UNIVERSITÀ DEGLI STUDI DI ROMA
TOR VERGATA

FACOLTÀ DI INGEGNERIA

CORSO DI LAUREA IN INGEGNERIA
DELL'AUTOMAZIONE

A.A. 2009/2010

Tesi di Laurea

L'Algoritmo di SLAM: il primo passo verso un robot autonomo

RELATORE

Prof.Francesco Martinelli

CANDIDATO

Antonio De Stasio

. . . dedicato a te che leggi

Indice

Introduzione

Uno degli aspetti fondamentali della robotica mobile è stimare la posizione del robot all'interno dell'ambiente operativo. Tale stima può essere effettuata in maniera differente, a seconda del contesto applicativo in cui il robot viene immerso. Talvolta è sufficiente conoscerne la sola variazione temporale, ossia lo spostamento relativo rispetto ad una posizione precedente. Altre volte viene richiesta la conoscenza della posizione assoluta all'interno dell'ambiente, soprattutto nei compiti in cui il robot deve raggiungere locazioni specifiche fornitegli secondo il sistema di riferimento globale. Il problema diventa ancora più arduo quando la conoscenza a priori dell'ambiente non è completamente disponibile. In queste situazioni, il robot deve essere in grado di estrapolare oltre alle informazioni riguardanti la sua posizione, anche le informazioni che riguardano la struttura metrica e topologica del sito operativo. Tali informazioni, quindi, sono indispensabili se lo scopo finale è quello di fornire al robot l'autonomia necessaria a renderlo un valido supporto per il genere umano. Basti pensare a situazioni post-terremoto, in cui i gravi danni subiti dagli edifici creano pericoli insidiosi alle squadre di soccorso. L'avere a disposizione una serie di robot ben equipaggiati, capaci di muoversi in maniera autonoma o semi-autonoma, può risultare di grande aiuto per l'individuazione delle vittime, di possibili corridoi ostruiti o finanche di cedimenti strutturali molto pericolosi. Per ottenere ciò, è indispensabile che il robot sia in grado di conoscere la sua posizione all'interno dell'edificio e costruirne una mappa

Figura 1: Alcuni esempi in cui un algoritmo di SLAM è necessario

chiara che contenga le informazioni necessarie per un intervento mirato e molto più efficace.

Per avere un'idea della necessità ma al tempo stesso della complessità di raccogliere tali informazioni, si immagini, ad esempio, di trovarsi all'interno di una città sconosciuta e di voler raggiungere un punto specifico (per esempio via Roma). Una soluzione immediata sarebbe quella di consultare una piantina della città, individuare la propria posizione (magari navigando per un po' di tempo alla ricerca di elementi disambiguanti) e infine pianificare il percorso per raggiungere l'obiettivo preposto. Nel campo della robotica mobile tali operazioni sono conosciute con i termini di **localizzazione** e **path-planning**. Il problema della localizzazione consiste nell'ottenere la posizione del robot avendo una conoscenza a priori dell'ambiente (la vostra piantina della città) , il path-planning nel pianificare il percorso. Nel caso in cui non sia disponibile una piantina della città, l'unica soluzione è quella di esplorare l'ambiente circostante, sperando di raggiungere l'obiettivo preposto. Durante quest'esplorazione, però è utile non ritornare nei punti già visitati, costruendo mentalmente una mappa grossolana della zona e cercando di capire qual è la posizione occupata al suo interno. Analogamente, nella robotica mobile, esiste il problema di costruire la mappa di un ambiente sconosciuto e individuare la propria posizione al suo interno. Tale problema è conosciuto col termine **SLAM** (*Simultaneous Localization And Mapping*) ed è uno

dei problemi maggiormente affrontati negli ultimi anni dalla comunità robotica internazionale. Il lavoro di questa tesi mira a fornire uno dei mattoni fondamentali per la costruzione di sistemi robotici avanzati. Per fare ciò sono stati affrontati i problemi di Localizzazione e di SLAM. Il primo riguarda la conoscenza della posizione del robot all'interno di un ambiente quando la mappa di tale ambiente è conosciuta a priori. Il secondo si occupa del problema più generale di stimare congiuntamente la posizione del robot e la mappa dell'ambiente.

I contesti applicativi in cui è possibile utilizzare le tecniche discusse in questa tesi sono molteplici. In linea di principio, ogni volta che viene utilizzato un robot mobile in applicazioni dove è necessario navigare all'interno di un ambiente, dotarlo di un algoritmo di localizzazione robusto e affidabile si rivela molto utile. In particolare è possibile fare una distinzione tra situazioni in cui è possibile acquisire una conoscenza a priori della mappa, e situazioni in cui l'obiettivo del robot è quello di operare in maniera autonoma all'interno di un ambiente sconosciuto. Nel primo caso è possibile costruire la mappa off-line e utilizzarla come input per un algoritmo di localizzazione, sebbene un feedback on-line durante la fase di costruzione porti a risultati migliori nella fase di esplorazione. Nel secondo caso, invece, una stima della mappa on-line è essenziale per i processi più ad alto livello che controllino il moto del robot, e che lo rendano veramente autonomo.

Capitolo 1

Robot: un modello cinematico

In questo capitolo viene fornita una struttura di massima del robot. Questa è stata pensata per affrontare sia i problemi di localizzazione che quelli di mapping. Verrà fatto un breve accenno sui tipi di sensori generalmente utilizzati in questo ambito con le loro peculiarità. Si identifica dunque il modello cinematico che verrà utilizzato nei diversi studi affrontati in questa tesi.

1.1 Sensori

Il disporre di una buona dotazione sensoristica è di fondamentale importanza quando si lavora con piattaforme robotiche. In caso di compiti complessi che richiedono l'interazione con l'ambiente, l'aver a disposizione un'adeguata sensoristica è una precondizione essenziale al fine del successo dell'applicazione. Per fare ciò il robot deve essere a conoscenza dei suoi movimenti, un'informazione che gli viene fornita da una strumentazione capace di acquisire informazioni riguardo la cinematica e la dinamica. Inoltre deve essere capace di ottenere informazioni riguardo la struttura metrica e topologica dell'ambiente in cui lavora, in modo da percepire la presenza di ostacoli (come possono essere dei muri o oggetti presenti nell'ambiente) e di configurazioni amiche che gli permettano di individuare la propria posizione nell'ambiente. Infine, per po-

ter costruire una mappa dell'ambiente, il robot deve obbligatoriamente raccogliere le informazioni necessarie per poi fonderle, con varie tecniche di fusione sensoriale, in una struttura che sia utile per la navigazione successiva o per analisi off-line anche da parte di un operatore umano.

Possiamo distinguere due categorie di sensori con due diverse finalità: **sensori di spostamento relativo** e **sensori di prossimità**.

I primi permettono di conoscere lo spostamento relativo del robot, ossia le piccole variazioni di orientamento e posizione all'interno dell'ambiente in cui il robot si sta muovendo. Tutti i sensori di questo tipo compiono esclusivamente misure cinematiche e dinamiche rilevabili a bordo del robot. Solitamente le informazioni di questo tipo più a basso livello che è possibile ottenere, quando si ha a che fare con robot mobili dotati di ruote, sono le rotazioni compiute dalle ruote stesse. Tali informazioni vengono analizzate per poter ottenere lo spostamento effettivo con una metodologia di analisi chiamata **odometria** . La rotazione delle ruote è registrata tramite dei dispositivi che vengono montati sulle ruote stesse e che rivelano piccoli spostamenti angolari. Tali dispositivi sono chiamati *encoder*, i più diffusi dei quali sono gli encoder ottici.

I secondi sono necessari qualora il robot debba interagire con l'ambiente esterno, in particolare se deve affrontare dei problemi di mapping. Vengono dunque rilevate, ad esempio, distanza e angolazione di oggetti esterni rispetto al robot. La maggior parte di questi sensori emette energia nell'ambiente e utilizza la quantità riflessa (tramite misurazioni di sfasamento o tempi di eco) al fine di ottenere la distanza degli oggetti esterni. Tali sensori vengono definiti *attivi*. Un'altra classe di sensori, definiti *passivi*, sfrutta invece l'energia che è già presente nell'ambiente, come ad esempio i sistemi di visione.

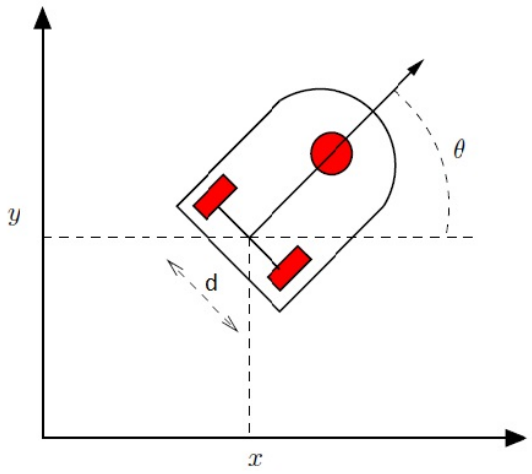

Figura 1.1: Schema di un uniciclo a guida differenziale

1.2 Modello cinematico

In questa sezione si descrive il modello cinematico del robot utilizzato in questa tesi, soffermandosi in particolare sullo schema dell'uniciclo a guida differenziale (Figura 1.1), derivandolo dai vincoli anolonomi delle ruote.
Infine viene introdotta l'odometria come tecnica per ricostruire la posa del robot a partire dai dati sensoriali.

La realizzazione del modello di un uniciclo a guida differenziale è costituita da due ruote motrici allineate tra di loro, più una o più ruote indipendenti a sostegno della base mobile. La direzione di traslazione, individuata da θ, è perpendicolare all'asse che collega i centri delle due ruote e rappresenta l'orientamento del robot nel piano. In questo modello è presente il solo vincolo di puro rotolamento, che è esprimibile come segue:

$$\begin{bmatrix} \sin\theta & -\cos\theta & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = 0 \quad . \tag{1.2.1}$$

Il vincolo esprime il fatto che il robot può traslare nel piano nella sola direzione individuata da θ. Nonostante ciò, per questo modello il piano è uno spazio completamente raggiungibile, nel senso che date due qualsiasi configurazioni q_0 e q_1, il robot può sempre raggiungere q_1 partendo da q_0 attraverso una serie di rotazioni intorno all'asse e di traslazioni lungo θ. Il robot viene controllato tramite i motori fissati sulle due ruote, per cui le equazioni cinematiche sono:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\cos\theta & \frac{1}{2}\cos\theta \\ \frac{1}{2}\sin\theta & \frac{1}{2}\sin\theta \\ \frac{1}{d} & -\frac{1}{d} \end{bmatrix} \begin{bmatrix} v_R \\ v_L \end{bmatrix} \tag{1.2.2}$$

dove v_R rappresenta la velocità lineare della ruota destra, v_L quella della ruota sinistra e d lo spiazzamento fra le due ruote. Una volta che il modello cinematico del robot è noto e si hanno a disposizione strumentazioni in grado di rilevare lo spostamento delle ruote, è possibile ottenere la posa del robot nell'ambiente. Questo processo è noto con il nome di *dead reckoning* o *odometria*.

1.2.1 Odometria

L'odometria è un metodo di fusione dei dati sensoriali utilizzato per ottenere informazioni riguardo la posa del robot, a partire da misure di spostamento incrementali fornite dai sensori. La conoscenza, nel caso di robot mobili su ruote, della cinematica del robot e della rotazione compiuta dalle ruote nell'intervallo di tempo ΔT_i permette di conoscere lo spostamento effettuato dal robot nell'intervallo considerato. Tale spostamento può essere considerato nelle sue due componenti: la componente traslazionale $[\Delta x_i \ \Delta y_i]$ e quella rotazionale $\Delta\theta_i$. Possiamo dunque definire il vettore di spostamento relativo nell'intervalo ΔT_i come $\Delta u_i = [\Delta x_i \ \Delta y_i \ \Delta\theta_i]^T$.

