UNIVERSITA' DEGLI STUDI DI ROMA
"TOR VERGATA"

FACOLTA' DI INGEGNERIA

CORSO DI LAUREA IN INGEGNERIA
DELL'AUTOMAZIONE

"Uso del filtro di Kalman esteso per la localizzazione e la pianificazione del moto del robot Nomad 150"

Relatore:	Laureanda:
Ing.Francesco Martinelli	**Giulia Franchi**

Anno Accademico 2005/2006

Ai miei genitori

Introduzione

Il problema della localizzazione di robot mobili, in ambienti non strutturati, rappresenta uno dei filoni di ricerca tra i più attivi ed importanti. Poter conoscere con sufficiente precisione il posizionamento del robot nel mondo reale costituisce un'esigenza imprescindibile per gli sviluppi futuri della robotica mobile. Il modo più semplice per rispondere alla domanda fondamentale nel problema della localizzazione, ovvero "dove mi trovo?", consiste nello sfruttamento di informazioni odometriche (mediante encoder calettati sui motori di trazione è possibile calcolare il numero di giri compiuti dalle ruote del robot in un lasso di tempo unitario e stimare quindi lo spazio percorso) e dei dati provenienti dai sonar. L'odometria così come le informazioni sensoriali sono tuttavia affette da errori che possono facilmente causare nel tempo una deriva della posizione stimata del robot rispetto a quella reale con conseguente perdita totale dell'orientamento. Le principali componenti d'errore che determinano tali derive sono raggruppabili in due categorie distinte: errori sistematici (anche detti deterministici) e errori non sistematici (anche detti non deterministici). Rientrano nella prima categoria gli errori causati da disallineamento fra le ruote motrici, differenze fra diametro reale delle ruote e diametro utilizzato per la stima, limitata risoluzione dei dispositivi sensoriali utilizzati per la rilevazione di posizione. Della categoria degli errori non sistematici fanno invece parte gli errori causati ad esempio da movimento del robot su superfici non perfettamente piane, profili di velocità tali da determinare slittamento delle ruote, ecc.. Mentre i primi possono essere eliminati ricorrendo ad una corretta calibrazione del sistema, gli errori non deterministici sono rappresentabili attraverso modelli di rumore e dunque determineranno una sempre presente incertezza sulla posizione stimata. L'incertezza sul reale posizionamento del robot cresce molto più che linearmente all'aumentare della distanza percorsa. Addirittura in presenza di movimenti compositi (lineari e rotazionali) la regione di incertezza non solo si incrementa ma non rimane neppure perpendicolare alla direzione di moto. Se si sceglie il metodo della fusione sensoriale, per avere delle misure accurate, bisogna anche considerare attentamente la cinematica del veicolo su cui la dobbiamo usare. Infatti determinate scelte progettuali (dal punto di vista cinematico) possono aumentare o diminuire la sensibilità dell'odometria e l'accuratezza dei dati ricevuti dai sonar rispetto a fenomeni quali lo slittamento delle ruote o gli urti.

Tecniche di fusione sensoriale possono essere utilizzate quando l'insieme dei rilevamenti abbia caratteristiche di ridondanza e complementarietà. Quando più sensori forniscono informazione si ha un'accrescimento dell'affidabilità delle misure poichè ogni sensore offre una misura ridondante , ma allo stesso tempo si ha un'informazione più accurata, contenente caratteristiche dell'ambiente impossibili da rilevare con un solo sensore, in minor tempo e ad un costo inferiore. È difficile ipotizzare metodi generali per la progettazione di un sistema di fusione sensoriale, in quanto fortemente vincolato all'architettura di elaborazione e rilevamento dati. Si devono valutare accuratamente gli utilizzi per i quali il robot è stato progettato, il tipo di ambiente nel quale opererà, le tipologie e le prestazioni dei vari sensori presenti nonché le caratteristiche di movimento quali, ad esempio, la massima velocità o raggio d'azione.

Nella tesi sviluppata sono stati elaborati dati odometrici e rilevamenti acustici, con una metodologia basata sul filtro di Kalman Esteso. L'impiego di questo filtro è giustificato se si considera la rumorosità di sensori acustici di bassa qualità e i risultati grossolani derivanti dall'integrazione diretta di dati tachimetrici, da cui deriva la necessità d'impiego di tecniche di fusione sensoriale. La semplice integrazione numerica dei rilevamenti odometrici, infatti, porta rapidamente ad un disallineamento tra la posizione effettiva del robot e quella misurata. La relazione è strutturata in una sezione in cui si descrivono le caratteristiche del robot, in una in cui si introduce la teoria del filtro di Kalman discreto, la sua evoluzione in filtro di Kalman Esteso ed infine l' applicazione dello stesso al Nomad 150. Nel'ultimo capitolo si riportano i risultati sperimentali ottenuti dall'implementazione del filtro.

Indice

Capitolo 1

Caratteristiche del robot Nomad 150

Il lavoro simulativo e sperimentale è stato condotto con l'ausilio del robot mobile Nomad 150, realizzato dalla Nomadic Software Inc. (vedi figura1).
Il robot è equipaggiato di un computer portatile ASUS sul quale sono presenti delle librerie software necessarie per il funzionamento del robot stesso e un simulatore con il quale si possono costruire virtualmente ambienti e situazioni.

1.1 Descrizione Hardware

Il Nomad 150 è un robot dotato di un sistema meccanico di spostamento su di un piano costituito da tre ruote simmetriche azionate da due motori, uno per spostamenti traslazionali e uno per quelli di rotazione. Data la loro simmetria le ruote non sono indipendenti ma traslano insieme e ruotano tutte nello stesso verso. Un terzo motore permette la rotazione della torretta. Il raggio di rotazione è pari a zero, cioè il robot può ruotare su se stesso senza traslare. Il sistema è *anolonomo* in quanto presenta dei vincoli non integrabili nella velocità.

Figura 1.1: Robot Nomad 150

Il controllo dei motori è affidato ad un microprocessore MOTOROLA MC68008/ASIC che permette al robot di raggiungere una velocità traslazionale massima di 20 inch/sec (0.508 m/sec) mentre la velocità di rotazione delle ruote e della torretta non può superare i 45 gradi/sec.

Il sistema sensoriale del Nomad 150 è composto da tre tipi di sensori: tattili, odometrici e acustici. Le informazioni tattili sono date dal SENSUS 100^{TM} TACTILE SYSTEM, un complesso di 20 sensori di pressione (switch),detti BUMPER, disposti in due anelli di 10 sensori che coprono la circonferenza del robot con una risoluzione di 18 gradi. I dati acustici provengono dal SENSUS 200^{TM} SONAR RANGING SYSTEM, un sistema di 16 sensori ad ultrasuoni, detti SONAR, alimentati a 5 volts che restituiscono valori da 6 a 255 pollici con una precisione dell'1'% sul range totale. Il sistema odometrico è composto da ENCODER, posizionati sulle ruote del robot, che acquisendo dati sul movimento dei tre motori permettono di conoscere la posizione del robot definita mediante le coordinate (x, y) relative al sistema di coordinate avente centro nella posizione in cui si trova inizialmente il robot.

Il Nomad 150 è alimentato per mezzo di tre batterie ricaricabili ACCORD NC 12120 al piombo secco da 12 V con una capacità di 12 Ah. Tali batterie hanno una durata massima di circa 6 ore.

Il robot non è dotato di alcun computer di bordo, è necessario quindi interfaccialo con il PC portatile ASUS L3400 Tp con processore Intel Pentium 4 con una frequenza di 2 GHz e 256 MB di memoria RAM. La connessione con il portatile può avvenire in due modi: tramite cavo seriale o con collegamento wireless. Nel secondo caso si fa uso di un radio-modem *Mercury*-EN prodotto dalla Nomadic Communications Inc, che si collega alla porta seriale del PC e lavora ad una frequenza di 2,4 GHz con una connessione TCP. Di questo tipo di collegamento è stata effettuata solo qualche sporadica prova, mentre tutti i test sono stati realizzati posizionando il computer sulla torretta del Nomad effettuando quindi la connessione tramite cavo seriale. L'unico inconveniente è dovuto alla scarsa autonomia delle batterie del computer per cui nelle prove con il robot reale è stato necessario l'utilizzo di una prolunga collegata alla rete elettrica per alimentarle, mentre invece le batterie del robot si sono dimostate molto efficienti e in grado di farlo funzionare per più di qualche giorno.
Il robot può essere inoltre comandato anche tramite joystick.

1.2 Descrizione Software

Il *Nomad*150 possiede delle librerie di funzioni scritte in C (ma utilizzabili anche in C++) che forniscono un'architettura di sistema a basso livello. Sono: Nclient.h e Ndirect.h e permettono di impartire al robot tutte le principali funzioni di movimento (rotazione e traslazione), nonché di gestione dei

sensori.
In aggiunta viene fornito un software di gestione chiamato Nserver che funziona da server-simulatore: permette di vedere a schermo i dati che il robot ottiene dall'ambiente, quali la sua posizione e le rilevazioni dei sensori, e agisce anche da simulatore per testare il robot senza doverlo far muovere realmente. Infatti esso genera un'interfaccia grafica (vedi figura1.2) con i dati dei sensori, la posizione del robot e la mappa dello spazio in cui esso si muove.

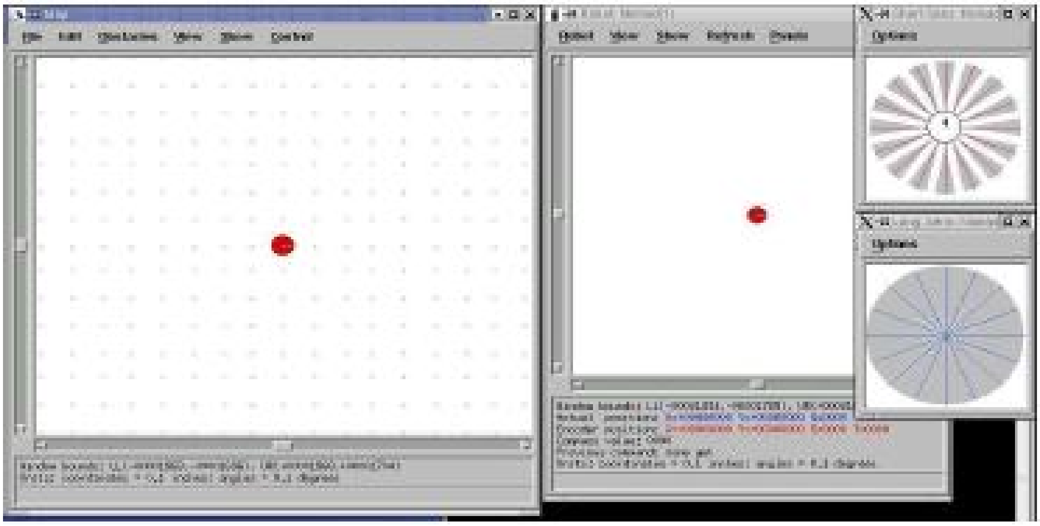


Figura 1.2: Ambiente simulativo Nserver

Esistono due possibili modi per eseguire un programma sul robot:

1. utilizzare un terminale di emulazione (*Nomad* sotto Linux o *hyperterminal*sotto Windows) per far eseguire al robot semplici operazioni;

2. eseguire operazioni complesse (un programma C/C++) grazie alle librerie fornite o ad *Nserver*, in due modi diversi:

   - creare un sistema *client-server* tra l'eseguibile e Nserver, utilizzando quest'ultimo come interfaccia con il robot stesso (ciò permette di visualizzare a schermo in *real-time* le operazioni che esso svolge);

- interfacciarsi direttamente al Nomad senza dover avviare alcun server.

Come già detto esistono due librerie di gestione: Nclient.h e Ndirect.h. La prima se compilata insieme al programma come file oggetto renderà il programma utente *client* e l'Nserver farà da server (nella compilazione è necessario includere il file *Nclient.o*). La seconda libreria invece serve nel caso sia già testato il programma e si è certi del corretto funzionamento, infatti permette un collegamento diretto tra programma e robot senza l'ausilio di alcun server (si include invece *Ndirect.o*).

Capitolo 2

Introduzione al filtro di Kalman discreto

2.1 Generalità sul filtro

Nel 1960 R. E. Kalman pubblicò un articolo in cui viene descritta una soluzione ricorsiva per il problema del filtraggio lineare di dati discreti. Da quel momento, anche grazie ai notevoli progressi nella computazione digitale, il filtro di Kalman è stato oggetto di ricerche e applicazioni in larga scala nel calcolo numerico e in particolare nel campo della navigazione autonoma o assistita. Il filtro risulta particolarmente efficiente per alcuni aspetti: esso determina stime degli stati passati, presenti e futuri e tali risultati possono essere conseguiti persino quando la natura del sistema modellato non è conosciuta con precisione.

Il filtro di Kalman è un algoritmo utilizzato per il filtraggio dei dati costruito sulla base di una media ragionata tra il prossimo valore predetto e il prossimo valore stimato. Questo filtro è spesso utilizzato per ottenere una migliore valutazione di un dato ottenuto dalla lettura di più sensori, ognuno caratterizzato da un rumore di misura avente caratteristiche differenti nel tempo (e quindi nella frequenza).

