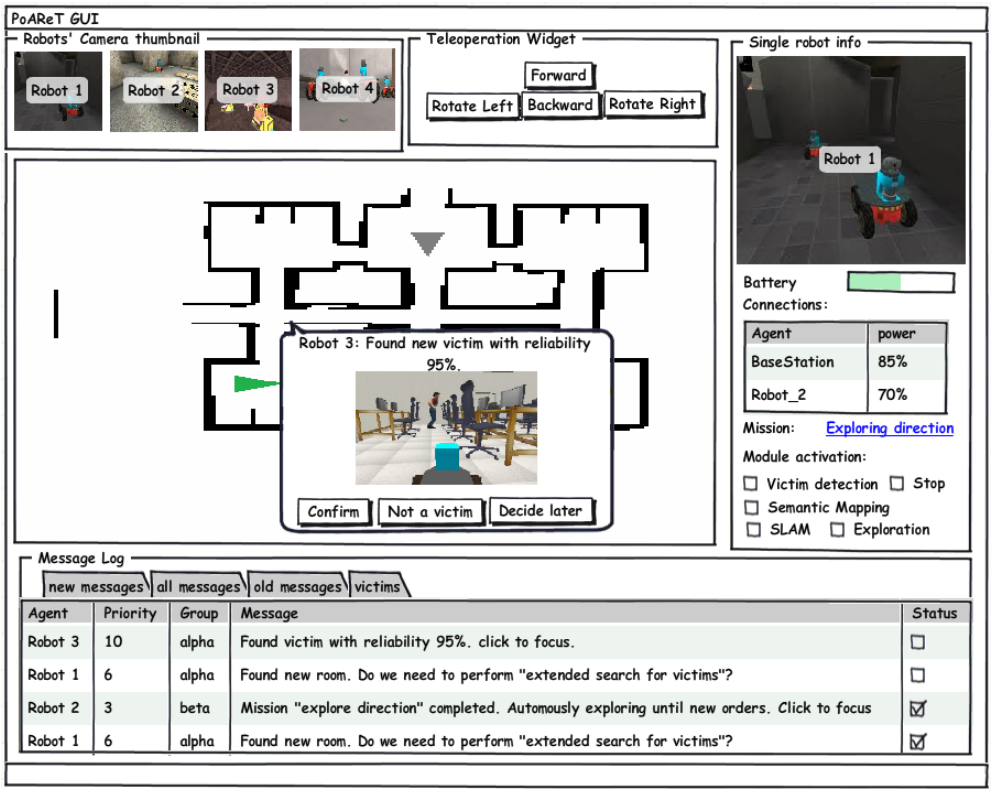

Figura 3.11: Mockup della GUI del sistema da noi implementato, considerante tutti gli aspetti discussi nella Sezione 3.1.2 e le innovazioni presentate nelle Sezioni 3.4 e 3.5.

razione più consapevole e proficua dell'operatore con il sistema. L'operatore può concentrarsi sulla sua strategia di alto livello e gestire le eccezioni quando vengono sollevate, riducendo il tempo di inattività dei robot ed evitando controlli ciclici alla ricerca di eventuali problemi.

3.6 L'interfaccia utente

L'interfaccia grafica utente realizzata in questa tesi tiene conto delle considerazioni fatte per i sistemi multirobot nella Sezione 3.1.2, e implementa le due innovazioni descritte in questo capitolo: HLC e NFS.
L'interfaccia che abbiamo creato è modulare, sia dal punto di vista grafico, rendendola pienamente configurabile dall'operatore, sia dal punto di vista logico, favorendone la scalabilità a livello progettuale. Analizziamo meglio l'architettura del sistema nel Capitolo 4.
Un mockup del progetto concettuale dell'interfaccia grafica è visibile in Figura 3.11. Notevole importanza è stata data alla collocazione dei componenti

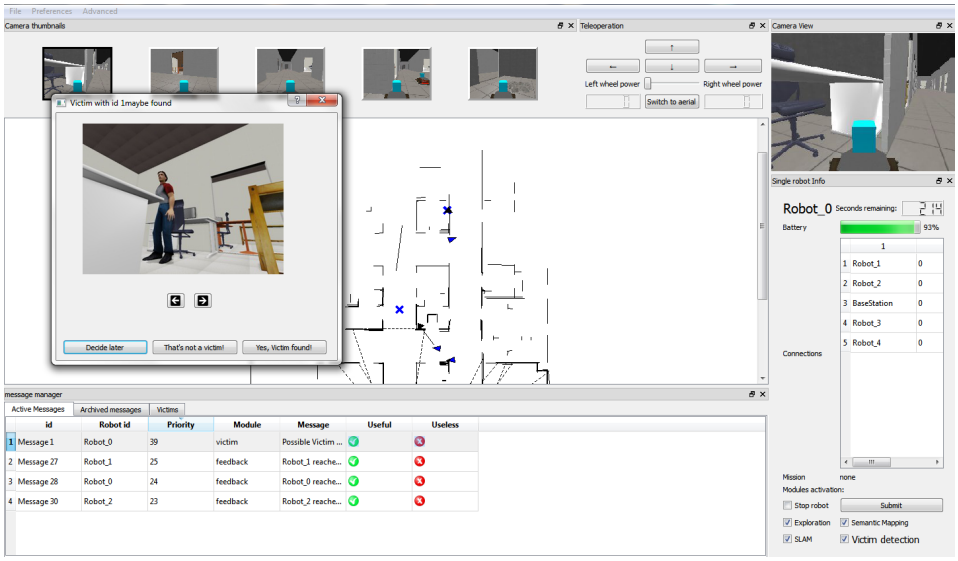

Figura 3.12: Interfaccia utente del sistema PoAReT, così come da noi impiegata nella RoboCup Rescue Simulation League 2012.

dell'interfaccia. Il ruolo centrale è riservato alla mappa, tramite la quale si interagisce con i robot, sia assegnando i waypoint da raggiungere, sia specificando gli HLC che i robot dovranno adempiere in maniera autonoma. Le informazioni sui singoli robot vengono visualizzate su una banda a destra dell'interfaccia, integrandole il più possibile. Sono visualizzate in particolare informazioni sulla connettività, sulla missione attuale e sullo stato della batteria. Le telecamere vengono gestite da un altro modulo, aggregando i flussi video dei robot non selezionati in piccoli riquadri aggregati (chiamati *thumbnails*), mentre il flusso video del robot selezionato ha dimensioni maggiori, maggior risoluzione e framerate più elevato, per garantire prestazioni ottime durante le azioni di tele-operazione. La tele-operazione viene eseguita tramite un apposito componente grafico, oppure tramite comandi a tastiera. Infine, vengono presentate le informazioni relative alle notifiche dei robot, integrate, filtrate ed aggregate secondo i principi del NFS descritti nella Sezione 3.5. Tali informazioni sono mantenute anche per i messaggi archiviati, rendendo disponibile uno storico dei messaggi ricevuti. Le informazioni sulle vittime sono raccolte in un'ulteriore componente per facilitarne l'estrazione dopo la fase di ricerca e per procedere a quella di recupero.
Nella Figura 3.12 una schermata della GUI realizzata seguendo i principi finora discussi.

3.7 Sommario

Abbiamo descritto in questo capitolo l'obiettivo di ricerca e le principali misure adottate in letteratura per misurare la bontà di un sistema multirobot. Abbiamo quindi analizzato i principali problemi nell'adozione di un sistema multirobot per missioni USAR: il workload cui è sottoposto l'operatore e la sua situation awareness, entrambi critici all'aumentare del numero di robot. Abbiamo descritto come abbiamo impostato la soluzione da un punto di vista tecnico, con il dichiarato scopo di ridurre al minimo tali problematiche e porci in linea con lo stato dell'arte sulle tematiche affrontate. Abbiamo quindi analizzato le due fondamentali innovazioni introdotte in questa tesi, volte a migliorare le procedure d'interazione sia da sistema verso operatore sia viceversa: NFS e HLC. Nel prossimo capitolo descriviamo l'architettura complessiva del sistema, e in particolare della stazione di controllo e della GUI annessa implementate per intero in questo lavoro di tesi.

Capitolo 4

Architettura del sistema

In questo capitolo presentiamo l'architettura del sistema PoAReT (Politecnico di Milano Autonomous Robotic Rescue Team), vincitore della RoboCup Rescue Simulation League 2012 a Città del Messico [18].
Al sistema nel suo complesso hanno lavorato il professore F. Amigoni e sei studenti della laurea specialistica in ingegneria informatica del Politecnico di Milano. Con PoAReT siamo in grado di gestire tre diversi tipi di robot, simulati in USARSim e più di dieci robot contemporaneamente attivi. Ogni robot è dotato di procedure autonome di esplorazione e coordinamento, di capacità di mappatura e di riconoscimento di vittime, nonché dei sensori che sono indispensabili per la sua operatività in un ambiente sconosciuto (almeno inizialmente): telecamera, range scanner, IMU (Inertial Measurement Unit), sensori odometrici, sonar di prossimità.
Il progetto è stato realizzato grazie al contributo della Fondazione Banca del Monte di Lombardia e dell'azienda QuitAndMove, che ci hanno permesso di presentare i risultati nella competizione RoboCup Rescue Simulation League [58].
Le innovazioni introdotte in questo lavoro di tesi, descritte nel Capitolo 3, sono state implementate ed utilizzate in PoAReT, integrate in una GUI che tiene conto dello stato dell'arte in fatto di sistemi multirobot per missioni di USAR. Durante questo lavoro di tesi è stata sviluppata nella sua interezza la stazione di comando dei robot e la GUI con la quale l'operatore controlla il sistema.

4.1 Sistema PoAReT

L'intero sistema PoAReT è modulare. Ogni modulo è un componente a "tenuta stagna", con un'interfaccia ben definita con il resto del sistema. Questo

approccio garantisce due principali vantaggi: è più semplice modificare un modulo, sostituirlo o eliminarlo, rendendo tali modifiche trasparenti al resto del sistema pur di attenersi alle interfacce specificate, e, inoltre, la realizzazione del sistema è più semplice, poiché viene snellito il coordinamento fra i responsabili dei vari moduli.
Due sono le principali parti del sistema: la *base station*, ovvero la stazione di comando da cui l'operatore controlla il sistema, e i *singoli robot*. Ogni robot e la base station è gestito in un processo indipendente. I diversi processi possono quindi comunicare solamente attraverso una rete wireless simulata, che si appoggia ai modelli di USARSim per valutare la connettività fra gli agenti. Tale rete, chiamata WSS [59], può funzionare secondo diversi modelli, dal più semplice che garantisce sempre la piena connettività fra agenti, al più complesso e realistico, che considera la dispersione del segnale wireless e l'attenuazione provocata dagli ostacoli che attraversa. Abbiamo implementato un sistema di routing basato sul protocollo *distance vector* [60] in modo da permettere all'operatore d'interagire con il sistema il più a lungo e il più affidabilmente possibile: i robot possono infatti assumere il ruolo di relè di comunicazione per trasmettere le informazioni da un robot ad un altro non direttamente collegato.
L'affidabilità e la robustezza del sistema sono fattori fondamentali. La suddetta suddivisione in moduli ha favorito la fase di test dell'applicazione, riducendo i tempi di debugging, ed allo stesso tempo ha permesso di realizzare procedure di recupero dell'operatività in caso di problemi durante le missioni. A titolo di esempio, grazie alla suddivisione in processi ed al protocollo di routing dinamico implementato, in caso di problemi alla base station che ne causino un arresto i singoli robot possono continuare ad operare in autonomia, ovviamente senza l'ausilio dell'operatore. Una volta risolti i problemi alla base station e riavviato l'applicativo, questa si collega di nuovo ai robot, ne riceve gli aggiornamenti e li visualizza nella GUI dell'operatore, ristabilendo la situazione com'era prima dei problemi. Questo tipo di procedure di recupero dell'operatività in breve tempo e sul campo sono ancor più rilevanti in ambito USAR, e il nostro sistema si è comportato egregiamente quando è stato messo alla prova su tali aspetti durante la competizione RoboCup Rescue Simulation League 2012.
Altro aspetto interessante del sistema PoAReT è la possibilità di eseguirlo su macchine diverse. La struttura modulare e la separazione in processi diversi per ogni robot e la base station permettono infatti di avere dell'hardware specifico per ogni singolo processo e semplificano l'adattabilità del sistema da un'applicazione basata sul simulatore USARSim a un'applicazione con robot reali, richiedendo solamente l'introduzione di un layer software per

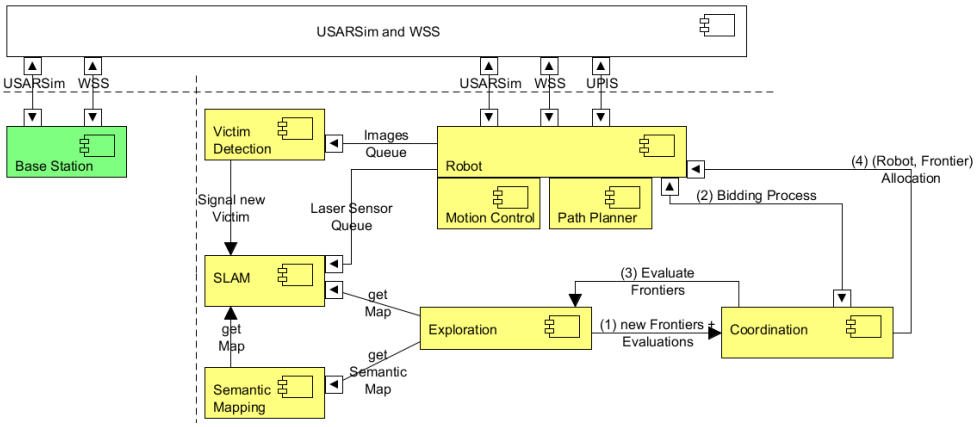

Figura 4.1: Struttura modulare del sistema PoAReT. Il processo base station è indipendente da quelli dei singoli robot. Da notare la suddivisione in moduli che garantisce semplicità di realizzazione, robustezza e facilita le procedure di debugging. Le frecce che connettono i moduli rappresentano le interazioni fra essi. Questo schema è stato presentato in [61].

interfacciarsi all'hardware specifico dei robot.
Forniamo una panoramica generale del sistema PoAReT, così come è stato presentato alla RoboCup Rescue Simulation League 2012, con la sua struttura modulare e la netta separazione in processi diversi fra robot e base station nella Figura 4.1. Viene evidenziata la separazione fra il processo base station e i singoli robot, la cui comunicazione avviene tramite WSS. Nella figura vengono mostrate anche le interazioni più rilevanti fra i diversi moduli all'interno di ogni robot, interazioni che evidenzieremo nel seguito durante la descrizione di ognuno dei moduli principali.
Entriamo nel dettaglio delle singole parti del sistema, in particolare i singoli robot e la base station, descrivendo brevemente le funzionalità di ogni modulo per fornire una chiara idea delle caratteristiche del sistema.

4.1.1 Robot mobili

PoAReT è in grado di gestire tre tipi diversi di robot. Il più comune è un robot di tipo Pioneer, nella fattispecie il modello *P3AT* [62] [63]. Si tratta di un semplice robot con ruote gommate con attuazione differenziale, ossia in cui l'asse destro e sinistro possono essere attuati a velocità differenti permettendo di curvare senza un vero e proprio sterzo, tecnica detta anche *simply tank style* o *skid steering* in quanto usata da cingolati e carrelli elevatori e basata sulle dinamiche di scivolamento degli pneumatici o cingoli. Il P3AT è idoneo all'esplorazione di ambienti pericolosi ma in cui non vi siano

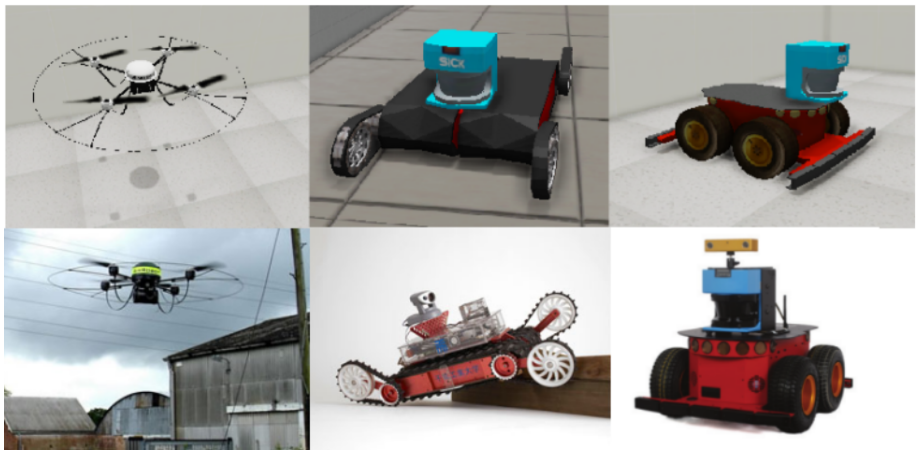

Figura 4.2: Robot supportati dal sistema PoAReT. Sopra le versioni simulate in USARSim, sotto le immagini dei robot reali, dai rispettivi siti ufficiali [62] [64] [66].

gravi difficoltà nella mobilità. Il secondo veicolo utilizzabile è chiamato *Kenaf* [64]: si tratta di un robot cingolato in grado di superare agevolmente gli ostacoli e navigare anche in ambienti in cui la mobilità è difficoltosa. Ultimo tipo di robot gestito è l'*AirRobot* [65] [66], un robot volante molto stabile ma ingombrante, utilizzato per missioni di ricognizione in ambito USAR. Nella Figura 4.2 sono mostrati i tre robot utilizzati, sia nella loro versione reale che in quella simulata. Come già sottolineato, ogni robot è gestito da un processo separato, per garantire la robustezza a guasti nei singoli robot, e le comunicazioni fra robot e con la base station avvengono attraverso la rete wireless WSS. Oltre a garantire robustezza ed affidabilità al sistema, questo approccio permette di esplorare anche aree ove non è disponibile la connettività con la base station: i robot, essendo logicamente separati dal resto del sistema, possono restare isolati ed esplorare autonomamente zone altrimenti irraggiungibili per poi tornare connessi e comunicare le informazioni raccolte (la mappa dell'ambiente, eventuali elementi d'interesse, vittime o pericoli). Le funzioni autonome di cui dispone ogni singolo robot per l'esplorazione dell'ambiente, la mappatura e il riconoscimento delle vittime vengono utilizzate anche quando il robot non è isolato dal sistema per permettere ad un solo operatore di gestire un maggior numero di robot contemporaneamente.

Ogni robot dispone di diversi moduli che vengono collegati al nucleo del suo processo permettendogli di sfruttare funzionalità avanzate. Oltre ai moduli base, indispensabili per la comunicazione wireless e con il simulatore, che sono strettamente collegati a tali strumenti e, quindi, sono di scarso interesse per lo scopo di questa tesi, ogni singolo robot contiene diversi moduli

con funzioni logiche ognuna innovativa nella sua area. Di seguito una breve descrizione dei principali moduli.

SLAM - Simultaneous Localization And Mapping

Il modulo di SLAM permette di mappare l'ambiente in cui si trova il robot e contemporaneamente localizzare il robot all'interno di tale ambiente. Questo componente è alla base della navigazione nelle missioni di USAR e non solo, in quanto permette al robot di orientarsi e poter quindi intraprendere una politica di esplorazione autonoma efficiente, nonché di evitare eventuali ostacoli o muri.
La caratteristica principale del modulo di SLAM del nostro sistema PoAReT è la struttura dati che utilizza: esso tiene conto delle caratteristiche dell'ambiente usando dei segmenti di retta per rappresentarle, una struttura dati molto poco ingombrante e idonea all'utilizzo in ambienti urbani e indoor che per loro natura sono perlopiù composti da muri o oggetti rettilinei. Si accantona così la tradizionale mappa a griglia creata a partire dai punti ricavati dal telemetro laser, molto ingombrante in termini di occupazione di memoria. Si garantisce così snellezza nel salvataggio della mappa, che quindi occupa molto meno spazio in memoria, nella sua visualizzazione all'utente e nella trasmissione della mappa fra robot e/o base station.
Questo modulo sfrutta come sensori di riferimento il *Range Scanner*, un sensore laser di distanza con visuale di 360° posizionato sopra i robot terrestri (P3AT e Kenaf) e un'unità IMU, che permette di avere un feedback sui movimenti del robot. In alternativa alla IMU, è possibile utilizzare un odometro per ricavare i dati dell'odometria del robot, ma le prestazioni sono inferiori.
Nella Figura 4.3 è riportato un esempio di mappa creata dal modulo di SLAM del sistema PoAReT, dove le linee rappresentano gli ostacoli o i muri incontrati. La figura mette in evidenza l'idoneità della struttura dati utilizzata per rappresentare gli ostacoli, date le caratteristiche dell'ambiente.

Path planner

Il modulo di path planner calcola il percorso che un robot deve seguire per raggiungere una destinazione senza urtare ostacoli o muri. Questo modulo è utilizzato sia nell'esplorazione autonoma sia per i comandi a punto di destinazione, situazioni in cui il robot deve autonomamente calcolare il percorso da seguire fino alla destinazione imposta rispettivamente dall'esplorazione autonoma o dall'operatore.

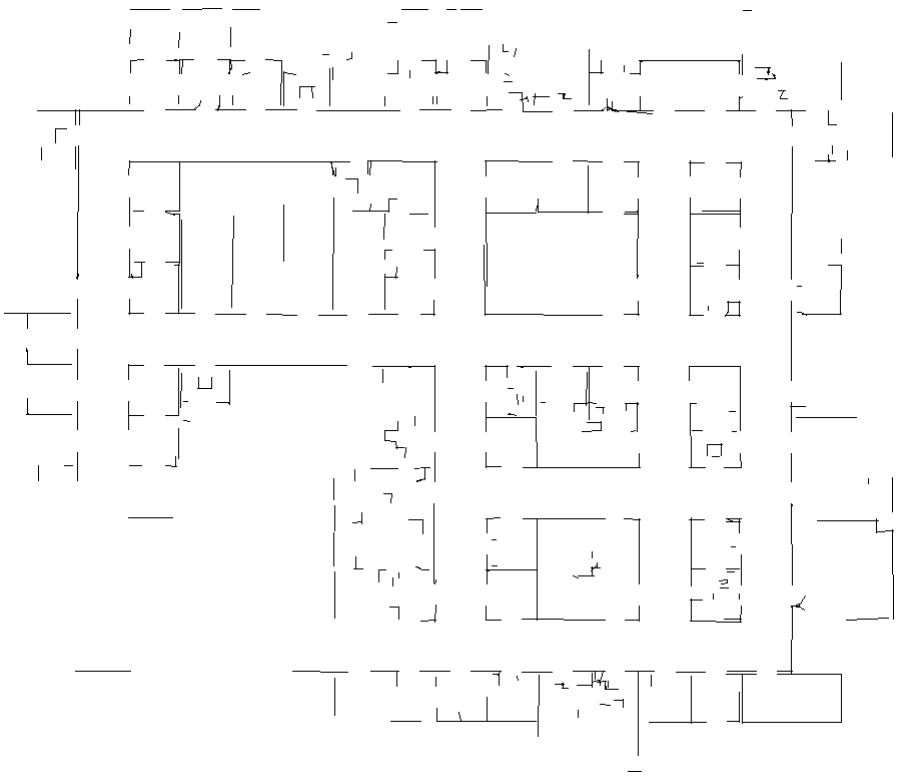

Figura 4.3: Esempio di mappa realizzata dal modulo di SLAM del sistema PoAReT. Ogni linea rappresenta un ostacolo o un muro. Questa mappa in particolare è stata estratta da una delle arene di prova usata nei test sperimentali.