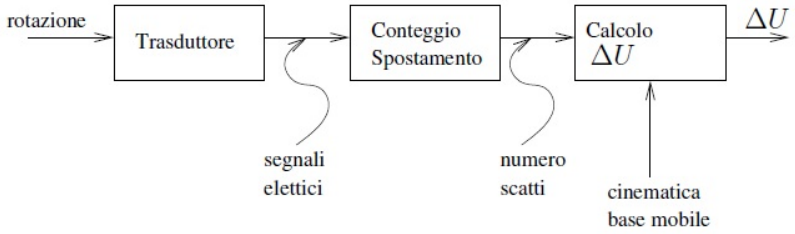

Figura 1.2: Processo odometrico

L'intero processo odometrico si svolge attraverso tre fasi principali. La prima si basa sulla trasformazione in segnali elettrici della rotazione fisica che effettuano le ruote, tramite gli encoder. Queste grandezze vengono poi elaborate per estrapolare numericamente la rotazione relativa o assoluta compiuta da ogni ruota. Infine viene calcolato Δu_i partendo da uno stadio in cui è nota la cinematica del robot. Per ottenere lo spostamento Δu_i del robot nell'intervallo ΔT_i è possibile usare la versione discretizzata dell'equazione (1.2.2), riferendola ai passi encoder compiuti dalle singole ruote piuttosto che alle loro velocità. In questo modo abbiamo che:

$$\begin{bmatrix} \Delta x_i \\ \Delta y_i \\ \Delta \theta_i \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\cos\theta & \frac{1}{2}\cos\theta \\ \frac{1}{2}\sin\theta & \frac{1}{2}\sin\theta \\ \frac{1}{d} & -\frac{1}{d} \end{bmatrix} \begin{bmatrix} \Delta u_{R_i} \\ \Delta u_{L_i} \end{bmatrix} \tag{1.2.3}$$

dove Δu_{R_i} e Δu_{L_i} rappresentano rispettivamente gli spostamenti della ruota destra e della ruota sinistra ottenuti a partire dalle misurazioni fornite dagli encoder.
Lo spostamento lineare di una singola ruota è uguale a:

$$\Delta u_i = c_m N_i \tag{1.2.4}$$

dove N_i rappresenta il numero di tick che l'encoder ha misurato a partire dall'ultima lettura e c_m è il fattore di conversione tra distanze percorse e tick dell'encoder.

Capitolo 2

Localizzazione tramite Filtro di Kalman

I problemi di localizzazione, mapping e SLAM possono essere visti come problemi di stima dello stato per un sistema dinamico discreto. Sebbene esistano molte tecniche specifiche per stimare lo stato del sistema a partire da un insieme di misurazioni, la maggior parte di queste non considera in maniera esplicita la natura rumorosa delle misurazioni. Tale rumore tipicamente è descritto da statistiche, il che porta a dover utilizzare metodi stocastici per affrontare il problema. In questo capitolo viene descritto il processo di stima stocastica dello stato.

Inizialmente vengono presentate le basi del filtraggio bayesiano, fornendo una breve derivazione matematica di come sia possibile effettuare una stima dello stato in maniera incrementale e on-line.

In seguito viene presentata una trattazione matematica dell'Algoritmo di Kalman per la localizzazione, estendendolo successivamente al caso di sistemi non lineari con l'Extended Kalman Filter.

Infine viene presentata l'implementazione in Matlab di un caso di localizzazione tramite EKF in presenza di Landmark, riportando i passi svolti ed i risultati ottenuti.

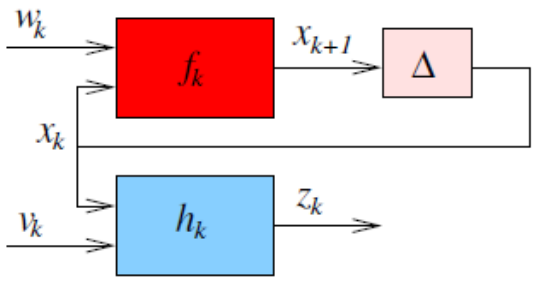

Figura 2.1: Schema di un sistema dinamico discreto

2.1 Cenni sul Filtraggio Bayesiano

Un filtro bayesiano è uno strumento matematico il cui scopo è stimare l'evoluzione dello stato del sistema, date le misurazioni disponibili. Questo strumento richiede:

- la conoscenza analitica della funzione di transizione f_t e la conoscenza stocastica del rumore dello stato ω_t
- la conoscenza analitica della funzione di uscita h_t e la conoscenza stocastica del rumore di osservazione v_t
- la realizzazione dell'uscita del sistema $z_{1:t}$ fino al tempo t.

Disponendo di questi dati, il filtro bayesiano è in grado di stimare la funzione densità di probabilità:

$$p(x_{0:t}|z_{1:t}) \tag{2.1.1}$$

dove quest'espressione rappresenta *la densità di probabilità dello stato x dall'istante zero all'istante t conscendo le misurazioni z dall'istante 1 all'istante t.*
Spesso si è interessati a stimare la densità marginale dello stato corrente, chiamata anche distribuzione filtrata:

$$p(x_t|z_{1:t}). \tag{2.1.2}$$

Nella maggior parte delle applicazioni pratiche, inoltre, viene aggiunta l'assunzione che il processo osservato sia Markoviano. In termini probabilistici questo equivale a dire che l'osservazione corrente è stocasticamente indipendente dalle osservazioni passate dato lo stato corrente, ossia:

$$p(z_t|x_t, z_{1:t-1}) = p(z_t|x_t). \tag{2.1.3}$$

Una volta formalizzato il problema della stima dello stato come un problema di filtraggio bayesiano, quello che resta da fare è trovare una formulazione matematica che ci permetta di risalire alla densità di probabilità (2.1.1) attraverso la conoscenza del sistema e delle osservazioni che abbiamo. Per prima cosa abbiamo bisogno di modellare sia l'evoluzione interna del sistema in oggetto, sia il modo in cui esso si presenta a noi attraverso le sue uscite osservabili. A questo scopo vengono introdotte due distribuzioni condizionate. La prima, $p(z_t|x_t)$, è il modello di osservazione e rappresenta la densità della misurazione z dato lo stato del sistema x. La seconda, $p(x_t|x_{t-1})$, è il modello d'evoluzione e rappresenta come il sistema evolve nel tempo. Mantenendo l'assunzione di catena Markoviana e utilizzando la regola di Bayes è possibile ottenere una densità di probabilità a posteriori dello stato in maniera incrementale. Dato che i dati arrivano progressivamente nel corso del tempo, si utilizza una formulazione ricorsiva della regola di Bayes che viene chiamata *Filtro di Bayes*. In questa formulazione la probabilità a posteriori dello stato viene progressivamente

affinata man mano che arrivano le misurazioni:

$$\begin{aligned} p(x_{0:t+1}|z_{1:t+1}) &= \frac{p(z_{t+1}|x_{0:t+1}, z_{1:t})p(x_{0:t+1}|z_{1:t})}{p(z_{t+1}|z_{1:t})} \\ &= \frac{p(z_{t+1}|x_{t+1})p(x_{0:t+1}|z_{1:t})}{p(z_{t+1}|z_{1:t})} \\ &= \frac{p(z_{t+1}|x_{t+1})p(x_{t+1}|x_t)}{p(z_{t+1}|z_{1:t})}p(x_{0:t}|z_{1:t}). \end{aligned} \tag{2.1.4}$$

Se si è interessati alla sola distribuzione filtrata (2.1.2) l'equazione diventa:

$$\begin{aligned} p(x_{t+1}|z_{1:t+1}) &= \frac{p(z_{t+1}|x_{t+1})p(x_{t+1}|z_{1:t})}{p(z_{t+1}|z_{1:t})} \\ &= \frac{p(z_{t+1}|x_{t+1})\int p(x_{t+1}|x_t)p(x_t|z_{1:t})dx_t}{p(z_{t+1}|z_{1:t})} \\ &= \frac{p(z_{t+1}|x_{t+1})\int p(x_{t+1}|x_t)p(x_t|z_{1:t})dx_t}{\int p(z_{t+1}|z_{1:t}, x_{t+1})p(x_{t+1}|z_{1:t})dx_{t+1}} \\ &= \eta p(z_{t+1}|x_{t+1})\int p(x_{t+1}|x_t)p(x_t|z_{1:t})dx_t \end{aligned} \tag{2.1.5}$$

dove η è un fattore di normalizzazione per assicurare che l'equazione (2.1.5) rappresenti una funzione densità di probabilità.

Solitamente la valutazione dell'equazione (2.1.5) avviene in due passaggi. Nella fase di predizione viene calcolato lo stato x_{t+1} a partire dallo stato x_t, attraverso l'*applicazione del modello di transizione*. Successivamente l'osservazione z_t viene incorporata all'interno della funzione di densità di probabilità precedentemente calcolata, attraverso la *fase di aggiornamento*. Entrambe le due fasi del processo di stima possono essere rintracciate nelle realizzazioni dei filtri descritti in seguito. Lo schema di filtraggio bayesiano descritto in questa sezione è esatto e può essere adoperato

in tutte le situazioni in cui è mantenuta l'assunzione Markoviana. Sfortunatamente nelle equazioni precedenti appaiono alcune integrazioni sullo spazio di stato che non permettono una realizzazione pratica nella maggior parte dei casi. Questo succede soprattutto nei problemi di SLAM, in cui lo spazio di stato comprende la posa del robot e la mappa, caratterizzando una dimensione molto elevata (dell'ordine delle centinaia o migliaia di elementi). Per questa ragione vengono utilizzati dei metodi approssimati.

2.2 Filtro di Kalman

Tra gli strumenti matematici che possono essere utilizzati per il filtraggio bayesiano, uno dei più famosi e più utilizzati è il *Filtro di Kalman*. Il filtro prende il nome da R.E.Kalman che nel 1960 pubblicò il suo famoso articolo sul filtraggio ricorsivo di sistemi lineari discreti. Il filtro di Kalman è composto sostanzialmente da un insieme di equazioni matematiche che implementano l'equazione (2.1.5) attraverso le due fasi di *predizione* e *aggiornamento*. Il filtro è ottimo, nel senso che minimizza la covarianza d'errore stimata, quando alcune assunzioni vengono soddisfatte. Per poter avere l'ottimalità, il sistema deve essere *lineare* e con *rumore Gaussiano*. Nel caso in esame ciò significa che lo stato è una variabile aleatoria gaussiana e il sistema può essere descritto da:

$$\begin{aligned} x_{t+1} &= F_t x_t + \omega_t \\ z_{t+1} &= H_t x_t + \upsilon_t \\ x_t &\sim N(\mu_t, \Sigma_t) \end{aligned} \tag{2.2.1}$$

in cui abbiamo che l'errore di sistema ω_t e l'errore di misura υ_t sono variabili

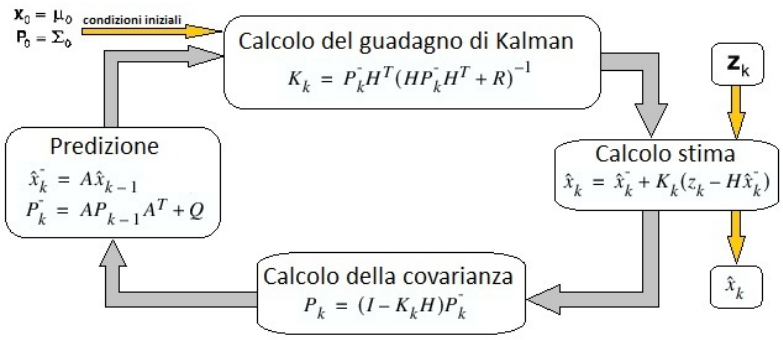

Figura 2.2: Schema del Filtro di Kalman

aleatorie distribuite normalmente e a media nulla

$$\begin{aligned} \omega_t &\sim N(0, \Sigma_{\omega_t}) \\ \upsilon_t &\sim N(0, \Sigma_{\upsilon_t}). \end{aligned} \tag{2.2.2}$$

Uno dei vantaggi principali del filtro di Kalman è che rappresenta una distribuzione in maniera compatta, facendo riferimento alla sola media e varianza. L'aggiornamento può essere portato avanti con la stessa complessità del calcolo di un'inversa di matrice (ossia in $O(n^3)$ dove n è la dimensione dello spazio di stato). L'algoritmo iterativo è mostrato in figura 2.2 e consta, come già accennato, di due passi fondamentali. Nel passo di **predizione** la funzione densità di probabilità dello stato evolve tramite la trasformazione gaussiana rappresentata dal sistema. In questo modo abbiamo che $\hat{x}_t \sim N(\hat{\mu}_t, \hat{\Sigma}_t)$ con

$$\begin{aligned} \hat{\mu}_t &= F_t \mu_t \\ \hat{\Sigma}_t &= F_t \Sigma_t F_t^T + \Sigma_{\omega_t}. \end{aligned} \tag{2.2.3}$$

In seguito, nel passo di **aggiornamento** viene calcolata la densità di probabilità $p(x_t|z_t)$. Per far ciò viene prima ricavata la densità congiunta $p(x_t, z_t)$

$$\begin{bmatrix} x_t \\ z_t \end{bmatrix} \sim N\left(\begin{bmatrix} \hat{\mu}_t \\ H_t \hat{\mu}_t \end{bmatrix}, \begin{bmatrix} \hat{\Sigma}_t & \hat{\Sigma}_t {H_t}^T \\ H_t \hat{\Sigma}_t & H_t \Sigma_t {H_t}^T + \Sigma_{vt} \end{bmatrix} \right). \tag{2.2.4}$$

In seguito si ottiene la densità condizionata $\mathrm{p}(x_t|z_t) = N(\mu_t, \Sigma_t)$ con

$$\begin{aligned} \mu_{t+1} &= \hat{\mu}_t + \hat{\Sigma}_t {H_t}^T ({H_t \Sigma_t {H_t}^T} + \Sigma_{vt})^{-1} (z_t - H_t \hat{\mu}_t) \\ \Sigma_{t+1} &= \hat{\Sigma}_t - \hat{\Sigma}_t {H_t}^T ({H_t \Sigma_t {H_t}^T} + \Sigma_{vt})^{-1} H_t \hat{\Sigma}_t. \end{aligned} \tag{2.2.5}$$

Definendo il *guadagno di Kalman* come:

$$K_t = \hat{\Sigma}_t {H_t}^T ({H_t \Sigma_t {H_t}^T} + \Sigma_{vt})^{-1} \tag{2.2.6}$$

possiamo riscrivere le equazioni del passo di aggiornamento come:

$$\begin{aligned} \mu_{t+1} &= \hat{\mu}_t + K_t (z_t - H_t \hat{\mu}_t) \\ \Sigma_{t+1} &= (I - K_t H_t) \hat{\Sigma}_t. \end{aligned} \tag{2.2.7}$$

Comprensivamente le equazioni (2.2.3), (2.2.6), (2.2.7) rappresentano il filtro di Kalman.