2.2 Caratteristiche del filtro

Per costruire un filtro di Kalman è necessario conoscere le seguenti componenti:

- una serie di misure sul sistema da stimare;
- la conoscenza di un modello matematico lineare descrittivo del sistema;
- il modello statistico dei rumori sulle misure.

Per la descrizione del sistema dinamico, non stazionario e a tempo discreto, saranno utilizzati i seguenti simboli, sottolineando che i pedici k sono utilizzati per tenere conto dell'istante in cui si effettua l'operazione di aggiornamento della stima:

- X_k: vettore di stato;
- u_k: vettore degli ingressi;
- z_k: vettore delle misure: lo stato di un sistema è a volte direttamente misurabile, a volte deve essere misurato attraverso grandezze loro equivalenti o il cui valore rappresenta la combinazione di una o più variabili lineari di stato, possibilmente sporcato da errori di misura o da imprecisione nella modellizzazione matematica del sistema;
- w_k: disturbo sullo stato: rappresenta gli ingressi che agiscono in maniera non nota sullo stato del sistema;
- v_k: rumore sulle misure: rumore di lettura presente nel vettore delle misure rispetto al valore che avrebbero in conseguenza al valore attuale dello stato e delle variabili;
- $Q_k = E[w_k w_k^T]$: matrice di covarianza del disturbo sullo stato: rappresenta la varibilità statistica del vettore dei disturbi sullo stato (il vettore ha media nulla) ovvero la potenza del disturbo introdotto nel sistema che devia l'andamento delle variabili di stato rispetto a quello

prevedibile dalla conoscenza del vettore degli ingressi e dalla legge che ne governa l'evoluzione;

- $R_k = E[v_k v_k^T]$: matrice di covarianza del rumore sulle misure: rappresenta la variabilità statistica del vettore dei disturbi di misura (vettore a media nulla) e rappresenta la potenza del disturbo sulla misura introdotta su ciascuna delle misure accessibili;

- $P_k = E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]$: matrice di varianza dell'errore sullo stato: rappresenta la variabilità dell'errore sulla stima dello stato conseguente ai due fattori di disturbo (errore di misura e disturbo dello stato);

- K_k: matrice di correzione della stima: indica il livello di fiducia assegnata alla misura rispetto alla fiducia assegnata alla stima dello stato in base al valore precedente e al modello matematico che rappresenta; tanto maggiore è il valore di K_k, tanto minore fiducia merita la stima basata sul modello rispetto alla misura riportata;

- A_k: matrice di stato: matrice descrittiva dell'evoluzione libera della variabile di stato rispetto al suo valore attuale;

- B_k: matrice degli ingressi: matrice descrittiva dell'evoluzione forzata della variabile di stato rispetto al valore attuale dell'ingresso;

- H_k: matrice delle uscite: matrice descrittiva del valore assunto dalle variabili misurate in funzione del valore attuale della variabile di stato.

Il valore atteso degli errori delle grandezze appena descritte viene indicato con $E[.]$. Queste grandezze hanno le seguenti proprietà e definizioni:

- $Q_k = E[w_k w_k^T]$;

- $R_k = E[v_k v_k^T]$;

- $E[w_j v_k^T] = 0 \forall j, k$;

- $P_0 = E[(X_0 - \hat{X}_0)(X_0 - \hat{X}_0)^T]$;

2.2.1 Il processo da stimare

Si consideri di voler stimare lo stato $x \in \Re^n$ di un processo a tempo discreto governato dall' equazione alle differenze lineare stocastica

$$x_{k+1} = A_k x_k + B_k u_k + w_k. \tag{2.1}$$

La stima viene effettuata per mezzo di successive misurazioni $z \in \Re^m$ tali che

$$z_k = H_k x_k + v_k \tag{2.2}$$

Le variabili casuali w_k e v_k rappresentano rispettivamente i rumori sul processo e sulle misurazioni. Tali rumori sono assunti come indipendenti, bianchi e con distribuzione di probabilità normale.

$$\begin{aligned} p(w) &\sim N(0, Q) \\ p(v) &\sim N(0, R). \end{aligned} \tag{2.3}$$

La matrice $n \times n$ A dell'equazione differenziale (2.1) lega lo stato al tempo k con lo stato al tempo $k+1$ in assenza di funzioni di comando e di rumore di processo. La matrice $n \times l$ B lega l'input $u \in \Re^l$ con lo stato x. La matrice $m \times n$ H nell'equazione delle misurazioni (2.2) lega lo stato x alla misurazione z_k.

2.2.2 Le origini del filtro

Possiamo definire $\hat{x}_k^- \in \Re^n$ come la stima dello stato eseguita a priori, ovvero al passo k data la conoscenza del processo prima dello step k, e $\hat{x}_k \in \Re^n$ come la stima dello stato eseguita a posteriori, cioè al passo k una volta nota la misurazione z_k.

Si può così definire l'errore di stima a priori ed a posteriori come:

$$\begin{aligned} e_k^- &\equiv x_k - \hat{x}_k^- \\ e_k &\equiv x_k - \hat{x}_k \end{aligned}$$

La covarianza dell'errore a priori è perciò:

$$P_k^- = E[e_k^- {e^-}_k^T] \tag{2.4}$$

e la covarianza dell'errore a posteriori:

$$P_k = E[e_k e_k^T]. \tag{2.5}$$

Derivando le equazioni del filtro di Kalman, si procede ricercando un'equazione che computi una stima a posteriori $\hat{x}_k$ come una combinazione lineare di $\hat{x}_k^-$ stimato a priori e di una differenza pesata tra la misura attuale z_k ed una previsione per la misurazione $H_k\hat{x}_k^-$.
La stima a posteriori risulta pertanto

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H_k\hat{x}_k^-). \tag{2.6}$$

La differenza $(z_k - H_k\hat{x}_k^-)$ è detta innovazione di misurazione o residuo. Essa riflette la discrepanza tra la previsione di misurazione $H_k\hat{x}_k^-$ e la misurazione attuale z_k. Un residuo nullo indica che le due quantità sono in perfetto accordo.
La matrice K $n \times m$ nell'equazione (2.6), detta guadagno di Kalman o blending factor, si determina in modo da minimizzare la covarianza dell'errore a posteriori (2.5). Questa minimizzazione può essere effettuata sostituendo la (2.6) nella precedente definizione di e_k, introducendo la relazione così ricavata nella (2.5), derivando rispetto a K, eguagliando a zero ed infine risolvendo rispetto a K. Una possibile forma del guadagno che minimizza la (2.5) è data dalla seguente espressione:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1}. \tag{2.7}$$

Osservando la (2.7) si può vedere che se la covarianza dell'errore di misura R_k tende a zero, il guadagno diventa più grande, quindi tende a far pesare di più il termine d'innovazione

$$\lim_{R_k \to 0} K_k = H_k^{-1}$$

Nel caso in cui, invece, sia la covarianza sull'errore di stima P_k^- a tendere a zero, il guadagno K_k diminuisce e fa pesare meno il residuo:

$$\lim_{P_k^- \to 0} K_k = 0.$$

Un altro modo di pensare l'azione di K è che quando la covarianza dell'errore R_k tende a zero, la misurazione z_k diviene sempre più vera, mentre la previsione della misurazione sempre meno. D'altro canto, quando la covarianza dell'errore di stima a priori P_k^- tende a zero, la misurazione attuale z_k si discosta sempre di più dalla realtà, mentre la previsione della misurazione si avvicina sempre più al valore corretto.

Attraverso alcune semplici operazioni di calcolo matriciale è altresì possibile esprimere il guadagno di Kalman nella seguente forma:

$$K_k = P_k H_k R^{-1}$$

con

$$P_k = P_k^- (I + H_k^T R^{-1} H_k P_k^{-1})^{-1}.$$

Se $H_k = 1$, è immediato che

$$K_k = P_k R^{-1}$$

e si può quindi osservare che K_k è direttamente proporzionale alla covarianza dell'errore di stima ed inversamente proporzionale alla varianza dell'errore di misurazione:

- tanto maggiore è l'errore che si è commesso nella stima precedente e tanto maggiore è l'affinamento della stima attuale (guadagno) indotto dal filtraggio;

- tanto meno le misurazioni si discostano dal valore reale, tanto più sarà possibile ottenere stime precise.

2.2.3 Le origini probabilistiche del filtro

La giustificazione per la (2.6) è radicata nella probabilità della stima a priori $\hat{x}_k^-$ condizionata da tutte le precedenti misurazioni z_k. Per ora è sufficiente sottolineare che il filtro di Kalman mantiene i primi due momenti della distribuzione degli stati:

$$E[\hat{x}_k] = x_k$$
$$E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] = P_k$$

La stima dello stato a posteriori (2.6) riflette la media (il primo momento) della distribuzione degli stati. Essa è normalmente distribuita se le condizioni espresse da (2.3) sono verificate. La covarianza dell'errore di stima a posteriori (2.5) riflette la varianza della distribuzione degli stati:

$$p(x_k \mid z_k) \sim N(E[x_k], E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]) = N(\hat{x}_k, P_k).$$

2.3 L'algoritmo del filtro di Kalman discreto

Il filtro di Kalman fornisce una stima di un processo utilizzando una forma di controllo in feedback: il filtro stima il processo ad un certo istante e ottiene un feedback sotto forma di misurazione affetta da rumore.

Le equazioni del filtro di Kalman possono essere suddivise in due gruppi: equazioni di *time update* ed equazioni di *measurement update*. Le prime sono responsabili della previsione dello stato attuale e della covarianza dell'errore, valutate per ottenere una stima a priori per lo step successivo. Le equazioni di *measurement update* sono responsabili del feedback: esse vengono impiegate per unire una nuova misurazione con la stima a priori al fine di ottenere una migliore stima a posteriori.

Le equazioni di *time update* possono anche essere pensate come equazioni

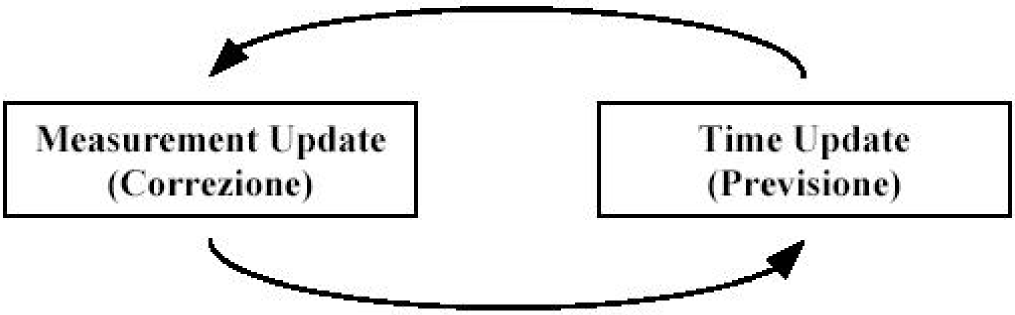


Figura 2.1: L'algoritmo del filtro di Kalman visto come una procedura ricorsiva in cui le equazioni di *time update* forniscono una previsione e quelle di *measurement update* determinano un miglioramento della stima introducendo l'informazione contenuta nella misurazione.

di predizione, mentre le equazioni di *measurement update* rappresentano le equazioni di correzione. Le equazioni di *predizione* fanno una stima a *priori* (cioè al tempo k) dello stato del sistema al tempo $k+1$, mentre le equazioni di *stima* fano una correzione a *posteriori* (cioè al tempo $k+1$) del valore che era stato stimato al tempo k sfruttando le misure che tornano in retroazione. Quindi l'algoritmo di stima finale rassomiglia ad un algoritmo previsore-correttore per la risoluzione di problemi numerici.

Sono riportate di seguito le equazioni della fase di predizione

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \tag{2.8}$$

$$P_k^- = AP_{k-1}A^T + Q_k \tag{2.9}$$

e quelle della fase di stima

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \tag{2.10}$$

$$\hat{x}_k = \hat{x}_k^- + K(z_k - H_k \hat{x}_k^-) \tag{2.11}$$

$$P_k = (I - K_k H_k) P_k^- \tag{2.12}$$

le matrici A e B sono le stesse della (2.1), mentre la Q è la matrice di covarianza del sistema.Nella fase di stima, prima di tutto si calcola il guadagno K_k tramite la (2.10), equazione identica alla (2.7). Nella (2.11) si calcola la stima a posteriori dello stato, cioè si fa la correzione della stima a priori usando i dati delle misure z_k. Infine si calcola la matrice di covarianza dell'errore a posteriori(2.12).
Questo metodo ricorsivo ad ogni ciclo calcola prima le equazioni della fase di stima e poi fa la predizione sul tempo successivo.
La covarianza sul rumore di misura K_k può essere calcolata prima di far partire l'algoritmo di Kalman, in quanto è plausibile poter rilevare qualche misura OFF-LINE per calcolare la varianza sul rumore di misura. La covarianza sul rumore del processo Q_k è invece più difficile da determinare in quanto in genere non si può osservare direttamente il processo che vogliamo stimare. Di fatto molte volte il filtro può essere più performante se si regolano le matrici Q_k e R_k. Nella condizione in cui Q_k e R_k restano costanti, si può notare che la covarianza sull'errore di stima P_k e il guadagno di Kalman K_k si stabilizzano rapidamente e restano costanti.
Proprio la natura ricorsiva dell'algoritmo è una delle caratteristiche più importanti e attraenti del filtro di Kalman, essa rende l'implementazione molto più agevole di quanto non lo sia, ad esempio, quella del filtro di Wiener, che è realizzato per operare su tutti i dati direttamente ad ogni stima. Il filtro di Kalman invece condiziona ricorsivamente la stima corrente e tutte le stime passate.