Il path planner deve essere in grado di affrontare almeno due possibili situazioni:

1. Il robot deve recarsi in una zona già visitata. Si utilizza in questo caso l'algoritmo A* su uno spazio degli stati con nodi le posizioni passate del robot e lati i percorsi di visibilità semplici: due nodo sono connessi da un lato se, con solo una rotazione ed una traslazione, dalla posizione del primo si giunge a quella del secondo;
2. Il robot deve raggiungere un punto non ancora mappato. In questo caso si utilizza RRT (Rapidly exploring random tree [67]) per esplorare la parte sconosciuta del percorso.

Il path planner è necessario anche in azioni di copertura di una zona, ad esempio per la ricerca approfondita di elementi d'interesse o vittime.

Motion control

Il modulo di motion control trasmette i comandi di attuazione ai robot per compiere effettivamente il movimento richiesto. Una volta stabilito il modello di moto dei vari robot, l'implementazione del modulo è stata banale.

Semantic mapping

Durante l'esplorazione di ambienti pericolosi, le squadre di soccorso umane sono in grado di cogliere elementi semantici importanti da ciò che vedono e dalla struttura stessa dell'ambiente. Ad esempio, una squadra di soccorso che entra in un hotel e si trova in un ambiente stretto e lungo può dedurre che si stia muovendo lungo un corridoio, e di conseguenza può ragionevolmente essere sicura che lungo i lati del corridoio ci saranno delle stanze con probabili vittime. Questa informazione non viene estratta da un sistema multirobot per lo USAR tradizionale e deve essere aggiunta dall'operatore umano. Un componente che gestisca tali informazioni semantiche, le estragga dai dati a disposizione e le integri ad altre aggiunte dall'operatore per facilitare le operazioni di gestione dei robot può portare a un uso molto efficace del sistema nelle operazioni USAR. In ambiti indoor, il modulo di semantic mapping è in grado di identificare le singole stanze, classificarle a seconda della loro struttura e dimensione e fare previsioni sulle stanze circostanti, ad esempio sul tipo di stanze e sul loro numero. Il sistema autonomo di coordinamento dei robot e quello di esplorazione possono sfruttare queste informazioni senza la necessità d'intervento dell'operatore, rendendo l'esplorazione autonoma più efficace e simile a quella di una squadra di soccorritori umani.

Ad esempio, con riferimento al caso precedente sull'hotel: il modulo di semantic mapping può identificare un ambiente come corridoio, trasmettere questa informazione al modulo di coordinamento che, consapevole delle implicazioni di tale informazione, può decidere di mandare un numero maggiore di robot lungo il corridoio in previsione delle stanze che saranno accessibili lungo di esso. Questa politica non sarebbe adottata da un sistema autonomo senza le informazioni semantiche, che avrebbe ragionevolmente mandato meno robot lungo il corridoio in quanto sarebbe stato interpretato come un qualsiasi ambiente angusto e non come una via d'accesso ad ambienti ampi e con probabili vittime.
Il modulo di semantic mapping è anche in grado di riconoscere le porte delle stanze, permettendo al path planner di creare dei percorsi più precisi per attraversarle, evitando così molti incidenti di navigazione.

Exploration

Il compito del modulo di exploration è stabilire la bontà delle possibili destinazioni del robot con lo scopo ultimo di coprire la maggior area interessante (ossia con possibili vittime) possibile. Le destinazioni fra cui scegliere vengono chiamate *frontiere* in quanto dividono l'ambiente noto e mappato da quello ignoto da esplorare. Queste frontiere vengono determinate dal modulo di SLAM e mandate al modulo di exploration, che si occupa di valutarle assegnandovi un punteggio, in funzione di:

- Distanza dalla frontiera: minore la distanza, migliore la frontiera;
- Information gain: guadagno di informazione atteso da una percezione alla data frontiera. Una frontiera più grande presuppone generalmente un guadagno maggiore;
- Informazioni semantiche a disposizione del robot che rendono più appetibile una destinazione piuttosto di un'altra;
- Indicazioni dell'utente tramite gli HLC, discussi nel Capitolo 3. L'operatore fornisce delle indicazioni ai robot, che le introducono nelle procedure autonome d'esplorazione;
- Livello di batteria: robot con poca carica residua nella batteria rischiano di non riuscire a portare a termine missioni lunghe e rimanere bloccati distanti dalla base station, rischiando di non essere più recuperabili. Robot con livello di batteria basso dovrebbero ragionevolmente tornare alla base station o operare in sua prossimità;
- Connettività con altri robot e/o con la base station: un robot con un livello di connettività criticamente basso con la base station rischia di isolarsi, non ricevendo più indicazioni dall'operatore. Quando possi-

bile si tenta di evitare la disconnessione per rimanere a disposizione dell'operatore.

I suddetti criteri vengono combinati dal modulo di exploration sfruttando il framework MCDM [57]. I pesi dei diversi criteri sono stati assegnati per avere prestazioni adeguate e tenere in forte considerazione le indicazioni dell'operatore, garantendo così un buon livello di controllo durante gli HLC.

Coordination

Il modulo di coordination assegna le destinazioni, valutate dal modulo di exploration, ai robot con un meccanismo market-based: esso gestisce delle aste alle quali ogni robot invia offerte per le proprie destinazioni preferite, secondo le valutazioni date dal modulo di exploration [68]. Quindi decide le destinazioni effettive dei robot massimizzando l'utilità del sistema nel complesso, ad esempio, massimizzando l'area potenzialmente esplorabile con una data allocazione di destinazioni ai robot.
L'utilizzo delle aste rende il sistema robusto ai problemi di connettività e malfunzionamenti dei robot.
Nella Figura 4.4 mostriamo una successione di immagini relative all'esplorazione autonoma di un ambiente all'interno di un edificio, senza alcun intervento dell'operatore. Si possono notare sia le dinamiche di coordinamento, che fanno sì che i robot si rechino in destinazioni diverse per massimizzare l'area esplorata, sia quelle di esplorazione, che valutano le frontiere da esplorare in funzione degli aspetti già citati. Altro aspetto interessante è l'apporto del semantic mapping nella procedura di navigazione per superare le porte, procedura altrimenti costosa e complessa. Nella figura sono rappresentate le traiettorie seguite da due robot nell'esplorazione dell'ambiente in rosso e in blu, mentre le linee nere sono ostacoli o muri. I triangoli, rispettivamente rosso e blu, sono le posizioni correnti dei robot.

Victim detection

Il modulo di victim detection è in grado d'individuare le vittime in immagini catturate dalla telecamera. Il processo d'individuazione delle vittime si compone di due fasi:

1. Skin detection: tramite lo spazio di colori HSV (Hue, Saturation and Value) si esamina l'immagine per determinare delle porzioni contenenti tonalità assimilabili a pelle umana; se rilevate porzioni di possibile pelle si procede al punto 2, altrimenti si scarta l'immagine e si analizza la successiva;

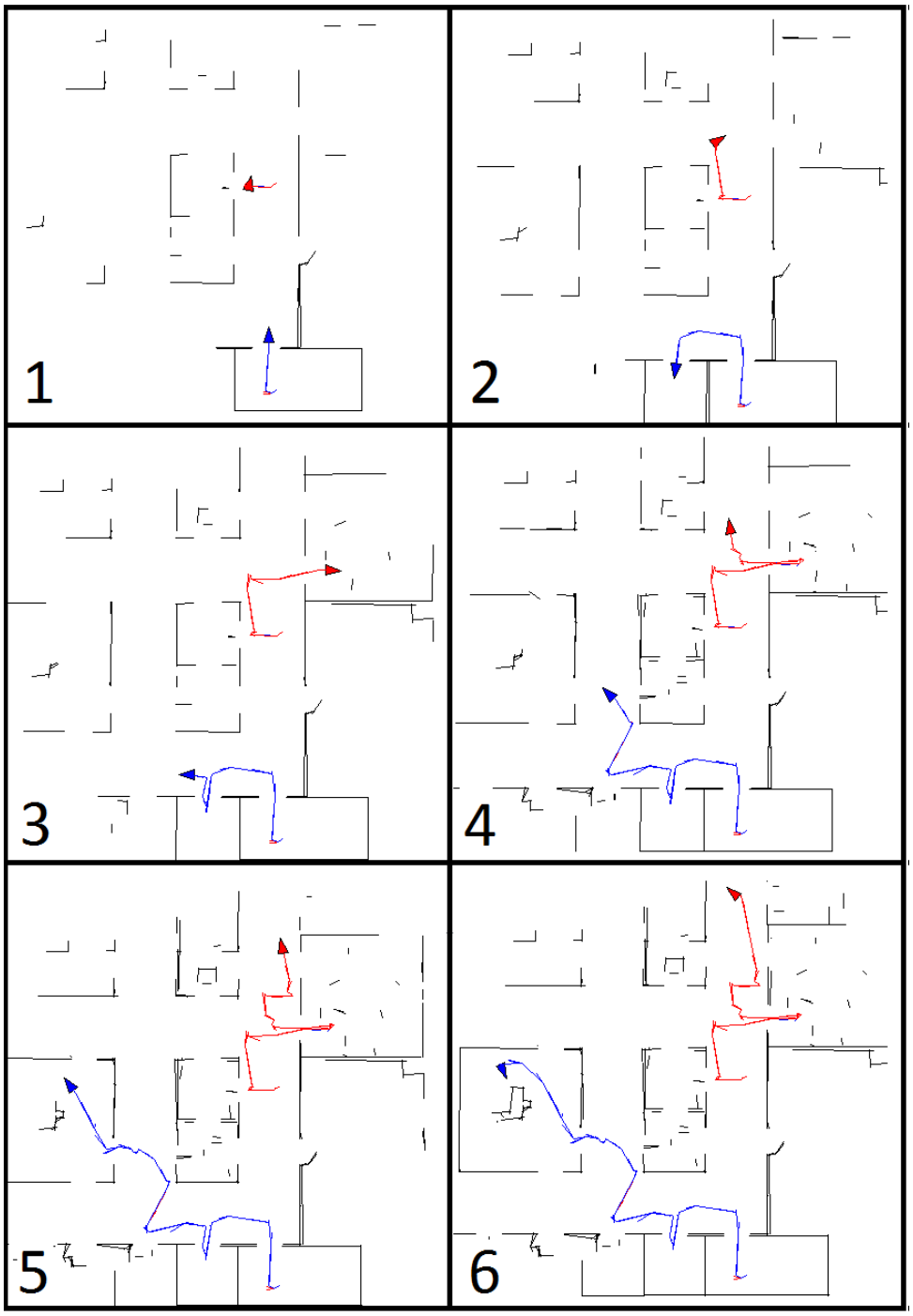

Figura 4.4: Esempio di esplorazione autonoma all'interno di un edificio ad opera di due robot col sistema PoAReT.

Figura 4.5: Vittima rilevata dal modulo di victim detection di PoAReT. A sinistra l'immagine catturata dalla telecamera. A destra il risultato della fase di skin detection dove le zone bianche sono state classificate come pelle.

2. Viene usato l'algoritmo Viola-Jones [69], metodo di analisi delle immagini ben noto ed utilizzato da numerosi team nella competizione RoboCup Rescue Simulation League, per verificare l'effettiva presenza di una vittima in base alla forma delle porzioni di pelle individuate al punto 1; si ottiene in uscita da questa fase un livello di confidenza sulla presenza o meno di vittime nell'immagine.

Quando viene individuata una possibile vittima con confidenza superiore ad una soglia, viene notificata la sua presenza all'operatore che si occupa di verificare l'effettiva correttezza della segnalazione e la colloca con precisione sulla mappa a partire dalla posizione del robot al momento della sua individuazione. In Figura 4.5 è mostrata un'immagine in cui è stata individuata una porzione di pelle dal modulo di victim detection.

4.1.2 Base station

La base station è la struttura attraverso cui i robot e l'operatore comunicano. La GUI è parte della base station, e la logica delle due è integrata per fornire all'operatore una visione quanto più possibile accurata ed aggiornata del sistema. Sia la base station che la GUI sono state interamente progettate e implementate durante questo lavoro di tesi.

Ogni robot comunica alla base station tutte le informazioni utili all'operatore ed al team di soccorso: la mappa dell'ambiente costruita fino a quel momento, le notifiche, l'eventuale presenza di vittime, il flusso video della camera in dotazione, la propria posizione e la destinazione futura. Ogni informazione viene trasmessa alla base station a intervalli di tempo differenti e regolati dinamicamente a seconda della natura dell'informazione, della sua occupazione di banda e della sua importanza per l'operatore. Ad esempio,

un robot selezionato dall'operatore, ossia su cui sta concentrando i suoi sforzi, probabilmente per tele-operarlo, trasmette alla base station il flusso video ad una frequenza di 6 immagini/secondo per evitare effetti visivi fastidiosi per l'operatore e migliorarne le prestazioni [4] [70]. Quando non è selezionato, invece, il robot manda il flusso video a 2 immagini/secondo, per ridurre il consumo di banda. Anche le mappe vengono inviate ciclicamente, mentre altre informazioni, quali ad esempio le notifiche, sono inviate al verificarsi dell'evento scatenante.
Anche la base station, così come i singoli robot, è costruita su una struttura modulare. Oltre ai moduli indispensabili per la gestione della comunicazione, presenti ovviamente anche nei singoli robot, la base station dispone di un insieme di moduli necessari per realizzare le funzioni logiche collegate alla GUI. Nella Figura 4.6 una panoramica della struttura della base station: vi si può notare come sia modulare la struttura, con elementi tra loro isolati e comunicanti solo con la parte centrale, la base station core, che realizza l'integrazione delle componenti e gestisce gli eventi, collegandoli ai metodi che se ne devono occupare.
L'interfaccia di comunicazione fra i singoli moduli della base station è realizzata tramite il sistema di *Signals & Slots* proposto dalle librerie QT [71] ogni classe è in grado di emettere dei segnali in corrispondenza di un evento e tali segnali possono essere connessi a metodi (chiamati slot) che li gestiscono. Questo tipo di approccio è particolarmente utile nello sviluppo di interfacce utente proprio per permetterne una gestione guidata dagli eventi. Il paradigma è molto simile ad un publish/subscribe, dove i moduli pubblicano o si iscrivono alla ricezione di eventi. Il sistema di Signals & Slots è realizzato basandosi su funzioni di C++, per cui il sistema risultante è cross piattaforma, aspetto che abbiamo potuto provare sviluppando il sistema su ben quattro sistemi operativi diversi: Windows 7, Mac OS, Linux Fedora e Debian.
Nel seguito entriamo nel dettaglio dei singoli moduli logici all'interno della base station, trascurando le loro componenti grafiche, che vengono meglio analizzate nella Sezione 4.2.

Map manager

Il map manager gestisce le mappe dei singoli robot. Ogni robot manda ciclicamente la mappa dell'ambiente da lui esplorato alla base station che inoltra le mappa al map manager il quale integra le mappe ricevute dai vari robot per renderle disponibili all'operatore. La visualizzazione avviene tramite un widget grafico descritto nella Sezione 4.2.

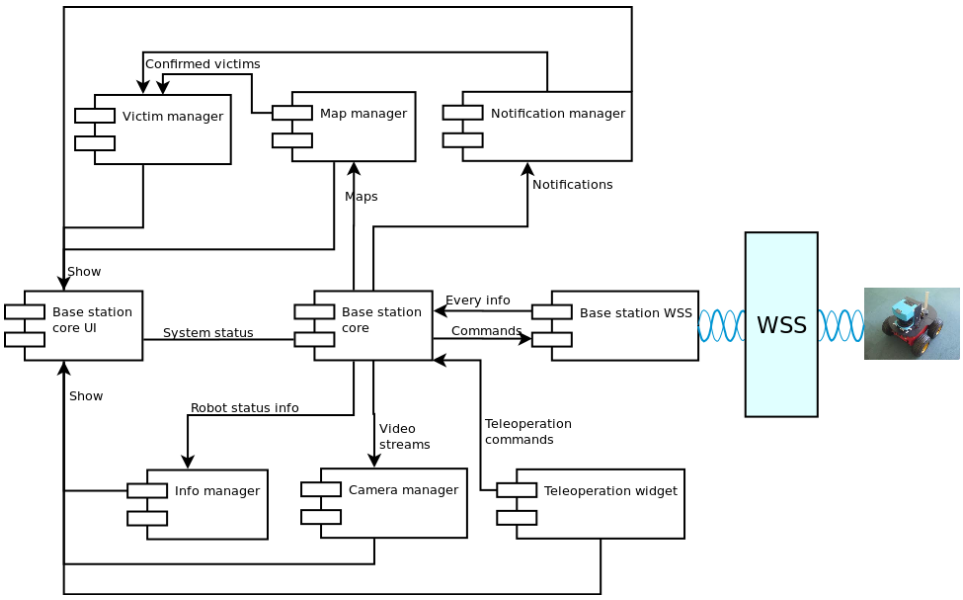

Figura 4.6: Struttura modulare della base station. Ogni modulo comunica solo con base station core, che si occupa di smistare le informazioni ricevute dai robot e mandare agli stessi robot i comandi tramite base station WSS. Base station core UI si occupa della visualizzazione grafica delle informazioni.

Il map manager si occupa anche di gestire i comandi HLC e waypoint dell'operatore. È in grado quindi di ricevere le informazioni sull'interazione con l'operatore dal widget grafico che visualizza la mappa e tradurre tale interazione in comandi da gestire e spedire ai robot. Viene mandato un feedback al widget grafico per informare l'operatore dell'esito dell'interazione e nel contempo viene inviato il comando alla base station core, che si occupa di mandarlo al robot d'interesse, ossia quello attualmente selezionato.
In Programma 4.1 mostriamo lo pseudo-codice per la gestione dell'interazione dell'operatore con la mappa. Le azioni possibili riguardano la selezione dei robot e la specifica di comandi agli stessi semplicemente interagendo con la mappa. Per brevità e semplicità non sono state riportate nello pseudo-codice tutte le possibili azioni: è possibile anche interagire con la mappa per inserire etichette informative (posizioni di vittime, pericoli o etichette personalizzate), per scorrere la mappa, zoomare su una posizione o ridurre il livello di zoom.

Camera manager

Ogni robot manda il flusso video della sua telecamera alla base station per permettere all'operatore di osservare l'ambiente in cui si trovano i robot. Il

```
Gestione interazione dell'operatore:
robot selezionato = ROBOT;
operatore esegue INTERACTION con la mappa nella posizione POS;
if(INTERACTION == click_destro)
    segnala comando waypoint(POS,ROBOT);
else if(INTERACTION == doppio_click_destro)
    segnala comando HLC esplora_area(POS,ROBOT);
else if(INTERACTION == drag&drop_destro){
    direzione DIR = f(POS_iniziale, POS_finale);
    segnala HLC esplora_direzione(DIR, ROBOT);
}
else if(INTERACTION == click_sinistro and POS == POS(ROBOT_W))
    segnala selezione di ROBOT_W;
```

Programma 4.1: Pseudo-codice relativo alla gestione dell'interazione dell'operatore con la mappa.

camera manager si occupa di gestire tali flussi video, associarli al robot che li trasmette, variarne le dimensioni per inserirlo in un thumbnail e gestire gli eventuali errori di trasmissione. Gestisce anche l'interazione dell'operatore con i thumbnail delle telecamere, permettendo di selezionare un robot semplicemente cliccando sull'immagine della sua telecamera.

Info manager

Ogni robot trasmette ciclicamente le informazioni di connettività, batteria e status della missione alla base station. L'info manager si occupa d'integrare tali informazioni, salvarle per la consultazione e trasmetterne una visualizzazione aggregata alla GUI.

Notification manager

Il notification manager implementa il NFS descritto nel Capitolo 3. Alla ricezione di una notifica, il notification manager avvia la procedura di gestione, eventualmente calcolando la priorità del messaggio per proporlo all'operatore. In Programma 4.2 mostriamo lo pseudo-codice relativo alla gestione dei messaggi di errore: il notification manager si occupa di aggregare i messaggi simili provenienti dallo stesso robot e di filtrarli e ordinarli a seconda della priorità stimata del messaggio. In Programma 4.3 si mostra invece lo pseudo-codice riferito alla gestione di un'interazione da parte dell'utente con un messaggio a schermo: a seconda del tipo di interazione

e del tipo di messaggio il notification manager esegue diverse azioni, comunicando a base station core le informazioni importanti per la gestione della notifica, quali ad esempio le informazioni sulle vittime confermate piuttosto che l'indicazione del robot sul quale si vuole focalizzare l'attenzione. Per approfondimenti sulle procedure di calcolo della priorità dei messaggi e per una più dettagliata analisi delle azioni intraprese dal notification manager si veda il Capitolo 3: l'implementazione del notification manager mantiene tutte le funzionalità del NFS.
Il notification manager, dunque, si occupa sia della gestione delle notifiche in arrivo e della relativa presentazione all'operatore, sia della gestione dell'interazione dell'operatore con le notifiche stesse, avviando l'archiviazione dei messaggi e notificando la selezione dei robot e il focus su una posizione nella mappa quando dettato dalle azioni dell'operatore.

```
Gestione messaggio di errore:
ricevuto messaggio di errore da robot_x;
if(robot_x ha un messaggio di errore pendente){
    unisci i due messaggi di errore;
    riavvia il timer per l'archiviazione automatica;
    return;
}
else{
    calcola la priorità per il nuovo messaggio;
    if(priorità > soglia){
        visualizza il messaggio;
        avvia il timer per l'archiviazione automatica;
        return;
    }
    else{
        scarta il messaggio;
        return;
    }
}
```

Programma 4.2: Pseudo-codice della gestione tipo di un messaggio di errore da parte del notification manager.

```
Gestione messaggio confermato:
utente interagisce con un messaggio di tipo TIPO_MEX mandato
    da ROBOT con un'interazione di tipo INTERACTION;
if(INTERACTION == conferma){
    archivia messaggio;
    diminuisci il numero di messaggi a video;
    aggiorna valore di workload;
    if(TIPO_MEX == vittima){
        aumenta affidabilità del modulo di victim detection;
        segnala la vittima a base station core;
    }
    else{
        segnala a base station core di focalizzarsi su ROBOT;
        elimina altre richieste pendenti di feedback o errore
            da parte di ROBOT;
        aumenta affidabilità di ROBOT;
    }
}
else{
    elimina messaggio a video;
    riduci affidabilità di ROBOT per TIPO_MEX;
}
return;
```

Programma 4.3: Gestione dell'interazione dell'utente con un messaggio, pseudo-codice.