2.3 Filtro di Kalman Esteso (EKF)

Spesso accade che il sistema che si vuole stimare sia *non-lineare*, mentre il filtro di Kalman appena presentato funziona con sistemi di tipo lineare. Per poter utilizzare il filtro di Kalman in situazioni del genere è necessario che il sistema venga linearizzato. Un filtro che include questa linearizzazione è il Filtro di Kalman Esteso (Extended Kalman Filter). In modo analogo ad un'espansione di Taylor è possibile linearizzare le relazioni tra la media e la varianza attraverso derivate parziali delle funzioni di transizione e di osservazione. Per poter derivare le equazioni del filtro, bisogna considerare dapprima il modello del sistema, fornito da

$$\begin{aligned} x_{t+1} &= f_t(x_t) + \omega_t \\ z_{t+1} &= h_t(x_t) + \upsilon_t \end{aligned} \tag{2.3.1}$$

dove ω_t e υ_t rappresentano sempre il rumore del processo e dell'osservazione. Lo stato è invece rappresentato tramite media e covarianza. In particolare è possibile ottenere un'approssimazione dello stato attraverso la media della distribuzione, ricavabile dalla formula

$$\hat{\mu}_t = f_t \mu_t. \tag{2.3.2}$$

Tale approssimazione viene utilizzata per linearizzare il sistema. Le funzioni di transizione e le osservazioni vengono infatti linearizzate intorno a tale punto dello spazio di stato

$$\begin{aligned} \tilde{F}_t &= \nabla_x f_t|_{\hat{\mu}_t} \\ \tilde{H}_t &= \nabla_x h_t|_{\hat{\mu}_t}. \end{aligned} \tag{2.3.3}$$

Una volta linearizzato il sistema possiamo utilizzare le equazioni introdotte nella sezione precedente. L'algoritmo per il filtro di Kalman esteso può essere quindi espresso come

- Predizione:

$$\begin{aligned} \hat{\mu}_t &= f_t \mu_t \\ \hat{\Sigma}_t &= \tilde{F}_t \Sigma_t \tilde{F}_t^T + \Sigma_{\omega_t} \end{aligned} \tag{2.3.4}$$

- Aggiornamento:

$$\begin{aligned} \tilde{K}_t &= \hat{\Sigma}_t \tilde{H}_t^T (\tilde{H}_t \Sigma_t \tilde{H}_t^T + \Sigma_{v_t})^{-1} \\ \mu_{t+1} &\cong \hat{\mu}_t + \tilde{K}_t (z_t - h_t \hat{\mu}_t) \\ \Sigma_{t+1} &\cong (I - \tilde{K}_t \tilde{H}_t) \hat{\Sigma}_t. \end{aligned} \tag{2.3.5}$$

Il Filtro di Kalman esteso, sebbene sembri rilassare alcune assunzioni fatte dalla formulazione originale, soffre di alcune forti limitazioni a causa delle assunzioni di rumore gaussiano e linearizzabilità. Infatti, in molte applicazioni, assunzioni del genere non vengono rispettate essendo l'incertezza multi modale e di forma irregolare. Inoltre il processo di linearizzazione del sistema può portare ad errori sistematici. Infine non tutti i sistemi possono essere linearizzati, avendo ad esempio derivata prima nulla, non permettendo quindi l'utilizzo dell'EKF. In queste situazioni è stata proposta un'ulteriore estensione al filtro di Kalman basata sulla Unscented Trasformation. Tale approccio, chiamato *Unscented Kalman Filter*, è stato proposto da S.J.Julier e J.K.Uhlmann con una famosa pubblicazione del 1996, e risolve i problemi di linearizzazione, mantenendo comunque l'ipotesi di unimodalità dell'incertezza. L'idea di base è quella di individuare dei punti, chiamati σ-point, posti lungo l'ellissoide di covarianza, e ricalcolare la covarianza sulla base del motion model applicato ai σ-point stessi.

2.4 Localizzazione tramite EKF in Matlab

Dopo aver affrontato una trattazione teorica del problema di localizzazione tramite Filtro di Kalman esteso, possiamo dunque passare ai risultati ottenuti implementando questo stesso algoritmo in ambiente Matlab.

In prima istanza occorre definire l'ambiente utilizzato per la simulazione. Si considera il caso in cui il robot segue un percorso all'interno di una stanza di dimensioni 4m x 4m, in cui sono stati disposti uniformemente 4 landmark statici, ossia degli oggetti fissi che il robot riesce a riconoscere sempre, ma di cui misura la posizione relativa con un certo errore di tipo aleatorio, distribuito normalmente e a media nulla. Misurando la loro posizione, espressa tramite distanza RHO e angolo PHI (come mostrato in

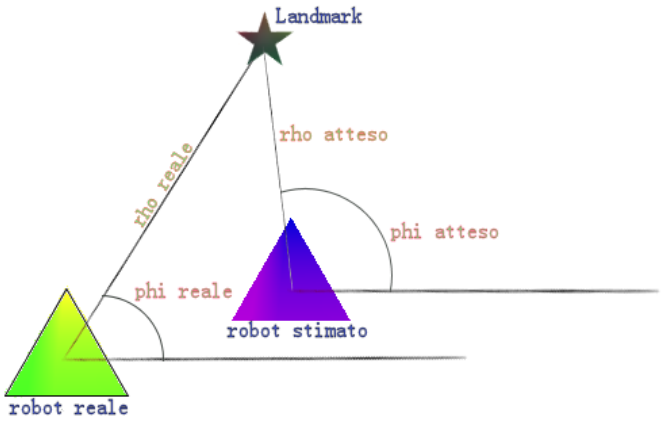

Figura 2.3: Rappresentazione istantanea di localizzazione tramite EKF

figura 2.3), il robot è in grado di applicare il filtro di Kalman, correggendo la stima della propria posizione. Possiamo infatti immaginare il robot che nel compiere un determinato percorso, non riesce a localizzare la sua effettiva posizione a causa degli errori sulle letture odometriche. Quindi, portandolo lungo il percorso verde in figura 2.4, se si utilizzasse la sola stima odometrica (che come abbiamo visto si basa sulla cinematica del robot ricavata dal numero di passi encoder compiuti dalle singole ruote nell'intervallo di tempo ΔT e che aggiorna lo stato del robot tramite la relazione (1.2.3)), essendo essa afflitta da errore, produrrebbe una stima della posa del robot del tipo mostrato dalla linea blu in figura 2.4. Impostando quindi la variabile *Nstep*, definiamo il numero di passi tra una correzione tramite filtro di Kalman e la successiva. Nel caso in cui non operiamo alcuna correzione, bisogna procedere con la stima della posizione successiva tramite odometria, aggiornando la matrice di covarianza. Per fare questo occorre calcolare:

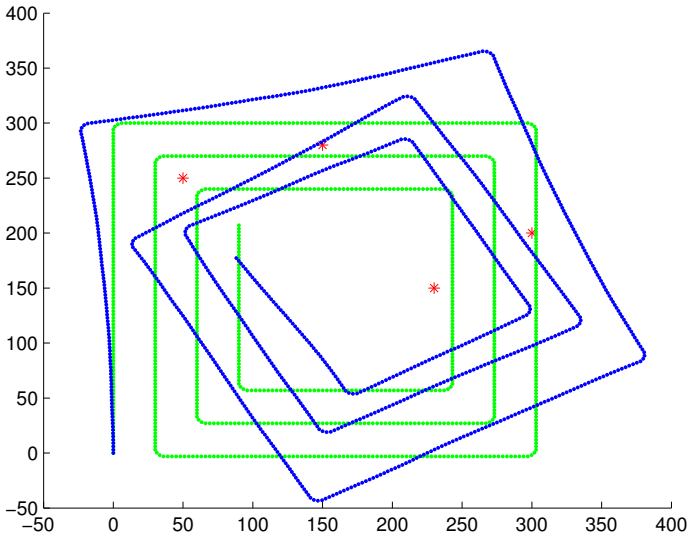

Figura 2.4: Rappresentazione della stima della posa del robot tramite sola odometria

- la matrice Jacobiana F della dinamica rispetto allo stato

$$F = \begin{bmatrix} \frac{\partial x_{t+1}}{\partial x_t} & \frac{\partial x_{t+1}}{\partial y_t} & \frac{\partial x_{t+1}}{\partial \theta_t} \\ \frac{\partial y_{t+1}}{\partial x_t} & \frac{\partial y_{t+1}}{\partial y_t} & \frac{\partial y_{t+1}}{\partial \theta_t} \\ \frac{\partial \theta_{t+1}}{\partial x_t} & \frac{\partial \theta_{t+1}}{\partial y_t} & \frac{\partial \theta_{t+1}}{\partial \theta_t} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \frac{-(u_{R,t}+u_{L,t})}{2} sin\theta_t \\ 0 & 1 & \frac{(u_{R,t}+u_{L,t})}{2} cos\theta_t \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4.1}$$

- la matrice Jacobiana W della dinamica rispetto al disturbo

$$W = \begin{bmatrix} \frac{\partial x_{t+1}}{\partial u_{R,t}} & \frac{\partial x_{t+1}}{\partial u_{L,t}} \\ \frac{\partial y_{t+1}}{\partial u_{R,t}} & \frac{\partial y_{t+1}}{\partial u_{L,t}} \\ \frac{\partial \theta_{t+1}}{\partial u_{R,t}} & \frac{\partial \theta_{t+1}}{\partial u_{L,t}} \end{bmatrix} = \begin{bmatrix} \frac{cos\theta_t}{2} & \frac{cos\theta_t}{2} \\ \frac{sin\theta_t}{2} & \frac{sin\theta_t}{2} \\ \frac{1}{d} & -\frac{1}{d} \end{bmatrix} \tag{2.4.2}$$

- la matrice Q di covarianza dell'errore odometrico

$$Q = \begin{bmatrix} K_R|u_{R,t}| & 0 \\ 0 & K_L|u_{L,t}| \end{bmatrix} \tag{2.4.3}$$

dove u_R e u_L rappresentano ancora gli spostamenti della ruota destra e sinistra rispettivamente in un certo intervallo di simulazione, mentre K_R e K_L rappresentano gli errori odometrici sulle due ruote, che sono proporzionali alla lunghezza del percorso compiuto nell'intervallo ΔT.

Possiamo quindi aggiornare la matrice di covarianza per il passo $t+1$ tramite la relazione

$$P_{t+1} = FP_tF^T + WQW^T. \tag{2.4.4}$$

Nel caso in cui si operi il passo di correzione, utilizzeremo le variabili *RhoReale* e *PhiReale*, che esprimono la distanza e l'angolazione del robot dal Landmark, misurata dal robot stesso tramite sensoristica (quindi contenente un certo errore) e le variabili *RhoAtteso* e *PhiAtteso* che invece rappresentano la distanza e l'angolazione che dividono la posizione del robot stimata tramite odometria e quella nota del Landmark (come mostrato in figura 2.3). Definiamo quindi la variabile Innovation, che è data dalla differenza tra Rho e Phi attesi e Rho e Phi Reali. Per procedere con la linearizzazione ci è utile definire inoltre la matrice Jacobiana H di dimensione n x 3, dove n è il numero di Landmark presenti.

Esplicitando le quantità

$$\begin{aligned} RhoAtteso_{i,t} &= \sqrt{(x_{Landmark_i} - x_{odo,t})^2 - (y_{Landmark_i} - y_{odo,t})^2} \\ PhiAtteso_{i,t} &= atan2(y_{Landmark_i} - y_{odo,t}, x_{Landmark_i} - x_{odo,t}) \end{aligned} \tag{2.4.5}$$

dove x_{odo}, y_{odo} e θ_{odo} rappresentano le coordinate del robot stimate tramite odometria, mentre $x_{Landmark_i}$ e $y_{Landmark_i}$ le coordinate dell'i-esimo Landmark note grazie ad una mappa costruita off-line e passata al robot che deve affrontare il problema di localizzazione.