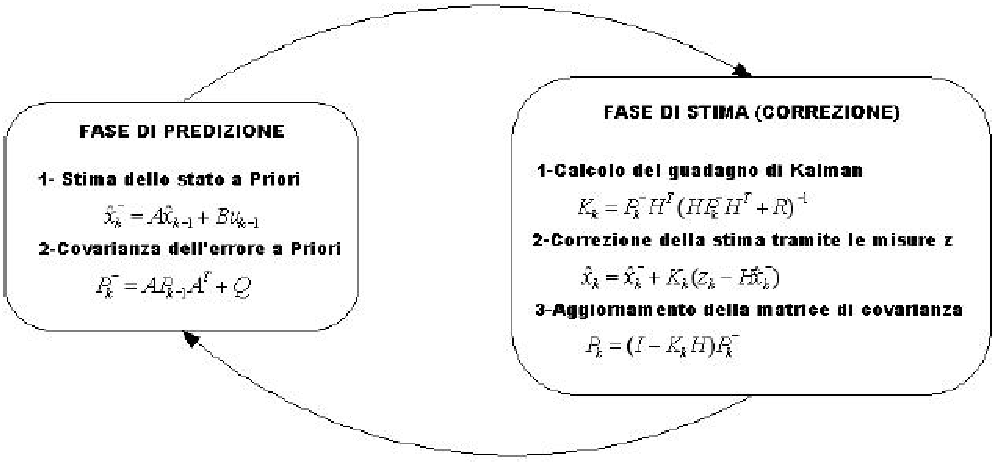


Figura 2.2: Quadro completo dell'algoritmo del filtro di Kalman discreto

2.3.1 Parametri del filtro e regolazione

In questa implementazione del filtro di Kalman le matrici R_k (covarianza dell'errore di misurazione) e Q_k (covarianza del rumore del processo) dovrebbero essere misurate prima che le operazioni del filtro vengano eseguite.
Spesso la scelta di Q_k è deterministica. Questa sorgente di rumore è ad esempio impiegata per rappresentare l'incertezza nel modello del processo. Talvolta modelli molto semplificati possono essere impiegati inserendo un certo grado di incertezza proprio attraverso la matrice Q_k. Certamente in questo caso è necessario che la misurazione dello stato e quindi del processo sia esatta.
Che si abbia o meno un fondamento razionale per scegliere il valore dei parametri, l'ottimizzazione delle performance del filtro possono essere ottenute regolando i parametri del filtro Q_k e R_k. La regolazione è generalmente ottenuta fuori linea, frequentemente con l'aiuto di altri distinti filtri di Kalman.
In conclusione si osserva che sotto le condizioni dove Q_k e R_k sono costanti, sia la covarianza dell'errore di stima P_k, sia il guadagno di Kalman K_k si stabilizzano rapidamente per poi rimanere costanti.

Capitolo 3

Filtro di Kalman esteso: EKF

3.1 Caratteristiche dell'EKF

Come già detto il filtro di Kalman si usa di solito per trovare una stima dello stato $X \in \Re^n$ di un sistema controllato tempo-discreto governato da una equazione differenziale lineare stocastica. Nel caso in cui il processo da stimare e/o le misure relative al processo sono non lineari, in maniera simile a quanto si fa per le serie di Taylor, possiamo linearizzare il sistema attorno alla stima corrente usando le derivate parziali delle funzioni di stato e di misura. Un filtro di Kalman che linearizza il sistema non lineare attorno alla stima ed alla covarianza correnti e detto filtro di Kalman esteso (Extended Kalman Filter).

Limiteremo lo studio al tempo discreto. Assumiamo che il processo abbia un vettore di stato $X \in \Re^n$ e sia governato da una equazione alle differenze stocastica non lineare:

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1}) \tag{3.1}$$

con misure $z \in \Re^n$ governate dall'equazione:

$$z_k = h(x_k, v_k) \tag{3.2}$$

dove x_k e z_k sono i vettori dello stato e delle misure, mentre le variabili aleatorie w_k e v_k rappresentano rispettivamente i rumori di processo e di misura,di cui non si conoscono i valori ad ogni ciclo di lavoro. Tali rumori saranno di tipo gaussiano e a media nulla. Le matrici di covarianza dei disturbi saranno definite nel modo seguente:

$$\begin{array}{ll} E[w_i] = 0 & E[v_i] = 0 \\ E[w_i w_i'] = Q \geq 0 & E[v_i v_i'] = R \geq 0 \\ E[w_i w_j'] = 0, i \neq j & E[v_i v_j'] = 0, i \neq j \end{array}$$

Nello studio effettuato le matrici di covarianza Q e R saranno considerate costanti, quindi non verranno aggiornate ad ogni reiterazione del procedimento. Poiché non conosciamo istante per istante i valori di w_k e v_k, possiamo approssimare le equazioni (3.1) e (3.2) con le seguenti:

$$\tilde{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0) \tag{3.3}$$

$$\tilde{z}_k = h(\tilde{x}_{k-1}, 0) \tag{3.4}$$

dove $\hat{x}_{k-1}$ rappresenta la stima a posteriori dello stato (relativo al precedente istante di campionamento $(k-1)$.
È importante notare che il difetto dell'EKF è che la distribuzione delle varie variabili aleatorie non è più normale(gaussiana) quando si passa alle rispettive trasformazioni non lineari. L'EKF è quindi soltanto un'osservatore dello stato che approssima l'ottimalità della regola di Bayes per linearizzazione.

3.2 Descrizione del filtro

Per stimare un processo governato da equazioni non lineari, si procede con la linearizzazione delle (3.1) e (3.2) intorno alla stima ottenendo:

$$\begin{gathered} x_k \approx \tilde{x}_k + A_k(x_{k-1} - \hat{x}_{k-1}) + W_k w_{k-1} \\ z_k \approx \tilde{z}_k + H_k(x_k - \hat{x}_k) + V_k v_k \end{gathered} \tag{3.5}$$

dove

- x_k e z_k sono i vettori dello stato e della misura attuale,
- $\tilde{x}_k$ e $\tilde{z}_k$ sono i vettori dello stato e della misura approssimata,
- $\hat{x}_k$ è una stima a *posteriori* dello stato al tempo k ,
- w_k e v_k sono ancora le variabili aleatorie che rappresentano i rumori di processo e di misura
- A_k è la matrice Jacobiana delle derivate parziali di f rispetto a x

$$A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_{[j]}}(\hat{x}_{k-1}, u_{k-1}, 0),$$

- W_k è la matrice Jacobiana delle derivate parziali di f rispetto a w

$$W_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_{[j]}}(\hat{x}_{k-1}, u_{k-1}, 0), \tag{3.6}$$

- H_k è la matrice Jacobiana delle derivate parziali di h rispetto a x

$$H_{[i,j]} = \frac{\partial h_{[i]}}{\partial x_{[j]}}(\tilde{x}_k, 0), \tag{3.7}$$

- V_k è la matrice Jacobiana delle derivate parziali di h rispetto a v

$$V_{[i,j]} = \frac{\partial f_{[i]}}{\partial v_{[j]}}(\tilde{x}_k, 0), \tag{3.8}$$

Il pedice k per le matrici Jacobiane appena descritte sta a significare che queste ultime variano ad ogni reiterazione dell'algoritmo. Adesso si può definire un nuovo errore di predizione

$$\tilde{e}_{x_k} \equiv x_k - \tilde{x}_k \tag{3.9}$$

e un nuovo errore di misura

$$\tilde{e}_{z_k} \equiv z_k - \tilde{z}_k \tag{3.10}$$

Poichè nella realtà non si conosce direttamente il valore reale di x_k e z_k, allora tali espressioni dell'errore vanno modificate come segue:

$$\tilde{e}_{x_k} \approx \tilde{A}(x_{k-1} - \hat{x}_{k-1}) + \epsilon_k, \tag{3.11}$$

$$\tilde{e}_{z_k} \approx \tilde{H}\tilde{e}_{x_k} + \eta_k \tag{3.12}$$

dove ϵ_k e η_k sono due nuove variabili aleatorie aventi media nulla e matrice di covarianza WQW^T e VRV^T con Q ed R matrici di covarianza dei processi aleatori w e v.

La (3.11) e la (3.12) sono equazioni lineari molto simili alla (2.1) e alla (2.2), equazioni generali del Filtro di Kalman lineare. Per questo si può sostituire l'errore di misura attuale $\tilde{e}_{z_k}$ nella (3.10) e si può introdurre un altro (ipotetico) filtro di Kalman per stimare l'errore di predizione $\tilde{e}_{x_k}$ dato dalla (3.11). Questa stima, che verrà indicata con $\hat{e}_k$, potrà essere usata per calcolare una stima a posteriori per il processo non lineare di partenza:

$$\hat{x}_k = \tilde{x}_k + \hat{e}_k \tag{3.13}$$

Il filtro di Kalman usato per stimare $\hat{e}_k$ è

$$\hat{e}_k = K_k\tilde{e}_{z_k}. \tag{3.14}$$

Sostituendo la (3.14) nella (3.13) e, usando la (3.10) si può constatare che il secondo ipotetico filtro di Kalman non serve:

$$\begin{aligned} \hat{x}_k &= \tilde{x}_k + K_k\tilde{e_{zk}} && (3.15) \\ &= \tilde{x}_k + K_k(z_k - \tilde{z}_k) && (3.16) \end{aligned}$$

A questo punto, l'equazione (3.16) può essere usata insieme al K_k della (2.10) (opportunamente modificata per la covarianza dell'errore di misura), per le equazioni di predizione del filtro di Kalman esteso. I termini $\tilde{x}_k$ e $\tilde{z}_k$ derivano dalle (3.3) e (3.4).

3.3 Implementazione dell'EKF

In questa sezione verrà descritto l'algoritmo del filtro di Kalman esteso. Sostanzialmente il processo di osservazione avviene in due fasi:

1. **Time update:**
   - vengono generate le proiezioni dello stato del sistema, a partire della conoscenza del modello e della stima dello stato precedente;
   - vengono aggiornati i parametri probabilistici di errore sulla misura dello stato.
2. **Measurement update:**
   - vengono corrette le stime prodotte al passo precedente, sulla base delle osservazioni delle uscite;
   - vengono aggiornati i parametri probabilistici di errore sulla misura delle uscite.

Quindi, dato un sistema nella forma:

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1}) \tag{3.17}$$

$$z_k = h(x_k, v_k) \tag{3.18}$$

l'algoritmo per EKF è:

- time update

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}) \tag{3.19}$$

$$P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T \tag{3.20}$$

- measurement update

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1} \tag{3.21}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0)) \tag{3.22}$$

$$P_k = (I - K_k H_k) P_k^- \tag{3.23}$$

dove:
$\hat{x}_{k-1}$: è lo stato stimato al passo precedente (stima corrente);
$\hat{x}_k^-$: è lo stato predetto noto $\hat{x}_{k-1}$;
K_k : é il guadagno del filtro al passo k;
z_k : sono le misure;
$h(\hat{x}_k^-, 0)$: è l'uscita predetta, noto $\hat{x}_k^-$;
$\hat{x}_k$: è lo stato stimato dal filtro;
A_k : é lo Jacobiano di f rispetto ad x, calcolato in $(\hat{x}_{k-1}, u_{k-1}, 0)$, cioè attorno alla stima corrente;
W_k : é lo Jacobiano di f rispetto ad w, calcolato in $(\hat{x}_{k-1}, u_{k-1}, 0)$;
H_k : è lo Jacobiano di h rispetto ad x, calcolato in $(\hat{x}_k^-, 0)$, cioè attorno alla predizione;
V_k : è lo Jacobiano di h rispetto ad v, calcolato in $(\hat{x}_k^-, 0)$;

Va sottolineato che la sequenza dei guadagni K_k deve necessariamente essere calcolata in tempo reale, non può essere calcolata off-line prima della lettura delle misure e poi immagazzinata in memoria.