Victim manager

Le vittime possono essere rilevate sia dai robot, tramite il modulo di victim detection, che dall'operatore. In ogni caso è l'operatore a confermare la presenza della vittima ed a validarne la posizione sulla mappa. Una volta confermate, le informazioni vengono inviate al victim manager, che si occupa di tenere traccia di tutte le vittime, visualizzandole in un elenco e sulla mappa e permettendo di visualizzare le immagini della vittima, per verificarne lo status. Le notifiche successive relative a possibili vittime vengono analizzate dal victim manager per verificare che non si tratti di vittime già individuate, nel qual caso la notifica viene scartata e le informazioni aggregate a quelle già disponibili per la vittima confermata. In questo modo si evita il problema di gestire manualmente notifiche multiple relative alla stessa vittima. Due vittime sono considerate la stessa se si trovano semplicemente nella stessa posizione, con un certo margine di errore. Questo non esclude il caso in cui vi sia un piccolo gruppo di vittime molto vicine, che verrebbe considerato dal sistema automatico come un'unica vittima. Tuttavia, visto lo scopo della fase di ricerca col sistema multirobot, ossia estrarre la mappa dell'ambiente e la posizione di vittime, tale comportamento è accettabile: semplicemente saranno indicate le posizioni delle vittime o di gruppi di vittime. Inoltre l'operatore all'atto della validazione dell'immagine delle vittime può manualmente inserire sulla mappa il numero di vittime presenti nella data posizione.

Base station WSS

Ogni singolo agente nel sistema PoAReT deve essere in grado di comunicare con gli altri attraverso l'infrastruttura WSS che simula la rete wireless. Per permettere tale comunicazione è stata creata un'infrastruttura di basso livello in grado di gestire la trasmissione e ricezione dei pacchetti di informazioni in maniera affidabile. Il modulo base station WSS si occupa della gestione ad alto livello delle informazioni ricevute e di quelle da trasmettere. Il modulo permette di gestire i singoli messaggi in arrivo, estrarne le informazioni rilevanti ed aggregarle in oggetti (istanze di classi appositamente definite) che vengono mandati alla base station core per essere smistati verso i moduli opportuni. Allo stesso tempo, tramite base station WSS è possibile mandare i messaggi ai robot, ad esempio i comandi da attuare: si ricevono da base station core le informazioni rilevanti e il tipo di azione da eseguire, quindi si crea un oggetto messaggio della classe idonea alle informazioni che devono essere trasmesse e si manda tale oggetto al gestore di pacchetti WSS che si occupa della trasmissione fisica dei dati. Le classi utilizzate per la trasmis-

sione e ricezione dei messaggi hanno la peculiarità di essere serializzabili, rendendo possibile la trasmissione dei pacchetti come flusso di dati e la loro aggregazione una volta giunti a destinazione.
Questo modulo è, dunque, la porta attraverso cui la base station comunica coi robot.

Base station core UI

Le informazioni nella base station devono essere presentate all'operatore in forma aggregata e consistente. Allo scopo di visualizzare tali informazioni nella forma più ordinata ed efficace possibile, il modulo di base station core UI aggrega tutti i componenti a video e li rende disponibili all'operatore. L'interfaccia risultante è uno specchio dell'organizzazione modulare della base station: ogni singola funzione è visualizzata in un componente indipendente (detto *widget*) che è riposizionabile e ridimensionabile a piacere dell'operatore, creando una GUI pienamente configurabile e duttile, adatta a funzionare su qualsiasi tipo di schermo. Nella Figura 4.7 è mostrata una schermata dell'interfaccia durante uno dei test eseguiti in laboratorio. Vi si può notare la modularità della GUI ed è evidente come ogni singolo componente assolva ad una particolare funzione fra quelle descritte in questa sezione, rimanendo nel contempo consistente col resto del sistema.

Configuration manager

L'ultimo modulo che presentiamo è il configuration manager. Esso ha lo scopo di raccogliere le preferenze dell'operatore e applicare i cambiamenti necessari sia alla base station che ai singoli robot. Ad esempio, è possibile cambiare i parametri di configurazione dell'HSV, utilizzato per il rilevamento della pelle delle vittime nel modulo di victim detection di ciascun robot. Fra le impostazioni configurabili vi sono anche la rilevanza dei tipi di messaggi ricevuti, che influisce sul calcolo delle priorità del notification manager, e impostazioni relative allo stato di funzionamento dei singoli robot, quali ad esempio l'attivazione o meno di certi moduli.

4.2 Interfaccia grafica utente

Come già evidenziato, la struttura modulare della base station permette di isolare logicamente le diverse funzioni. Ad ogni modulo viene pertanto associata, ovviamente ove necessario, una parte grafica, realizzata sotto forma di widget grafico. Queste singole parti grafiche compongono l'interfaccia. Il già descritto modulo base station core UI si occupa di unire insieme i moduli

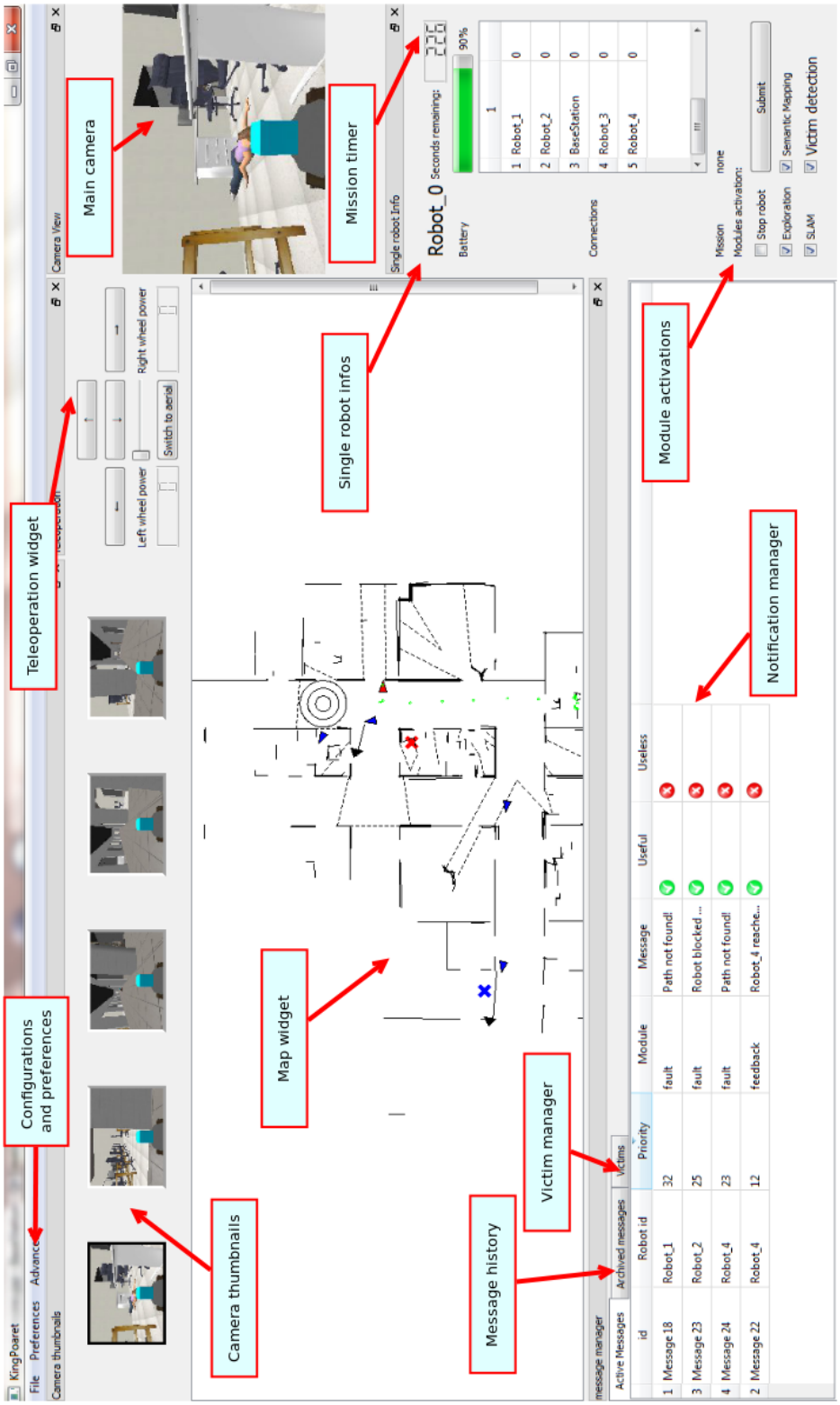

Figura 4.7: Interfaccia grafica del sistema PoAReT, con evidenziati i singoli moduli grafici.

per presentarli all'operatore, ma il contenuto di ognuno è dettato esclusivamente dal modulo logico relativo. La consistenza di ogni modulo è garantita dal nucleo della base station, la base station core, che si occupa di mandare ai vari moduli le informazioni necessarie e garantisce la congruenza di tutte le informazioni presentate.
Nella Figura 4.7 è visibile ogni singolo modulo grafico citato. Approfondiamo le caratteristiche interessanti dei principali moduli in questa sezione.

4.2.1 Map widget

Il map manager si appoggia ad un complesso componente grafico per la visualizzazione delle mappe. Il componente comprende elementi creati ad hoc per favorire l'interazione dell'operatore col sistema, quali i simboli dei robot e quelli delle vittime. È stata implementata una versione potenziata delle *scene grafiche* messe a disposizione da QT, per permettere la gestione delle dinamiche complesse d'interazione come ad esempio la specifica dei comandi waypoint tramite un click sulla mappa, oppure la realizzazione di comandi HLC con doppio click o tracciando una direzione col mouse sulla mappa. Grazie a questi componenti le dinamiche d'interazione con l'operatore risultano molto fluide, permettendo ad un operatore esperto di ottenere buone prestazioni nel controllo di gruppi numerosi di robot.
Tramite il map widget è possibile:

- Selezionare un robot cliccando sul suo simbolo nella mappa;
- Specificare comandi waypoint o HLC in maniera immediata;
- Interagire con le etichette che arricchiscono la mappa d'informazioni aggiuntive, come ad esempio le vittime o i segnali di pericolo, ricevendo informazioni dettagliate sull'elemento;
- Aggiungere etichette per segnalare pericoli, zone d'interesse, zone già esplorate, vittime o altre note personalizzate che supportino l'attività dell'operatore;
- Spostare la posizione di una vittima o eliminarla, qualora la vittima si sia mossa, si sia messa in salvo oppure vi sia stato un errore nella sua identificazione;
- Ingrandire o rimpicciolire la mappa a piacere, permettendo così di avere un maggior dettaglio durante fasi di navigazione critiche e un livello di dettaglio minore quando si vuole avere una visione d'insieme dell'ambiente;
- Scorrere la mappa, rendendo possibile l'analisi di zone già mappate per studiare la situazione attuale e prendere decisioni più accurate.

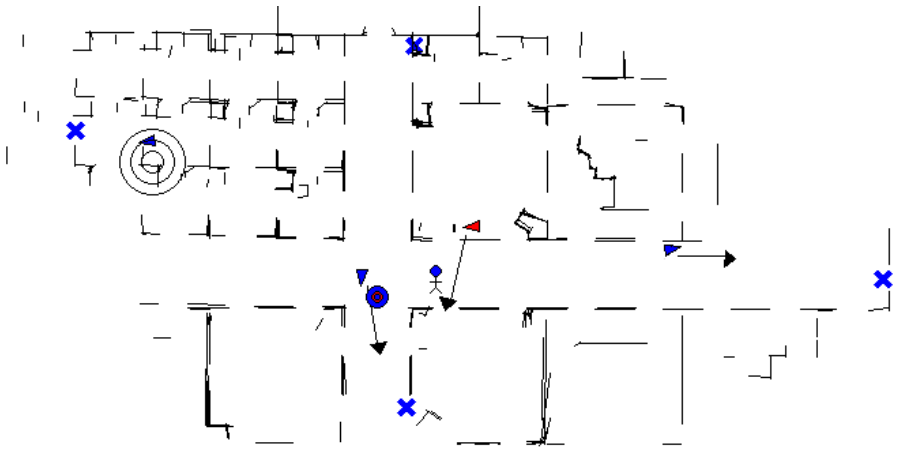

Figura 4.8: Map widget, così come visualizzato all'operatore. Le croci blu sono le destinazioni dei robot calcolate automaticamente dal sistema d'esplorazione autonoma; i cerchi blu sono comandi waypoint; i triangoli sono i robot, in rosso quello selezionato. Infine, le frecce vicino ai robot indicano le direzioni da esplorare preferenzialmente, mentre i cerchi concentrici indicano il comando HLC di esplorazione in una specifica area.

Il comportamento della mappa è integrato con alcuni comandi da tastiera (come, ad esempio, lo scorrimento della mappa) e con funzioni del mouse (ad esempio, è possibile zoomare sulla mappa tramite la rotella del mouse). Tutti questi dettagli, seppure scarsamente innovativi, influenzano notevolmente le dinamiche d'interazione dell'operatore e fanno la differenza fra un'interfaccia usabile ed una ostica e di conseguenza fra un sistema che riesce a sfruttare in maniera efficace il contributo della componente umana ed un sistema che non raggiunge tale obiettivo. Nella Figura 4.8 mostriamo un esempio di mappa visualizzata nel map widget. L'operatore vi può trovare le posizioni dei robot, i feedback ai comandi espressi (ad esempio la direzione preferita di esplorazione o il waypoint che il robot sta cercando di raggiungere) e le destinazioni decise autonomamente. Vi può inoltre aggiungere etichette personalizzate, segnali di pericolo o di vittime trovate.

4.2.2 Main camera

Il flusso video del robot selezionato viene visualizzato alla massima risoluzione e ad un framerate più elevato rispetto agli altri. Inoltre, ha dimensione maggiore nell'interfaccia, per permettere all'operatore di focalizzarsi meglio sul segnale video durante le sequenze di tele-operazione, nelle quali la telecamera rappresenta il principale strumento di orientamento. La finestra che visualizza il video principale è ridimensionabile a piacere dell'operatore, e

con essa viene automaticamente ridimensionato il video stesso, mantenendo le proporzioni. In questo modo, qualora l'operatore abbia a disposizione uno spazio maggiore e voglia dedicarlo al video principale, può farlo senza difficoltà.

4.2.3 Camera thumbnail

I flussi video dei robot non selezionati arrivano con un framerate minore, per risparmiare banda di rete. Per risparmiare spazio nella GUI tali video vengono proposti all'operatore a risoluzione ridotta e in piccole finestre dette thumbnail. L'operatore può così monitorare i robot ed ha nel contempo spazio nell'interfaccia per configurare nella maniera più comoda ogni singolo componente. Tramite i thumbnail è possibile anche selezionare i robot. Il segnale di selezione di un nuovo robot viene mandato alla base station core, che si occupa di attuare la selezione effettiva ed aggiornare tutti i moduli interessati.
Mostriamo in Figura 4.9 un confronto fra la main camera ed i camera thumbnail in cui si possono confrontare le proporzioni di default fra i due componenti. Fra le camera thumbnail, quella del robot selezionato ha un'ombreggiatura nera attorno al flusso video.

4.2.4 Teleoperation widget

Quando un robot richiede l'intervento diretto dell'operatore per uscire da una situazione complessa, l'operatore può eseguire delle azioni direttamente sull'attuazione dei robot tramite il teleoperation widget. Questo componente ha un aspetto diverso a seconda del tipo di robot selezionato, in accordo col tipo di moto di ognuno dei robot. Le azioni vengono eseguite cliccando sui tasti appositi del componente, oppure tramite i comandi da tastiera, permettendo una manovrabilità ancor migliore. La collocazione della telecamera sui robot è studiata per facilitare la navigazione tele-operata ed inquadra la parte anteriore dei robot per evitare di urtare inavvertitamente gli ostacoli. In Figura 4.10 mostriamo un confronto tra il teleoperation widget dell'AirRobot e quello dei robot terrestri (P3AT e Kenaf). Il primo permette all'operatore di muovere un AirRobot in avanti o all'indietro, in alto o in basso, ruotare oppure traslare lateralmente. Il secondo comanda ai robot terrestri le azioni di rotazione sul posto o movimento in avanti o indietro. In entrambe le versioni del teleoperation widget si può regolare la velocità d'attuazione dei comandi con la barra scorrevole posta in basso.

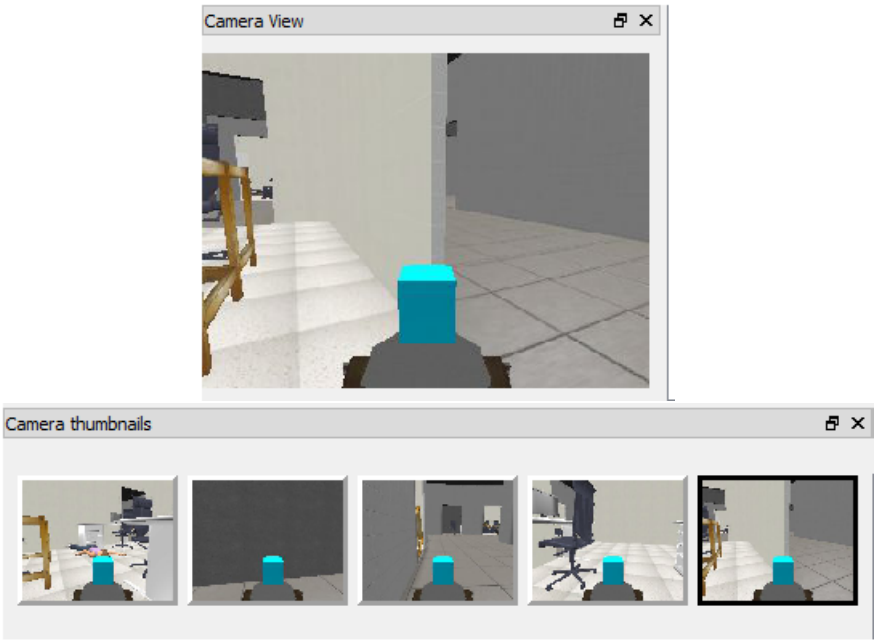

Figura 4.9: Confronto tra il visualizzatore della telecamera del robot selezionato (figura in alto) e i video delle telecamere degli altri robot raggruppati (figura in basso). Il rapporto fra le dimensioni dei thumbnail e della telecamera principale è lo stesso utilizzato di default nella GUI di PoAReT.

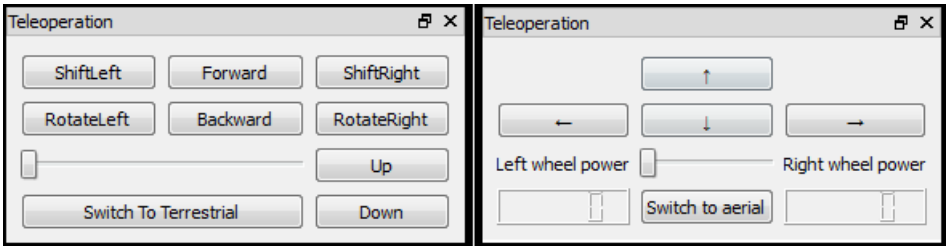

Figura 4.10: Teleoperation widget dei robot aerei (sinistra) e terrestri (destra).

4.2.5 Notification viewer

Le notifiche filtrate, aggregate e classificate dal notification manager vengono visualizzate tramite tabelle che compongono il notification viewer. I dati relativi alle notifiche vengono memorizzati in un'apposita struttura dati e visualizzati tramite un *QTableWidget*, un componente standard delle librerie QT. Per collegare i dati alla loro visualizzazione si utilizza un proxy, che permette di modificare la visualizzazione senza necessità di intervenire sui dati. Ad esempio, è possibile filtrare alcune tuple, selezionare solo messaggi di un certo tipo o provenienti da un certo robot, ordinare i messaggi per priorità, per identificativo o altro. Grazie a questa architettura di tipo MCV (Model-View-Controller) abbiamo reso disponibili funzioni avanzate in maniera molto efficiente dal punto di vista del carico computazionale.

4.2.6 Interazioni

Come abbiamo sottolineato all'inizio della sezione, ogni componente grafico della GUI è collegato al rispettivo modulo manager, che si occupa di aggiornare le informazioni presentate e di gestire le interazioni con l'operatore. La tipica procedura di gestione di una nuova informazione prevede diversi passi:

1. Un modulo segnala una nuova informazione o un evento nel sistema che bisogna gestire. Se tale informazione deriva da un robot, ad esempio una nuova mappa o una notifica, il modulo che solleva l'evento è base station WSS, altrimenti può essere qualsiasi modulo dell'interfaccia con cui l'operatore abbia interagito: ad esempio, un cambio della configurazione HSV viene segnalato dal configuration manager, mentre la selezione di un robot diverso può essere scatenata da map manager, notification manager o camera manager;
2. Base station core riceve il segnale e lo manda ai moduli interessati, evocando i metodi che devono gestire tale evento. Si attua uno smistamento verso tutti e soli i moduli interessati;
3. Ognuno dei moduli interessati esegue le elaborazioni che gli sono necessarie per gestire l'evento, quindi aggiorna il rispettivo componente grafico per essere congruente con lo stato del sistema implicato dall'evento gestito.