Abbiamo quindi che la jacobiana H assume la forma del tipo:

$$H = \begin{bmatrix} \frac{\partial RhoAtteso_{1,t}}{\partial x_{odo,t}} & \frac{\partial RhoAtteso_{1,t}}{\partial y_{odo,t}} & \frac{\partial RhoAtteso_{1,t}}{\partial \theta_{odo,t}} \\ \frac{\partial PhiAtteso_{1,t}}{\partial x_{odo,t}} & \frac{\partial PhiAtteso_{1,t}}{\partial y_{odo,t}} & \frac{\partial PhiAtteso_{1,t}}{\partial \theta_{odo,t}} \\ & ..... & \\ & ..... & \\ \frac{\partial RhoAtteso_{n,t}}{\partial x_{odo,t}} & \frac{\partial RhoAtteso_{n,t}}{\partial y_{odo,t}} & \frac{\partial RhoAtteso_{n,t}}{\partial \theta_{odo,t}} \\ \frac{\partial PhiAtteso_{n,t}}{\partial x_{odo,t}} & \frac{\partial PhiAtteso_{n,t}}{\partial y_{odo,t}} & \frac{\partial PhiAtteso_{n,t}}{\partial \theta_{odo,t}} \end{bmatrix}. \tag{2.4.6}$$

ed introducendo per comodità di notazione

$$q_i = (x_{Landmark_i} - x_{odo,t})^2 + (y_{Landmark_i} - y_{odo,t})^2 \tag{2.4.7}$$

possiamo dunque riscrivere H come

$$H = \begin{bmatrix} -\frac{x_{Landmark_1} - x_{odo,t}}{\sqrt{q_1}} & -\frac{y_{Landmark_1} - y_{odo,t}}{\sqrt{q_1}} & 0 \\ \frac{y_{Landmark_1} - y_{odo,t}}{q_1} & -\frac{x_{Landmark_1} - x_{odo,t}}{q_1} & -1 \\ & \cdot \quad \cdot \quad \cdot & \\ & \cdot \quad \cdot \quad \cdot & \\ -\frac{x_{Landmark_n} - x_{odo,t}}{\sqrt{q_n}} & -\frac{y_{Landmark_n} - y_{odo,t}}{\sqrt{q_n}} & 0 \\ \frac{y_{Landmark_n} - y_{odo,t}}{q_n} & -\frac{x_{Landmark_n} - x_{odo,t}}{q_n} & -1 \end{bmatrix}. \tag{2.4.8}$$

Abbiamo dunque tutti gli elementi per procedere con il filtraggio di Kalman esteso. Calcoliamo dunque il guadagno di Kalman come

$$K = P_{t+1} H^T (H P_{t+1} H^T + R)^{-1} \tag{2.4.9}$$

dove R è la matrice di covarianza dell'errore sensoriale che assume la forma

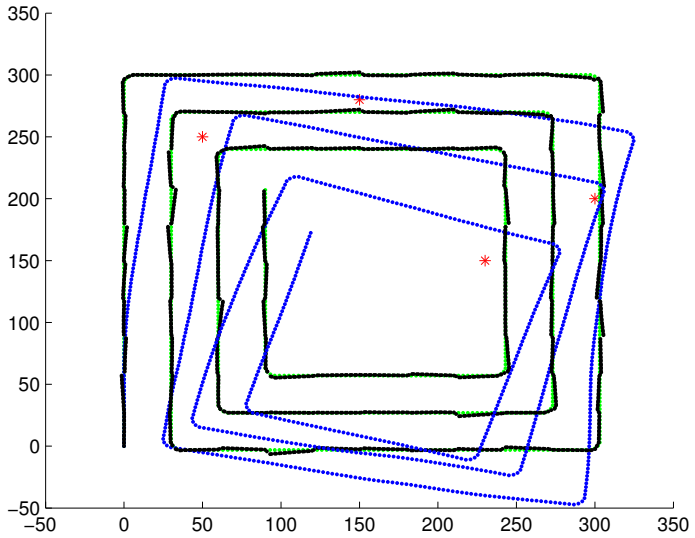

Figura 2.5: Rappresentazione della stima della posa del robot applicando una correzione tramite EKF

$$R = \begin{bmatrix} {\sigma_{RhoReale_1}}^2 & 0 & 0 & 0 \\ 0 & {\sigma_{PhiReale_1}}^2 & 0 & 0 \\ & & \cdot & \\ & & \cdot & \\ 0 & 0 & {\sigma_{RhoReale_n}}^2 & 0 \\ 0 & 0 & 0 & {\sigma_{PhiReale_n}}^2 \end{bmatrix}. \tag{2.4.10}$$

Aggiorniamo infine la media μ_t e la covarianza Σ_t della stima sulla posa del robot tramite l'*Innovation*

$$\begin{aligned} \mu_{t+1} &= \mu_t + K Innovation \\ \Sigma_{t+1} &= (I - KH)\Sigma_t \end{aligned} \tag{2.4.11}$$

concludendo così il ciclo di correzione tramite EKF.

Operando questo filtraggio, abbiamo che il robot stima la sua posizione lungo il percorso compiuto come mostrato dalla linea nera in figura 2.5. Viene dunque corretto l'errore odometrico, ottenendo una buona localizzazione tramite EKF.

Capitolo 3

SLAM

Il problema della Localizzazione e Mappatura Simultanea (Simultaneous Localization And Mapping) chiede se sia possibile per un robot mobile posto in un ambiente ignoto effettuare la costruzione incrementale di una mappa dell'ambiente in cui si sta muovendo e conoscere allo stesso tempo la sua posizione all'interno di quest'ambiente. L'importanza di una soluzione al problema dello SLAM per la comunità della robotica mobile è data dal fatto che questa fornisce i mezzi per la realizzazione di un robot veramente autonomo, a cui non deve essere passata alcuna mappa precompilata o debbano essere forniti mezzi esterni per la sua localizzazione.

Lo SLAM come problema teorico è stato formulato e risolto in diverse forme, è stato implementato in diversi tipi di ambienti, differenziando studi indoor e outdoor, sistemi subacquei e nell'aria. A livello teorico e concettuale lo SLAM può essere dunque considerato un problema risolto. Tuttavia, le questioni sostanziali rimangono la realizzazione pratica di soluzioni SLAM più generali, e in particolare la costruzione e l'utilizzo di mappe particolarmente ricche all'interno di un algoritmo di SLAM.

3.1 Cenni storici sullo SLAM

Storicamente, la nascita del problema probabilistico riguardante lo SLAM può essere collocata nel 1986 durante la Robotics and Automation Conference tenutasi a San Francisco, California. In quel periodo, infatti, metodi probabilistici cominciavano a essere appena introdotti nella robotica e nell'intelligenza artificiale. Un certo numero di scienziati aveva tentato di applicare metodi di stima teorica a problemi di mappatura e localizzazione. Alcuni nomi da ricordare sono sicuramente Peter Cheeseman, Jim Crowley, e Hugh Durrant-Whyte.

Nel corso della conferenza numerosi tovaglioli e intere tovaglie di carta furono riempiti di lunghe discussioni sulla mappatura coerente. Raja Chatila, Oliver Faugeras, Randal Smith e altri scienziati apportarono importanti contributi alle numerose conversazioni che si tennero in quella conferenza. Il risultato di queste conversazioni fu il riconoscimento che una mappatura probabilistica coerente è un problema basilare nel campo della robotica e che c'erano fondamentali questioni concettuali e computazionali che dovevano essere affrontate. Nel corso degli anni successivi furono prodotti un gran numero di importanti documenti. Il lavoro di Smith e Cheesman [5] e Durrant-Whyte [6] ha istituito una base per descrivere le relazioni che intercorrono tra i diversi *punti di riferimento* (**landmark**) e forniva validi strumenti per manipolare l'incertezza geometrica. Un elemento chiave di questo lavoro fu quello di mostrare che ci deve essere un alto grado di correlazione tra le stime della posizione di diversi landmark in una mappa e che, anzi, queste correlazioni devono crescere al crescere del numero di osservazioni successive.

Allo stesso tempo Ayache e Faugeras [7] avviarono i primi studi sulla navigazione visiva, mentre Crowley [8], Chatila e Laumond [9] iniziarono a lavorare sulla navigazione

basata sui sonar, utilizzando algoritmi del tipo Kalman Filter. Questi due filoni di ricerca avevano molto in comune e portarono alla pubblicazione di Smith [10], che è uno dei capisaldi dell'argomento. Questo lavoro ha dimostrato che per un robot mobile che si muove in un ambiente ignoto eseguendo osservazioni relative di landmark, le stime di questi landmark sono tutte necessariamente correlate tra loro a causa del comune errore di stima di posizione del veicolo. L'implicazione di questa scoperta è stata profonda: una soluzione completa coerente per un problema di localizzazione combinata a mappatura richiederebbe uno stato misto composto della posa del veicolo e dalla posizione di ognuno dei landmark, da essere aggiornata dopo ogni osservazione. A sua volta, ciò richiederebbe per lo stimatore l'utilizzo di un vettore di stato enorme (dell'ordine del numero di landmark considerati nella mappa) con una complessità computazionale dell'ordine del quadrato del numero di landmark considerati. Tuttavia questo lavoro non teneva conto delle proprietà di convergenza della mappa o del suo comportamento stazionario. Infatti, era opinione diffusa che gli errori di stima della mappa non convergessero e che il sistema avrebbe avuto un comportamento di tipo casuale con una crescita illimitata di errore. Così, data la complessità computazionale del problema di mappatura e senza le conoscenze sul comportamento di convergenza della mappa stessa, i ricercatori hanno invece focalizzato l'attenzione su una serie di approssimazioni al problema di mappatura coerente, che assumevano o addirittura imponevano che le correlazioni tra punti di riferimento fossero limitate o del tutto inesistenti, in modo da ridurre il filtro completo ad un sistema con una serie di landmark disaccoppiati rispetto ai filtri del veicolo. Anche per questi motivi, il lavoro teorico sulla localizzazione e mappatura combinate giunse ad un arresto temporaneo, che portò a numerosi lavori focalizzati invece sulla mappatura e sulla localizzazione come problemi separati.

La svolta concettuale si è avuta con la realizzazione che il problema di mapping e localizzazione combinati, formulato come un problema di stima unico, aveva in realtà proprietà di convergenza. Soprattutto fu riconosciuto che le correlazioni tra i landmark, che molti ricercatori avevano tentato di ridurre al minimo, erano in realtà la parte critica del problema e che al contrario, maggiore fosse stata questa correlazione, migliore sarebbe stata la soluzione ottenuta.

La struttura del problema dello SLAM, i risultati di convergenza e la coniatura della sigla SLAM furono presentati per la prima volta all'International Symposium on Robotics Research nel 1995 in un articolo di robotica mobile [11]. La fondamentale teoria sulla convergenza e molti dei primi risultati furono sviluppati da M.Csorba [12],[13]. Diversi gruppi già al lavoro sulla mappatura e sulla localizzazione, in particolare presso il Massachusetts Institute of Technology [14], Saragozza [15],[16], l'ACFR a Sydney [17],[18] e molti altri iniziarono dunque a lavorare in modo approfondito sullo SLAM, concentrandosi sulla sua implementazione in ambienti chiusi, all'aperto o in ambito sottomarino. Il lavoro si incentrò per lo più sul miglioramento dell'efficienza computazionale e sul problema dell'associazione dati. Il Simposio Internazionale sulla Ricerca Robotica del 1999 (ISRR'99) [19] è stato un importante punto d'incontro in cui si tenne la prima sessione dedicata allo SLAM e dove fu raggiunto un alto grado di intercorrelazione tra i metodi di SLAM basati sul Kalman-filter e la localizzazione e mappatura probabilistica con i metodi introdotti da Thrun [20].

Il Workshop della IEEE International Conference on Robotics and Automation (ICRA 2000) sullo SLAM attirò 15 ricercatori e fu incentrato su questioni come la complessità algoritmica, il Data Association e le diverse sfide di attuazione. Il workshop seguente sullo Slam nel 2002, attirò ben 150 ricercatori con una vasta gamma di proposte e applicazioni. Sempre nel 2002, la SLAM summer school organizzata da Henrik

Christiansen al KTH di Stoccolma attirò tutti i ricercatori chiave che, insieme a 50 studenti provenienti da tutto il mondo diedero un grande contributo al miglioramento e all'espansione dell'algoritmo di SLAM, e costituì un enorme successo internazionale. Negli ultimi anni l'interesse per lo SLAM è cresciuto esponenzialmente, e numerosi workshop continuano a essere tenuti sia dall'ICRA sia dall'IROS (International Conference on Intelligent Robots and Systems) per affinare sempre più questa interessante via della robotica mobile e i suoi innumerevoli sbocchi pratici.

3.2 Formulazione del problema di SLAM

Il principale vantaggio dello SLAM è che elimina la necessità della conoscenza topologica a priori di un certo scenario. Se il robot si muove in un ambiente prendendo misure di posizione relativa tra Landmark, si è osservato che le diverse stime di posizione sono necessariamente correlate l'una all'altra a causa degli errori sulla stima di posizione del robot stesso.