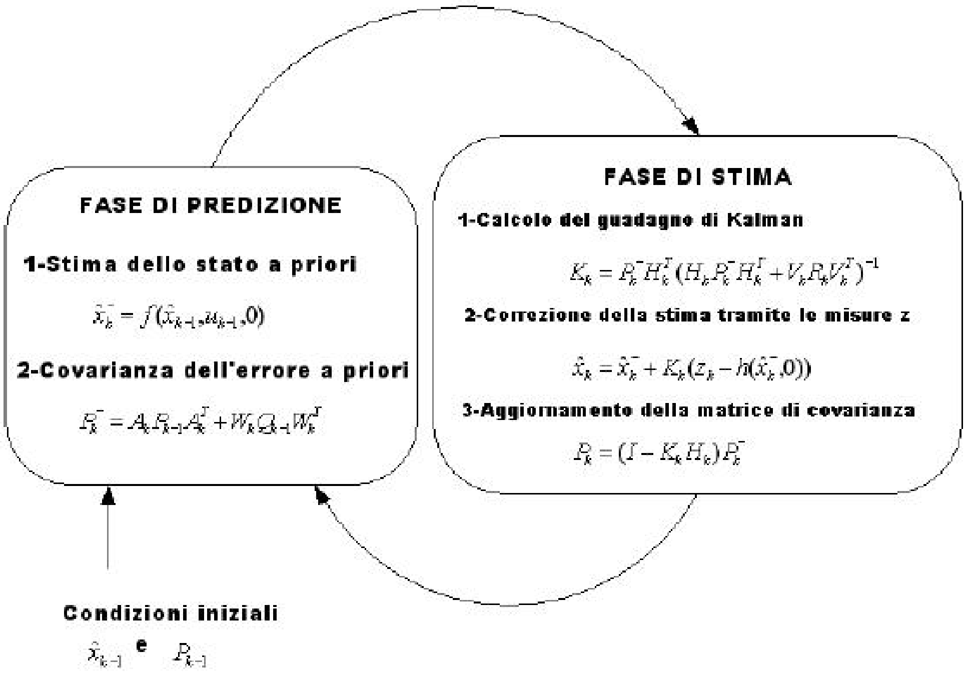


Figura 3.1: Schema generale per EKF

In Figura (3.1) si riporta uno schema riassuntivo delle operazioni che compongono l'algoritmo del filtro di Kalman esteso, suddiviso in *fase di predizione* e *fase di stima*. Nella fase di predizione si stima il valore che può avere lo stato al tempo corrente k utilizzando i dati del tempo $k-1$ e inoltre si calcola una matrice di covarianza dell'errore *a priori* P_k^- avendo a disposizione soltanto P_{k-1} e Q_{k-1} (covarianza sul rumore di processo). Nella fase di stima si calcola il guadagno di Kalman e si opera una correzione sulla stima fatta in precedenza. Si calcola il valore che dovrebbe avere z_k cioè $h(\hat{x}_k^-, 0)$ e poi lo si confronta con il z_k reale e si somma, opportunamente pesato grazie al guadagno K_k, alla stima *a priori* $\hat{x}_k^-$. Infine si aggiorna P_k.

Un'importante caratteristica del filtro EKF è che la matrice jacobiana H_k nell'equazione per il calcolo del guadagno del filtro serve a propagare correttamente o ad amplificare solo le componenti rilevanti dell'informazione di misura. Ad esempio se non c'è una corrispondenza uno ad uno tra le misure z_k e lo stato attraverso h, lo jacobiano H_k influisce sul guadagno di Kalman in modo da amplificare solo la porzione del residuo $z_k - h(\hat{x}_k^-, 0)$ che influisce sullo stato. Di certo, se tra tutte le misure z_k e lo stato h non c'è mai una corrispondenza uno a uno, allora si può affermare che il filtro divergerà rapidamente. In questo caso il processo sará inosservabile.

Capitolo 4

EKF applicato al caso del Nomad 150

La posizione di un robot può essere stimata prelevando informazioni dall'ambiente circostante: le informazioni necessarie alla stima differiscono in base al tipo di movimentazione e ambiente operativo, al livello di precisione necessario e alla qualità dei sensori utilizzati. È difficile ipotizzare una soluzione generale del problema che vada oltre la semplice intenzione di elaborare i dati sfruttandone "ridondanza" e "complementarietà". Per l'applicazione sviluppata in questo ambito il filtro ha una forma molto specifica adatta per le esigenze che si volevano soddisfare. Tali esigenze prevedono la stima della posizione e dell'orientamento del robot in tempi relativamente contenuti, utilizzando sensori odometrici (encoder sulle ruote e sulla torretta), e sensori acustici (16 sonar posti sulla circonferenza del robot).

4.1 Modello e rilevamenti

La soluzione più semplice per la stima di posizione consiste nell'integrazione diretta nel tempo delle informazioni di velocità. Supponendo un ambiente bidimensionale, si può definire la configurazione del robot rispetto ad un riferimento fisso solidale con l'ambiente, mediante un vettore di tre parametri.

$$X_r = [x \quad y \quad \theta]' \tag{4.1}$$

Nell'ipotesi di perfetta aderenza delle ruote al suolo, esatta calibrazione delle ruote, misure non rumorose (etc.), ovvero in assenza di errori odometrici, possono essere usate le equazioni:

$$\begin{aligned} \dot{x} &= v(t)cos\theta \\ \dot{y} &= v(t)sin\theta \\ \dot{\theta} &= \omega(t) \end{aligned}$$

dove $v(t)$ e $\omega(t)$ sono la velocità di traslazione e rotazione del robot misurate.
Scegliendo opportunamente il tempo di campionamento T, è possibile discretizzare le equazioni di cui sopra. Operando su una sola equazione

$$\begin{aligned} \tfrac{x((k+1)T)-x(kT)}{T} &= v(kT)cos\left(\theta(kT) + \tfrac{\omega(kT)T}{2}\right) \\ &\Longrightarrow \\ x((k+1)T) &= x(kT) + Tv(kT)cos\left(\theta(kT) + \tfrac{\omega(kT)T}{2}\right) \\ &\Longrightarrow \\ x(k+1) &= x(k) + Tv(k)cos\left(\theta(k) + \tfrac{\omega(k)T}{2}\right) \end{aligned}$$

ed estendendo il risultato a tutte le equazioni si ottiene

$$\begin{aligned} x(k+1) &= x(k) + Tv(k)cos\left(\theta(k) + \tfrac{\omega(k)T}{2}\right) \\ y(k+1) &= y(k) + Tv(k)sin\left(\theta(k) + \tfrac{\omega(k)T}{2}\right) \\ \theta(k+1) &= \theta(k) + T\omega(k) \end{aligned}$$

Il moto del robot presenta continui slittamenti, piccoli urti, discontinuità del suolo che portano rapidamente l'integrazione diretta dei dati odometrici a risultati erronei: si verifica un disallineamento tra la posizione effettiva e quella misurata. Il modello descritto dalle (4.1) non è sufficiente e deve essere completato con l'inserimento di disturbi aleatori:

$$\begin{aligned} x(k+1) &= x(k) + Tv(k)cos\left(\theta(k) + \tfrac{\omega(k)T}{2}\right) + v_x(k+1) \\ y(k+1) &= y(k) + Tv(k)sin\left(\theta(k) + \tfrac{\omega(k)T}{2}\right) + v_y(k+1) \\ \theta(k+1) &= \theta(k) + T\omega(k) + v_\theta(k+1) \end{aligned}$$

con $v(k) = [v_x \quad v_y \quad v_\theta]'$ disturbi gaussiani a media nulla, esplicitamente:

$$\begin{aligned} Ev(k) &= 0 \\ Ev(k)v'(h) &= Q(k, \Delta_s) \qquad h = k \\ Ev(k)v'(h) &= 0 \qquad h \neq k \end{aligned}$$

con $Q(k, \Delta_s)$, simmetrica, semidefinita positiva, dipendente dalla distanza (Δ_s) tra due step consecutivi.

Per migliorare l'accuratezza della stima e risolvere la sostanziale instabilità del metodo di integrazione diretta sono necessarie altre misure. Dato un generico vettore $z(k)$ di misure al tempo k si vuole un algoritmo di stima che minimizzi la differenza tra posizione reale e stimata.

$$\hat{X}_r(k+1) = \Lambda(\hat{X}_r(k), z(k)) \tag{4.2}$$

I limiti delle informazioni odometriche possono essere superati quando si considerino le informazioni complementari fornite da altri sensori. Nel caso specifico sono presenti 16 sensori sonar gestibili mediante un vettore dati $Z(k) = [z_1, ..., z_{16}]'$. Le misure sonar sono dipendenti dall'ambiente di lavoro e dalla posizione del robot:

$$Z(k) = h(k, X_r) + \omega(k) \tag{4.3}$$

dove con $h(k, X_r)$ si indica la relazione esistente tra la posizione del robot e il rilevamento dati sonar; $\omega(k)$ è un rumore gaussiano a media nulla, con

matrice di covavarianza diagonale, $R(k)$, utilizzato per modellare gli errori di misura.
Considerando il legame tra la posizione del robot e il rilevamento sonar è possibile porre formalmente il problema della localizzazione come la ricerca di una funzione Λ che fornisca in modo ricorsivo nel tempo la configurazione ottima del robot, combinando le equazioni (4.1) del modello del sistema e le informazioni ultrasoniche (4.3):

$$X_r(k+1) = \Lambda(X_r(k), Y(k+1)) \tag{4.4}$$

Ad ogni ciclo di lavoro l'algoritmo aggiorna la stima della posizione $X_r(k)$ e la matrice di covarianza dell'errore associato nelle corrispondenti variabili al passo successivo, $X_r(k+1)$ e relativa covarianza. L'algoritmo utilizzerà un filtro di Kalman esteso.

4.2 Applicazione del filtro

L'ambiente nel quale avviene la movimentazione del robot è descritto da una mappa planare costituita da segmenti rappresentanti le pareti principali (con coordinate relative al sistema di riferimento fisso). Il robot è in grado di misurare la velocità delle ruote e la variazione di orientamento ed utilizza un sistema di rilevatori ultrasonici (sonar) per valutare la distanza dalle pareti.
L'idea principale per risolvere il problema della localizzazione è la seguente: noto lo stato $X_r(k)$ del robot e date le misure di velocità lineare ed angolare si può ipotizzare la nuova posizione $X_{r,ipotesi}(k+1)$. Nella nuova posizione ipotetica possono essere calcolate le risposte dei sonar. Quando il robot andrà nello stato $X_r(k+1)$ si potranno confrontare le risposte calcolate sullo stato ipotetico, con le misure effettive dei sonar e correggere eventuali errori di posizione.
In questo modo si confronta il risultato dell'integrazione diretta delle velocità con le misure dei sensori, riducendo continuamente potenziali errori di stima.

Si esplicita la relazione (4.3):

$$\mathbf{h}(\mathbf{k}, \mathbf{X_r}) = \begin{bmatrix} h_1(k, X_r) \\ \vdots \\ h_{16}(k, X_r) \end{bmatrix}$$

$$h_i(k, X_r) = \sqrt{(S_{i,x} - x_p^2) + (S_{i,y} - y_p^2)} \tag{4.5}$$

La relazione (4.5) è la spiegazione dell'origine della misura sonar e definisce il rapporto intercorrente tra questa e la posizione del robot. Con S_i si indica la posizione del sensore i-esimo (nelle sue coordinate assolute x e y) che dipende dalla posizione e dall'orientamento del robot; x_p, y_p è la posizione di un generico punto della parete, individuato dalla retta ortogonale alla tangente della circonferenza nel punto di collocazione del sonar e la parete stessa. Si è omessa la dipendenza dal tempo k per semplicità di notazione. Definendo

$$\hat{X}_r(k|k) = [\hat{x}(k|k) \quad \hat{y}(k|k) \quad \hat{\theta}(k|k)]' \tag{4.6}$$

come la stima dello stato del robot al tempo k sui primi k vettori di misure dei sonar, possiamo riscrivere la soluzione del problema come:

$$\hat{X}_r(k+1|k+1) = \Lambda(\hat{X}_r(k|k), Z(k+1)). \tag{4.7}$$

Ad ogni passo l'algoritmo stima in funzione dei rilevamenti sperimentali lo stato del robot e la relativa matrice di covarianza. L'uso del filtro di Kalman nella progettazione dell'algoritmo permette di ridurre l'incertezza sulla stima dello stato dovuta alla presenza di rumore nelle misure e alla movimentazione del robot.

4.2.1 Fase di predizione

Si consideri noto lo stato $\hat{X}_r(k|k)$ (stimato al tempo k, dopo k misure sonar) e la matrice di covarianza

$$P(k|k) \quad = \quad E\{[\hat{x}_r(k|k) - \bar{\hat{x}}_r(k|k)][\hat{x}_r(k|k) - \bar{\hat{x}}_r(k|k)]'\} \tag{4.8}$$

si può fare una predizione al tempo $k+1$

$$\begin{aligned}
\hat{x}(k+1|k) &= \hat{x}(k|k) + Tv(k)cos\left(\hat{\theta}(k|k) + \tfrac{\omega(k)T}{2}\right)\\
\hat{y}(k+1|k) &= \hat{y}(k|k) + Tv(k)sin\left(\hat{\theta}(k|k) + \tfrac{\omega(k)T}{2}\right)\\
\hat{\theta}(k+1|k) &= \hat{\theta}(k|k) + T\omega(k)
\end{aligned}$$

$$P(k+1|k) = E\{[\hat{x}_r(k+1|k) - \hat{x}_r^-(k+1|k)][\hat{x}_r(k+1|k) - \hat{x}_r^-(k+1|k)]'\} \tag{4.9}$$

Lo stato $X_r(k+1|k)$ è facilmente calcolabile mantre la relazione (4.9) deve essere semplificata. Si linearizzano le equazioni (4.1) sul punto $X = \hat{x}_r(k|k)$:

$$\mathbf{f(k,X)} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} x(k) + Tv(k)cos\left(\theta(k) + \frac{\omega(k)T}{2}\right) \\ y(k) + Tv(k)sin\left(\theta(k) + \frac{\omega(k)T}{2}\right) \\ \theta(k) + T\omega(k) \end{bmatrix}$$

$$\mathbf{F(k)} = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial \theta} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial \theta} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial \theta} \end{bmatrix}_{x_r=\hat{x}_r(k|k)} = \begin{bmatrix} 1 & 0 & -Tv(k)sin\left(\theta(k) + \frac{\omega(k)T}{2}\right) \\ 0 & 1 & Tv(k)cos\left(\theta(k) + \frac{\omega(k)T}{2}\right) \\ 0 & 0 & 1 \end{bmatrix}_{X_r=\hat{X}_r(k|k)}$$