Nella Figura 4.11 rappresentiamo un esempio d'interazione di questo tipo, nella fattispecie il processo di gestione della selezione di un robot. Nella figura l'evento viene sollevato dal camera manager, gestito dalla base station core che si occupa di trasmettere l'evento ai moduli interessati (map manager, info manager, teleoperation manager e notification manager) che

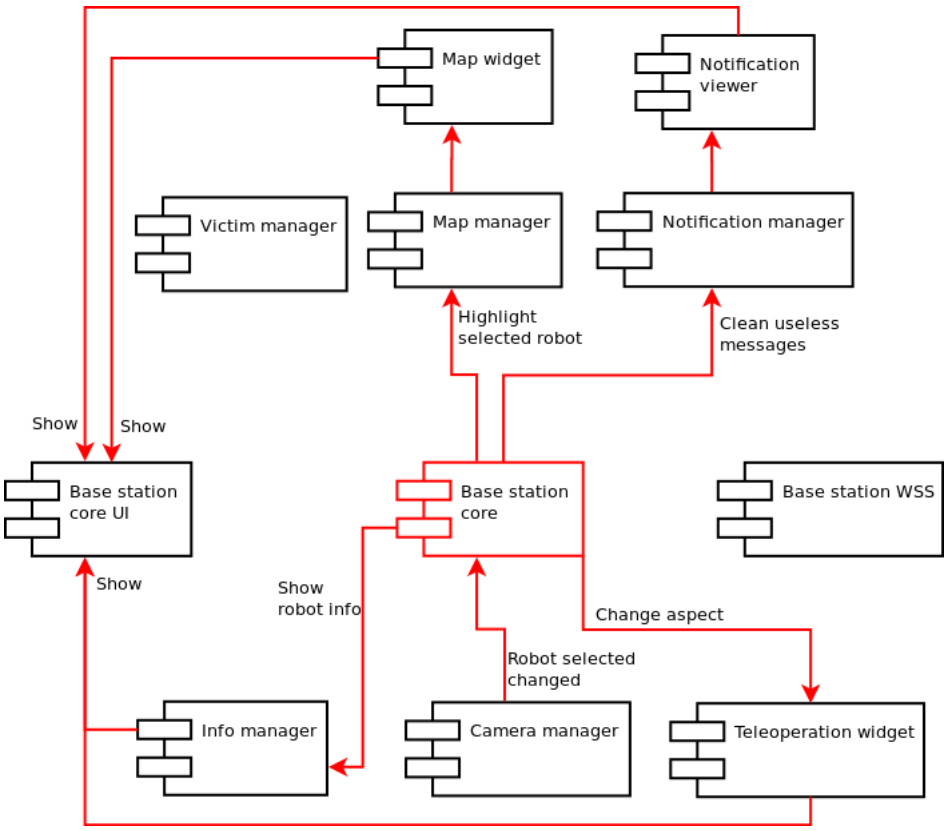

Figura 4.11: Esempio di procedura d'interazione fra i moduli della base station nella gestione dell'evento selezione di un robot nato dall'interazione con un camera thumbnail.

eseguono le azioni opportune ed aggiornano la loro componente grafica in risposta all'evento.
In tabella 4.1 sono mostratigli scambi di informazioni fra i diversi moduli nella base station. Ogni informazione viene scambiata solo con base station core, che si occupa eventualmente di inoltrarla verso altri moduli. Questa architettura permette di modificare, aggiungere o eliminare moduli in maniera semplice semplicemente modificando il solo modulo base station core, rendendo le modifiche trasparenti alle singole parti.

Descriviamo ora, brevemente, il sistema implementato e le sue modalità di utilizzo da parte di un operatore in missioni di USAR. Ci riferiamo in particolare all'implementazione utilizzata durante i test sperimentali che vengono descritti nel Capitolo 5.

		Base station core
Map manager	→	HLC
	→	Waypoint command
	↔	select robot
	←	victim position
	←	map data
	←	autonomous destination
	←	robot position
	←	focus on position
Camera manager	←	camera data
	↔	select robot
Victim manager	↔	victim position
	←	victim images
Notification manager	←	all messages (fault, victim and feedback)
	←	user preferences
	→	show notification
	↔	select robot
	→	focus on position
Base station WSS	→	all messages (fault, victim and feedback)
	→	map and camera data
	→	autonomous destination
	→	connectivity information
	→	battery information
	←	commands (Teleoperation, waypoint and HLC)
Info manager	←	connectivity information
	←	battery information
	←	mission information
	↔	select robot
Teleoperation widget	↔	select robot
	→	teleoperation command

Tabella 4.1: Scambi di informazioni tra i vari moduli. Ogni informazione passa attraverso base station core, che smista i messaggi verso i moduli interessati. Nella tabella indichiamo con ← le informazioni che da base station core vengono spedite al modulo interessato, con → le informazioni che viaggiano dal modulo a base station core e con ↔ indichiamo le informazioni che possono viaggiare in entrambi i versi.

4.3 Il sistema implementato

Tutti i moduli descritti in questo capitolo, per quanto riguarda sia il processo base station sia i singoli robot, sono stati implementati mantenendo le funzionalità descritte. Ove un modulo avesse prestazioni irragionevolmente basse o fosse necessario astrarre dall'aleatorietà legata al suo funzionamento sono state introdotte implementazioni che li rendessero più affidabili o che simulassero il comportamento del modulo sostituito mantenendo le funzionalità originarie. A titolo d'esempio, in ambito di test abbiamo ritenuto insostenibile l'aleatorietà legata al modulo di Victim Detection dei robot ed abbiamo deciso di sostituirlo con un modulo simulato, che fosse in grado di dare la quasi certezza d'individuazione delle vittime, grazie alla conoscenza a priori delle loro posizioni. Ovviamente queste modifiche sono state utilizzate esclusivamente durante i test in laboratorio.
Alcuni moduli sono stati utilizzati senza alcune funzionalità, in quanto non ancora implementate durante la stesura del presente lavoro. La principale fra le funzionalità mancanti è quella di previsione delle mappe a partire dalla semantica legata alla struttura dell'ambiente. Questo aspetto non deve essere interpretato come negativo: l'utilizzo di funzionalità avanzate all'interno dei moduli avrebbe in certa misura compromesso un confronto con altri sistemi implementati dal punto di vista delle tematiche affrontate in questo elaborato, ed in particolare l'effetto di HLC e NFS nelle interazioni con l'operatore e i conseguenti effetti sulle prestazioni dell'intero sistema. Avendo a disposizione un'implementazione dei singoli moduli più vicina all'attuale stato dell'arte e simile a quella usata dai sistemi presi come metro di paragone possiamo focalizzare il confronto sui soli elementi innovativi da noi introdotti. Nella Tabella 4.2 presentiamo l'elenco delle funzionalità attive in fase di test del sistema per ognuno dei moduli presentati e descritti nella prima parte di questo capitolo. Come si può notare la quasi totalità delle funzioni è stata mantenuta anche in ambito di test.
Nel seguito presentiamo alcune dinamiche d'utilizzo del sistema da parte dell'operatore, focalizzando l'attenzione sui processi legati all'interfaccia utente. Tuttavia, questa non vuole essere una guida utente e pertanto non verranno presentate tutte le funzioni ma solo quelle ritenute più significative.

4.3.1 Dinamiche d'interazione

Presentiamo un diagramma use case contenente i casi d'uso approfonditi nel seguito ed altri ritenuti interessanti nella Figura 4.12. Lo scopo di tale diagramma è solamente di sintesi e non vuole essere esaustivo. Esso eviden-

Modulo	Funzionalità attive.
SLAM	SLAM a segmenti di base ricevente il feedback della IMU.
Path planner	Implementazione completa del path planner sia con RRT che A* per poter raggiungere sia zone conosciute che non.
Motion control	Implementazione completa per tutti e tre i tipi di robot supportati (P3AT, Kenaf e AirRobot).
Semantic mapping	Implementazione del solo riconoscimento degli ambienti e rilevazione delle vie d'accesso.
Exploration	Implementazione completa con valutazione dei criteri di esplorazione tramite MCDM.
Coordination	Implementazione completa del meccanismo market-based per l'allocazione delle destinazioni ai robot.
Victim Detection	Implementazione completa, ma utilizzato simulatore per eliminare l'aleatorietà del processo di rilevamento delle vittime durante i test in laboratorio.

Tabella 4.2: Funzioni utilizzate durante i test per ogni modulo rilevante dei singoli robot.

zia le relazioni fra le principali funzioni che il sistema mette a disposizione dell'operatore per interagire coi robot ed un loro possibile utilizzo durante missioni di USAR per compiere dei macro-compiti.
Consideriamo nel seguito tre casi d'uso riguardanti le tre principali attività dell'operatore durante le missioni. Esse sono legate ai compiti più importanti della componente umana nel sistema multirobot: la gestione di alto livello della strategia di esplorazione, la gestione delle eccezioni sollevate dai robot ed infine il lavoro di continuo monitoraggio che l'operatore deve compiere per avere consapevolezza dello stato del sistema e poter intervenire su di esso in modo efficace.

Gestione della strategia d'esplorazione

Le capacità dell'operatore umano, come già argomentato nel Capitolo 2, sono complementari a quelle del sistema automatico; l'essere umano è più idoneo sia a compiere scelte di alto livello, strategiche per il compimento della missione, sia per risolvere problematiche al più basso livello, sull'attuazione vera e propria. Abbiamo spiegato nel Capitolo 3 come l'operatore nel nostro sistema può intervenire in maniera naturale sulle politiche di esplorazione combinando comandi tradizionali e comandi HLC, sfruttando al meglio tutte le funzioni automatiche a disposizione e mantenendo un saldo controllo del

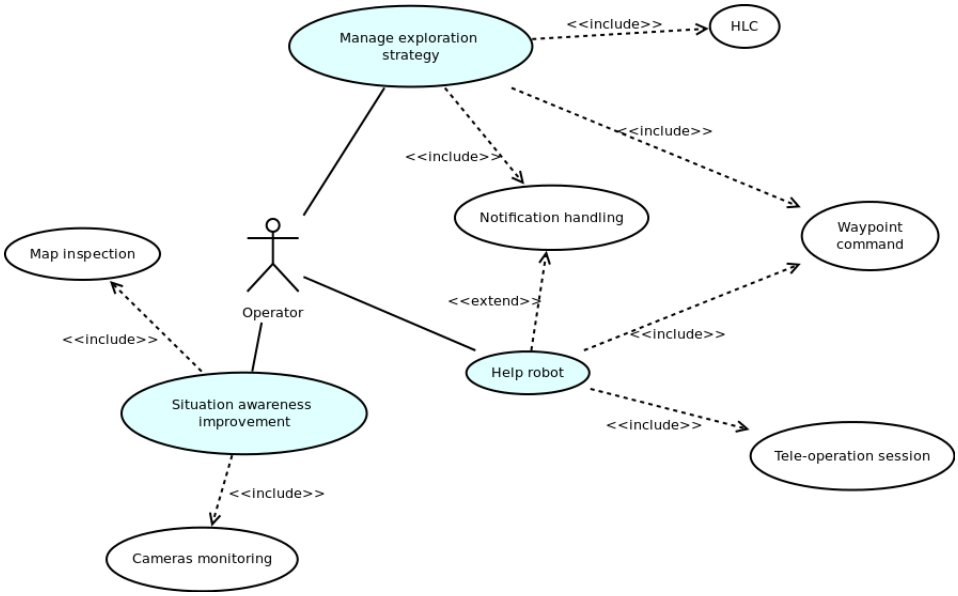

Figura 4.12: Diagramma use case delle tre principali attività dell'operatore (a sfondo azzurro). Le attività principali sono composte da sotto-attività più semplici.

sistema. Nella Figura 4.13 mostriamo un sequence diagram esemplificativo della procedura d'interazione dell'operatore col sistema per intervenire sulla strategia d'esplorazione. Ovviamente le dinamiche d'interazione sono molto complesse ed il diagramma ne è solo un esempio semplificato. Nella figura mostriamo una semplicistica sequenza di azioni: l'operatore seleziona un robot, il sistema reagisce opportunamente, l'operatore interagisce con la mappa per lanciare un HLC e il sistema lo trasmette al robot ed aggiorna la mappa con un feedback grafico. Per trasmettere il comando HLC al robot, il map manager segnala a base station core il comando, questa manda tali informazioni a base station WSS che crea un apposito messaggio e lo trasmette al robot destinatario tramite la rete wireless simulata. In una situazione più complessa la selezione del robot potrebbe avvenire per mezzo di una notifica, oppure potrebbe essere usata una combinazione di comandi HLC e waypoint.

Aiuto ai robot e gestione delle eccezioni

La componente umana nel sistema è indispensabile nella gestione delle eccezioni, quali errori nella navigazione, individuazione di possibili vittime o errori di sistema che comportano un diretto intervento dell'operatore per poter essere gestiti.
L'introduzione del NFS, descritto nel Capitolo 3, permette al nostro sistema

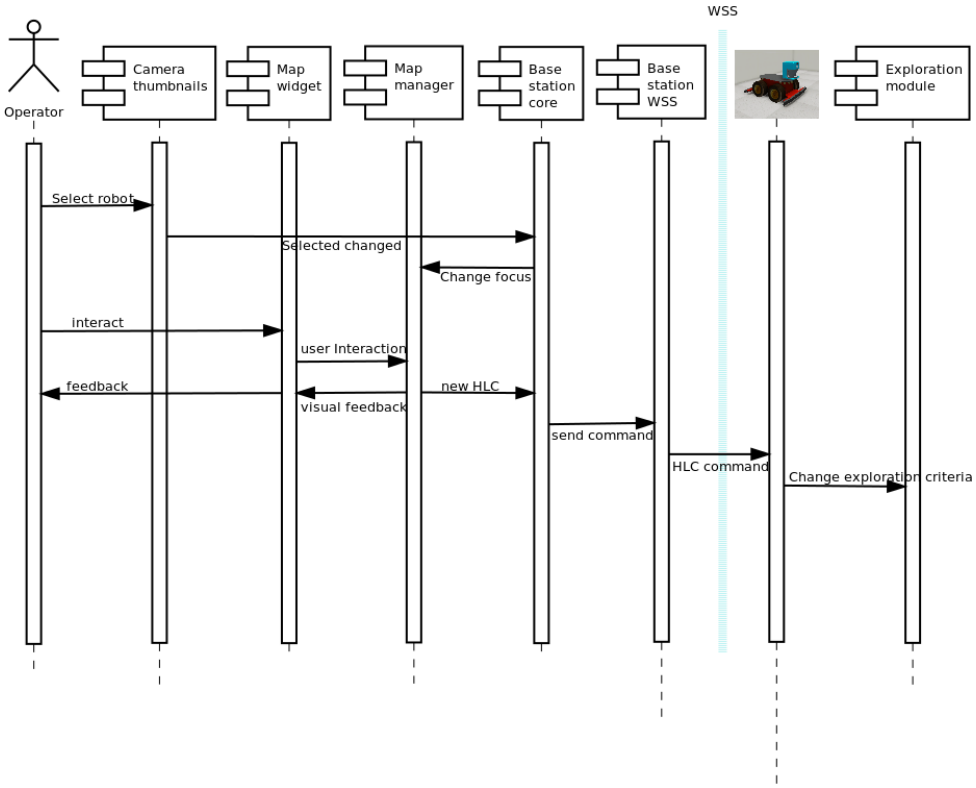

Figura 4.13: Esempio d'interazione fra l'operatore ed il sistema per l'intervento sulla strategia d'esplorazione tramite l'utilizzo di HLC.

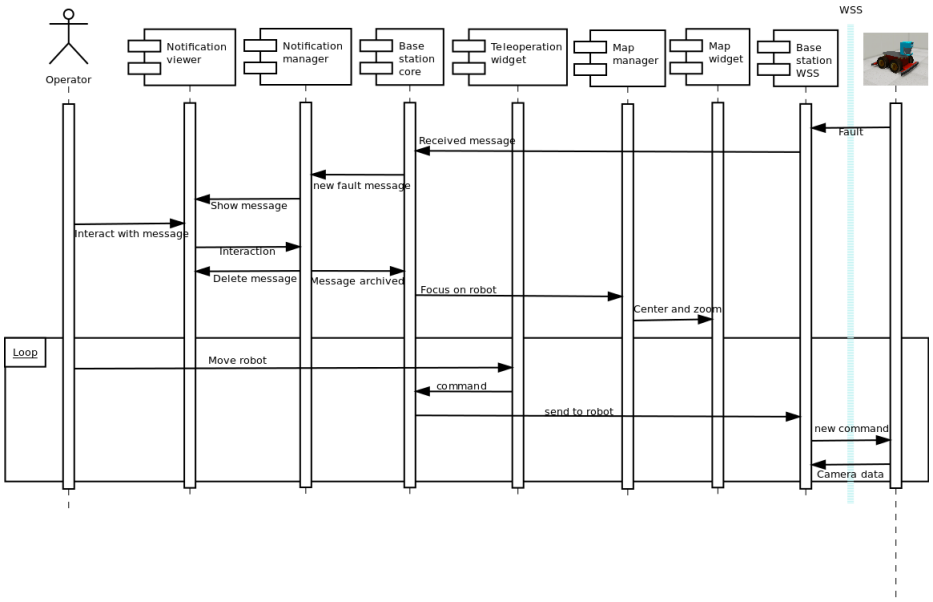

Figura 4.14: Esempio di procedura d'interazione per l'intervento sui robot a fronte di una loro richiesta.

una gestione avanzata delle notifiche dei robot, e quindi di quelle situazioni che i robot non riescono ad affrontare autonomamente. Nella Figura 4.14 è mostrato il sequence diagram relativo alle dinamiche di aiuto ai robot durante le missioni. Anche in questo caso il diagramma vuole essere solo esemplificativo e non è certo esaustivo: viene rappresentato l'arrivo di una richiesta d'intervento per un errore di navigazione, la sua gestione da parte della base station e del notification manager, l'effetto dell'interazione dell'operatore con la notifica e la risoluzione dell'eccezione tramite una sessione di tele-operazione. Le dinamiche sono ben più complesse nella realtà, e possono includere diversi tipi di intervento (soprattutto tele-operazione e waypoint), possono prevedere o meno la selezione di diversi robot per la risoluzione di una sola eccezione e possono anche non essere originati dalla gestione di una richiesta esplicita da parte dei robot.

Aumento della situation awareness

Per poter intervenire in maniera efficace sul sistema, l'operatore deve essere costantemente aggiornato sulla situazione dei singoli robot. I principali dati necessari per avere una consapevolezza sia dell'ambiente globale sia di quello circostante i singoli robot e poter così individuare sia le politiche di esplorazione ottimali sia i possibili problemi che i robot potrebbero incontrare sono

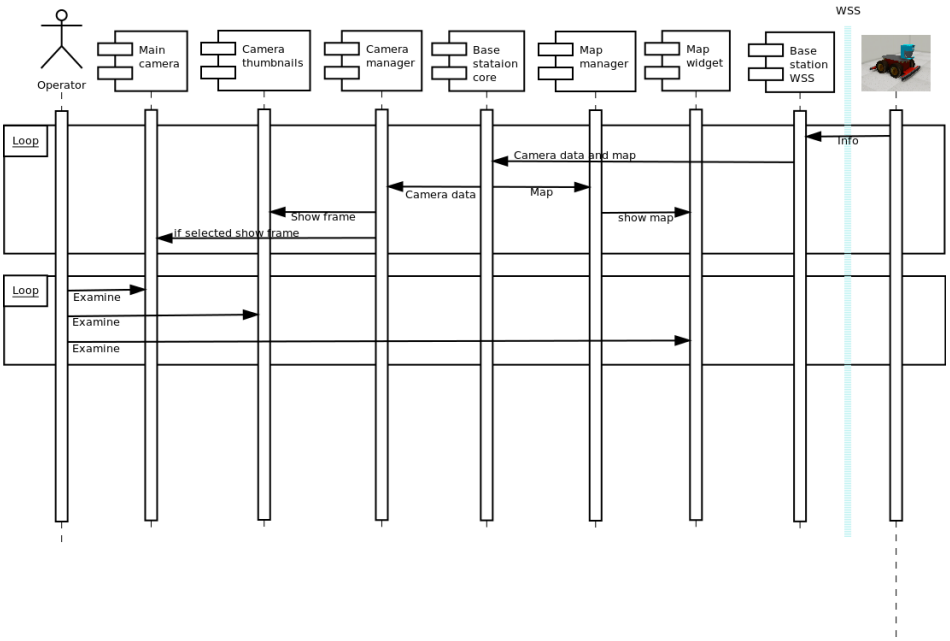

Figura 4.15: Esempio di creazione della SA da parte dell'operatore basato su flussi video e mappe. Non vengono, per semplicità, mostrate le altre informazioni che competono a migliorare la SA.

i flussi video e le mappe dei robot. Altre informazioni utili all'operatore sono lo stato della missione, le informazioni sul livello di batteria e di connettività ed altre informazioni accessorie, ma hanno un impatto ben minore rispetto a mappa e flussi video sulla creazione della SA.
Nella Figura 4.15 mostriamo i flussi informativi relativi a mappa e segnali video, flussi che l'operatore deve monitorare per avere un'opportuna situation awareness e poter intervenire in modo rapido ed efficace. Si tenta di enfatizzare nella figura la ciclicità di questa attività, che viene ripetuta costantemente durante tutta la missione di soccorso, con intervalli in cui l'operatore si dedica alle altre macro-attività descritte nei paragrafi precedenti.

Capitolo 5

Realizzazioni sperimentali e valutazioni

Analizziamo in questo capitolo i risultati sperimentali ottenuti dall'interfaccia implementata in questa tesi, sfruttando le misure già definite nel Capitolo 3. I dati provengono da due ambiti: i test in laboratorio, che rappresentano il più affidabile metro di valutazione delle performance, e i risultati ottenuti in una competizione internazionale, sicuramente non ripetibile ma che è indubbiamente obiettivo e permette di confrontare il nuovo sistema direttamente con sistemi e soluzioni concorrenti.