L'implicazione che ne consegue è molto profonda, in quanto una soluzione consistente al problema della localizzazione del robot e della produzione della mappa richiedono uno stato congiunto composto dalla dinamica del robot e dalla posizione di ogni singolo Landmark. Ciò richiede che lo stimatore implementi un vettore di stato di dimensioni ragguardevoli con un costo computazionale che va come il quadrato del numero del Landmark. Nella sua forma probabilistica il problema dello SLAM richiede che venga computata ad ogni nuova misura

$$p(z_k|x_k, m) \tag{3.2.1}$$

la densità di probabilità congiunta

$$p(x_k, m|Z_{0:k}, U_{0:k}, x_0) \tag{3.2.2}$$

dove

- x_k: è il vettore di stato che descrive la posizione e l'orientazione del robot
- m_i: è il vettore che descrive la posizione dell'i-esimo Landmark supposta tempo invariante
- $z_{i,k}$: è l'osservazione dell'i-esimo Landmark al tempo k
- $X_{0:k} = |x_0, ..., x_k| = |X_{0:k-1}, x_k|$: le posizioni del robot fino all'istante k
- $U_{0:k} = |u_0, ..., u_k| = |U_{0:k-1}, u_k|$: gli input del sistema fino all'istante k
- $m = |m_1, ..., m_k|$: l'insieme di tutti i landmark
- $Z_{0:k} = |z_0, ..., z_k| = |Z_{0:k-1}, z_k|$: l'insieme di tutte le osservazioni dei Landmark.

Ovviamente una soluzione ricorsiva è desiderabile e sarà fornita dall'applicazione di un filtro di Kalman esteso. Avendo già affrontato in questa tesi una trattazione matematica dell'EKF, possiamo applicare questo metodo di filtraggio per presentare un algoritmo di SLAM sviluppato in Matlab basato su di esso.

3.3 Algoritimo di SLAM basato su EKF con corrispondenze note in Matlab

L'ipotesi di corrispondenze note realizza il fatto che il robot riesca sempre a riconoscere i Landmark visti, escludendo la possibilità di confondere un Landmark già visto come uno nuovo. Questa ipotesi è molto forte, poiché risolve uno dei problemi più importanti che si possono presentare in una mappatura, ossia quello di non accorgersi che un punto che stiamo vedendo, in realtà è già stato stimato e catalogato e quindi

la parte di mappa che ci si accinge a realizzare, in realtà è già stata costruita in precedenza. Questo porta chiaramente ad un'alterazione della mappa, che si allontana dunque dalla realtà che si sta tentando di rappresentare. Nel paragrafo successivo questa ipotesi verrà quindi rimossa, affrontando il caso più generale.

3.3.1 Ipotesi di visibilità di tutti i Landmark

Per introdurre l'adattamento dell'EKF ad un problema di SLAM, partiamo dall'ipotesi che il robot visto nel capitolo sulla localizzazione sia in grado di vedere sempre i 4 Landmark presenti nell'ambiente in cui era stato introdotto. Questa ipotesi è poco reale, vista la limitatezza degli strumenti sensoristici reali, ma verrà rimossa nel sottoparagrafo successivo, rendendo la trattazione più vicina alla realtà. Partiamo dunque col definire il nuovo vettore di stato

$$X_k = [x(k), m(k)]^T = [x(k), m_1(k), ..., m_n(k)]^T. \tag{3.3.1}$$

Per lo stato del robot possiamo riprendere il modello introdotto nella presentazione del filtro di Kalman (vedi 2.3.1), mentre il modello dei Landmark è di tipo stazionario. Quindi il modello del vettore di stato avrà dimensione 3+2n e forma del tipo

$$X_{k+1} = X_k + \begin{bmatrix} \frac{U_{R_k}+U_{L_k}}{2} cos\theta_k \\ \frac{U_{R_k}+U_{L_k}}{2} sin\theta_k \\ \frac{U_{R_k}-U_{L_k}}{d} \\ 0 \\ . \\ . \\ . \\ 0 \end{bmatrix}. \tag{3.3.2}$$

Possiamo dunque ridefinire le matrici:

- Jacobiana F della dinamica rispetto allo stato di dimensione (3+2n)x(3+2n)

$$F = \begin{bmatrix} \frac{\partial x_{t+1}}{\partial x_t} & \frac{\partial x_{t+1}}{\partial y_t} & \frac{\partial x_{t+1}}{\partial \theta_t} & \frac{\partial x_{t+1}}{\partial x_{L_1}} & \dots & \frac{\partial x_{t+1}}{\partial y_{L_n}} \\ \frac{\partial y_{t+1}}{\partial x_t} & \frac{\partial y_{t+1}}{\partial y_t} & \frac{\partial y_{t+1}}{\partial \theta_t} & \frac{\partial y_{t+1}}{\partial x_{L_1}} & \dots & \frac{\partial y_{t+1}}{\partial y_{L_n}} \\ \frac{\partial \theta_{t+1}}{\partial x_t} & \frac{\partial \theta_{t+1}}{\partial y_t} & \frac{\partial \theta_{t+1}}{\partial \theta_t} & \frac{\partial \theta_{t+1}}{\partial x_{L_1}} & \dots & \frac{\partial \theta_{t+1}}{\partial y_{L_n}} \\ \frac{\partial x_{L_1}}{\partial x_t} & \frac{\partial x_{L_1}}{\partial y_t} & \frac{\partial x_{L_1}}{\partial \theta_t} & \frac{\partial x_{L_1}}{\partial x_{L_1}} & \dots & \frac{\partial x_{L_1}}{\partial y_{L_n}} \\ & & . \quad . \quad . & & & \\ \frac{\partial y_{L_n}}{\partial x_t} & \frac{\partial y_{L_n}}{\partial y_t} & \frac{\partial y_{L_n}}{\partial \theta_t} & \frac{\partial y_{L_n}}{\partial x_{L_1}} & \dots & \frac{\partial y_{L_n}}{\partial y_{L_n}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \frac{-(u_{R,t}+u_{L,t})}{2} sin\theta_t & 0 & \cdots & 0 \\ 0 & 1 & \frac{(u_{R,t}+u_{L,t})}{2} cos\theta_t & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ & & & & \ddots & \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \tag{3.3.3}$$

- Jacobiana W della dinamica rispetto al disturbo di dimensione (3+2n)x(2)

$$W = \begin{bmatrix} \frac{\partial x_{t+1}}{\partial u_{R,t}} & \frac{\partial x_{t+1}}{\partial u_{L,t}} \\ \frac{\partial y_{t+1}}{\partial u_{R,t}} & \frac{\partial y_{t+1}}{\partial u_{L,t}} \\ \frac{\partial \theta_{t+1}}{\partial u_{R,t}} & \frac{\partial \theta_{t+1}}{\partial u_{L,t}} \\ \frac{\partial x_{L_1}}{\partial u_{R,t}} & \frac{\partial x_{L_1}}{\partial u_{L,t}} \\ . \quad . \quad . & \\ \frac{\partial x_{L_n}}{\partial u_{R,t}} & \frac{\partial x_{L_n}}{\partial u_{L,t}} \end{bmatrix} = \begin{bmatrix} \frac{cos\theta_t}{2} & \frac{cos\theta_t}{2} \\ \frac{sin\theta_t}{2} & \frac{sin\theta_t}{2} \\ \frac{1}{d} & -\frac{1}{d} \\ 0 & 0 \\ . \quad . \quad . & \\ 0 & 0 \end{bmatrix} \tag{3.3.4}$$

Mentre la matrice Q di covarianza dell'errore odometrico resta nella stessa forma della (2.4.3). Nel caso in cui, dopo aver impostato la variabile NStep non si dovesse procedere al passo di correzione, siamo dunque in grado di aggiornare la matrice di covarianza per il passo t+1 tramite la relazione:

$$P_{t+1} = FP_tF^T + WQW^T. \tag{3.3.5}$$

Qualora si dovesse procedere con il passo di correzione, bisogna ricordare che le posizioni dei Landmark non sono più fornite a priori, ma vengono anch'esse stimate volta

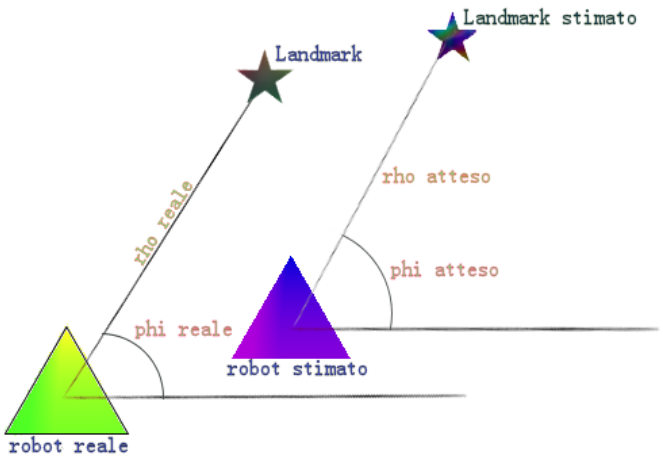

Figura 3.1: Rappresentazione istantanea di un problema di SLAM

per volta così come la posa del robot. Utilizzeremo dunque le variabili *RhoReale* e *PhiReale*, che esprimono ancora la distanza e l'angolazione del robot dal Landmark misurata dal robot stesso tramite sensoristica (quindi contenente un certo errore), mentre le variabili *RhoAtteso* e *PhiAtteso* rappresenteranno la distanza e l'angolazione che dividono la posizione del robot stimata tramite odometria e la posizione del Landmark, anch'essa stimata e non più effettiva, come mostrato in figura 3.1. Definiamo ancora una volta la variabile Innovation come differenza tra Rho e Phi attesi e Rho e Phi Reali. Procediamo dunque con la linearizzazione tramite la matrice Jacobiana H, che sarà sempre definita come in (2.4.6). Tuttavia dobbiamo ridefinire le quantità RhoAtteso e PhiAtteso come

$$\begin{aligned} RhoAtteso_{i,t} &= \sqrt{(x_{StimaLandmark_i,t} - \hat{x}_{robot,t})^2 - (y_{StimaLandmark_i,t} - \hat{y}_{robot,t})^2} \\ PhiAtteso_{i,t} &= atan2(y_{StimaLandmark_i,t} - \hat{y}_{robot,t}, x_{StimaLandmark_i,t} - \hat{x}_{robot,t}) \end{aligned} \tag{3.3.6}$$

ed introducendo per comodità di notazione

$$q_i = (x_{StimaLandmark_i,t} - \hat{x}_{robot,t})^2 + (y_{StimaLandmark_i,t} - \hat{y}_{robot,t})^2 \tag{3.3.7}$$

possiamo dunque riscrivere la matrice Jacobiana H come

$$H = \begin{bmatrix} -\frac{x_{SL_1,t}-\hat{x}_{R,t}}{\sqrt{q_1}} & -\frac{y_{SL_1,t}-\hat{y}_{R,t}}{\sqrt{q_1}} & 0 & \frac{x_{SL_1,t}-\hat{x}_{R,t}}{\sqrt{q_1}} & -\frac{y_{SL_1,t}-\hat{y}_{R,t}}{\sqrt{q_1}} & \cdots & 0 & 0 \\ \frac{y_{SL_1,t}-\hat{y}_{R,t}}{q_1} & -\frac{x_{SL_1,t}-\hat{x}_{R,t}}{q_1} & -1 & -\frac{y_{SL_1,t}-\hat{y}_{R,t}}{q_1} & -\frac{x_{SL_1,t}-\hat{x}_{R,t}}{q_1} & \cdots & 0 & 0 \\ & & & . & . & . & & \\ -\frac{x_{SL_n,t}-\hat{x}_{R,t}}{\sqrt{q_n}} & -\frac{y_{SL_n,t}-\hat{y}_{R,t}}{\sqrt{q_n}} & 0 & 0 & 0 & \cdots & \frac{x_{SL_n,t}-\hat{x}_{R,t}}{\sqrt{q_n}} & -\frac{y_{SL_n,t}-\hat{y}_{R,t}}{\sqrt{q_n}} \\ \frac{y_{SL_n,t}-\hat{y}_{R,t}}{q_n} & -\frac{x_{SL_n,t}-\hat{x}_{R,t}}{q_n} & -1 & 0 & 0 & \cdots & -\frac{y_{SL_n,t}-\hat{y}_{R,t}}{q_n} & -\frac{x_{SL_n,t}-\hat{x}_{R,t}}{q_n} \end{bmatrix} \tag{3.3.8}$$

Calcoliamo il guadagno di Kalman come mostrato in (2.4.9)

$$K = P_{t+1}H^T(HP_{t+1}H^T + R)^{-1} \tag{3.3.9}$$

notando che in questo caso avrà dimensione 2n x 2n, e aggiorniamo tramite Innovation oltre alla media μ_t e la covarianza Σ_t della stima sulla posa del robot, anche le pose stimate dei Landmark, essendo anch'esse parte dello stato del sistema in analisi:

$$X_{t+1} = \begin{bmatrix} \mu_{t+1} \\ m_{1,t+1} \\ \vdots \\ m_{n,t+1} \end{bmatrix} = X_t + KInnovation$$

$$\Sigma_{t+1} = (I - KH)\Sigma_t \tag{3.3.10}$$

concludendo così il ciclo di correzione tramite EKF.