Note le velocità di traslazione e rotazione del robot al tempo k (rilevamenti odometrici), $F(k)$ rimane definita. Da cui:

$$\begin{aligned}
P(k+1|k) &= E\{[x_r(k+1|k) - \bar{x}_r(k+1|k)][x_r(k+1|k) - \bar{x}_r(k+1|k)]'\}\\
&= E\{[(F(k)\hat{x}_r(k|k) + v_{xr}(k)) - (F(k)\bar{\hat{x}}_r(k|k))]
\end{aligned}$$

$$
\begin{aligned}
& [(F(k)\hat{x}_r(k|k) + v_{xr}(k)) - (F(k)\bar{\hat{x}}_r(k|k))]'\} \\
= \;& F(k)E\{[\hat{x}_r(k|k) - \bar{\hat{x}}_r(k|k)][\hat{x}_r(k|k) - \bar{\hat{x}}_r(k|k)]'\}F'(k) + \\
& + E\{v_{xr}(k)v'_{xr}(k)\} - F(k)E\{[\hat{x}(k|k) - \bar{\hat{x}}(k|k)][v'_{xr}(k)]\} + \\
& + E\{[\hat{x}(k|k) - \bar{\hat{x}}(k|k)]'[v_{xr}(k)]\}F'(k) \\
= \;& F(k)E\{[\hat{x}_r(k|k) - \bar{\hat{x}}_r(k|k)][\hat{x}_r(k|k) - \bar{\hat{x}}_r(k|k)]'\}F'(k) + \\
& E\{v_{xr}(k)v'_{xr}(k)\} \\
= \;& F(k)P(k|k)F'(k) + Q(k,l)
\end{aligned}
$$

con $\bar{\hat{x}} = E[\hat{x}]$ e $Q(k,l) = E\{v_{xr}(k)v'_{xr}(k)\}$ matrice simmetrica e semidefinita positiva, dipendente dalla distanza percorsa l.
Queste equazioni calcolano la stima dello stato e della matrice di covarianza al tempo k+1 basandosi sulle informazioni note al tempo k. I risultati trovati possono essere utilizzati per effettuare la predizione sulle misure dei sonar.

$$\hat{Y}(k+1) = h(k, \hat{x}_r(k+1|k). \tag{4.10}$$

Questo risultato, nella seconda parte dell'algoritmo, sarà confrontato con le misure reali dei sensori per ottenere un miglioramento della stima di posizione.

4.2.2 Misurazioni

Noto lo stato del robot X_r si calcola posizione e orientamento dell'i-esimo sensore. Considerando la figura (4.1):

$$
\begin{aligned}
S_{i,x} &= x + rcos(\theta_0 + \theta) \\
S_{i,y} &= x + rsin(\theta_0 + \theta) \\
S_{i,\theta} &= \theta_0 + \theta
\end{aligned}
$$

θ_0 con angolo del sensore rispetto all'orientamento $\theta = 0$ del robot. Definita la posizione e l'orientamento del sensore è importante capire quale sarà il

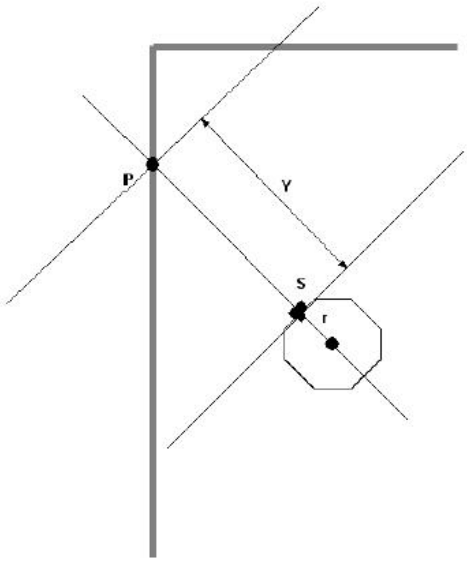


Figura 4.1: Interazione sonar-ambiente

punto sulla parete colpito. La parete è descritta mediante i punti estremi P_1 e P_2. Il segmento P_1P_2 è approssimabile, a meno dei suoi limiti, da una retta e risolvendo nelle incognite m_p, q_p

$$
\begin{aligned}
P_{1,y} &= m_p P_{1.x} + q_p \\
P_{2,y} &= m_p P_{2.x} + q_p
\end{aligned}
$$

si ottengono i parametri della retta cercata.
Il sensore misura la distanza ortogonale al suo orientamento. Anche in questo caso è possibile approssimare la direzione di misura con una retta

$$
S_{i,y} = \tan(S_{i.\theta}) S_{i,x} + q_s \tag{4.11}
$$

che deve essere risolta nell'incognita q_s.
Le due rette determinate si intersecano nel punto P cercato. Si deve risolvere nelle incognite x_p, y_p

$$x_p = m_p x_p + q_p$$

$$y_p = \tan(S_{i,\theta})S_{i,x} + q_s$$

ottenendo le coordinate del punto della parete.
La predizione dell'i-esimo sensore sarà quindi:

$$h_i(k, x_r) = \sqrt{(S_{i,x} - x_p)^2 + (S_{i,y} - y_p)^2}. \tag{4.12}$$

4.2.3 Fase di stima

La fase di stima dell'algoritmo inizia quando sono note le misure dei sonar al tempo k+1. Definito

$$\tilde{Y} = Y(k+1) - \hat{Y}(k+1). \tag{4.13}$$

la fase di stima consiste nella sua elaborazione volta all'aggiornamento della posizione del robot, $X_r(k+1|k+1)$. Come per lo stato X_r, si può definire la matrice di covarianza:

$$\begin{aligned} S(k+1) &= E\{\tilde{y}(k+1)\tilde{y}^{'}(k+1)\} \\ &= E\{[y(k+1) - \hat{y}(k+1)][y(k+1) - \hat{y}(k+1)]^{'}\} \end{aligned}$$

della quale si deve trovare una forma più semplice per il calcolo. I passaggi sono simili al caso precedente. Si linearizza la relazione $h(k, x_r)$

$$\mathbf{h}(\mathbf{k}, \mathbf{x_r}) = \begin{bmatrix} h_1(k, x_r) \\ h_2(k, x_r) \\ (...) \\ h_N(k, x_r) \end{bmatrix} = \begin{bmatrix} ((S_{1,x} - x_1)^2 + (S_{1,y} - y_1)^2)^{\frac{1}{2}} \\ ((S_{2,x} - x_2)^2 + (S_{2,y} - y_2)^2)^{\frac{1}{2}} \\ (...) \\ ((S_{N,x} - x_N)^2 + (S_{N,y} - y_N)^2)^{\frac{1}{2}} \end{bmatrix}$$

$$= \begin{bmatrix} ((x_{r,x} + r\cos(\theta_1 + x_{r,\theta}) - x_1)^2 + (x_{r,y} + r\sin(\theta_1 + x_{r,\theta}) - y_1)^2)^{\frac{1}{2}} \\ ((x_{r,x} + r\cos(\theta_2 + x_{r,\theta}) - x_2)^2 + (x_{r,y} + r\sin(\theta_2 + x_{r,\theta}) - y_2)^2)^{\frac{1}{2}} \\ (...) \\ ((x_{r,x} + r\cos(\theta_N + x_{r,\theta}) - x_N)^2 + (x_{r,y} + r\sin(\theta_N + x_{r,\theta}) - y_N)^2)^{\frac{1}{2}} \end{bmatrix}$$

Le righe sono tutte uguali tra loro:

$$\frac{\partial h_i}{\partial x_{r,x}} = \Big((x_{r,x} + r\cos(\theta_i + x_{r,\theta}) - x_i)^2 + (x_{r,y} + r\sin(\theta_i + x_{r,\theta}) - y_i)^2\Big)^{-\frac{1}{2}}$$
$$\Bigg((x_{r,x} + r\cos(\theta_i + x_{r,\theta}) - x_i)(1 - \frac{\partial x_i}{\partial x_{r,x}}) + (x_{r,y} + r\sin(\theta_i + x_{r,\theta}) - y_i)(-\frac{\partial y_i}{\partial x_{r,x})}\Bigg)$$

$$\frac{\partial h_i}{\partial x_{r,y}} = \Big((x_{r,x} + r\cos(\theta_i + x_{r,\theta}) - x_i)^2 + (x_{r,y} + r\sin(\theta_i + x_{r,\theta}) - y_i)^2\Big)^{-\frac{1}{2}}$$
$$\Bigg((x_{r,x} + r\cos(\theta_i + x_{r,\theta}) - x_i)(1 - \frac{\partial x_i}{\partial x_{r,y}}) + (x_{r,y} + r\sin(\theta_i + x_{r,\theta}) - y_i)(-\frac{\partial y_i}{\partial x_{r,y})}\Bigg)$$

$$\frac{\partial h_i}{\partial x_{r,\theta}} = \Big((x_{r,x} + r\cos(\theta_i + x_{r,\theta}) - x_i)^2 + (x_{r,y} + r\sin(\theta_i + x_{r,\theta}) - y_i)^2\Big)^{-\frac{1}{2}}$$
$$\Big[{}_{r,x} + r\cos(\theta_i + x_{r,\theta}) - x_i)\frac{\partial}{\partial x_{r,\theta}}(r\cos(\theta_i + x_{r,\theta} - x_i)\Big] +$$
$$+ \Big[x_{r,y} + r\sin(\theta_i + x_{r,\theta}) - y_i)\frac{\partial}{\partial x_{r,\theta}}(r\sin(\theta_i + x_{r,\theta}) - y_i)\Big]$$

Chiamando con d_i, per semplificare la notazione dei passaggi, la distanza misurata dall'i-esimo sensore ovvero

$$d_i = \Big((x_{r,x} + r\cos(\theta_i + \theta) - x_i)^2 + (x_{r,y} + r\sin(\theta_i + \theta) - y_i)^2\Big)^{\frac{1}{2}} \quad (4.14)$$

e considerando che

$$\begin{aligned}(x_{r,x} + r\cos(\theta_i + x_{r,\theta}) - x_i) &= d_i\cos(\theta_i + x_{r,\theta})\\ (x_{r,y} + r\sin(\theta_i + x_{r,\theta}) - y_i) &= d_i\sin(\theta_i + x_{r,\theta})\end{aligned}$$

posso riscrivere le derivate parziali:

$$\begin{aligned}\frac{\partial h_i}{\partial x_{r,x}} &= \cos(\theta_i + x_{r,\theta})(1 - \frac{\partial x_i}{\partial x_{r,x}}) + \sin(\theta_i + x_{r,\theta})(-\frac{\partial y_i}{\partial x_{r,x})}\\ \frac{\partial h_i}{\partial x_{r,y}} &= \cos(\theta_i + x_{r,\theta})(\frac{\partial x_i}{\partial x_{r,y}}) + \sin(\theta_i + x_{r,\theta})(1 - \frac{\partial y_i}{\partial x_{r,y})}\end{aligned}$$

$$\begin{aligned}\frac{\partial h_i}{\partial x_{r,\theta}} &= \cos(\theta_i + x_{r,\theta})\frac{\partial}{\partial x_{r,\theta}}(r\cos(\theta_i + x_{r,\theta}) - x_i) + \\ &+ \sin(\theta_i + x_{r,\theta})\frac{\partial}{\partial x_{r,\theta}}(r\sin(\theta_i + x_{r,\theta}) - y_i)\end{aligned}$$

Si devono ancora svolgere le derivate parziali di x_i, y_i, punti sulla parete, dipendenti dalla posizione e dall'orientamento del robot. Nel precedente paragrafo si è sviluppato il calcolo per determinare un generico punto (x_p, y_p) sulla parete quando fosse nota la configurazione del robot. Risolvendo i vari sistemi di equazioni proposti si ottiene la seguente soluzione:

$$\begin{aligned}y_p &= -\frac{-m_p x_{r,y}\cos(\theta_i + x_{r,\theta}) + m_p x_{r,x}\sin(\theta_i + x_{r,\theta}) + q_p \sin(\theta_i + x_{r,\theta})}{m_p\cos(\theta_i + x_{r,\theta}) - \sin(\theta_i + x_{r,\theta})} \\ x_p &= -\frac{q_p\cos(\theta_i + x_{r,\theta}) + x_{r,y}\cos(\theta_i + x_{r,\theta}) + x_{r,x}\sin(\theta_i + x_{r,\theta})}{m_p\cos(\theta_i + x_{r,\theta}) - \sin(\theta_i + x_{r,\theta})}\end{aligned}$$

con $(x_i, y_i) = (x_p, y_p)$ e

$$\begin{aligned}m_p &= \frac{P_{1,y} - P_{2,y}}{P_{1,x} - P_{2,x}} \\ q_p &= \frac{-P_{2,x}P_{1,x} + P_{2,x}P_{1,y}}{P_{1,x} - P_{2,x}}.\end{aligned}$$