Per valutare le prestazioni del sistema abbiamo realizzato un ambiente di test e coinvolto un numero di tester sufficiente a rendere significativi i dati raccolti, anche in accordo a test sperimentali eseguiti su sistemi simili nell'area dello USAR: in [3] vi sono 14 partecipanti ai test, in [8] partecipano 2 esperti e un piccolo gruppo di operatori non esperti e se ne confrontano le prestazioni, in [2] i test vengono eseguiti replicando le condizioni della competizione e sono eseguiti da soli 2 tester, in [72] si eseguono 10 test con 8 operatori. Riteniamo quindi idonee le nostre serie di 9 prove per ognuno dei 14 tester, di cui 7 esperti e 7 inesperti, per avere un'indicazione abbastanza affidabile delle prestazioni del sistema di controllo.
Ogni serie di 9 test è composta dalle singole prove in Tabella 5.1: ogni partecipante è sottoposto a prove con 2, 5 e 8 robot e con diversi livelli di funzionalità del sistema: un sistema *standard*, ossia con solo funzionalità presenti nello stato dell'arte, un sistema con solo l'introduzione degli *HLC* ed un sistema con piene funzionalità (Full Functionality, *FF*) che include anche il NFS.
Il compito dell'operatore in ognuna delle prove cui viene sottoposto è riuscire

N.robot	standard	HLC	FF
2	prova 1	prova 2	prova 3
5	prova 4	prova 5	prova 6
8	prova 7	prova 8	prova 9

Tabella 5.1: Prove eseguite per ogni partecipante ai test, una per ogni combinazione possibile fra numero di robot a disposizione e funzionalità del sistema.

ad individuare il maggior numero di vittime possibile nel tempo fornito (5 minuti). Una volta individuata una vittima l'operatore deve solo segnarne la posizione sulla mappa e quindi può procedere nell'esplorazione. L'operatore deve dunque dedicarsi sia a compiti legati all'esplorazione dell'ambiente, sia a compiti percettivi che gli permettano d'individuare le vittime.
Per ogni serie di tre prove eseguite con un certo numero di robot si utilizza una diversa arena di test, di dimensioni crescenti col numero di robot. Nella Figura 5.1 le planimetrie delle tre mappe da noi progettate, rispettivamente da utilizzarsi con 2, 5 e 8 robot. Le mappe sono dimensionate in maniera tale che solo con una *perfetta gestione* di tutti i robot sia possibile trovare tutte le vittime ed esplorare completamente l'ambiente. Per questo motivo nella mappa da 2 robot è ragionevole che gli operatori riescano ad esplorare tutto l'ambiente e trovare tutte le vittime, mentre è molto improbabile che ciò accada nella mappa da 8 robot data la difficoltà a gestire molti robot. Il metro di paragone per la definizione di perfetta gestione di un robot sono le prestazioni di un operatore esperto nella tele-operazione di un singolo robot per la durata della prova. Tale metro di paragone è stato scelto poiché rappresentante le attuali applicazioni sul campo dei sistemi robotici in ambito USAR.
Ad ogni singolo tester è stato proposto l'insieme delle 9 prove totali in ordine diverso, per ridurre gli eventuali effetti delle dinamiche di apprendimento. Sempre allo scopo di ridurre l'effetto di tali dinamiche, che potrebbero nel caso degli utenti inesperti falsare i risultati, si è proposto ad ogni tester una sessione di prova col sistema per prenderne confidenza, sessione conclusa solo quando il tester e l'esaminatore hanno ritenuto che se ne avesse pienamente appreso il funzionamento.
Ognuna delle 9 prove ha durata di 5 minuti.

5.1 Risultati ottenuti

Analizziamo i risultati ottenuti durante i test con riferimento alle misure di prestazione descritte nel dettaglio nel Capitolo 3. Nei casi interessan-

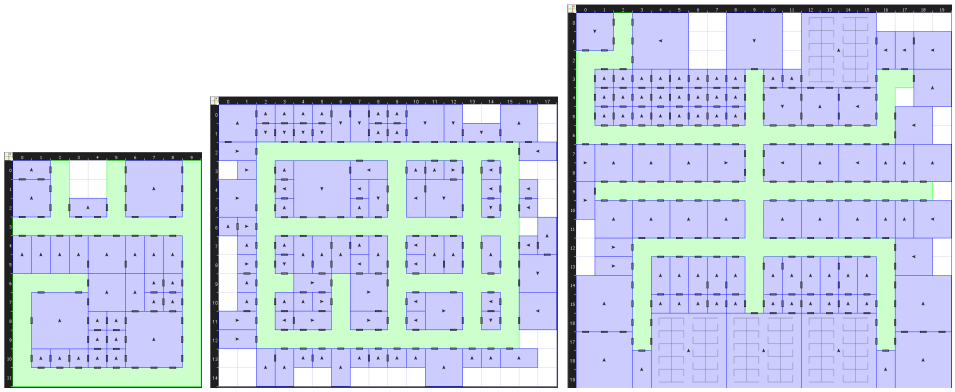

Figura 5.1: Mappe utilizzate nei test sperimentali. Nell'ordine le mappe per 2, 5 e 8 robot. Tutte e tre le mappe simulano complessi di uffici. In verde sono rappresentati i corridoi, in azzurro le singole stanze. Sugli assi di ogni figura sono rappresentate le dimensioni della mappa nell'unità di misura adottata dal simulatore e corrispondente a 3 metri. In metri, le dimensioni delle mappe sono, rispettivamente: 27m x 33m, 51m x 42m e 57m x 57m.

ti analizziamo le prestazioni del gruppo di partecipanti esperto e quelle dei partecipanti inesperti separatamente, al fine di fare un confronto degli effetti delle innovazioni sui due gruppi. Quando opportuno presentiamo un'analisi sui dati dell'intero gruppo nel suo complesso, per massimizzare la quantità di dati a sostegno dei risultati. Dove i risultati ottenuti sono particolarmente interessanti per questo lavoro di tesi eseguiamo dei test di tipo *one-way ANOVA* (*ANalysis Of VAriance*) [73] per determinare l'effettiva validità statistica dei risultati. Tutti i test eseguiti considerano due gruppi di osservazioni ed evidenziano la differenza fra di essi (considerando come soglia $p < 0.05$) o la loro somiglianza.

5.1.1 Fan out

Ricordiamo che il fan out è il numero di robot che l'operatore riesce a gestire in maniera efficace. Le misure che utilizziamo per valutare l'efficacia nella gestione dei robot sono l'area mappata e la distanza percorsa, che insieme danno una buona approssimazione dell'efficacia nella gestione, almeno per quanto riguarda la capacità di esplorazione. Si rimanda al Capitolo 3 per una definizione più dettagliata.
Nella Figura 5.2 mostriamo i risultati ottenuti per i tester inesperti. Si può notare un netto miglioramento delle prestazioni con gruppi numerosi di robot (5 e 8) con l'introduzione degli HLC (prove contrassegnate HLC e FF), grazie al contributo dell'autonomia. Gli utenti inesperti ottengono le mi-

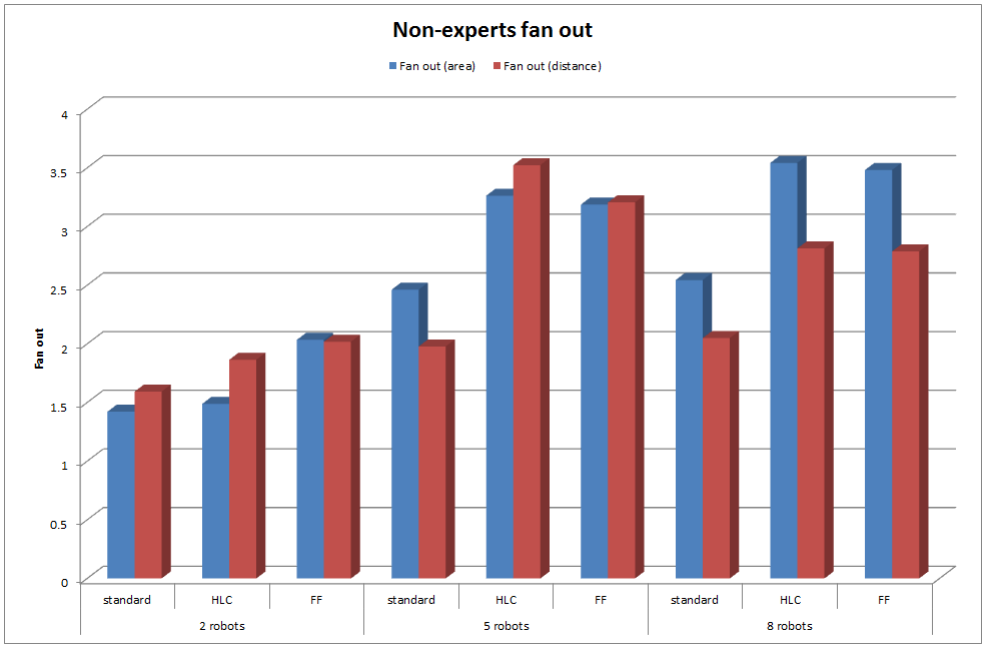

Figura 5.2: Fan out medio del gruppo di tester inesperti relativo alle nove prove effettuate.

gliori prestazioni con 5 robot, mentre con 8 robot hanno difficoltà maggiori a gestire il sistema e le prestazioni hanno una leggera flessione.
Nella Figura 5.3 mostriamo gli stessi dati per i tester esperti. Da notare come le prestazioni siano generalmente migliori e come vi siano miglioramenti anche in questo caso con l'introduzione degli HLC. Nella configurazione FF le prestazioni tendono ad aumentare ulteriormente, contrariamente al caso degli utenti inesperti. Questo effetto è da attribuire ad un più consapevole ed efficace utilizzo del NFS. Per gli utenti esperti notiamo un effetto interessante: le prestazioni nella configurazione standard peggiorano (seppur di poco) nel passaggio da 5 a 8 robot, ma quelle in HLC e FF migliorano in maniera significativa. Questo risultato dimostra che un'interfaccia utente che utilizzi la nostra soluzione è in grado di gestire efficacemente un numero maggiore di robot rispetto ad una GUI tradizionale e quindi permette prestazioni complessivamente migliori nella missione.
Nella Figura 5.4 mostriamo i risultati di tutti i tester globalmente. Il grafico evidenzia come le prestazioni migliorino notevolmente con l'introduzione degli HLC per tutti i tester nelle prove con 5 robot ($F_{1,26} = 10.18$, $p = 0.00369$ nelle osservazioni basate sulla distanza e $F_{1,26} = 6.21$, $p = 0.0194$ nelle osservazioni sull'area mappata) e 8 robot ($F_{1,26} = 4.66$, $p = 0.0403$ per la distanza e $F_{1,26} = 7.43$, $p = 0.0113$ per l'area). Con il NFS i risultati sono simili a

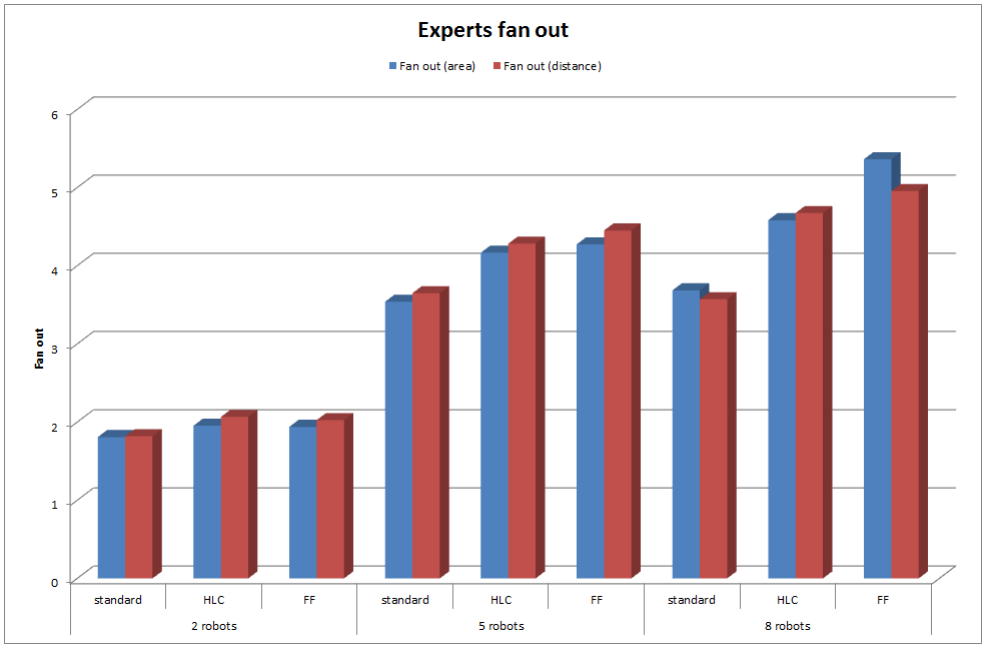

Figura 5.3: Fan out medio ottenuto dai 7 tester esperti.

quelli ottenuti in modalità HLC sia con 5 robot ($F_{1,26} = 0.04$, $p = 0.843$ sulla distanza e $F_{1,26} = 0$, $p = 1$ sull'area) sia con 8 robot ($F_{1,26} = 0.07$, $p = 0.793$ sulla distanza e $F_{1,26} = 0.7$, $p = 0.410$ sull'area). Con 5 robot, le prestazioni in FF hanno una leggera flessione, da collegarsi alle prestazioni scarse degli operatori inesperti nell'uso del NFS, probabilmente dovute al cambio di strategia di gestione del sistema che il NFS implica, cambiamento questo che in alcuni casi ha disorientato i tester.

5.1.2 Vittime trovate

Il numero di vittime trovate, seppure intrinsecamente contenente una rilevante componente aleatoria, è l'obiettivo di maggior interesse durante le missioni di USAR. Ricordiamo che le dimensioni della mappa ed il numero di vittime per ognuna aumentano col numero di robot a disposizione, ragion per cui la percentuale di vittime trovate decresce col numero di robot a parità di funzionalità. I dati sono perciò da interpretare e confrontare considerando separatamente le prove con lo stesso numero di robot.
In Figura 5.5 riportiamo la percentuale di vittime trovate mediamente dai tester inesperti. L'introduzione degli HLC porta notevoli vantaggi anche in questo ambito, ma le prestazioni migliori si ottengono in FF, grazie al

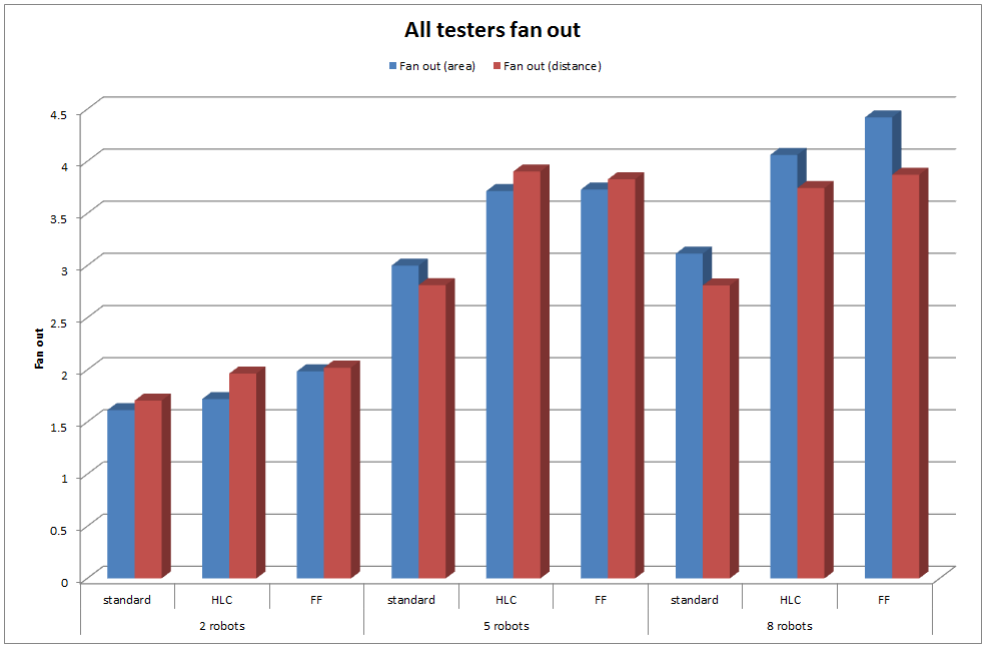

Figura 5.4: Fan out globale medio, calcolato su tutti i tester.

contributo del NFS che permette di sfruttare anche l'aiuto del modulo di victim detection dei singoli robot. Lo stesso trend è riscontrato anche per i tester esperti, di cui riportiamo in Figura 5.6 i risultati. In questo caso si riscontrano, com'è ragionevole, prestazioni genericamente migliori, esaltate soprattutto dal NFS che i tester esperti hanno saputo utilizzare al meglio. Infine in Figura 5.7 è visibile l'andamento globale dei 14 tester. Complessivamente le prestazioni sono notevolmente migliori in modalità FF rispetto a standard per 5 robot ($F_{1,26} = 13.65$, $p = 0.00103$) e per 8 robot ($F_{1,26} = 25.97$, $p < 0.0001$). Tale miglioramento è sicuramente dovuto all'effetto del NFS e non all'introduzione degli HLC: le prove con soli HLC, infatti, non portano a risultati statisticamente diversi rispetto alla configurazione standard né per 5 robot ($F_{1,26} = 0.85$, $p = 0.365$) né per 8 robot ($F_{1,26} = 0.05$, $p = 0.825$)

5.1.3 Distribuzione delle modalità di controllo dei robot

Come abbiamo visto, l'introduzione di HLC e NFS migliora le performance del sistema grazie ad una migliore integrazione della componente umana nel ciclo di controllo sia per utenti esperti che non. Tramite l'analisi dell'utilizzo che i tester fanno dei robot possiamo capire meglio dove queste innovazioni influiscono e qual è l'uso che ne fa l'operatore.

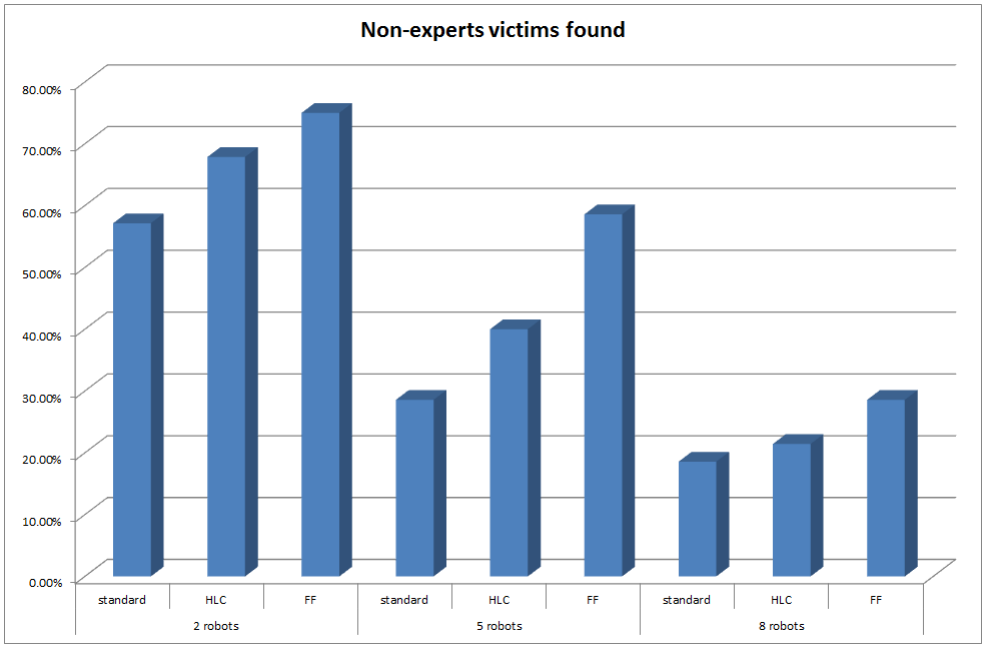

Figura 5.5: Percentuale di vittime individuate mediamente dai tester inesperti.

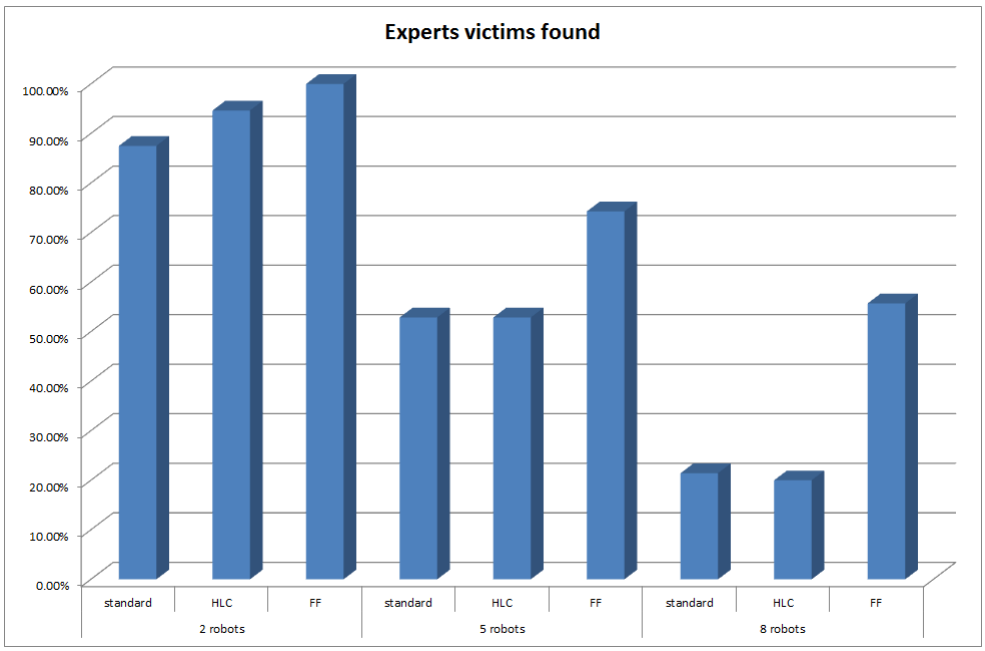

Figura 5.6: Percentuale di vittime mediamente individuate da tester esperti.