Operando questo filtraggio, abbiamo che il robot stima la sua posizione lungo il percorso compiuto come mostrato dalla linea nera in figura 3.2. Viene dunque corretto

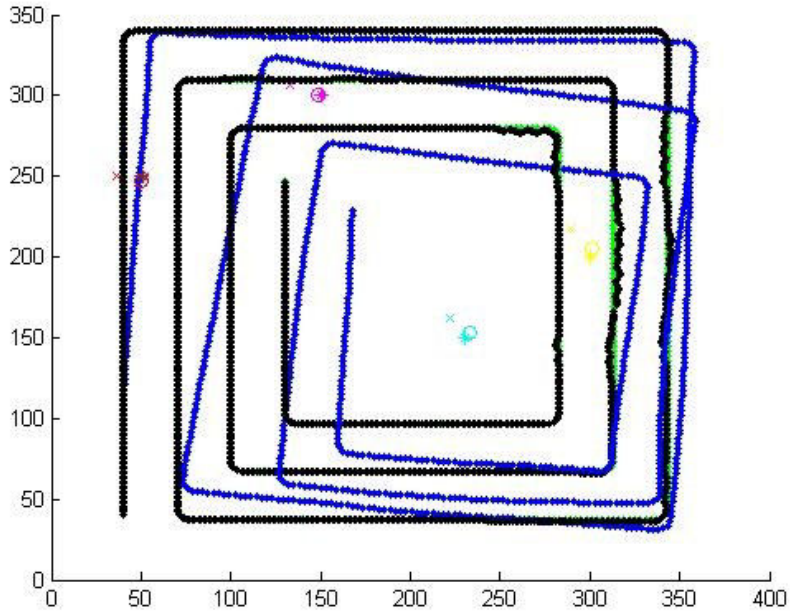

Figura 3.2: Rappresentazione della stima della posa del robot e dei Landmark applicando l'algoritmo di SLAM tramite EKF

l'errore odometrico, ottenendo un buon lavoro di localizzazione. Vengono inoltre riportate le stime sui 4 Landmark, rappresentati in figura con il simbolo '*', presenti nell'ambiente, graficando le loro stime al primo istante di elaborazionedove tramite il simbolo 'x' e le stime all'ultimo istante tramite il simobolo 'o'. Si vede dunque che la stima, con l'applicazione dell'algortimo di SLAM, si avvicina alla posizione effettiva del Landmark, ottenendo dunque un buon lavoro di mappatura.

3.3.2 Introduzione limiti sensoriali

Come annunciato, l'ipotesi di continua visibilità di tutti i Landmark è molto lontana dal caso reale, in cui i sensori hanno un loro range di sensibilità. Sono dunque state introdotte delle limitazioni sulla distanza massima a cui un Landmark in un certo istante di tempo può trovarsi per essere visibile dal robot, individuando quindi una

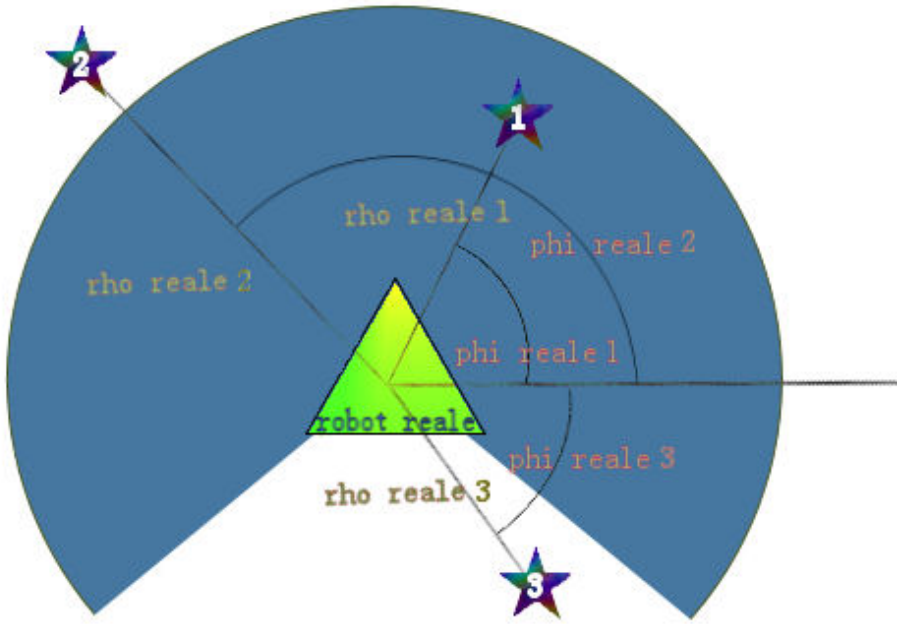

Figura 3.3: Rappresentazione del settore circolare di visibilità del robot di raggio Rho Massimo e ampiezza Phi Massimo. Viene dunque visto il solo Landmark 1.

circonferenza di visibilità del robot stesso. In seguito sono state introdotte anche delle limitazioni sull'angolo di visibilità, ossia di quanto la posizione di un Landmark può discostarsi dal θ lungo cui il robot si sta muovendo per essere ancora visibile, restringendo la circonferenza ad un settore circolare. Questi controlli sono effettuati verificando ad ogni iterazione se i RhoReali e i PhiReali calcolati per ogni Landmark (che ricordiamo rappresentare la distanza e l'angolazione del robot dal Landmark misurata dal robot stesso tramite sensoristica) risultano inferiori a dei fissati **RhoMassimo** e **PhiMassimo** (come mostrato in figura 3.3).

E' chiaro che una modifica di questo tipo costringe a valutare il numero di Landmark visti in un certo istante di tempo e su cui basare la correzione tramite EKF. Sono state quindi sviluppate quattro diverse funzioni che effettuano un filtraggio basandosi su 1,2,3 o 4 Landmark, per coprire tutti i possibili casi ottenibili nell'ambiente in cui

stiamo simulando.

Valutiamo il caso in cui vediamo un solo Landmark.

E' stata dunque sviluppata una funzione $SLAM1$, che prende in ingresso oltre a tutti gli elementi necessari per la stima odometrica anche le variabili RhoReale e PhiReale dell'unico Landmark visibile, ed effettua tramite esso il filtraggio basato su EKF. Sono dunque state implementate le matrici di dimensioni adeguate per un problema di questo tipo. Ipotizzando di vedere il Landmark j, abbiamo dunque:

$$F = \begin{bmatrix} \frac{\partial x_{t+1}}{\partial x_t} & \frac{\partial x_{t+1}}{\partial y_t} & \frac{\partial x_{t+1}}{\partial \theta_t} & \frac{\partial x_{t+1}}{\partial x_{L_j}} & \frac{\partial x_{t+1}}{\partial y_{L_j}} \\ \frac{\partial y_{t+1}}{\partial x_t} & \frac{\partial y_{t+1}}{\partial y_t} & \frac{\partial y_{t+1}}{\partial \theta_t} & \frac{\partial y_{t+1}}{\partial x_{L_j}} & \frac{\partial y_{t+1}}{\partial y_{L_j}} \\ \frac{\partial \theta_{t+1}}{\partial x_t} & \frac{\partial \theta_{t+1}}{\partial y_t} & \frac{\partial \theta_{t+1}}{\partial \theta_t} & \frac{\partial \theta_{t+1}}{\partial x_{L_j}} & \frac{\partial \theta_{t+1}}{\partial y_{L_j}} \\ \frac{\partial x_{L_j}}{\partial x_t} & \frac{\partial x_{L_j}}{\partial y_t} & \frac{\partial x_{L_j}}{\partial \theta_t} & \frac{\partial x_{L_j}}{\partial x_{L_j}} & \frac{\partial x_{L_j}}{\partial y_{L_j}} \\ \frac{\partial y_{L_j}}{\partial x_t} & \frac{\partial y_{L_j}}{\partial y_t} & \frac{\partial y_{L_j}}{\partial \theta_t} & \frac{\partial y_{L_j}}{\partial x_{L_j}} & \frac{\partial y_{L_j}}{\partial y_{L_j}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \frac{-(u_{R,t}+u_{L,t})}{2} sin\theta_t & 0 & 0 \\ 0 & 1 & \frac{(u_{R,t}+u_{L,t})}{2} cos\theta_t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3.11}$$

$$W = \begin{bmatrix} \frac{\partial x_{t+1}}{\partial u_{R,t}} & \frac{\partial x_{t+1}}{\partial u_{L,t}} \\ \frac{\partial y_{t+1}}{\partial u_{R,t}} & \frac{\partial y_{t+1}}{\partial u_{L,t}} \\ \frac{\partial \theta_{t+1}}{\partial u_{R,t}} & \frac{\partial \theta_{t+1}}{\partial u_{L,t}} \\ \frac{\partial x_{L_j}}{\partial u_{R,t}} & \frac{\partial x_{L_j}}{\partial u_{L,t}} \\ \frac{\partial y_{L_j}}{\partial u_{R,t}} & \frac{\partial y_{L_j}}{\partial u_{L,t}} \end{bmatrix} = \begin{bmatrix} \frac{cos\theta_t}{2} & \frac{cos\theta_t}{2} \\ \frac{sin\theta_t}{2} & \frac{sin\theta_t}{2} \\ \frac{1}{d} & -\frac{1}{d} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.3.12}$$

$$R = \begin{bmatrix} {\sigma_{RhoReale_j}}^2 & 0 \\ 0 & {\sigma_{PhiReale_j}}^2 \end{bmatrix} \tag{3.3.13}$$

$$H = \begin{bmatrix} -\frac{x_{SL_j,t}-\hat{x}_{R,t}}{\sqrt{q_1}} & -\frac{y_{SL_j,t}-\hat{y}_{R,t}}{\sqrt{q_1}} & 0 & \frac{x_{SL_j,t}-\hat{x}_{R,t}}{\sqrt{q_1}} & -\frac{y_{SL_j,t}-\hat{y}_{R,t}}{\sqrt{q_1}} \\ \frac{y_{SL_j,t}-\hat{y}_{R,t}}{q_1} & -\frac{x_{SL_j,t}-\hat{x}_{R,t}}{q_1} & -1 & -\frac{y_{SL_j,t}-\hat{y}_{R,t}}{q_1} & -\frac{x_{SL_j,t}-\hat{x}_{R,t}}{q_1} \end{bmatrix} \tag{3.3.14}$$

mentre Q resta nella forma (2.4.3). Si procede dunque all'aggiornamento della matrice di covarianza. Questo passo è il più delicato del nostro problema di adattamento del caso di corrispondenze note a quello di limitazioni sensoristiche. Infatti, in generale, la matrice di covarianza P ha dimensione (3+2n)x(3+2n), dove n è il numero di Landmark presenti nel sistema. Nel nostro caso, essendo presenti 4 Landmark, ha dimensione 11x11. Dovendo affrontare un problema in uno spazio di stato di dimensione inferiore, dato che possiamo vedere un solo Landmark, è stata introdotta una matrice filtro, che adatta la dimensione della matrice di covarianza stessa, portandola ad una 5x5 ((3+2*1)x(3+2*1)). A seconda del Landmark visto, la matrice filtro assume la forma:

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.3.15}$$

dove gli elementi diversi da zero in quarta e quinta riga dipendono dal Landmark visto. In particolare per la quinta riga è l'elemento 3+2j, mentre per la quarta riga è l'elemento 3+2j-1, dove j ricordiamo essere l'indice del Landmark visto. Nella matrice rappresentata in (3.3.15) si è dunque supposto visibile il Landmark 2.

Possiamo quindi ottenere la matrice di covarianza ridotta tramite la relazione

$$P_j = GPG^T. \tag{3.3.16}$$

E' questa la matrice che passiamo alla nostra funzione SLAM1, che viene dunque aggiornata tramite la relazione (3.3.5), dove le matrici interessate saranno quelle appena ridefinite, che assumono dunque le dimensioni adeguate. Nel caso in cui, avendo impostato la variabile Nstep, bisogna procedere al passo di correzione tramite EKF, si eseguono le relazioni (3.3.9) e (3.3.10) sempre con le matrici di dimensione ridotta. Vengono dunque aggiornate oltre alla posa stimata del robot e quella dell'unico Landmark visibile, anche la matrice di covarianza P. Per quest'ultimo aggiornamento è stato necessario riutilizzare la matrice filtro G, che riporta le dimensioni utilizzate nella funzione SLAM1 a quelle del sistema generale. Tuttavia questa trasformazione pone a zero tutte le componenti della matrice di covarianza che riguardano i Landmark non visti in questa iterazione. Bisogna quindi sommare ad essa una matrice di re-inizializzazione per tutti quei Landmark non visti, in modo da preparare la matrice di covarianza per il caso in cui dovesse risultare visibile nell'istante successivo uno dei Landmark non visti in questa iterazione. Questa matrice ha la forma del tipo

$$L = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{Rho_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{Phi_1} & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{Rho_3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{Phi_3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{Rho_4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \sigma^2_{Phi_4} \end{bmatrix} \tag{3.3.17}$$

dove gli elementi da impostare a zero a partire dalla quarta riga in poi dipendono dal Landmark visto. In particolare le righe 3+2j e 3+2j-1 devono risultare tutte nulle. In quella rappresenta in (3.3.17) si è dunque supposto visibile ancora il Landmark 2.

