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

IMPLEMENTAZIONE E CONFRONTO SPERIMENTALE DI METODI PER LA RIDUZIONE DI SEGMENTI RIDONDANTI IN MAPPE INDOOR COSTRUITE DA ROBOT MOBILI

Relatore: prof. Francesco AMIGONI

Tesi di Laurea di:
Marco Deambrogi, matricola 747558
Matteo Ferrario, matricola 739351

Anno Accademico 2011-2012

Sommario

L'interesse nei confronti delle mappe basate su segmenti è in continuo aumento. Esse rappresentano una soluzione efficiente per la rappresentazione di ambienti indoor. Un problema aperto riguardante questa tipologia di mappa è quello di ridurre il numero di segmenti ridondanti, introdotti dalle operazioni di acquisizione, che rappresentano la stessa porzione di ambiente e che aumentano la dimensione della mappa causando problemi nella sua gestione. In questa tesi vengono confrontati diversi metodi di riduzione presenti in letteratura. Le prove sperimentali mostrano potenzialità e limiti di ciascun metodo. I risultati del lavoro di ricerca permettono di ricavare alcune linee guida per la progettazione e lo sviluppo di soluzioni future per la riduzione di segmenti ridondanti.

Indice

Capitolo 1

Introduzione

La capacità di costruire una mappa dell'ambiente è una delle caratteristiche fondamentali dei robot mobili autonomi. Nel corso degli anni sono stati sviluppati diversi *sistemi di mappatura* [1], tuttavia, ultimamente, una particolare tipologia di rappresentazione per ambienti indoor, caratterizzata dall'utilizzo dei segmenti, ha suscitato sempre maggiore interesse [4, 15, 16, 17, 19, 20, 21, 22, 29, 30, 31, 32, 33, 42, 43, 44, 45, 48]. Un problema aperto, riguardante questa tipologia di sistemi di mappatura, è la capacità di costruire una rappresentazione compatta dell'ambiente, eliminando quei segmenti che rappresentano uno stesso oggetto e che costituiscono quindi informazione ridondante.
Lo scopo di questa tesi è confrontare tra loro diversi metodi di riduzione dei segmenti ridondanti, già esistenti in letteratura, in modo da stabilire quale sia il metodo di fusione che fornisce le prestazioni migliori e in quali condizioni e in modo da gettare le basi per lo sviluppo di nuovi metodi. Dopo una prima analisi dei metodi esistenti, sono stati scelti, attraverso opportune considerazioni, alcuni metodi significativi su cui concentrare il lavoro di ricerca. I metodi selezionati sono stati implementati e sono stati applicati a quattro insiemi di dati provenienti da esplorazioni reali. Per ogni metodo è stata condotta un'analisi sia quantitativa che qualitativa. In particolare, per ciascun metodo, sono stati presi in considerazione i tempi computazionali, le percentuali di riduzione dei segmenti e la qualità delle mappe finali ridotte. La misura di qualità rappresenta uno degli aspetti innovativi di questo lavoro di ricerca. Nelle analisi condotte fino ad oggi, infatti, il confronto tra i metodi è stato fatto in gran parte basandosi sulle percentuali di riduzione e mai prendendo in considerazione la qualità delle mappe finali risultanti. I risultati ottenuti nelle prove sperimentali hanno portato alla conclusione

che le prestazioni di uno specifico metodo di fusione sono correlate alle caratteristiche dei dati sui quali il metodo viene applicato: in alcuni ambienti è risultato più efficace un particolare metodo di fusione, mentre in altri ambienti diversi metodi di fusione sono stati in grado di fornire prestazioni simili.