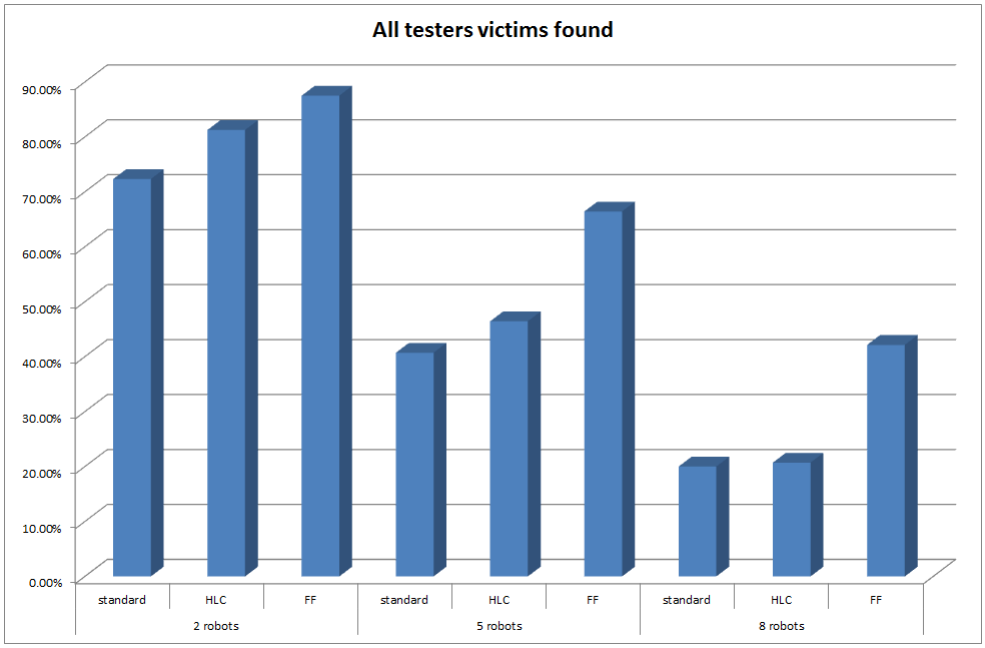

Figura 5.7: Percentuale di vittime mediamente individuate da tutti i partecipanti.

Misuriamo nei test il tempo speso da ogni robot in ognuna delle cinque modalità di lavoro:

- Free AI: il robot sta esplorando l'ambiente seguendo la politica d'esplorazione autonoma, senza alcun intervento dell'operatore;
- HLC: il robot sta esplorando autonomamente l'ambiente seguendo le direttive di alto livello impartite dall'operatore tramite HLC;
- Idle: il robot non sta eseguendo alcun compito utile;
- Teleoperation: il robot viene direttamente tele-operato dall'operatore;
- Waypoint: il robot sta eseguendo un comando di tipo waypoint.

Nella Figura 5.8 è mostrata la distribuzione percentuale media sui robot nelle diverse modalità di funzionamento nelle nove prove effettuate per tutti gli operatori coinvolti nei test. Due sono gli aspetti evidenti:

1. L'introduzione dell'autonomia abbatte l'idle time: sull'intero campione di 14 tester, l'idle time viene ridotto nelle prove HLC e FF in maniera rilevante rispetto alle prove standard nell'uso di 2 robot ($F_{1,26} = 13.14$, $p = 0.00123$), 5 robot ($F_{1,26} = 40.95$, $p < 0.0001$) e 8 robot ($F_{1,26} = 33.44$, $p < 0.0001$). Le differenze fra le prove HLC e FF non sono, invece, statisticamente rilevanti per nessun numero di robot (2 robot: $F_{1,26} = 0.03$, $p = 0.864$; 5 robot: $F_{1,26} = 0.01$, $p = 0.921$; 8 robot: $F_{1,26} = 0.65$, $p = 0.427$);
2. Gli operatori preferiscono di gran lunga l'utilizzo di HLC piuttosto che

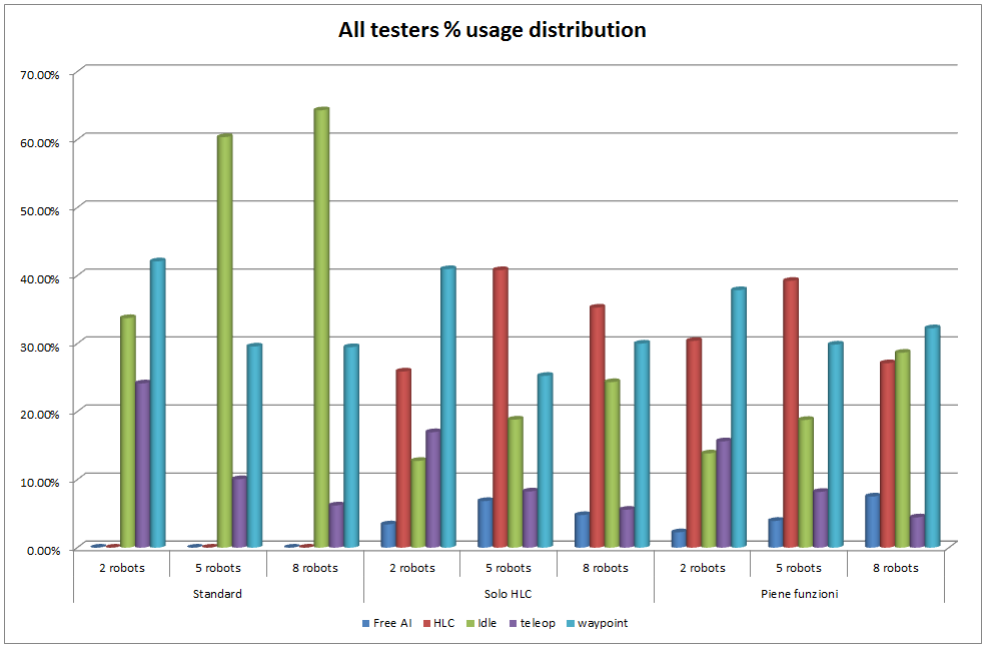

Figura 5.8: Distribuzione delle modalità di funzionamento sulla base dei dati raccolti da tutti i partecipanti.

lasciare il robot esplorare in modalità pienamente autonoma. Diamo maggior rilevanza a questo risultato aggregando le osservazioni sulle 84 prove (6 per ognuno dei 14 tester) che implicano l'utilizzo dell'autonomia e la scelta fra le modalità HLC e Free AI: la preferenza è stata accordata di gran lunga agli HLC con una media del 33.1% del tempo complessivo di funzionamento dei robot contro Free AI al 4.79% ($F_{1,167} = 300$, $p < 0.0001$)

Grazie agli HLC gli operatori possono facilmente applicare la loro strategia d'esplorazione di alto livello ed ottenere un maggior controllo sul sistema. Nelle Figure 5.9 e 5.10 sono riportati i grafici relativi alla distribuzione delle modalità di controllo dei robot rispettivamente per i tester esperti e inesperti. Si nota come gli aspetti appena riscontrati si verifichino in entrambi i gruppi di tester. Da notare anche una differenza notevole fra gli operatori esperti ed inesperti nell'idle time percentuale, differenza che riteniamo giustifichi le prestazioni globalmente migliori degli utenti esperti rispetto agli inesperti, da collegare ad una miglior gestione di ogni funzionalità del sistema. Questo aspetto è ancor più evidente in Figura 5.11 che mostra nel dettaglio il confronto fra gli idle time percentuali per i due gruppi di operatori nelle diverse prove.

Nella Figura 5.12 mostriamo nello specifico i risultati ottenuti nelle serie da

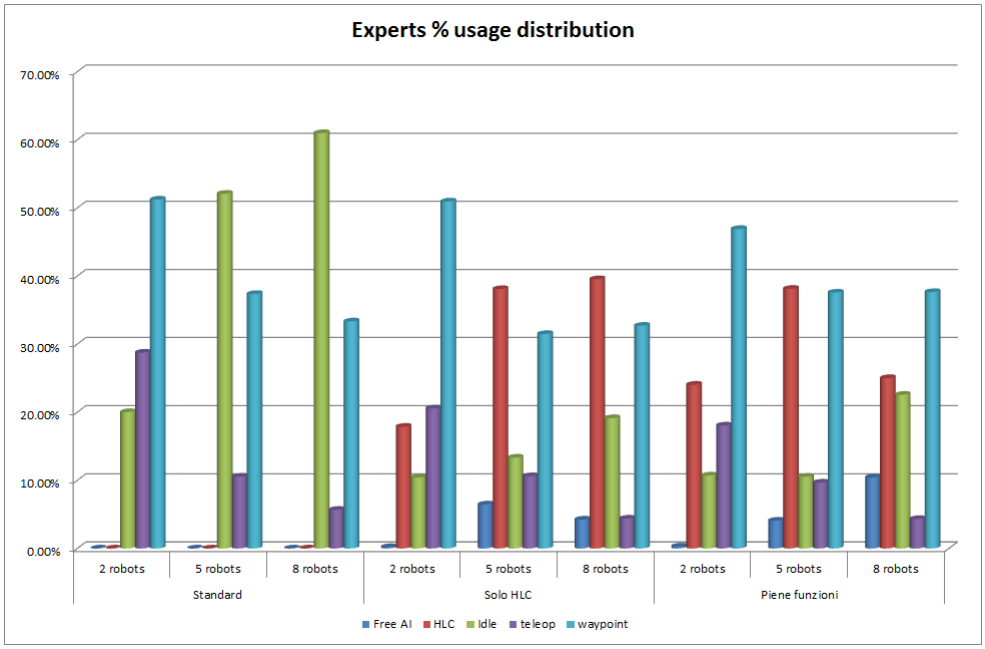

Figura 5.9: Distribuzione delle modalità di funzionamento dei robot sulla base dei dati raccolti dagli operatori esperti.

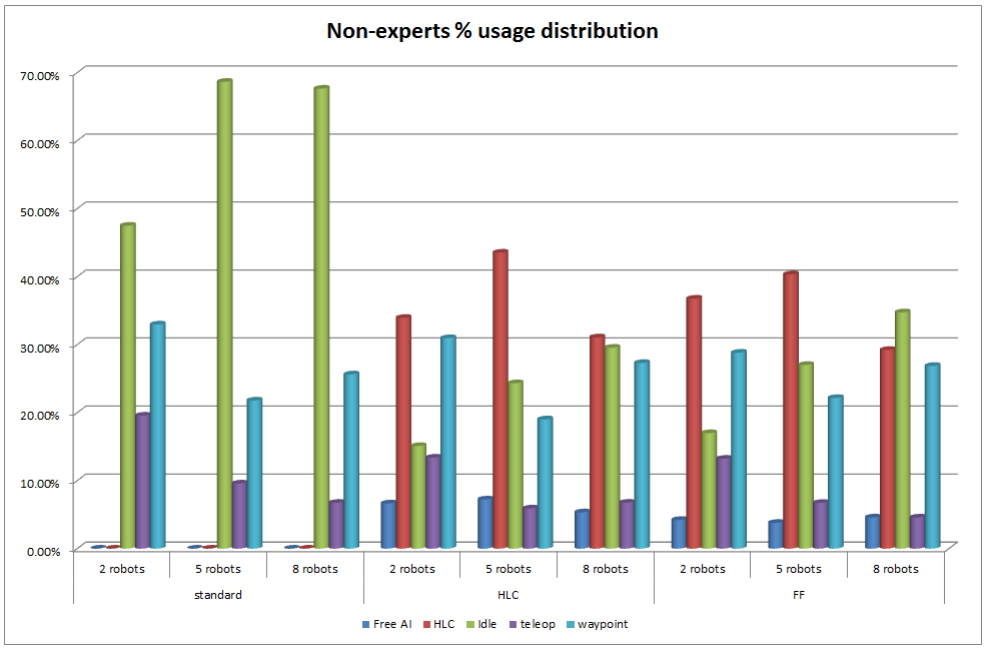

Figura 5.10: Distribuzione delle modalità di funzionamento dei robot sulla base dei dati raccolti dagli operatori inesperti.

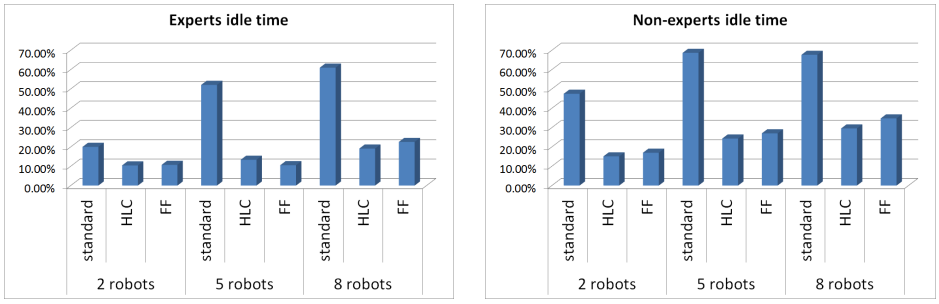

Figura 5.11: Confronto fra gli idle time nei test degli utenti esperti (grafico a sinistra) e non esperti (grafico a destra).

tre prove con lo stesso numero di robot, sia per utenti esperti che inesperti. Valgono ancora gli aspetti sottolineati in questa sezione.

5.1.4 Rapporto tempo operatore fratto tempo dei robot

Il tempo operatore viene considerato pari alla durata delle prove, mentre quello dei robot è il tempo totale che tutti i robot hanno dedicato alla missione (fondamentalmente pari al numero di robot moltiplicato per la durata della prova meno la somma degli idle time dei robot). La misura è tanto migliore quanto più è bassa.
Com'è ragionevole, questa misura migliora all'aumentare del numero di robot, come possiamo vedere dalla Figura 5.13 dove è rappresentata la media dei risultati ottenuti per tutti i tester. Un notevole miglioramento si ha grazie agli HLC e alla conseguente introduzione dell'autonomia che abbatte l'idle time.
Una leggera flessione delle prestazioni, seppure non significativa ($F_{1,26} = 0.13$, $p = 0.721$) si ha nella prova FF con 8 robot rispetto alla prova HLC. Questo può ancora una volta essere dovuto alla difficoltà di utilizzo del NFS da parte degli utenti inesperti, che si traduce in un tempo di attesa superiore da parte dei robot e dunque in idle time superiori.

5.1.5 Workload

Abbiamo identificato, nel Capitolo 3, il workload complessivo cui è sottoposto l'operatore come uno dei principali problemi dei sistemi multirobot in ambito USAR. Nello stesso capitolo abbiamo descritto come misuriamo il workload partendo dagli eventi cognitivi, ossia che richiedono elaborazione cognitiva da parte dell'operatore.
Presentiamo qui i risultati ottenuti per utenti esperti ed inesperti, analiz-

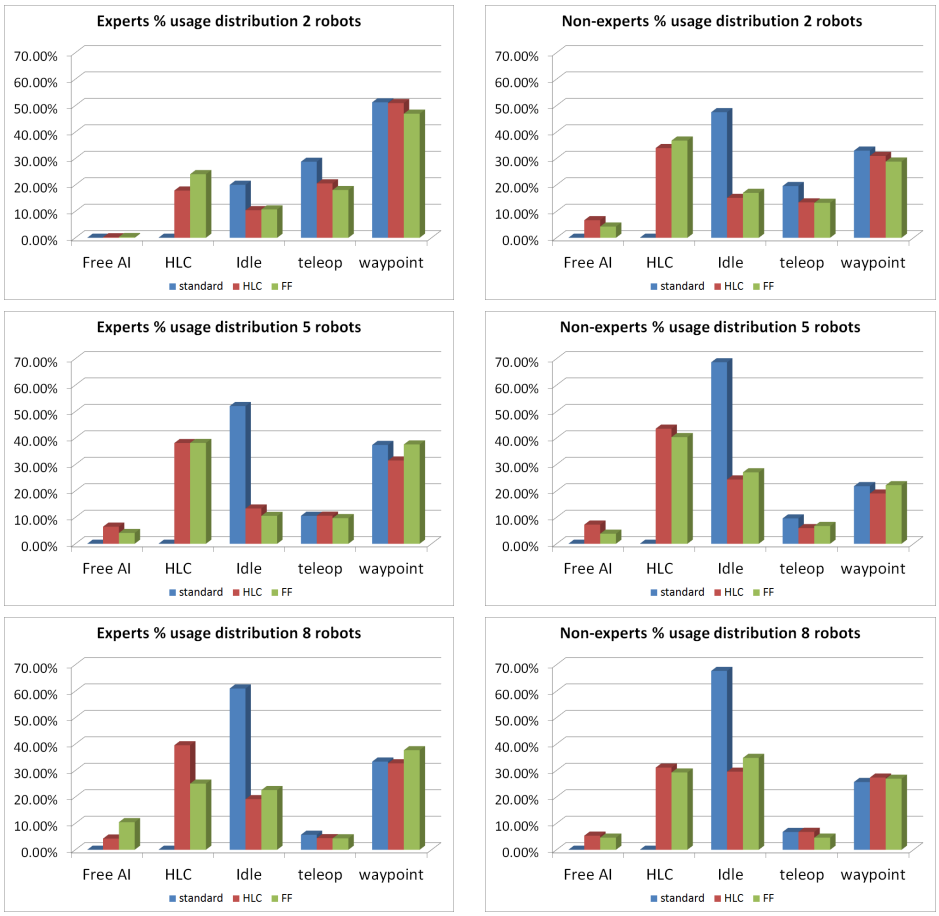

Figura 5.12: Distribuzione delle modalità di funzionamento percentuali dei robot. I grafici nella colonna di sinistra sono riferiti ad utenti esperti, quelli nella colonna di destra ad utenti inesperti. Ogni riga contiene i dati relativi alle prove con un certo numero di robot: la prima riga contiene le prove sottoposte ai due gruppi di tester con 2 robot, la seconda riga quelle con 5 e l'ultima le prove con 8 robot.

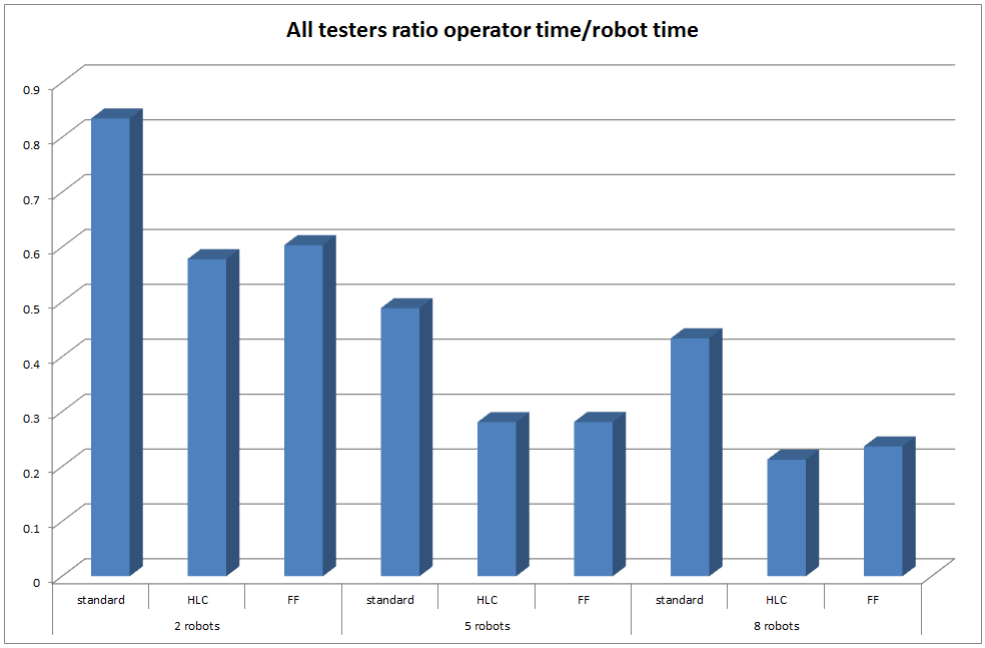

Figura 5.13: Rapporto tempo operatore su tempo robot medio, basato sui dati di tutti i 14 tester.

zando le serie di tre prove a pari numero di robot separatamente per avere una più chiara idea dei risultati e delle loro implicazioni.
In Figura 5.14 si presentano i dati relativi al workload sia degli utenti esperti che di quelli inesperti nelle tre serie di prove. Abbiamo definito nel Capitolo 3 gli eventi cognitivi come qualsiasi evento che comporti un carico cognitivo per l'operatore. Possiamo notare come l'introduzione dell'autonomia dei comandi di alto livello comporti un workload maggiore, dovuto in parte ad un maggior numero di notifiche inviate dai robot ed in parte all'utilizzo più intensivo dei robot durante le missioni grazie appunto alle procedure di esplorazione autonoma. L'aumento del workload comporta una minore SA, e può portare ad errori nella strategia d'esplorazione e nella gestione dei robot in genere. L'introduzione del NFS riesce ad abbattere il workload rispetto alla configurazione HLC (sui 14 tester, con 5 robot $F_{1,26} = 40.95, p < 0.0001$, con 8 robot $F_{1,26} = 27.76$, $p < 0.0001$), riportandolo ai livelli riscontrati in un sistema standard (sui 14 tester, con 5 robot $F_{1,26} = 0.13$, $p = 0.721$, con 8 robot $F_{1,26} = 0.09$, $p = 0.766$), favorendo così la gestione del sistema. Questo risultato ci porta a considerare le due innovazioni, HLC e NFS, complementari per il buon utilizzo del sistema in quanto una permette di alleviare i problemi dell'altra.
Un confronto fra i workload medi in ogni prova fra utenti esperti ed inesperti

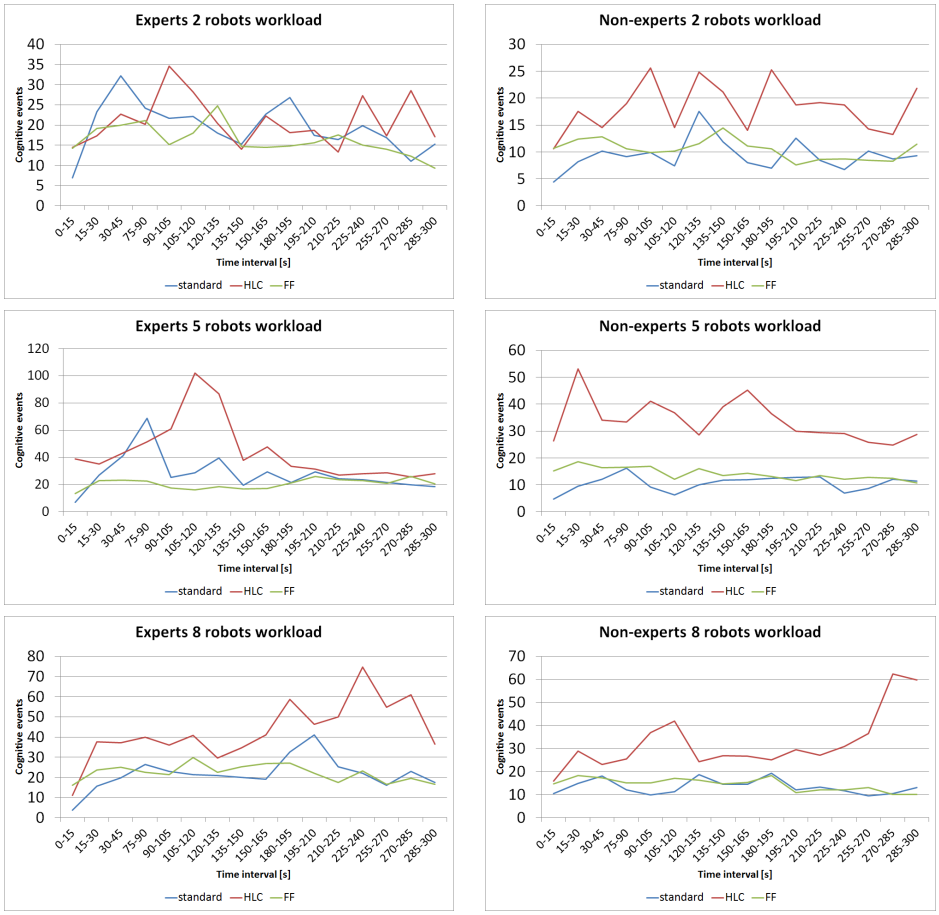

Figura 5.14: Workload. Nella colonna di sinistra i dati relativi agli utenti esperti, in quella di destra i dati relativi a quelli inesperti. Per ogni riga si presentano i dati relativi ad una specifica mappa e relativo numero di robot usati, nell'ordine 2, 5 e 8.