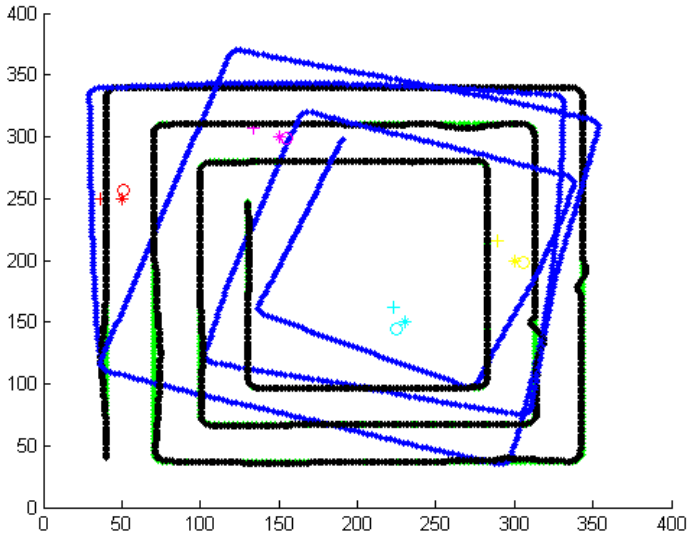

Figura 3.4: Rappresentazione della stima della posa del robot e dei Landmark applicando le limitazioni sensoriali su Rho

La relazione che restituisce la matrice di covarianza corretta è quindi

$$P = G^T P_j G + L \tag{3.3.18}$$

concludendo così il ciclo di correzione tramite EKF. Operando questo filtraggio, abbiamo che il robot stima la sua posizione lungo il percorso compiuto come mostrato dalla linea nera in figura 3.4. Viene ancora una volta corretto l'errore odometrico, ottenendo un buon lavoro di localizzazione. Si vede inoltre che la stima della posizione dei Landmark con l'applicazione dell'algoritmo di SLAM, si avvicina alla posizione effettiva del Landmark, ottenendo ancora una volta un buon lavoro di mappatura.

In figura 3.5 è presentato l'andamento del robot impostando RhoMassimo a 70 cm. Sebbene il risultato sembri un filtraggio non efficace, questo ci è utile per notare alcune dinamiche del filtraggio tramite EKF. In particolare notiamo che nella prima parte del percorso, il robot non vedendo alcun Landmark, segue perfettamente l'odometria. Nel momento in cui il Landmark 1 entra nel range di visibilità del robot stesso, la

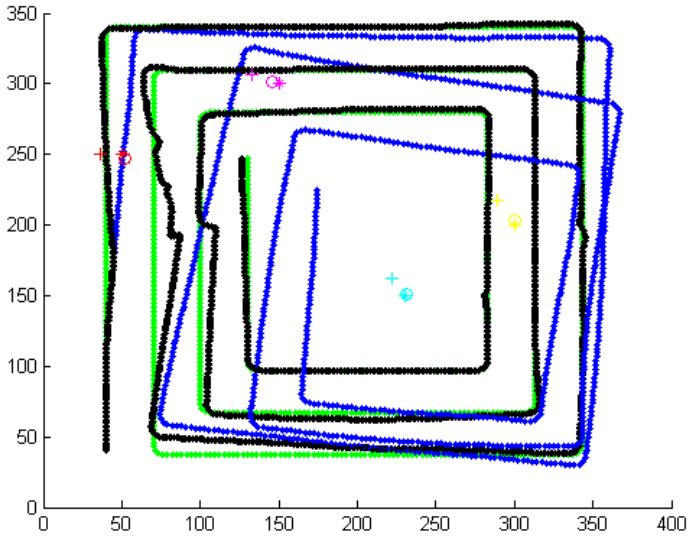

Figura 3.5: Rappresentazione della stima della posa del robot e dei Landmark applicando la limitazione su Rho a 70cm

sua posa stimata si corregge correttamente verso l'andamento reale. Possiamo notare un comportamento simile nella parte bassa del grafico. Abbiamo, infatti, nuovamente una situazione in cui il robot non vede alcun Landmark e continua il suo percorso seguendo l'andamento dell'odometria. Quando il Landmark 1 entra nel range di visibilità del robot, esso ritorna verso l'andamento reale.

Aggiungiamo ora la limitazione anche su Phi.

Riportiamo il limite di Rho a 130 cm, dove avevamo visto che il filtraggio funzionava in modo soddisfacente, ed impostiamo il limite su Phi a 260 gradi. Il robot stima la sua posizione lungo il percorso compiuto come mostrato dalla linea nera in figura 3.6. Rispetto alla sola limitazione su Rho, notiamo una maggiore imprecisione nella parte bassa del percorso. Prima della curva in basso a destra infatti, il robot non riesce a vedere il Landmark giallo. Effettua quindi la curva in modo impreciso,

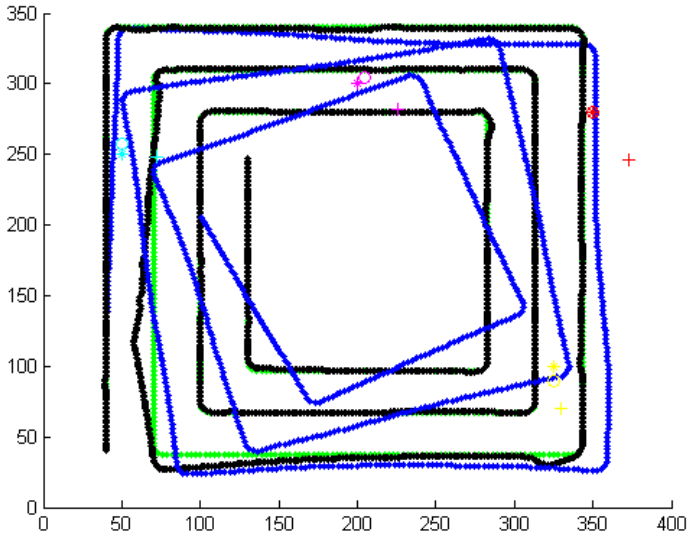

Figura 3.6: Rappresentazione della stima della posa del robot e dei Landmark applicando le limitazioni su Rho e su Phi

influenzando anche l'andamento successivo che nella prima parte viene corretto, ma poi, non avendo alcun Landmark tramite cui filtrare la posizione, segue l'andamento odometrico, allontanandosi leggermente dal percorso reale. Continuando il percorso è evidente il momento in cui il robot ricomincia a vedere il Landmark celeste e corregge il suo andamento di tipo odometrico ritornando verso il percorso reale. Per il resto il filtraggio funziona molto bene, seguendo precisamente il percorso reale. Anche la mappatura dei Landmark è molto vicina alla posizione reale, e migliora di molto la prima stima dei Landmark stessi.

In figura 3.7 vengono riportate una serie di simulazioni ottenute mantenendo la stessa odometria e variando solo il valore di Phi Massimo. E' così possibile valutare l'influenza di questo parametro sul filtraggio tramite EKF dell'algoritmo di SLAM. Commentiamo dunque queste rappresentazioni.

In figura a) è impostato Phi Massimo a 230 gradi. Notiamo un accentuarsi delle im-

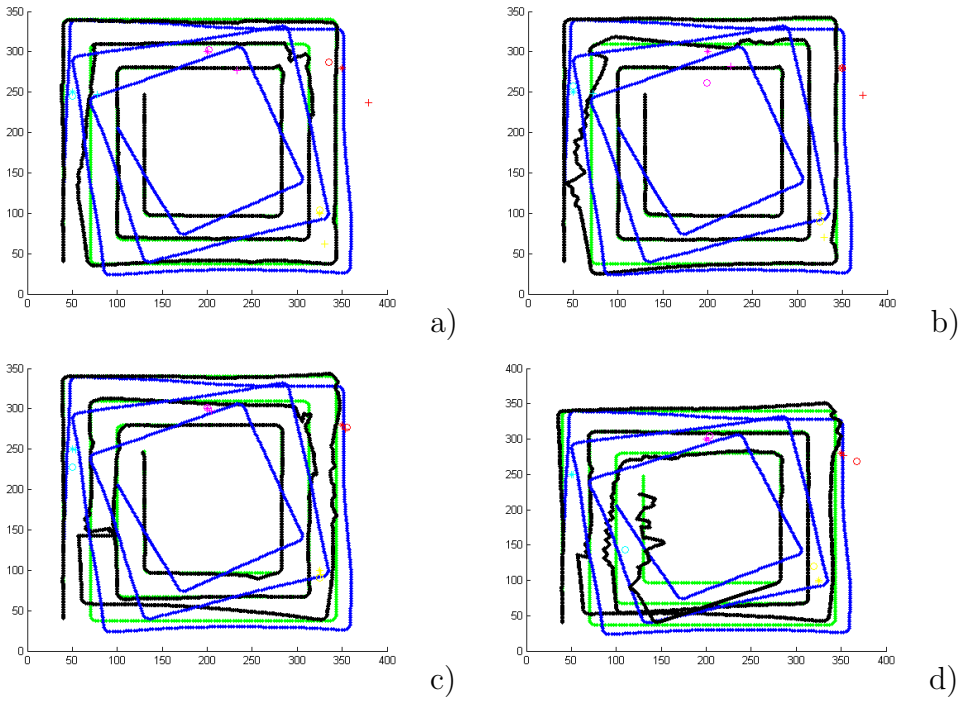

Figura 3.7: Andamento del filtraggio al variare di Phi Massimo.
In a)PhiMax=230, b)PhiMax=200, c) PhiMax=180, d) PhiMax=150.

perfezioni commentate in figura 3.6. Resta comunque un filtraggio abbastanza fedele al percorso reale.

In figura b) è impostato Phi Massimo a 200 gradi. Rispetto alla figura a) notiamo che quando il robot vede per la seconda volta il Landmark celeste e corregge tramite esso la traiettoria odometrica dovuta all'assenza di Landmark nella parte bassa del percorso, non riesce ad assestarsi sulla retta corretta come faceva in figura a), superando il Landmark celeste stesso di un angolo maggiore di quello massimo ammissibile. Questo porta ad una curva non perfetta, che comunque si assesta rapidamente per la presenza del Landmark viola.

In figura c) è impostato Phi Massimo a 180 gradi. Qui è facile vedere che appena il robot sorpassa il Landmark nel suo avanzare, ha uno scostamento evidente dal percorso reale. Si vede al primo passaggio del rosso e del giallo, si vede al secondo passaggio

del viola.

In figura d) è impostato Phi Massimo a 150 gradi. Qui oltre all'accentuarsi di tutti gli andamenti anomali presentati finora, abbiamo anche un notevole scostamento nel tratto finale del percorso. Questo perché nell'affrontare la penultima curva, il robot non vede più il Landmark giallo, portando ad un andamento totalmente odometrico, che tenta poi di correggere quando comincia a rivedere il Landmark celeste prima del termine del percorso stesso.

3.4 SLAM tramite EKF con corrispondenze non note in Matlab

In questo paragrafo elimineremo l'ipotesi di corrispondenze note, presentando le tecniche usate per affrontare questa situazione molto vicina alla realtà. E' stata mantenuta l'imposizione di un Rho Massimo, mentre è stata rimossa la limitazione su Phi. Questo ci porta a mantenere le matrici filtro prima presentate per ricondurci di volta in volta nel sottospazio di dimensione pari al numero di Landmark visibili in un certo istante di simulazione e su cui basare il filtraggio tramite EKF. Inoltre è stata aggiunta l'ipotesi che in un certo istante di tempo, è possibile aggiungere alla lista dei Landmark visti (quindi su cui basare il filtraggio) un solo Landmark alla volta. Il criterio impostato per scegliere quale Landmark aggiungere alla lista, tra quelli che entrano nella circonferenza di visibilità del robot, seleziona il Landmark effettivamente più vicino al robot stesso. In particolare, in questo ciclo di identificazione del Landmark entrato nella visibilità del robot, vengono aggiornati due vettori: **visto** e **lookup**, entrambi di dimensione *n+1* (quindi nel nostro caso composti da 5 elementi), di cui il primo indica quali dei Landmark presenti nell'ambiente sono stati visti ed inseriti nella lista, mentre il secondo rappresenta proprio la lista dei Landmark visti, posti

per ordine di visione durante il percorso. Viene quindi codificato il vettore *visto* in binario, per poi tradurlo in decimale, per scindere il caso in cui non è stato visto alcun Landmark da quello in cui almeno un Landmark è entrato nella lista. Infatti nel primo caso, dovremo procedere con l'odometria, mentre in tutti gli altri casi avremo almeno un Landmark su cui basare il nostro filtraggio tramite EKF. Viene inoltre introdotto l'indice N_t, che indica il numero di Landmark visti e catalogati, su cui è stata quindi costruita una mappa temporanea.