Le derivate parziali sono:

$$\begin{aligned}\frac{\partial x_i}{\partial x_{r,x}} &= -\frac{\sin(\theta_i + x_{r,\theta})}{m_p\cos(\theta_i + x_{r,\theta}) - \sin(\theta_i + x_{r,\theta})} \\ \frac{\partial x_i}{\partial x_{r,y}} &= \frac{\cos(\theta_i + x_{r,\theta})}{m_p\cos(\theta_i + x_{r,\theta}) - \sin(\theta_i + x_{r,\theta})} \\ \frac{\partial x_i}{\partial x_{r,\theta}} &= -\frac{q_p - x_{r,y}) + m_p x_{r,x}}{m_p^2\cos(\theta_i + x_{r,\theta})^2 - 2\sin(\theta_i + x_{r,\theta})m_p\cos(\theta_i + x_{r,\theta}) + 1 - \cos(\theta_i + x_{r,\theta})^2} \\ \frac{\partial y_i}{\partial x_{r,x}} &= -\frac{m_p\sin(\theta_i + x_{r,\theta})}{m_p\cos(\theta_i + x_{r,\theta}) - \sin(\theta_i + x_{r,\theta})} \\ \frac{\partial y_i}{\partial x_{r,y}} &= \frac{m_p\cos(\theta_i + x_{r,\theta})}{m_p\cos(\theta_i + x_{r,\theta}) - \sin(\theta_i + x_{r,\theta})} \\ \frac{\partial y_i}{\partial x_{r,\theta}} &= -\frac{m_p(q_p - x_{r,y}) + m_p x_{r,x})}{m_p^2\cos(\theta_i + x_{r,\theta})^2 - 2\sin(\theta_i + x_{r,\theta})m_p\cos(\theta_i + x_{r,\theta}) + 1 - \cos(\theta_i + x_{r,\theta})^2}\end{aligned}$$

dalle quali sostituendo

$$\begin{aligned}
\frac{\partial h}{\partial x_{r,x}} &= \frac{-m_p}{m_p\cos(\theta_i + x_{r,\theta}) - \sin(\theta_i + x_{r,\theta})}\\
\frac{\partial h}{\partial x_{r,y}} &= \frac{1}{m_p\cos(\theta_i + x_{r,\theta}) - \sin(\theta_i + x_{r,\theta})}\\
\frac{\partial h}{\partial x_{r,\theta}} &= +\frac{q_p\cos(\theta_i + x_{r,\theta}) - x_{r,y}\cos(\theta_i + x_{r,\theta})) + m_p x_{r,x}\cos(\theta_i + x_{r,\theta})}{m_p^2\cos(\theta_i + x_{r,\theta})^2 - 2\sin(\theta_i + x_{r,\theta})m_p\cos(\theta_i + x_{r,\theta}) + 1 - \cos(\theta_i + x_{r,\theta})^2}\\
&\quad + \frac{q_p\sin(\theta_i + x_{r,\theta})m_p - x_{r,y}\sin(\theta_i + x_{r,\theta})m_p + m_p^2 x_{r,x}\sin(\theta_i + x_{r,\theta})}{m_p^2\cos(\theta_i + x_{r,\theta})^2 - 2\sin(\theta_i + x_{r,\theta})m_p\cos(\theta_i + x_{r,\theta}) + 1 - \cos(\theta_i + x_{r,\theta})^2}
\end{aligned}$$

Queste equazioni costituiscono gli elementi della matrice H.

$$\mathbf{H} = \begin{bmatrix} \frac{\partial h_1}{\partial x_{r,x}} & \frac{\partial h_1}{\partial y_{r,y}} & \frac{\partial h_1}{\partial x_{r,\theta}} \\ \frac{\partial h_2}{\partial x_{r,x}} & \frac{\partial h_2}{\partial y_{r,y}} & \frac{\partial h_2}{\partial x_{r,\theta}} \\ \dots & & \\ \frac{\partial h_N}{\partial x_{r,x}} & \frac{\partial h_N}{\partial y_{r,y}} & \frac{\partial h_N}{\partial x_{r,\theta}} \end{bmatrix}$$

Inserendo nelle espressioni i vari coefficienti numerici si ottiene la matrice H calcolata sul punto X_r.

Nota numericamente H, il calcolo della matrice di covarianza $S(k+1)$ si semplifica notevolmente. Infatti

$$\begin{aligned}
S(k+1) &= E\{[y(k+1) - \hat{y}(k+1)][y(k+1) - \hat{y}(k+1)]'\}\\
&= E\{\hat{y}(k+1) + \hat{y}'(k+1)\} + E\{y(k+1) + y'(k+1)\}\\
&\quad - E\{y(k+1) + \hat{y}'(k+1)\} - E\{\hat{y}(k+1) + y'(k+1)\}\\
&= E\{[H(k+1)\hat{x}_r(k+1|k)][H(k+1)\hat{x}_r(k+1|k)]'\} + E\{y(k+1)y'(k+1)\}\\
&= H(k+1)E\{\hat{x}_r(k+1|k)\hat{x}_r'(k+1|k)\}H'(k+1) + E\{y(k+1)y'(k+1)\}\\
&= H(k+1)P(k+1|k)H'(k+1) + R(k+1)
\end{aligned}$$

dove la terza uguaglianza vale nell'ipotesi di incorrelazione tra $y(k+1)$ e $\hat{y}(k+1)$.

La matrice di covarianza trovata è un indice sulla qualitàdel confronto effettuato.
L'ultimo passo dell'algoritmo, note le due matrici di covarianza $S(k+1)$ e $F(k+1)$, consiste nel calcolare i guadagni di $K(k+1)$ del filtro di Kalman.

$$\begin{aligned}
\hat{x}(k+1|k+1) &= \hat{x}(k+1|k) + K_{k+1}\tilde{y}(k+1) \\
P(k+1|k+1) &= [I - K_{k+1}H(k+1)]P(k+1|k) \\
K_{k+1} &= P(k+1|k)H^{'}(k+1)\left(R(k+1) + H(k+1)P(k+1|k)H^{'}(k+1)\right)^{-1} \\
&= P(k+1|k)H^{'}(k+1)S^{-1}(k+1)
\end{aligned}$$

Queste equazioni completano la fase di stima.

Capitolo 5

Risultati Simulativi

5.1 Ambiente di simulazione

Tutti gli esperimenti sono stati condotti sul simulatore del Nomad 150: Nserver. L'architettura dell'ambiente simulativo consiste (come già descritto nel primo capitolo) in un rapporto client-server; il client è il programma che implementa il filtro di Kalman esteso (in $C++$), mentre il server si occupa di attuare le disposizioni del client mostrando i risultati su un'opportuna interfaccia grafica. Il robot ed il server condividono gli stessi comandi, la differenziazione tra simulatore e robot si attua nelle relazioni con l'ambiente esterno in quanto nel simulatore sussistono evidenti semplificazioni. Nella figura (5.1) è riprodotto bidimensionalmente lo spazio di lavoro del robot, e in particolare si vede l'ambiente in cui esso viene fatto muovere durante le simulazioni. Le linee nere rappresentano le pareti della stanza dalle dimensioni 290×290 cm. Il robot può muoversi solo all'interno di questo quadrato, qualsiasi posizione al di fuori di esso è per il robot irraggiungibile, quindi la simulazione non inizia per niente. Le due piccole finestre di destra riportano graficamente l'andamento delle misure sonar. Le misure degli encoder e dei sonar nel simulatore non sono perturbate da alcun errore, sono esatte: per valutare l'algoritmo di localizzazione sono stati introdotti quindi disturbi gaussiani.

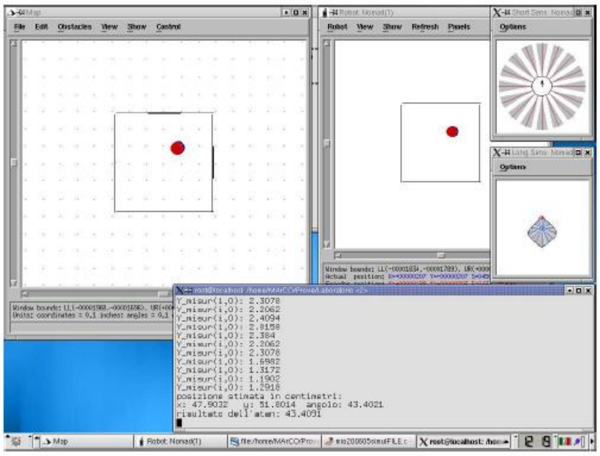


Figura 5.1: Ambiente di lavoro riprodotto sull'N-Server

5.2 Prove condotte

Nella pratica si è riprodotto l'ambiente dell'N-server in Matlab, per facilità di implementazione. Il programma prevede quindi la creazione di una stanza a forma di parallelogramma, all'interno della quale sono state condotte varie simulazioni. L'algoritmo consta di due parti molto importanti: la prima costituita dalle equazioni del filtro di Kalman e dall'implementazione dell'algoritmo di localizzazione e la seconda dalll'algoritmo di controllo. Per quanto riguarda quest'ultimo i passi sono i seguenti:

- calcolo del δ_ρ e del δ_θ in base alla $\hat{x}_k$ e alla $\hat{y}_k$;
- definizione di $\delta_\rho^r = \delta_\rho + \eta_\rho$ e di $\delta_\theta^r = \delta_\theta + \eta_\theta$
  dove η_ρ e η_θ sono errori gaussiani a media nulla proporzionali alla distanza;
- calcolo delle equazioni del sistema

$$\begin{aligned} x_{k+1} &= x_k + \delta_\rho^r \cos\theta_k \\ y_{k+1} &= y_k + \delta_\rho^r \cos\theta_k \\ \theta_{k+1} &= \theta_k + \delta_{\theta k}^r \end{aligned}$$

.

Nel calcolo del δ_ρ e del δ_θ vanno tenute presenti le seguente equazioni

$$
\begin{aligned}
\delta_\rho &= \alpha_\rho \quad (norm[(\hat{x},\hat{y}) - (x_{Des}, y_{Des})]) \\
\delta_\rho &= min\{\delta_{\rho max}, \delta_\rho\} \\
\delta_\theta &= \alpha_\theta \quad \Delta_\theta \\
\delta_\theta &= sign(\delta_\theta) \quad min\{\delta_{\theta max}, \mid \delta_\theta \mid\}.
\end{aligned}
$$

Dove α_ρ e α_θ sono costanti non maggiori di 1, mentre $\delta_{\rho max}$ e $\delta_{\theta max}$ sono costanti scelte opportunamente in base alla velocità del robot. È preferibile che il loro valore rimanga al di sotto dell'unità e il loro ruolo è fondamentale sia nel raggiungimento del goal che nel funzionamento del filtro.

L'applicazione del filtro concatenato all'algoritmo di localizzazione permette al robot di raggiungere la posizione desiderata in modo soddisfacente e inoltre fa si che la posizione reale insegua quella stimata, concorrendo così ad un buon funzionamento del programma. Il filtro viene implementato utilizzando le equazioni di cui si è parlato ampiamente nel capitolo precedente: si calcolano quindi le matrici di covarianza della stima iniziale, delle misure e del rumore sulla dinamica. Inizia poi la fase di predizione che viene associata in un ciclo while all'algoritmo di localizzazione e contemporaneamente viene, ad ogni passo, aggiornata la stima.

5.3 Risultato delle simulazioni

Il grafico che segue è un esempio di traiettoria che il robot deve raggiungere. Le coordinate di partenza sono (0,0) e quelle desiderate (60,40).