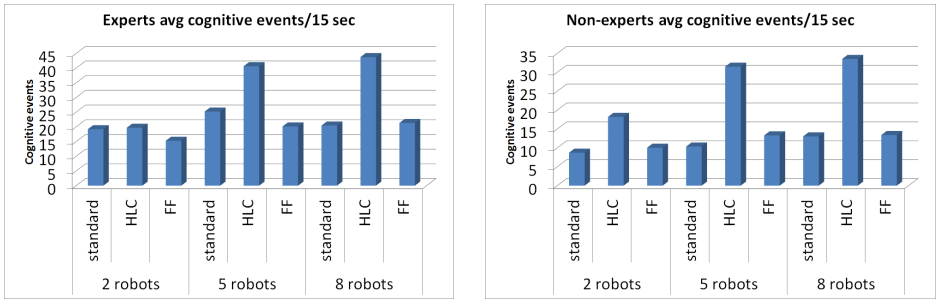

Figura 5.15: Workload medio per gli utenti esperti (sinistra) e non (destra).

è presentato in Figura 5.15, dove si può notare un workload mediamente più alto per gli utenti esperti. Riteniamo che ciò sia dovuto ancora una volta alla miglior conoscenza del sistema, che permette agli operatori esperti un'interazione più veloce ed efficace, con conseguente aumento del workload.

5.1.6 Distribuzione differenze di utilizzo dei singoli robot

Quando il numero di robot a disposizione è troppo elevato per essere gestito in maniera efficace, l'operatore potrebbe essere tentato di abbandonare il controllo di alcuni robot per ottenere prestazioni migliori dal controllo di un sottogruppo di essi. Questo tipo di comportamento si traduce in un utilizzo molto diverso da robot a robot e in prestazioni non ottimali dato il numero di robot a disposizione. Questo tipo di dinamiche si instaura, ovviamente, solo in prove con un numero elevato di robot.
Riportiamo in Figura 5.16 l'utilizzo medio dei robot durante le prove con 8 robot sia per operatori esperti che inesperti. La prima cosa che si può notare è una differenza sostanziale nel comportamento fra i due gruppi di operatori: se da una parte gli utenti inesperti abbandonano da subito alcuni robot per dedicarsi ad un sottogruppo, gli operatori esperti tentano un controllo di tutti i robot a disposizione, salvo poi abbandonarli quando non è più possibile una loro gestione efficace. I grafici di utilizzo dei robot da parte degli operatori esperti mostra quindi un utilizzo genericamente più omogeneo dei robot.
Nella Figura 5.17 mostriamo la differenza media percentuale nell'utilizzo dei robot in tutte e nove le prove, sia per operatori esperti che inesperti. La misura viene ricavata dalla formula seguente:

$$\delta usage = \frac{max(robot\ distance) - min(robot\ distance)}{\sum_{r=1}^{N} robot_r\ distance}$$

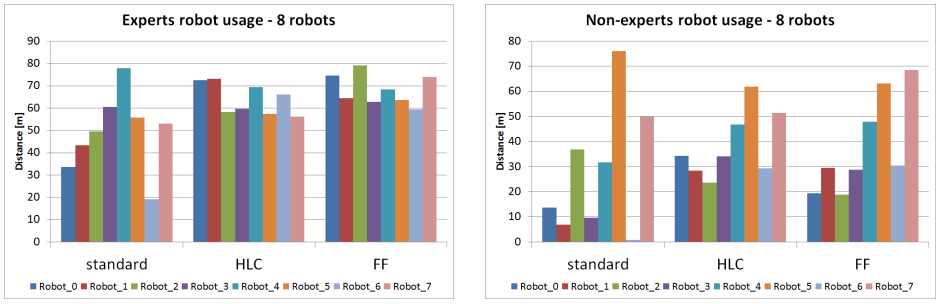

Figura 5.16: Utilizzo dei robot nelle tre prove con 8 robot. A sinistra i dati degli operatori esperti, a destra quelli degli inesperti.

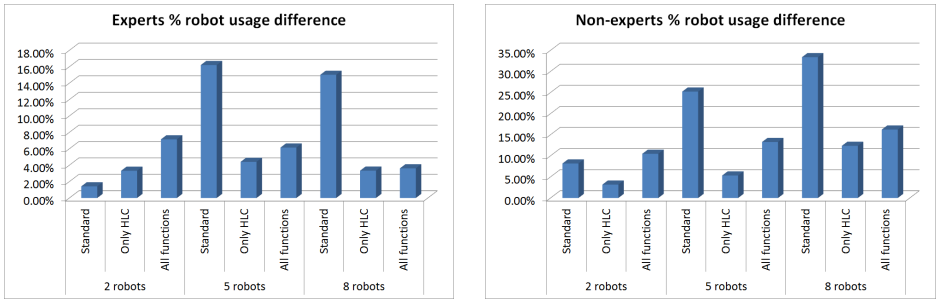

Figura 5.17: Differenza di utilizzo massima percentuale fra i robot mediata sui gruppi di tester esperti (sinistra) e inesperti (destra).

Si considera dunque la differenza massima di utilizzo fra i robot rispetto al totale della distanza percorsa.
Possiamo notare che con l'introduzione dell'autonomia, entrambi i gruppi di tester utilizzano i robot in modo più omogeneo sia con 5 ($F_{1,26} = 21.7$, $p < 0.0001$) che con 8 robot ($F_{1,26} = 18.5$, $p < 0.0001$). Possiamo collegare questo risultato al fatto che i robot trascurati dall'operatore esploreranno l'ambiente autonomamente invece di restare inutilizzati.
Un peggioramento delle prestazioni con l'introduzione del NFS si ha per gli utenti non esperti, probabilmente causata ancora una volta da una padronanza insufficiente dello strumento. Per gli utenti esperti le prestazioni in FF e HLC sono, invece, del tutto assimilabili.

5.2 Confronto con altri sistemi

I risultati ottenuti dal nostro sistema durante i test effettuati sottolineano come l'uso congiunto di HLC e NFS permetta di raggiungere prestazioni più elevate rispetto ad un sistema tradizionale, sia grazie ad una riduzione del

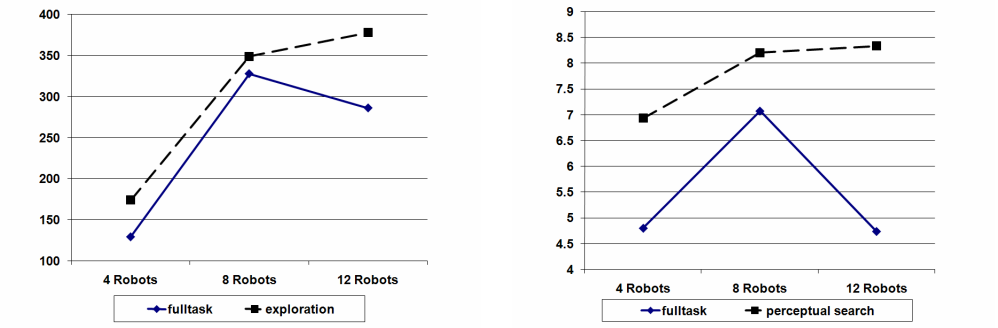

Figura 5.18: Prestazioni del sistema Steel in termini di area esplorata (a sinistra) e vittime trovate (a destra). La curva di nostro interesse è segnata in blu e a tratto continuo, e corrisponde a prove che simulano missioni USAR in cui l'operatore ha le stesse responsabilità che ha nel nostro sistema. Grafici tratti da [7].

workload che ad un uso migliorato dei robot e delle strategie d'esplorazione. In questa sezione confrontiamo i risultati ottenuti con quelli a disposizione dai sistemi presentati come paragone nel Capitolo 2.

5.2.1 Progetto Steel, Carnegie Mellon University

Disponiamo di dati relativi alle performance del sistema Steel in un ambito di utilizzo simile al nostro, con l'obiettivo di valutare la scalabilità del sistema multirobot dotato di autonomia col numero di robot [7]. Il test prevede il controllo del sistema multirobot Steel su una mappa ampia, usata nella RoboCup Rescue Simulation League nell'edizione 2006. Un solo operatore deve gestire 4, 8 e quindi 12 robot alla ricerca di vittime in una tipica missione di USAR. Altre prove sono state effettuate riducendo i compiti dell'operatore al solo esaminare i flussi video alla ricerca di vittime ed alla mera esplorazione, ma risultano di scarso interesse dati i nostri obiettivi.
Dai test eseguiti su 45 partecipanti non esperti di sistemi robotici risultano i dati presentati in Figura 5.18: le prestazioni del sistema aumentano fino alla soglia degli 8 robot da gestire contemporaneamente, oltre la quale le prestazioni precipitano. La linea blu continua è quella di nostro interesse, e rappresenta la condizione di test in cui l'operatore assolve agli stessi compiti previsti per gli operatori del nostro sistema. Notiamo come il deperimento di prestazioni coinvolga sia il numero di vittime trovate, sia l'area esplorata.

L'andamento delle prestazioni al crescere del numero di robot è comparabile con quello del nostro sistema, con un aumento di prestazioni fra 2 e 5 robot ed il mantenimento delle prestazioni fino ad 8 robot, con leggera flessione

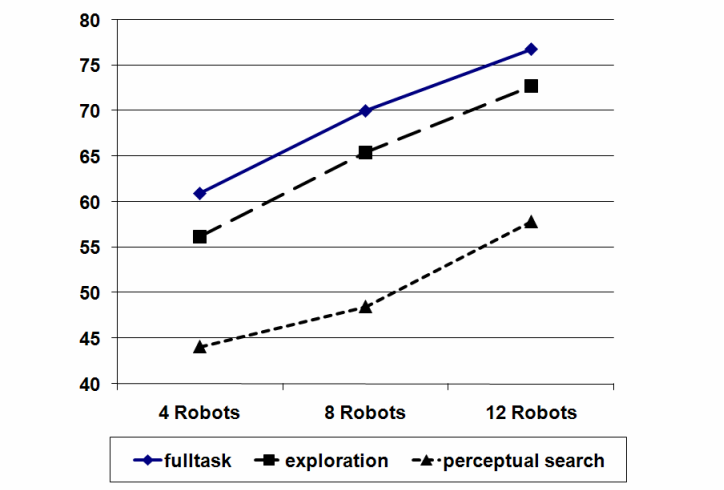

Figura 5.19: Workload misurato per il sistema Steel. Grafico tratto da [7].

nel caso di sistema non dotato di autonomia nell'esplorazione.
Altro dato di rilievo sottolineato in [7] è l'andamento del workload misurato. Lo strumento adottato per la sua misurazione è chiamato *NASA-TLX workload survey* e consiste di un questionario alla fine di ogni prova, quindi si tratta di una valutazione soggettiva da parte dell'operatore. I risultati confermano quanto verificato sul nostro sistema: in Figura 5.19 si mostra come il workload aumenti in maniera lineare col numero di robot e diventi presto insostenibile per l'operatore.

5.2.2 Team Michigan, University of Michigan

Il team Michigan ha presentato un sistema multirobot fortemente incentrato sull'autonomia e con un'interfaccia verso gli operatori innovativa. In [2] vengono presentati alcuni dati sulle prestazioni del sistema durante la competizione MAGIC 2010. I dati sono relativi ad un'unica prova ed hanno dunque scarsa significatività, ma rimangono interessanti per fare un confronto col nostro sistema.
Il focus sull'autonomia risulta evidente dalle modalità di utilizzo del sistema, fra le quali spiccano la piena autonomia (corrispondente alla nostra Free AI, dove i robot autonomamente decidono le politiche di esplorazione) e il funzionamento semi-automatico (corrispondente al nostro controllo tramite waypoint): queste sono le due uniche modalità d'interazione usate dall'operatore, con l'unica alternativa di fermare il robot. In Figura 5.20 riportiamo le modalità di utilizzo dei robot durante la competizione MAGIC 2010 nelle sue tre prove (o fasi) a difficoltà crescente.

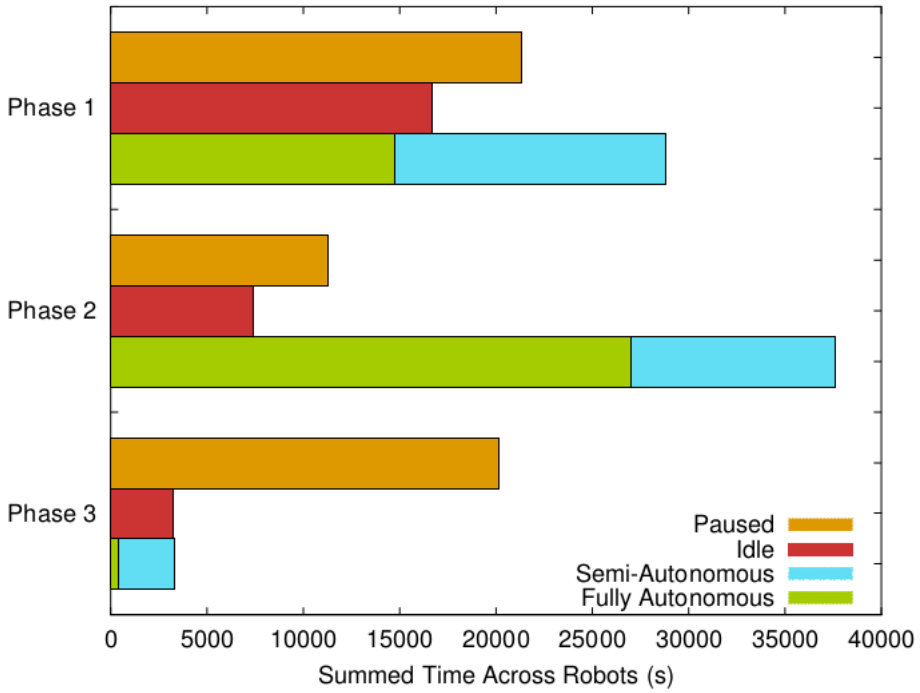

Figura 5.20: Modalità di funzionamento globale nelle 3 prove della competizione MAGIC 2010 del Team Michigan. Tratto da [2].

Possiamo notare come, nonostante il sistema sia dotato di procedure molto avanzate di esplorazione autonoma, il tempo speso dai robot nello stato idle e paused sia ancora molto rilevante.
Nella Figura 5.21 si mostra come sia distribuito l'utilizzo dei singoli robot, sempre nelle 3 prove eseguite. Notiamo una distribuzione non omogenea dell'utilizzo dei robot, ancora una volta a dispetto della quantità e qualità delle procedure autonome di esplorazione. Possiamo azzardare l'ipotesi che la sola autonomia, senza un'opportuna integrazione della componente umana nel ciclo di controllo del sistema (non favorita dalle regole della competizione MAGIC 2010) non consenta un utilizzo ottimale di tutte le potenzialità del sistema. Tuttavia, come già ribadito, è richiesta una mole di dati ben superiore per poter effettuare un'analisi affidabile.
Per ultimo mostriamo i dati relativi al numero di interventi al minuto del sistema del Team Michigan in Figura 5.22. Com'è ragionevole, dato il forte focus sull'autonomia del loro progetto, il numero di interventi è abbastanza limitato rispetto al nostro sistema. Risulta interessante constatare come l'andamento del numero di interventi al minuto sia impulsivo, con momenti di calma alternati a momenti di forte sforzo per l'operatore. La stessa dina-

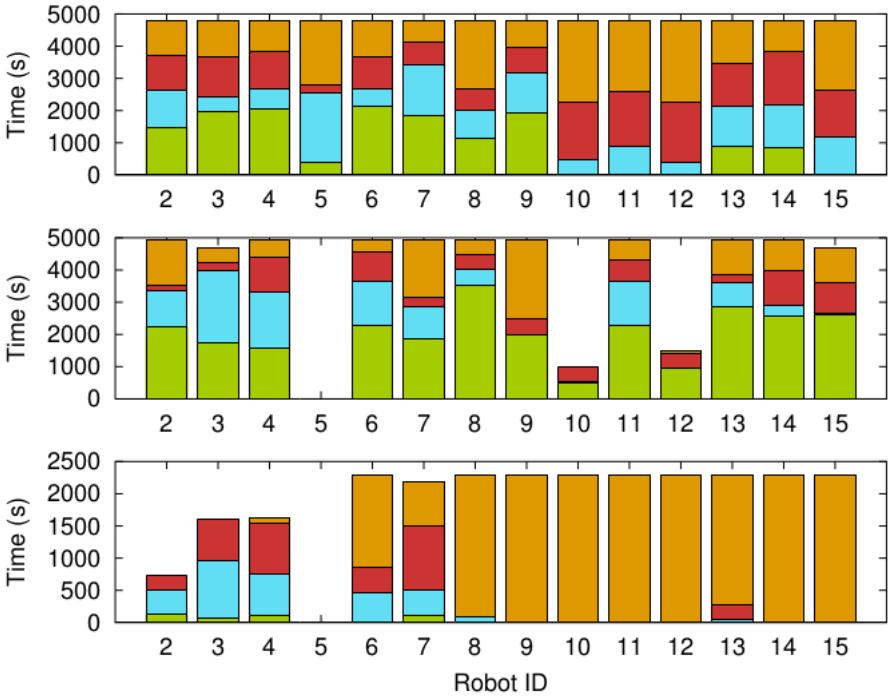

Figura 5.21: Modalità di funzionamento di ogni singolo robot nelle 3 prove della competizione MAGIC del Team Michigan. Tratto da [2].

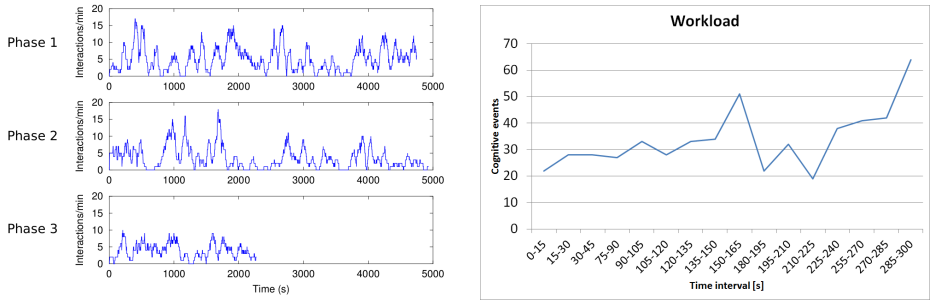

Figura 5.22: A sinistra: andamento workload delle singole fasi della competizione MAGIC 2010 per il Team Michigan. A destra un esempio di workload ricavato durante una delle prove sperimentali del nostro sistema.

mica si riscontra nei dati sugli eventi cognitivi del nostro sistema per ogni singola prova. Sempre in Figura 5.22 mostriamo a titolo d'esempio i dati relativi ad un tester esperto nella gestione di 8 robot col nostro sistema a confronto con i dati del Team Michigan. La natura impulsiva del workload è ragionevolmente dovuta alla natura stessa del compito, che prevede interventi intensi ma limitati nel tempo nella gestione delle eccezioni.

Grazie ai dati ricavati dai test siamo stati in grado di valutare le prestazioni del sistema e l'incremento delle stesse collegabile alle nostre innovazioni. Per confrontare direttamente la nostra soluzione con i sistemi attualmente concorrenti nella stessa area abbiamo partecipato ad una competizione internazionale, di cui descriviamo i risultati ottenuti nella prossima sezione.

5.3 RoboCup Rescue Simulation League

Il secondo banco di prova per il sistema PoAReT e per l'interfaccia d'interazione e relativa GUI, oggetto del presente lavoro di tesi, è stato la competizione internazionale RoboCup Rescue Simulation League 2012, che valuta le prestazioni di sistemi multirobot in missioni di USAR con l'utilizzo del simulatore USARSim.
L'edizione 2012 della competizione ha visto i team qualificati sfidarsi in 5 prove, in ognuna delle quali un singolo operatore era chiamato a controllare il sistema multirobot realizzato per trovare nel minor tempo possibile ed entro il limite di 20 minuti imposto, tutte le vittime presenti. Una volta trovata una vittima bisognava collocare un robot vicino ad essa (entro due metri di distanza). Ogni prova è caratterizzata da un numero diverso di

robot da gestire, nell'ordine di esecuzione delle prove: 4 robot, 7 robot, 4 robot, 8 robot e per finire 5 robot. La gestione efficace di gruppi numerosi di robot è dunque fondamentale per ottenere un buon punteggio in ogni prova. Due sono i tipi di mappe in cui si sono testate le capacità dei sistemi partecipanti.
Tre delle prove erano scenari tipici di ambienti urbani, strutturati e ampi. In questo tipo di mappa la principale sfida è riuscire a manovrare un gruppo numeroso di robot per trovare le vittime in un ambiente relativamente vasto. In queste prove il sistema PoAReT si è dimostrato superiore ai concorrenti, vincendo con ampio margine tutte e tre le prove e trovando tutte le vittime in un tempo minore al limite imposto: complessivamente, nelle 3 prove (cronologicamente la prima, la seconda e la quarta), sono state trovate tutte le 18 vittime in soli 39 minuti (su 60 minuti disponibili) contro le 16 vittime trovate in 59 minuti del secondo classificato.
Le altre due mappe testavano le prestazioni del sistema rispetto a problematiche di navigazione, sottoponendo ai robot ostacoli di vario genere da superare. Per questo tipo di mappe è stato necessario l'utilizzo massiccio della tele-operazione. La GUI e le innovazioni realizzate in questa tesi, così come i principi che hanno portato alla realizzazione dell'intero sistema, non sono stati pensati per questo tipo di applicazione, ma le prestazioni sono state comunque soddisfacenti, permettendo al team PoAReT di piazzarsi in queste prove al secondo e terzo posto, con un totale di 5 vittime trovate in 40 minuti contro le 7 vittime trovate in 33 minuti dei secondi classificati (i migliori in queste due prove), rendendo così possibile la vittoria finale della competizione.
Riteniamo fondamentale, nell'ambito della competizione, l'apporto della GUI e delle innovazioni che introduce, che hanno permesso, nelle mappe idonee, di ottenere prestazioni nettamente superiori a qualsiasi concorrente nella gestione di gruppi numerosi di robot.