Il passo di *predizione* ha la stessa forma presentata con le corrispondenze note. Sono infatti le misure a subire dei controlli aggiuntivi per determinare se un Landmark che è all'interno della visibilità del robot è già stato considerato o è un nuovo Landmark da catalogare. Questo è reso possibile con il **criterio di massima verosimiglianza**. In particolare, per tutti gli *i* Landmark osservati in un certo istante di iterazione, si suppone ognuno di essi come un nuovo Landmark da catalogare. Viene quindi operato per questo nuovo presunto Landmark il passo di predizione tramite la relazione:

$$\begin{bmatrix} x_{StimaLandmark_{N_t+1}} \\ \\ y_{StimaLandmark_{N_t+1}} \end{bmatrix} = \begin{bmatrix} x_{StimaRobot} \\ \\ y_{StimaRobot} \end{bmatrix} + r_i \begin{bmatrix} cos(\phi_i + \theta_{StimaRobot}) \\ \\ sin(\phi_i + \theta_{StimaRobot}) \end{bmatrix} \tag{3.4.1}$$

dove r_i e ϕ_i rappresentano distanza e angolazione rilevata tramite sensoristica. A questo punto vengono calcolati per tutti i k Landmark stimati tramite predizione, con k che va da 1 a N_{t+1} (quindi anche il Landmark stimato tramite la 3.4.1) le quantità:

$$\begin{aligned} \delta_k &= \begin{bmatrix} \delta_{k,x} \\ \delta_{k,y} \end{bmatrix} = \begin{bmatrix} x_{StimaLandmark_k} - x_{StimaRobot} \\ y_{StimaLandmark_k} - y_{StimaRobot} \end{bmatrix} \\ \\ q_k &= \delta_k^T \delta_k \end{aligned} \tag{3.4.2}$$

permettendoci di definire il vettore delle misure nella forma

$$z_k = \begin{bmatrix} \sqrt{q_k} \\ atan2(\delta_{k,y}, \delta_{k,x}) - \theta_{StimaRobot} \end{bmatrix}. \tag{3.4.3}$$

Operiamo la linearizzazione calcolandoci la Jacobiana H che espressa rispetto alla forma compatta delle misure sarà del tipo

$$H_k = \frac{1}{q_k} \begin{bmatrix} -\sqrt{q_k}\ \delta_{k,x} & -\sqrt{q_k}\ \delta_{k,y} & 0 & \sqrt{q_k}\ \delta_{k,x} & \sqrt{q_k}\ \delta_{k,y} \\ \delta_{k,y} & -\delta_{k,x} & -q_k & -\delta_{k,y} & \delta_{k,x} \end{bmatrix} \tag{3.4.4}$$

che ci permette di calcolare la quantità

$$\pi_k = (z_i - z_k)^T (H_k P_{t+1} H^T + R)^{-1} (z_i - z_k) \tag{3.4.5}$$

dove z_i rappresenta il vettore delle misure ottenute tramite sensoristica, mentre R è ancora la matrice di covarianza dell'errore sensoriale presentata nella relazione (3.3.13). Questa quantità è chiamata **distanza di Mahalanobis**. In particolare, se dopo aver calcolato le Mahalanobis distance per tutti i Landmark, nessuna di queste è inferiore ad una certa soglia di tolleranza α (per cui un Landmark stimato in un'area centrata in uno dei Landmark già catalogati e di raggio α può essere ricondotto al Landmark stesso) questo è indice del fatto che il Landmark stimato è effettivamente un elemento nuovo e non catalogato della mappa e del vettore di stato. E' chiaro che la scelta del valore α di tolleranza è molto delicata ed è da valutare in base all'errore che la sensoristica può compiere. Appurata quindi la presenza di un nuovo Landmark da catalogare, viene aggiornato l'indice N_t tramite le relazioni:

$$\begin{aligned} j(i) &= argmin_k \pi_k \\ N_t &= max\{N_t, j(i)\}. \end{aligned} \tag{3.4.6}$$

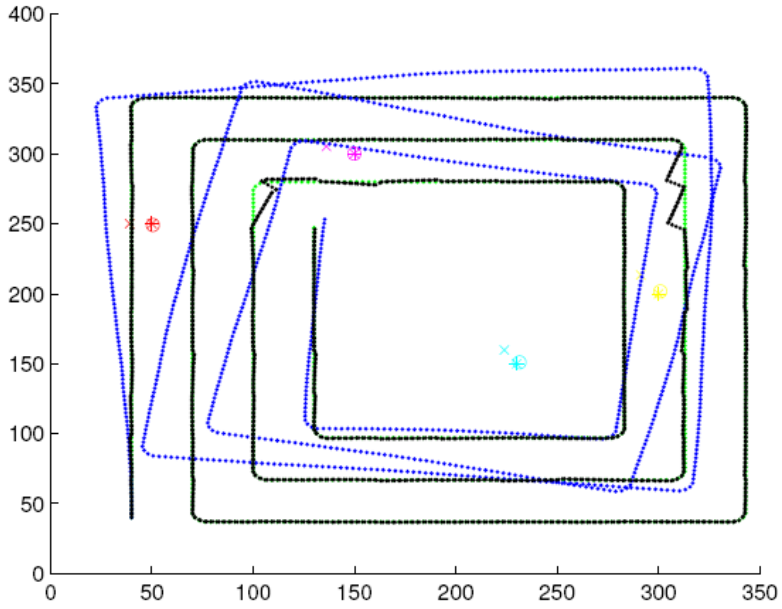

Figura 3.8: Rappresentazione della stima della posa del robot e dei Landmark applicando un filtraggio tramite EKF con corrispondenze non note

Vengono dunque utilizzate le matrici filtro per adattare la dimensione delle matrici utilizzate alla dimensione dello spazio di stato attuale. Si può quindi procedere alla fase di aggiornamento tramite le relazioni mostrate in (3.3.9) e (3.3.10) concludendo il ciclo di correzione tramite EKF. Operando questo filtraggio, abbiamo che il robot stima la sua posizione lungo il percorso compiuto come mostrato dalla linea nera in figura 3.8. Viene ancora una volta corretto l'errore odometrico, ottenendo un buon lavoro di localizzazione. Si vede inoltre che la stima della posizione dei Landmark con l'applicazione dell'algoritmo di SLAM, si avvicina alla posizione effettiva del Landmark, ottenendo ancora una volta un buon lavoro di mappatura.

Capitolo 4

Conclusioni e sviluppi futuri

In questo lavoro di tesi si è dunque presentato il problema di SLAM affrontato con l'approccio del Filtro di Kalman Esteso. Abbiamo mostrato che in un ambiente limitato e con un esiguo numero di Landmark, questa tecnica porta a degli ottimi risultati. Anche i tempi di esecuzione di tutti i programmi rimangono contenuti. In particolare per lo SLAM con corrispondenze note sono dell'ordine dei secondi mentre per lo SLAM con corrispondenze non note, delle decine di secondi. Ma è chiaro che questo approccio non è adeguato per una situazione con un alto numero di Landmark, che porterebbe al crescere delle dimensioni dello spazio di stato, con tutto quello che ne consegue. Per queste situazioni abbiamo già accennato ad un'ulteriore estensione al filtro di Kalman: l'*Unscented Kalman Filter*, che si basa sull'idea di individuare dei punti, chiamati σ-point, posti lungo l'ellissoide di covarianza, e ricalcolare la covarianza sulla base del motion model applicato ai σ-point stessi.

Chiaramente questa tesi ha sviluppato degli algoritmi che sono stati testati in Matlab. Interessante sarebbe applicare questi stessi algoritmi su un robot reale, con tutte le modifiche che questo passaggio comporta, per comparare i risultati ottenuti in un ambiente reale rispetto ad un ambiente simulativo. Più in generale si era già accennato al fatto che lo SLAM come problema teorico fosse stato formulato e risol-

to in diverse forme (diversi tipi di ambienti, indoor e outdoor, sistemi subacquei e nell'aria). A livello teorico e concettuale lo SLAM può essere dunque considerato un problema risolto. Tuttavia, le questioni sostanziali rimangono la realizzazione pratica di soluzioni SLAM più generali, e in particolare la costruzione e l'utilizzo di mappe particolarmente ricche.

Ringrazio particolarmente te che sei arrivato a leggere fino in fondo il racconto del mio lavoro compiuto, sperando di essere stato chiaro ed esauriente nello spiegare il primo passo verso la realizzazione di un robot autonomo.

Elenco delle figure

Bibliografia

[1] S.Thrun - W.Burgard - D.Fox, *"Probabilistic Robotics"*, the MIT Press,2005.

[2] E.Di Giampaolo - F.Martinelli, *"Robot localization by sparse and passive RFID tags"*, ISIE, 2010.

[3] Hugh Durrant - Whyte and Tim Bailey, *"Simultaneous Localization and Mapping: Tutorial"*, IEEE Robotics and Automation Magazine, 2006.

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

[5] R. Smith - P. Cheesman, *"On the representation of spatial uncertainty"*, Int. J. Robot. Res., vol. 5, no. 4, pp. 56-68, 1987.

[6] H.F. Durrant - Whyte, *"Uncertain geometry in robotics"*, IEEE Trans. Robot. Automat., vol. 4, no. 1, pp. 23-31, 1988.

[7] N. Ayache - O. Faugeras, *"Building, registrating, and fusing noisy visual maps"*, Int. J. Robot. Res., vol. 7, no. 6, pp. 45-65, 1988.

[8] J. Crowley, *"World modeling and position estimation for a mobile robot using ultra-sonic ranging"*, in Proc. IEEE Int. Conf. Robot. Automat., 1989, pp. 674-681.

[9] R. Chatila - J.P. Laumond, *"Position referencing and consistent world modeling for mobile robots"*, in Proc. IEEE Int. Conf. Robot. Automat., 1985, pp. 138-143.

[10] R. Smith - M. Self - P. Cheeseman, *"Estimating uncertain spatial relationships in robotics"*, in Autonomous Robot Vehicles, I.J. Cox and G.T. Wilfon, Eds. New York: Springer-Verlag, pp. 167-193, 1990.

[11] H. Durrant-Whyte - D. Rye - E. Nebot, *"Localisation of automatic guided vehicles"*, in Robotics Research: The 7th International Symposium (ISRR'95), G. Giralt and G. Hirzinger, Eds. New York: Springer Verlag, pp. 613-625, 1996.

[12] M. Csorba, *"Simultaneous Localisation and Map Building"*, Ph.D. dissertation,Univ. Oxford, 1997.

[13] M. Csorba - H.F. Durrant-Whyte, *"A new approach to simultaneous localisation and map building"*, in Proc. SPIE Aerosense, Orlando, FL, 1996.

[14] J.J. Leonard - H.J.S. Feder, *"A computational efficient method for large-scale concurrent mapping and localisation"*, in Robotics Research, The Ninth International Symposium (ISRR'99), J. Hollerbach and D. Koditscheck, Eds. New York: Springer-Verlag, pp. 169-176, 2000.

[15] J.A. Castellanos - J.M. Martnez - J. Neira - J.D. Tardós, *"Experiments in multisensor mobile robot localization and map building"*, in Proc. 3rd IFAC Sym. Intell. Auton. Vehicles, 1998, pp. 173-178.

[16] J.A. Castellanos - J.D. Tardós - G. Schmidt, *"Building a global map of the environment of a mobile robot: The importance of correlations"*, in Proc. IEEE Int. Conf. Robot. Automat., 1997, pp. 1053-1059.

[17] J. Guivant - E.M. Nebot - S. Baiker, *"Localization and map building using laser range sensors in outdoor applications"*, J. Robot. Syst., vol. 17, no. 10, pp. 565-583, 2000.

[18] S.B. Williams - P. Newman - G. Dissanayake - H.F. Durrant-Whyte, *"Autonomous underwater simultaneous localisation and map building"*, in Proc. IEEE Int. Conf. Robot. Automat. (ICRA), San Francisco, CA, Apr. 2000, pp. 1793-1798.

[19] J. Hollerbach - D. Koditscheck, Eds., *Robotics Research, The Ninth International Symposium (ISRR'99)*, New York: Springer-Verlag, 2000.

[20] S. Thrun, D. Fox, and W. Burgard, *"A probabilistic approach to concurrent mapping and localization for mobile robots"*, Mach. Learning, vol. 31, no. 1, pp. 29-53, 1998.