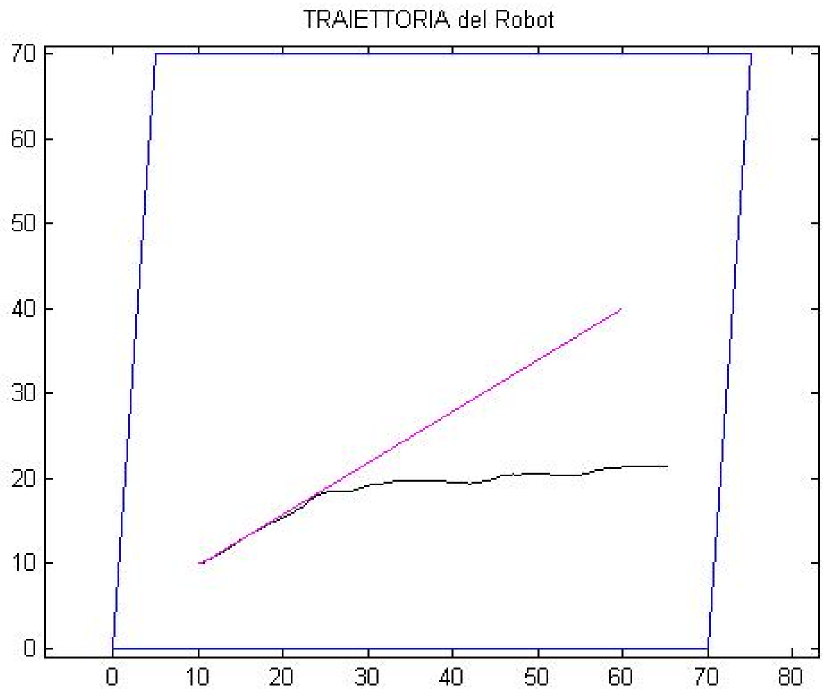
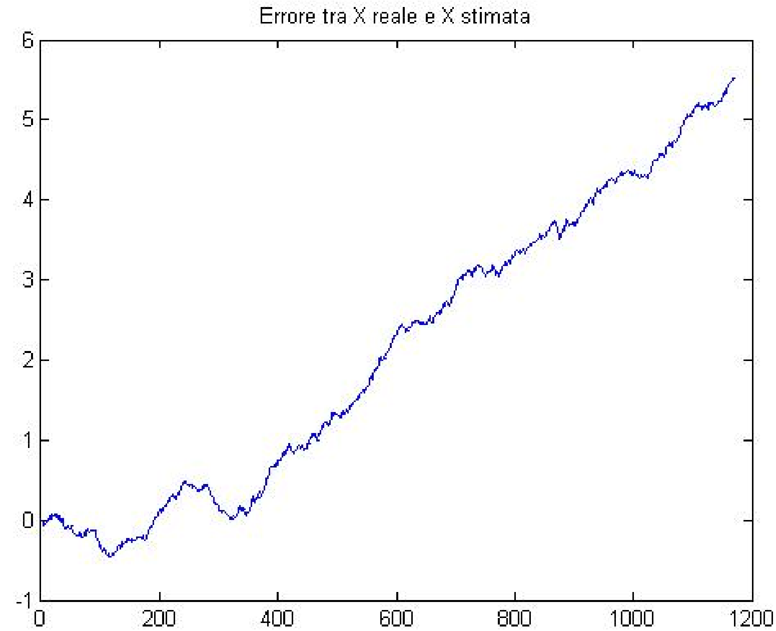
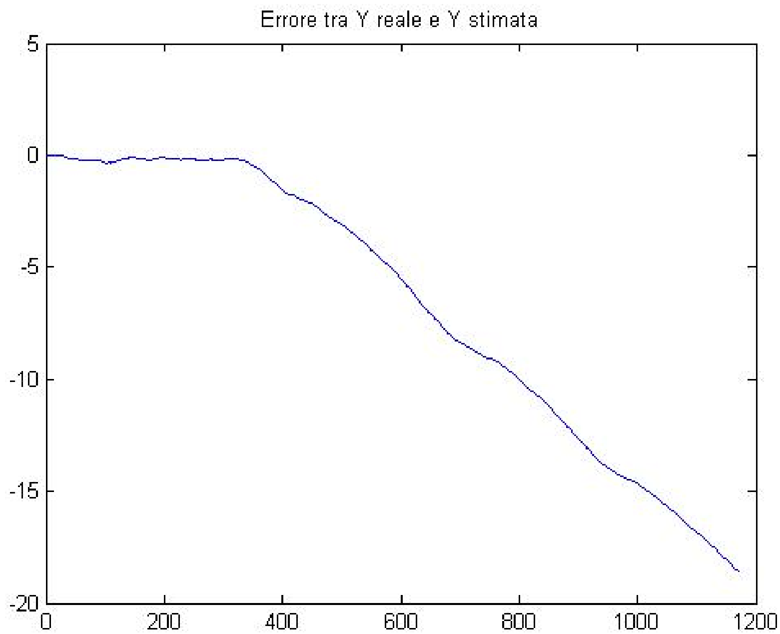


Figura 5.2: In questa simulazione è stato utilizzato l'algoritmo di controllo senza l'applicazione del filtro di Kalman. Come si può vedere abbiamo una forte divergenza nel tempo tra la posizione stimata (in colore magenta) e la posizione reale (rappresentata in nero) tanto che alla fine il robot reale non riesce a raggiungere la posizione desiderata. Per questo esempio sono stati scelti: $\delta\rho_{max} = 0.05$, $\alpha\rho = 0.1$, $\delta\theta_{max} = 0.05$ e $\alpha\theta = 0.1$.


Errore tra X reale e X stimata
6
5
4
3
2
1
0
-1
0
200
400
600
800
1000
1200


Errore tra Y reale e Y stimata
5
0
-5
-10
-15
-20
0
200
400
600
800
1000
1200

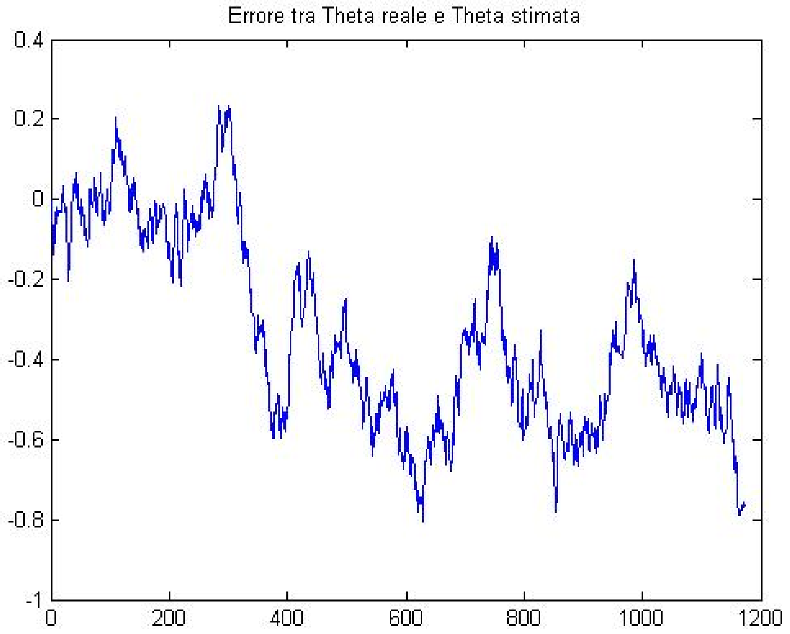


Figura 5.3: Nei tre grafici precedenti sono invece riportati gli errori sulle coordinate x, sulle coordinate y e sull'angolo θ, ossia la differenza tra le coordinate reali e quelle stimate e tra l'angolo reale e quello stimato.

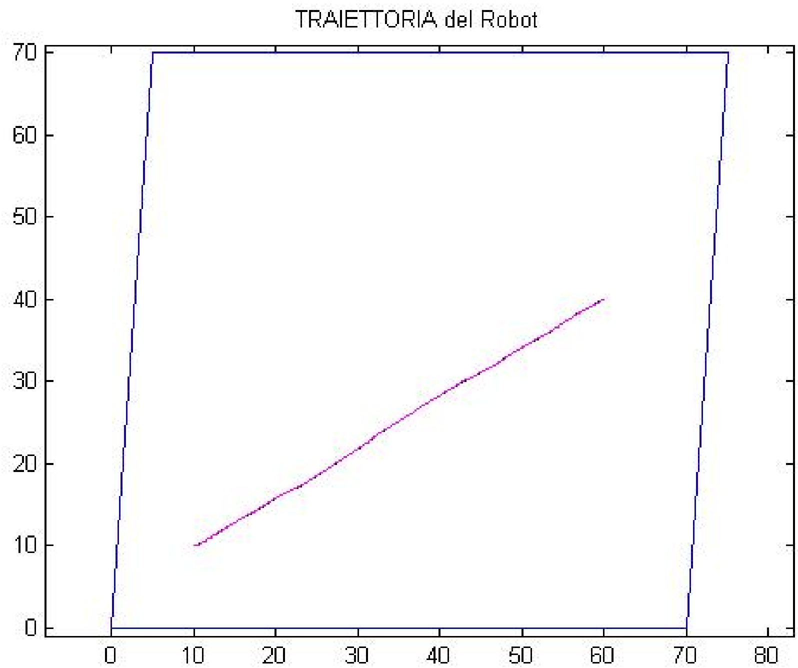
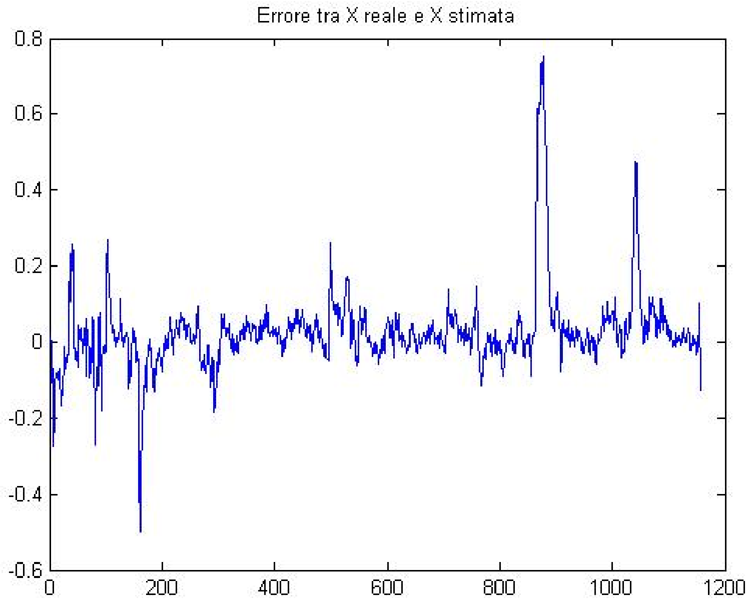
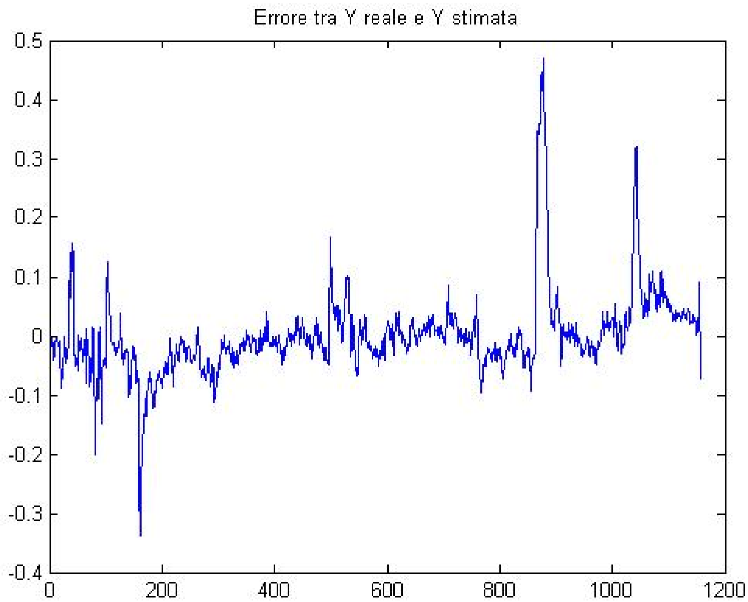


Figura 5.4: Si può vedere come, grazie all'uso del filtro di Kalman Esteso, diminuisca notevolmente la divergenza nel tempo tra la posizione stimata (in colore magenta) e la posizione reale (rappresentata in nero), che vanno quasi a coincidere, e inoltre l'algoritmo di localizzazione ci permette di ottenere una traiettoria accettabile. Come spiegato precedentemente la scelta dei parametri $\delta\rho_{max}$, $\delta\theta_{max}$, $\alpha\rho$ e $\alpha\theta$ è fondamentale. Per questo esempio sono stati scelti: $\delta\rho_{max} = 0.05$, $\alpha\rho = 0.1$, $\delta\theta_{max} = 0.05$ e $\alpha\theta = 0.1$.


Errore tra X reale e X stimata
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
200
400
600
800
1000
1200


Errore tra Y reale e Y stimata
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
0
200
400
600
800
1000
1200

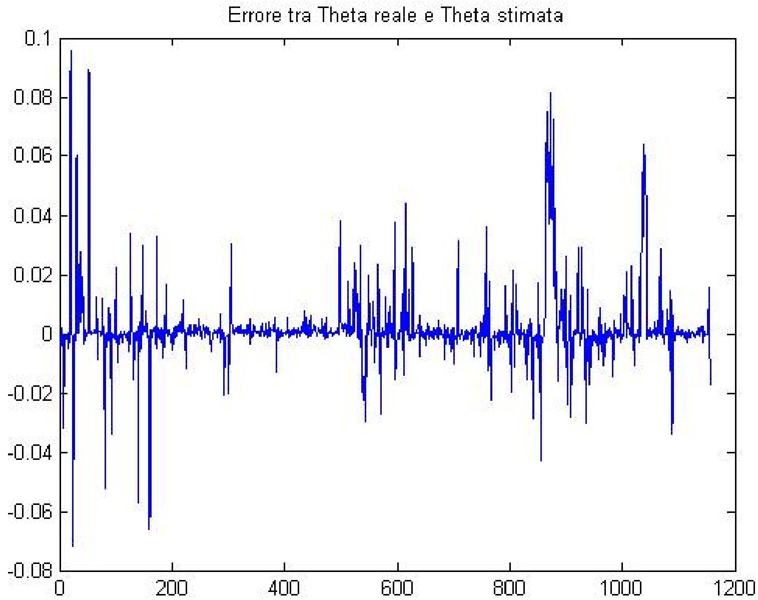


Figura 5.5: Nei tre grafici precedenti sono invece riportati gli errori sulle coordinate x, sulle coordinate y e sull'angolo θ, ossia la differenza tra le coordinate reali e quelle stimate e tra l'angolo reale e quello stimato. Il numero di passi necessari al raggiungimento del goal sono 1199, come si nota dai grafici.