L'esigenza di ottenere una mappa accurata, facilmente aggiornabile e consultabile rappresenta la sfida da affrontare nella realizzazione di un sistema di mappatura. La costruzione di una mappa richiede da parte del robot l'acquisizione, attraverso i sensori di cui è dotato, come ad esempio telemetri laser, sonar o telecamere, di scansioni che contengono informazioni sull'ambiente. Tali informazioni devono essere successivamente interpretate e integrate per la composizione di un'unica mappa globale. Durante il processo di costruzione della mappa, però, le scansioni acquisite possono riferirsi alla stessa porzione di ambiente e quindi contenere informazione ridondante che, senza aggiungere nuova conoscenza, non fa altro che aumentare la dimensione della mappa e rendere le operazioni di aggiornamento inutilmente più difficoltose. Sono necessari quindi dei meccanismi che individuino ed eliminino tale informazione ridondante. Nel caso delle mappe basate su segmenti, questo si traduce nella necessità di meccanismi in grado di ridurre insiemi di segmenti che rappresentano una stessa porzione di ambiente. Tale compito non è banale, principalmente a causa di errori introdotti durante il processo di acquisizione. A causa di questi può succedere che segmenti che descrivono la stessa porzione di ambiente abbiano caratteristiche abbastanza differenti. Ad oggi in letteratura sono state realizzate differenti soluzioni per risolvere il problema della ridondanza nelle mappe basate su segmenti [4, 15, 29, 30, 31, 33, 42, 43, 44, 45, 48]: queste soluzioni differiscono tra loro per il modo in cui i segmenti ridondanti vengono individuati e conseguentemente ridotti. La nostra analisi segue quella in [52] e ha l'obiettivo di migliorare la valutazione e il confronto tra i metodi, aggiungendo nuovi termini di paragone.
Il lavoro svolto in questa tesi è rivolto proprio in questa direzione.

La tesi è strutturata nel modo seguente.
Nel Capitolo 2 si mostra lo stato dell'arte riguardante il lavoro svolto, inquadrando dapprima l'ambito dell'esplorazione robotica e le varie tipologie di mappe e, successivamente, illustrando il problema della fusione nelle mappe a segmenti attraverso la descrizione dei metodi esistenti in letteratura.
Nel Capitolo 3 si affronta il problema della costruzione di mappe bidimensionali basate su segmenti, illustrando dapprima alcuni dei metodi esistenti

per l'estrazione dei segmenti e, successivamente, alcuni dei metodi esistenti per l'allineamento dei segmenti.
Nel Capitolo 4 si descrivono in dettaglio i metodi di fusione presi in considerazione nel lavoro di ricerca.
Nel Capitolo 5 si descrivono le implementazioni dei metodi di fusione considerati mostrando lo pseudocodice degli algoritmi.
Nel Capitolo 6 si mostrano le varie prove sperimentali, eseguite su data set provenienti da acquisizioni in ambienti reali, e i risultati ottenuti nella fusione delle mappe applicando i diversi metodi di fusione considerati.
Nel Capitolo 7 si riassumono gli scopi del lavoro svolto, le valutazioni effettuate e i possibili sviluppi futuri.

Capitolo 2

Stato dell'arte

2.1 L'esplorazione robotica e la costruzione di mappe per l'ambiente

Una delle aree di ricerca più attive della robotica mobile è quella relativa al problema della *navigazione* [2, 3], definito come il problema, per un robot mobile, di raggiungere una posizione finale a partire da una posizione iniziale, evitando gli ostacoli presenti nell'ambiente.
Il problema della navigazione è strettamente correlato a tre specifici problemi [4, 5, 6, 7]:

- *localizzazione*: il robot deve essere in grado di determinare la propria posizione all'interno dell'ambiente;
- *mappatura*: il robot deve essere in grado di costruire una mappa dell'ambiente circostante. Una mappa è una rappresentazione semplificata dello spazio [1];
- *pianificazione*: data una descrizione dello stato iniziale e una descrizione dello stato finale, il robot, basandosi sulle conoscenze a sua disposizione, deve essere in grado di generare il percorso migliore per giungere a destinazione.