Capitolo 6

Conclusioni e sviluppi futuri

In questo lavoro di tesi è stata realizzata una interfaccia di controllo innovativa per il controllo di sistemi multirobot. Si è inoltre progettata e implementata una GUI che permette a un operatore umano di gestire un numero elevato di robot in missioni USAR grazie all'aiuto dell'autonomia dei singoli robot.
Abbiamo introdotto l'argomento sottolineando come le attuali applicazioni di sistemi robotici in aree critiche (ambito USAR e militare soprattutto) sia limitata a soli robot tele-operati, mentre la ricerca accademica focalizza le sue forze su sistemi completamente autonomi. Sosteniamo che questo divario sia in buona parte dovuto allo scarso controllo che un operatore umano può esercitare sul sistema autonomo e alla conseguente "sfiducia" nello stesso.
Abbiamo anche sottolineato come, grazie alle diverse e complementari capacità di sistemi automatici ed esseri umani, un controllo mixed initiative porti a prestazioni migliori sia rispetto a sistemi comandati manualmente sia a quelli puramente autonomi.
Il passo successivo è stato progettare un'interfaccia d'interazione fra operatore e sistema che garantisse un buon livello di controllo e permettesse contemporaneamente di sfruttare a pieno l'autonomia dei singoli robot, asservendola all'essere umano responsabile del sistema. In questa fase abbiamo individuato i due aspetti di maggior rilievo nell'affrontare le problematiche citate: workload e situation awareness.
Analizzando i flussi di informazioni che viaggiano da sistema a operatore e viceversa abbiamo progettato due innovazioni per migliorare l'interfaccia di controllo: NFS, un sistema di filtraggio delle notifiche e delle richieste provenienti dai robot, e HLC, un metodo per la specifica di comandi di alto livello che permettano di modificare efficacemente le politiche di esplorazio-

ne autonoma.
Una volta individuate le misure prestazionali da adottare abbiamo eseguito dei test sperimentali per validare la bontà della GUI e l'effetto delle innovazioni introdotte. Per confrontare il sistema con altri progetti con obiettivi simili, abbiamo partecipato alla competizione RoboCup Rescue Simulation League 2012 con il team PoAReT, ottenendo la vittoria grazie anche alla gestione efficiente dei robot consentita dalla GUI implementata.

L'intero lavoro di tesi ha portato a dei buoni e incoraggianti risultati sugli effetti delle innovazioni introdotte per il controllo del sistema, permettendo, in particolare ad operatori esperti, di gestire un numero più elevato di robot rispetto ad una interfaccia tradizionale.
Il livello di controllo garantito sul sistema con gli HLC si è dimostrato superiore a quello fornito dalla normale autonomia. A riprova di tale risultato, tutti i tester hanno, in ogni prova, preferito assegnare dei comandi di alto livello ai robot piuttosto che lasciarli esplorare liberamente senza indicazioni sulla politica di esplorazione da adottare.
Il livello di workload dell'operatore, che a causa dell'introduzione dell'autonomia tende ad aumentare per la più intensa attività dei robot, è stato notevolmente ridotto dal NFS, permettendo una gestione più agevole e consapevole del sistema. Inoltre si è favorita con il NFS l'adozione di politiche di controllo avanzate guidate dagli eventi sollevati dai robot, utili in particolare quando si è reso necessario gestire gruppi di robot numerosi.
Ovviamente, sia le prestazioni di NFS che di HLC dipendono dalle funzionalità dei singoli robot. Il NFS è stato testato con notifiche molto affidabili da parte dei robot, e in particolare quelle relative al modulo di victim detection. Se le prestazioni dei moduli che inviano le notifiche fossero pessime, il NFS non potrebbe in ogni caso migliorare molto le prestazioni del sistema. Lo stesso vale per gli HLC, le cui prestazioni dipendono dalla bontà e affidabilità dell'autonomia dei singoli robot.

In questo lavoro di tesi si è compiuto un piccolo passo avanti nella ricerca di metodi per l'integrazione dell'operatore umano all'interno del ciclo di controllo di sistemi multirobot. Possibili sviluppi futuri riguardano:

Miglioramenti tecnici alla GUI realizzata. Nell'ambito della HRI qualsiasi miglioria alla GUI tramite cui l'operatore controlla il sistema può avere effetti rilevanti ai fini delle prestazioni del sistema nel suo complesso. Miglioramenti in questa direzione permetterebbero di migliorare ulteriormente le prestazioni raggiungibili con l'impostazione da noi data al sistema, basata su HLC e NFS. A titolo d'esempio si po-

trebbe integrare la soluzione proposta in questa tesi con l'utilizzo di una mappa 3D dell'ambiente, come proposto da [10], eventualmente arricchendo la stessa con informazioni sui flussi video, innovazione introdotta da [11]. Si potrebbero così valutare la compatibilità delle diverse soluzioni e le prestazioni complessive.

Valutazione del sistema. I test da noi eseguiti hanno permesso di evidenziare alcuni aspetti positivi della nostra soluzione ed alcuni miglioramenti rispetto al caso base delle interfacce di controllo tradizionali nello stesso ambito d'utilizzo. Test più approfonditi, con un maggior numero di partecipanti ed utilizzando misure più avanzate ed affidabili per cogliere gli aspetti di maggiore importanza possono fornire un'idea più chiara dei vantaggi che la nostra soluzione potrebbe apportare. In particolare, uno sviluppo futuro può prevedere l'esecuzione di test raccogliendo dati sulla situation awareness tramite il metodo SAGAT [44] e la raccolta di informazioni sul workload tramite segnali fisiologici [9]. L'allungamento della durata delle prove potrebbe permettere di valutare anche le dinamiche di affaticamento dell'operatore durante la missione, soprattutto quando sottoposto a forte stress, e l'ampliamento del numero di partecipanti potrebbe dare una maggior affidabilità statistica ai risultati.

Test sul campo. Il sistema sviluppato è basato su un simulatore, seppure realistico e di eccellente qualità. Per quanto riguarda le dinamiche di controllo del sistema questo aspetto potrebbe influenzare i risultati in modo rilevante. Sarebbe dunque interessante testare il sistema sul campo, adattandolo all'utilizzo con robot reali, e ponendo l'operatore in una condizione simile a quella reale di una missione USAR. Si potrebbe così valutare l'efficacia delle innovazioni introdotte sul campo e verificare che l'operatore ne tragga dei vantaggi nella ricerca e salvataggio delle possibili vittime.

Sistemi multirobot sul campo. Come abbiamo evidenziato in questa tesi, l'applicazione di sistemi robotici sul campo in missioni USAR è limitata a pochi casi di robot molto avanzati e costosi esclusivamente tele-operati. Uno sviluppo futuro necessario per un'eventuale applicazione pratica deve riguardare necessariamente test sul campo di sistemi multirobot con robot piccoli, agili e poco costosi.

Questo lavoro costituisce dunque un piccolo tassello in un ben più lungo e impervio percorso che ha come futuro ideale un mondo in cui non sarà più necessario rischiare la vita delle persone per il salvataggio di altre persone o addirittura di beni materiali, ma il compito sarà interamente delegato a innovativi ed autonomi sistemi robotici che permettano ai soccorritori di com-

piere solo decisioni strategiche ed eventualmente risolvere le problematiche impreviste che sorgono durante le missioni.

Bibliografia

[1] Siciliano, B., Khatib, O.: Chapter 50. In: Springer Handbook of Robotics. Springer (2008) 1151–1173

[2] Olson, E., Strom, J., Morton, R., Richardson, A., Ranganathan, P., Goeddel, R., Bulic, M., Crossman, J., Marinier, B.: Progress towards multi-robot reconnaissance and the MAGIC 2010 competition. Journal of Field Robotics **29**(5) (2012) 762–792

[3] Wang, J., Lewis, M.: Human control for cooperating robot teams. In: Proceedings of the ACM/IEEE International conference on Human-Robot Interaction. (2007) 9–16

[4] Prewett, M.S., Johnson, R.C., Saboe, K.N., Elliott, L.R., Coovert, M.D.: Managing workload in human-robot interaction: A review of empirical studies. Computers in Human Behavior **26**(5) (2010) 840–856

[5] Petersen, K., von Stryk, O.: Towards a general communication concept for human supervision of autonomous robot teams. In: Proceedings of the Fourth International Conference on Advances in Computer-Human Interactions. (2011) 228–235

[6] Chen, J.Y.C., Barnes, M.J., Harper-Sciarini, M.: Supervisory control of multiple robots: Human-performance issues and user-interface design. IEEE Transactions on Systems, Man, and Cybernetics, Part C **41**(4) (2011) 435–454

[7] Lee, W.H., Lewis, M., Velagapudi, P., Scerri, P., Sycara, K.: How search and its subtasks scale in n robots. In: Proceedings of the ACM/IEEE International conference on Human Robot Interaction. (2009) 141–147

[8] Nevatia, Y., Stoyanov, T., Rathnam, R., Pfingsthorn, M., Markov, S., Ambrus, R., Birk, A.: Augmented autonomy: Improving human-robot team performance in urban search and rescue. In: Proceedings of the

IEEE/RSJ International Conference on Intelligent Robots and Systems. (2008) 2103–2108

[9] Steinfeld, A., Fong, T.W., Kaber, D., Lewis, M., Scholtz, J., Schultz, A., Goodrich, M.: Common metrics for human-robot interaction. In: Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-Robot Interaction. (2006) 33–40

[10] Crossman, J., Marinier, R., Olson, E.: A hands-off, multi-robot display for communicating situation awareness to operators. In: Proceedings of the International Conference on Collaboration Technologies and Systems. (2012) 109–116

[11] Velagapudi, P., Kleiner, A., Brooks, N., Scerri, P., Lewis, M., Sycara, K.: RoboCup Rescue Simulation League 2010 Team STEEL (USA). Technical report, Carnegie Mellon University, Pittsburgh (2010)

[12] LeBlanc, P., Fahy, R.: Firefighter Fatalities in the United States - 2005. National Fire Protection Association (2006)

[13] Tridata Corporation National Fire: Firefighter Fatality Retrospective Study. General Books (2011)

[14] Casper, J., Murphy, R.R.: Workflow study on Human-Robot Interaction in USAR. In: Proceedings of IEEE International Conference on Robotics and Automation. (2002) 1997–2003

[15] Wang, J., Lewis, M., Hughes, S., Koes, M., Carpin, S.: Validating USARsim for use in HRI research. In: Proceedings of the 49th Annual Meeting of the Human Factors and Ergonomics Society. (2005) 457–461

[16] RoboCup. `http://www.robocup.org/` (2012) [Online; accessed 5-August-2012].

[17] RoboCup Robot League - RoboCup Federation Wiki. `http://wiki.robocup.org/wiki/Robot_League` (2012) [Online; accessed 7-August-2012].

[18] RoboCup Rescue Simulation League - RoboCup Federation Wiki. `http://wiki.robocup.org/wiki/Rescue_Simulation_League` (2012) [Online; accessed 7-August-2012].

[19] DSTO Events MAGIC 2010: Super-smart robots wanted for international challenge. `http://www.dsto.defence.gov.au/MAGIC2010/` (2012) [Online; accessed 2-September-2012].

[20] Elrob-Website. http://www.elrob.org/ (2012) [Online; accessed 2-September-2012].

[21] Anderson, M., Jenkins, O.C., Osentoski, S.: Recasting robotics challenges as experiments. IEEE Robotics and Automation Magazine **18**(2) (2011) 10–11

[22] Defense Advanced Research Projects Agency - DARPA. http://www.darpa.mil/ (2012) [Online; accessed 2-September-2012].

[23] Sullins, J.P.: RoboWarfare: can robots be more ethical than humans on the battlefield? Ethics and Information Technology **12**(3) (2010) 263–275

[24] Sellers, D.P., Ramsbotham, A.J., Bertrand, H., Karvonides, N.: International assessment of unmanned ground vehicles. Technical report, Institute for Defence Analyses (2008)

[25] Lin, P., Bekey, G., Abney, K.: Autonomous military robotics: Risk, ethics, and design. Technical report, US Department of Navy, Office of Naval Research (2008)

[26] Yanco, H.A.: Classifying human-robot interaction: An updated taxonomy. In: Proceedings of the 2004 IEEE International Conference on Systems, Man, and Cybernetics. (2004) 2841–2846

[27] Scholtz, J.: Theory and evaluation of human robot interactions. In: Proceedings of the 36th Annual Hawaii International Conference on System Sciences. (2003) 125

[28] Sierhuis, M., Bradshaw, J.M., Acquisti, A., van Hoof, R., Jeffers, R., Uszok, A.: Human-agent teamwork and adjustable autonomy in practice. In: Proceedings of the Seventh International Symposium on Artificial Intelligence, Robotics and Automation in Space. (2003)

[29] Parasuraman, R., Sheridan, T.B., Wickens, C.D.: A model for types and levels of human interaction with automation. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans **30**(3) (2000) 286–297

[30] Endsley, M.R.: Level of automation effects on performance, situation awareness and workload in a dynamic control task. Ergonomics **42**(3) (1999) 462–492

[31] Kaber, D.B., Endsley, M.R.: The effects of level of automation and adaptive automation on human performance, situation awareness and workload in a dynamic control task. Theoretical Issues in Ergonomics Science **5**(2) (2004) 113–153

[32] Huang, H.M.: Autonomy levels for unmanned systems (ALFUS) framework: safety and application issues. In: Proceedings of the 2007 Workshop on Performance Metrics for Intelligent Systems. (2007) 48–53

[33] Kaber, D.B., Onal, E., Endsley, M.R.: Design of automation for telerobots and the effect on performance, operator situation awareness, and subjective workload. Human Factors and Ergonomics in Manufacturing & Service Industries **10**(4) (2000) 409–430

[34] Hardin, B., Goodrich, M.A.: On using mixed-initiative control: a perspective for managing large-scale robotic teams. In: Proceedings of the 4th ACM/IEEE International Conference on Human Robot Interaction. (2009) 165–172

[35] Velagapudi, P., young Kwak, J., Scerri, P., Lewis, M., Sycara, K.: RoboCup Rescue Simulation League 2008 Team STEEL (USA). Technical report, Carnegie Mellon University, Pittsburgh (2008)

[36] Yanco, H.A., Baker, M., Casey, R., Keyes, B., Thoren, P., L.Drury, J., Few, D., Nielsen, C., Bruemmer., D.: Analysis of Human-Robot Interaction for urban search and rescue. In: Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics. (2006)

[37] Graber, T., Kohlbrecher, S., Meyer, J., Petersen, K., von Stryk, O.: RoboCup Rescue Robot League 2011 Team Hector Darmstadt (Germany). Technical report, Technische Universität Darmstadt (2011)

[38] Graber, T., Kohlbrecher, S., Meyer, J., Petersen, K., von Stryk, O., Klingauf, U.: RoboCup Rescue Robot League 2012 Team Hector Darmstadt (Germany). Technical report, Technische Universität Darmstadt (2012)

[39] Hinze, A., Sachs, K., Buchmann, A.: Event-based applications and enabling technologies. In: Proceedings of the Third ACM International Conference on Distributed Event-Based Systems. (2009) 1–15

[40] Kadous, M.W., Sheh, R.K.M., Sammut, C.: Effective user interface design for rescue robotics. In: Proceedings of the 1st

ACM SIGCHI/SIGART conference on Human-robot interaction. (2006) 250–257

[41] Drury, J.L., Keyes, B., Yanco, H.A.: LASSOing HRI: analyzing situation awareness in map-centric and video-centric interfaces. In: Proceedings of the ACM/IEEE international conference on Human-robot interaction. (2007) 279–286

[42] Driewer, F., Sauer, M., Schilling, K.: Design and evaluation of an user interface for the coordination of a group of mobile robots. In: Proceedings of the 17th International Symposium on Robot and Human Interactive Communication. (2008) 237–242

[43] Lewis, M., Sycara, K.: Effects of automation on situation awareness in controlling robot teams. In: Proceedings of the Fourth International Conference in Advances in Human Computer Interaction. (2011)

[44] Endsley, M.R.: Situation awareness global assessment technique (SAGAT). In: Proceedings of the IEEE National Aerospace and Electronics Conference. (1988) 789–795

[45] Endsley, M.R.: Direct measurement of situation awareness:validity and use of SAGAT. In Endsley, M.R., Garland, D.J., eds.: Situation awareness analysis and measurement. Lawrence Erlbaum Associates (2000) 147–173

[46] Selcon, S.J., Taylor, R.M.: Evaluation of the situational awareness rating technique (SART) as a tool for aircrew systems design. In: Proceedings of the Situational Awareness in Aerospace Operations. (1990) 23–53

[47] Endsley, M.R.: A comparative analysis of SAGAT and SART for evaluations of situation awareness. In: Proceedings of the 42th Annual Meeting of the Human Factors and Ergonomics Society. (1998) 82–86

[48] Endsley, M.R.: Measurement of situation awareness in dynamic systems. Human Factors: The Journal of the Human Factors and Ergonomics Society **37**(1) (1995) 65–84

[49] Endsley, M.R., Sollenberger, R., Stein, E.: Situation awareness: a comparison of measures. In: Proceedings of the Human Performance, Situation Awareness and Automation: User Centered Design for the New Millenium Conference. (2000)

[50] Jex, H.R.: Measuring mental workload: Problems, progress, and promises. In Hancock, P.A., Meshkati, N., eds.: Human Mental Workload. Volume 52 of Advances in Psychology. North-Holland (1988) 5 – 39

[51] Nygren, T.E.: Psychometric properties of subjective workload measurement techniques: implications for their use in the assessment of perceived mental workload. Human Factors: The Journal of the Human Factors and Ergonomics Society **33**(1) (1991) 17–33

[52] Lee, W.H., S, C., Lewis, M., P, V., Scerri, P., Sycara, K.: Human teams for large scale multirobot control. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, IEEE (2009) 1306–1311

[53] Wang, J., Lewis, M.: Assessing cooperation in human control of heterogeneous robots. In: Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction. (2008) 9–16

[54] Van Der Vecht, B., Dignum, F., Meyer, J.J.C., Neef, M.: A dynamic coordination mechanism using adjustable autonomy. In: Proceedings of the 2007 international conference on Coordination, Organizations, Institutions, and Norms in agent systems III. (2008) 83–96

[55] Toups, Z.O., Kerne, A.: Implicit coordination in firefighting practice: design implications for teaching fire emergency responders. In: Proceedings of the SIGCHI conference on Human factors in computing systems. (2007) 707–716

[56] Branlat, M., Fern, L., Voshell, M., Trent, S.: Understanding coordination challenges in urban firefighting: A study of critical incident reports. In: Proceedings of the 53th Human Factors and Ergonomics Society Annual Meeting. (2009) 284–288

[57] Basilico, N., Amigoni, F.: Exploration strategies based on multi-criteria decision making for searching environments in rescue operations. Autonomous Robots **31**(4) (2011) 401–417

[58] PoAReT. `http://home.dei.polimi.it/amigoni/research/PoAReT.html` (2012) [Online; accessed 2-August-2012].

[59] Pfingsthorn, M.: Robocup rescue virtual robots: Wireless simulation server documentation. Technical report (2008)

[60] Comer, D.E.: Internetworking with TCP/IP Vol.1: Principles, Protocols, and Architecture. 4th edn. Prentice Hall (2000)

[61] Amigoni, F., Caltieri, A., Cipolleschi, R., Conconi, G., Giusto, M., Luperto, M., Mazuran, M.: PoAReT team description paper. In: RoboCup 2012 (CDROM Proceedings), Team Description Paper, Rescue Simulation League. (2012)

[62] MobileRobots Pioneer 3-AT (P3AT) research robot platform. http://www.mobilerobots.com/researchrobots/p3at.aspx [Online; accessed 2-September-2012].

[63] Adept Technology, I.: Pioneer 3 - AT Datasheet. Technical report (2011)

[64] fuRo:Kenaf. http://furo.org/en/works/kenaf.html [Online; accessed 2-September-2012].

[65] Meyer, T.: Micro unmanned aerial vehicle with autonomous flight and navigation capabilities and modular payloads. Technical report (2007)

[66] Air Robot Products - AirRobot AR100B, AirRobotUK® - Supplier of Unmanned Aerial Vehicles (UAV's) and Unmanned Aerial Systems (UAS's) Worldwide. Air Robot. http://www.airrobot-uk.com/air-robot-products.htm [Online; accessed 2-September-2012].

[67] Lavalle, S.M., Kuffner, J.J.: Rapidly-Exploring Random Trees: Progress and Prospects. In Peters, A.K., ed.: Algorithmic and Computational Robotics: New Directions. Donald, B. R. and Lynch, K. M. and Rus, D. (2001) 293–308

[68] Zlot, R., Stentz, A., Dias, M.B., Thayer, S.: Multi-robot exploration controlled by a market economy. In: Proceedings 2002 IEEE International Conference on Robotics and Automation. (2002) 3016–3023

[69] Viola, P., Jones, M.J.: Robust Real-Time Face Detection. International Journal of Computer Vision **57**(2) (2004) 137–154

[70] Chen, J.Y.C., Thropp, J.E.: Review of low frame rate effects on human performance. IEEE Transactions on Systems, Man, and Cybernetics, Part A **37**(6) (2007) 1063–1076

[71] Qt - Cross-platform application and UI framework – Qt - A cross-platform application and UI framework. http://qt.nokia.com/ (2012) [Online; accessed 1-September-2012].

[72] Kane, B., Velagapudi, P., Scerri, P.: Asking for help through adaptable autonomy in robotic search and rescue. In Bai, Q., Fukuta, N., eds.: Advances in Practical Multi-Agent Systems. Volume 325 of Studies in Computational Intelligence. Springer Berlin / Heidelberg (2011) 339–357

[73] Neter, J., Kutner, M.H., Nachtsheim, C.J., Wasserman, W.: Applied Linear Statistical Models. McGraw-Hill/Irwin (1996)