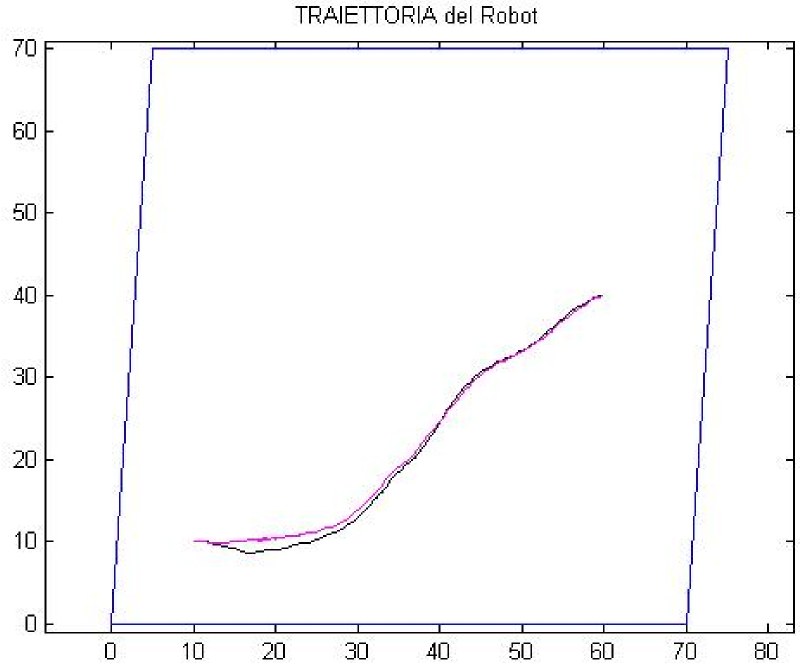
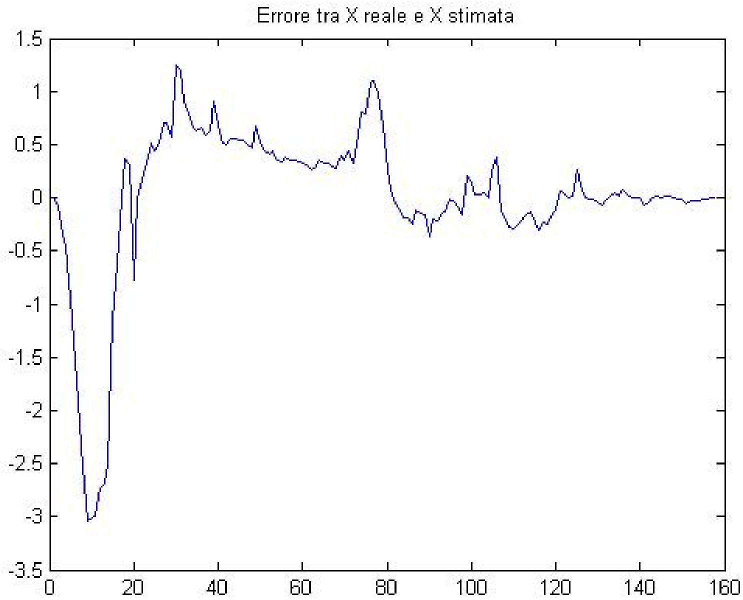
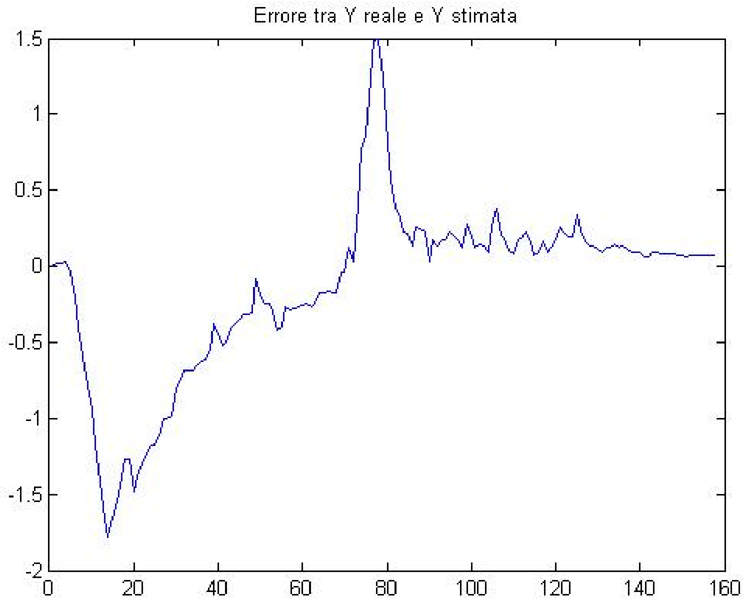


Figura 5.6: Con la seguente simulazione si vuole evidenziare la differenza dei risultati, sempre sulla stessa traiettoria ossia con partenza da (0,0) e arrivo nel punto (60,40), a causa di una cattiva scelta dei parametri $\delta\rho_{max}$, $\delta\theta_{max}$, $\alpha\rho$ e $\alpha\theta$. A seguito di svariate prove si è dimostrato che più aumenta il valore di $\delta\rho_{max}$ e $\delta\theta_{max}$, minore risulta essere l'efficienza dell'algoritmo di localizzazione, poichè gli errori sull'odometria e sui dati sensoriali diventano troppo grandi da far si che in alcuni parti della traiettoria si riscontri un graduale allontanamento tra la posizione stimata e quella effettiva del robot. Per questo esempio sono stati scelti: $\delta\rho_{max} = 0.5$, $\alpha\rho = 0.08$, $\delta\theta_{max} = 0.5$ e $\alpha\theta = 0.8$.


Errore tra X reale e X stimata
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
0
20
40
60
80
100
120
140
160


Errore tra Y reale e Y stimata
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
20
40
60
80
100
120
140
160

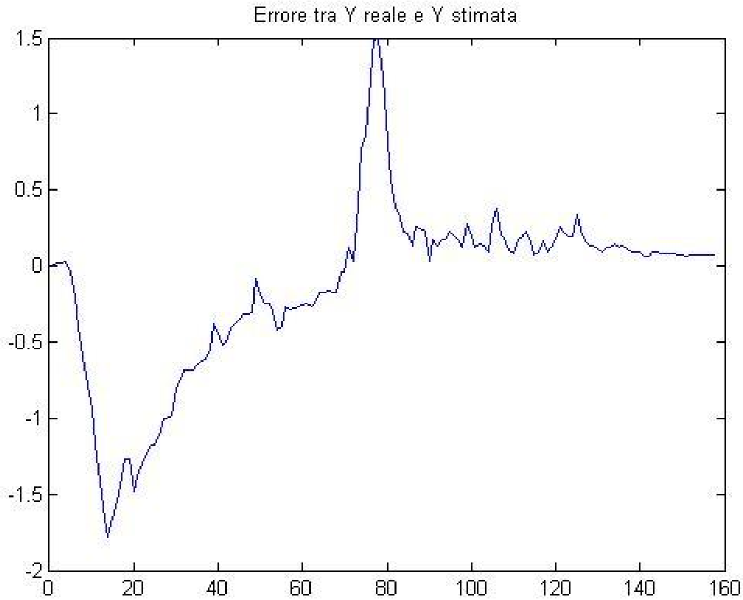


Figura 5.7: Anche gli errori sulle coordinate x, sulle coordinate y e sull'angolo θ aumentano, come si denota dai tre grafici sopra e il numero di passi necessari al raggiungimento del goal diminuisce vertiginosamente arrivando a 150, rimanendo però costante il tempo impiegato. Accade quindi che il campionamento viene fatto più a lungo.

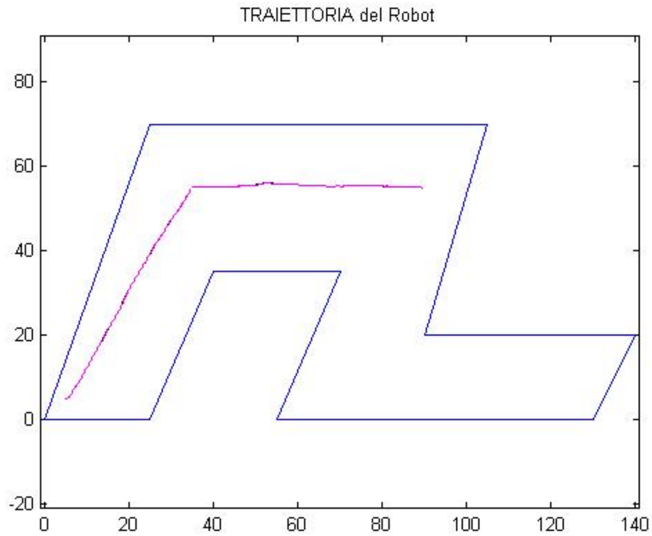


Figura 5.8: Esempio di traiettoria guidata all'interno di una stanza con varie pareti. La posizione di partenza è (5,5) la prima posizione da raggiungere è (35,55) e l'ultima (90,55)

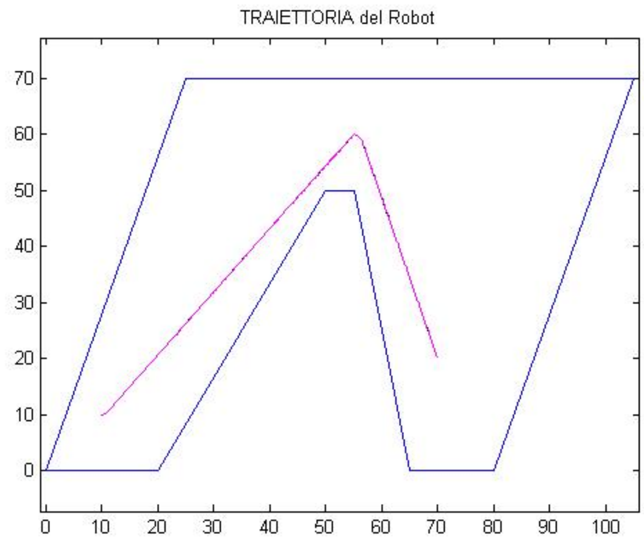


Figura 5.9: Esempio di traiettoria guidata all'interno di una stanza con vertice nel centro. La posizione di partenza è (10,10) la prima posizione da raggiungere è (60,60) e l'ultima (70,20). Con le simulazione delle traiettorie guidate si è voluto testare il funzionamento del programma e i risultati ottenuti sono stati buoni.

Capitolo 6

Conclusioni

La fusione delle informazioni dei sensori ultrasonici con i dati odometrici, mediante filtro di Kalman esteso, risolve il problema della localizzazione, portando ad una sostanziale coincidenza lo stato stimato con quello effettivo. Il filtro di Kalman utilizzato per la localizzazione è stato ottimizzato per un uso sul simulatore del robot, mai sul robot reale e si è dimostrato essere una soluzione valida per risolvere il problema della fusione sensoriale. I limiti dell'approccio considerato sono costituiti proprio dalle oscillazioni dovute agli errori di misura: la stima non diverge dalla posizione effettiva ma diviene nervosamente variabile e attuare un controllo in funzione di questa non è facile. Per migliorare la risposta complessiva è necessario agire sulle caratteristiche hardware del robot:

- sensori acustici con maggiore risoluzione;
- maggiore rapidità di rilevamento e lettura degli stessi;
- numero più elevato dei sensori sonar per aumentare la risoluzione angolare.

La stima dell'orientamento è la più delicata in quanto le maggiori vibrazioni sulla stima dello stato, relativamente alle coordinate, sono dovute alla rumorosità della misura dei sensori ultrasonici. La stima dell'orientamento è fortemente dipendente dalla bassa risoluzione angolare degli stessi per questo

non si sono ottenuti buoni risultati nel raggiungimento di un punto prefissato della stanza.
La qualità della risposta, infine, è dovuta anche alle notevoli approssimazioni lineari effettuate. Il filtro viene applicato su un modello fortemente non lineare, linearizzato e trattato come un sistema lineare non stazionario, come si nota dai calcoli sulle relative matrici di covarianza.

Bibliografia

[1] Robert Grover Brown, Patrick Y. C. Hwang, "Introduction to random signals and applied Kalman filtering", John Willey & Sons, Inc, 1992.

[2] E.Fornasini, G.Marchesini. *Appunti di Teoria dei Sistemi.* Edizioni Libreria Progetto, 1994

[3] Greg Welch, Gary Bishop, "An Introduction to the Kalman Filter", Department of Computer Science University of North Carolina at Chapel Hill, 2004.

[4] M. Gerosi, "Navigazione e localizzazione del robot Nomad 150 mediante filtro di Kalman", Tesi di Laurea in Ingegneria Informatica, Università di Roma Tor Vergata, 2005.

[5] O.M.Grasselli. *Proprietà strutturali dei sistemi lineari e stazionari.* Pitagora Editrice, 1978.

[6] F.Grandoni, A.Martinelli, F.Martinelli, S.Nicosia, P.Valigi. *Sensor fusion for robot localisation.* Università di Roma Tor Vergata.

[7] Definizione e sviluppo algoritmi di pianificazione e navigazione. Technical report, Università di Roma Tor Vergata, 2001. Accordo di Programma ENEA-NURST L.95/95, Calcolo parallelo con applicazioni alla robotica; Relazione 2.1: stato di avanzamento al 31.12.2001.

[8] F.Romanelli. *Robot Nomad 150: nozioni di base.* Laboratorio di Automatica. (F.Martinelli)

[9] F.Forni, [2004],“Filtro di Kalman Esteso per la localizzazione di Robot Mobili”, Tesi di laurea in Ingegneria Informatica, Università di Roma Tor Vergata.

[10] Nomadic Technologies, Inc. [1997],*Nomad150-User's Manual*, Mountain View, CA, USA.

[11] Nomadic Technologies, Inc. [1997],*Nomad150-Hardware Manual*, Mountain View, CA, USA.