I tre processi descritti in precedenza sono strettamente correlati fra di loro [19, 20, 21, 22]. Come abbiamo visto, un aspetto fondamentale è la costruzione di una mappa dell'ambiente, ossia la costruzione di un modello spaziale dell'ambiente in cui il robot si trova a dover operare [6, 11, 12, 13]. La costruzione di una mappa avviene in una fase detta di *esplorazione* [2]:

durante tale fase il robot raccoglie dati sull'ambiente necessari alla costruzione della mappa. Questi dati, forniti dai sensori di cui il robot è dotato, indicano la posizione degli oggetti fisici che occupano lo spazio circostante. In letteratura si distinguono due approcci nella costruzione delle mappe: *topologico* e *geometrico*. Nell'approccio topologico (Figura 2.1) un ambiente è rappresentato in termini di punti di interesse per il robot e delle relazioni intercorrenti fra di essi. Un punto di interesse è un oggetto naturale o artificiale che ha importanza per la navigazione del robot (ad esempio: pareti, porte...) [5, 11, 12, 13, 14]. Nell'approccio geometrico (Figura 2.2) gli oggetti

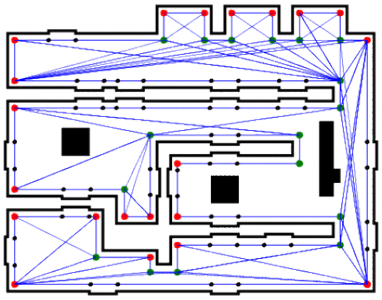

Figura 2.1: Esempio di mappa topologica [13].

dell'ambiente sono rappresentati metricamente in base alle loro dimensioni e posizioni rispetto ad un sistema di riferimento assoluto [4, 15, 16, 17, 18, 19].

Figura 2.2: Esempio di mappa geometrica [13].

2.2 Tipologie di mappe

Nella precedente sezione abbiamo introdotto una prima distinzione tra mappe topologiche e geometriche: le prime forniscono una rappresentazione dell'ambiente in termini di punti di interesse per il robot; le seconde riportano dimensioni e posizioni di oggetti presenti all'interno dell'ambiente.
I principali metodi per la rappresentazione di mappe geometriche sono due: l'approccio basato su *griglia di occupazione* [7, 10, 23, 24, 25, 26] e quello basato sulla descrizione con primitive geometriche degli oggetti circostanti. Nel primo caso (Figura 2.3) un ambiente è rappresentato tramite una griglia

Figura 2.3: Esempio di mappa ottenuta con griglia di occupazione [13].

bidimensionale o tridimensionale. A ogni elemento della griglia è assegnato un valore che indica lo stato della cella: libera o occupata. La risoluzione della mappa è definita dalla granularità della griglia. Le mappe basate su

Figura 2.4: Esempio di mappa composta da nuvole di punti [33].

griglia di occupazione (Figura 2.4) sono semplici da costruire, tuttavia richiedono elevate risorse per la loro memorizzazione e il loro accesso. Queste particolari mappe, inoltre, diventano impraticabili nel caso di rappresentazioni tridimensionali. Nel secondo caso, invece, un ambiente è rappresentato mediante l'utilizzo di primitive geometriche. Un caso particolare consiste nell'utilizzare direttamente i punti forniti da alcuni sensori per descrivere i contorni degli oggetti. Il risultato è una mappa composta da *nuvole di punti* che definiscono i bordi degli ostacoli [22, 27, 28]: tale approccio però soffre degli stessi problemi delle mappe basate su griglia di occupazione. Un'altra

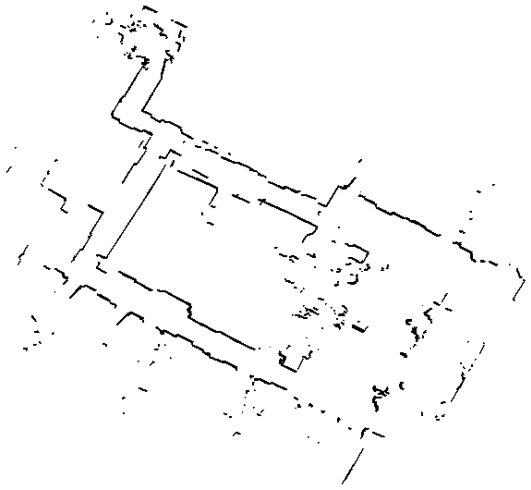

Figura 2.5: Esempio di mappa basata su segmenti [35].

possibilità consiste nell'utilizzare i segmenti di retta come primitiva geometrica per descrivere i contorni degli ostacoli: si parla quindi di mappe *basate sui segmenti* [4, 15, 16, 17, 19, 20, 21, 22, 29, 30, 31, 32, 33, 34] e su tale tipologia di mappe abbiamo concentrato il nostro lavoro di tesi.
Nella prossima sezione illustreremo vantaggi e svantaggi di questo tipo di rappresentazione.

2.3 Mappe basate su segmenti

L'interesse nei confronti delle mappe basate su segmenti è in continuo aumento. Ciò è dovuto ai numerosi vantaggi che derivano dal loro utilizzo, tra i quali ricordiamo:

- l'approccio basato sui segmenti permette di catturare informazioni strutturali, le quali vanno oltre le informazioni ottenibili mediante

mappe basate sui soli punti forniti dai sensori. Le mappe basate sui segmenti combinano un insieme di punti in un singolo segmento: questo permette di estrarre, ad esempio, la *direzione* di un segmento, la quale può essere utilizzata per aumentare la robustezza degli eventuali processi di allineamento e di fusione successivi;

- l'approccio basato sui segmenti è veloce. Una tipica scansione di un robot consiste in un numero di segmenti inferiore a venti, mentre il numero di punti è tipicamente di uno o due ordini di grandezza superiore;

- l'approccio basato sui segmenti è efficiente dal punto di vista della memoria utilizzata. Ogni segmento richiede la memorizzazione dei soli suoi due estremi;

- l'approccio basato sui segmenti è preciso. Gli estremi del segmento non devono essere adattati ad un parametro di risoluzione: questo comporta l'assenza di errori di quantizzazione.

Tra gli svantaggi, invece, ricordiamo la possibilità di utilizzare tale approccio solo per rappresentare ambienti rettilinei, come ad esempio gli ambienti indoor, e la maggiore complessità del processo di costruzione della mappa, che richiede l'interpretazione dei dati forniti dai sensori per estrarne le proprietà geometriche. I numerosi lavori presenti oggi in letteratura permettono di superare agevolmente quest'ultimo svantaggio. Nel sistema di mappatura in [36] è stato implementato un processo ricorsivo che cerca di far aderire una polilinea[1] alle misurazioni effettuate da un sonar. I sistemi in [16] e [37] mostrano come sia possibile estrarre polilinee dalle scansioni di un laser. In [27], [28] e [38] sono stati studiati metodi che permettono di creare mappe geometriche composte da oggetti primitivi. In [39] viene utilizzata una rete neurale per l'estrazione di segmenti dalle letture di un sonar. Un ulteriore approccio in [30] permette la costruzione di mappe senza utilizzare alcuna informazione relativa alla posizione del robot. In [19] e [29] vengono implementati sistemi di mappatura in un contesto di cooperazione multiagente. Infine esistono in letteratura numerosi metodi che permettono di estrarre un insieme di segmenti a partire da un insieme di punti: i più popolari sono confrontati in [40] sulla base di diversi criteri con lo scopo di sottolineare vantaggi e svantaggi di ciascun metodo.

[1] Insieme ordinato di segmenti orientati ordinatamente consecutivi (cioè tali che il secondo estremo di un segmento coincida con il primo estremo del successivo) e non adiacenti (cioè tali che un segmento e il suo successivo non appartengano alla stessa retta).

2.4 Costruzione di mappe basate su segmenti

Un generico robot mobile costruisce una mappa bidimensionale di ambienti indoor basata su segmenti secondo il seguente processo:

1. acquisizione di una nuova scansione S^t nella posizione corrente p^t del robot nell'ambiente (t è il tempo del passo corrente). La scansione S^t rappresenta la mappa locale della porzione di spazio percepita dai sensori in p^t. I dati contenuti in questa scansione vengono successivamente approssimati con segmenti utilizzando algoritmi ad hoc per la particolare tipologia di sensori [16, 36, 39, 53] oppure utilizzando metodi generali di estrazione [17, 33, 44, 46];

2. registrazione di S^t con la mappa globale corrente M^t. In molti casi la registrazione avviene tra S^t e la scansione precedente S^{t-1}. Tuttavia, in generale si può fare riferimento esclusivamente a M^t poiché essa contiene tutte le informazioni sulle precedenti scansioni e quindi anche sulla scansione S^{t-1}. La registrazione consiste nel calcolo della rototraslazione r^t che deve essere applicata ai segmenti di S^t per allinearli ai segmenti presenti in M^t. Alcuni approcci utilizzano le informazioni odometriche per avere una stima iniziale di r^t [47], altri approcci invece utilizzano esclusivamente le informazioni geometriche presenti in S^t e M^t per trovare r^t [30];

3. aggiornamento della mappa globale con i segmenti della scansione S^t per ottenere la mappa aggiornata M^{t+1};

4. pianificazione della nuova posizione di scansione del robot p^{t+1} e del percorso necessario per arrivarci. Alcuni approcci utilizzano strategie di pianificazione per massimizzare la quantità e la qualità attesa dalle informazioni ottenibili nelle posizioni successive [4];

5. navigazione verso la posizione p^{t+1}. Infine il processo riparte dal passo 1 per acquisire nuove scansioni.

2.5 Il problema dell'allineamento delle scansioni

Durante la fase di esplorazione, ogni scansione acquisita è accompagnata da una stima della posizione corrente del robot, espressa rispetto a un sistema di riferimento globale. Il robot mantiene la conoscenza circa la sua posizione attraverso i dati forniti dall'odometria[2]. Tuttavia, le informazioni fornite

[2]L'odometria è la tecnica per stimare la posizione di un veicolo su ruote che si basa su informazioni provenienti da sensori che misurano lo spazio percorso.

dai sensori odometrici non sono affidabili: gli errori accumulati durante la navigazione (dovuti a scivolamenti, rotazioni e approssimazioni) rendono inaffidabile il tracciamento della posizione corretta del robot. Sono necessari quindi dei metodi di allineamento che, dopo l'acquisizione di una nuova scansione, provvedono a correggere la stima della posizione del robot sulla base del contenuto delle scansioni.

2.5.1 Panoramica dei metodi di allineamento

In [41] viene descritto il metodo di allineamento meglio noto come *Iterative Closest Point*: l'algoritmo cerca di allineare due insiemi di punti eseguendo iterativamente due passi. Nel primo passo l'algoritmo, basandosi su una determinata misura di distanza, cerca coppie di punti corrispondenti appartenenti a due differenti scansioni. Nel secondo passo l'algoritmo calcola la trasformazione (rotazione e traslazione) che minimizza la distanza tra i punti corrispondenti.

L'approccio descritto in [31] è derivato dal precedente e adattato in modo tale da allineare una nuova scansione a una mappa globale, entrambe composte da segmenti. Il punto chiave dell'algoritmo è la definizione di una misura di distanza che permette di stabilire quando due segmenti possono essere considerati corrispondenti. Nell'approccio descritto due segmenti vengono considerati corrispondenti quando la loro distanza angolare è inferiore a una soglia T_θ e la loro distanza spaziale è inferiore a una soglia T_D. L'algoritmo di allineamento ripete iterativamente i seguenti passi:

1. rappresenta la nuova scansione S nel sistema di riferimento della mappa globale M, applicando a S le trasformazioni fornite dalla stima odometrica della sua posizione;

2. cerca la rotazione $\overline{\theta}$ che, applicata a S, minimizza la distanza tra i segmenti di S e quelli di M:

$$D(S, M) = \frac{1}{N_m + N_n} \cdot \sum_{i=1}^{N_m} D'(s_i) + N_n \cdot T_d \tag{2.1}$$

   dove N_m è il numero di segmenti di S che corrispondono ad almeno un segmento di M e N_n è il numero di segmenti di S che non corrispondono ad alcun segmento di M. Per ogni segmento s della scansione S viene definito l'insieme C_s dei segmenti di M che corrispondono a s (naturalmente l'insieme C_s può essere vuoto). $D'(s_i)$ è la distanza

media di un segmento s_i di S dai segmenti in C_{s_i} pesati con le loro lunghezze;

3. applica la rotazione $\overline{\theta}$ a S;
4. cerca la traslazione $(\overline{x}, \overline{y})$ che, applicata a S, minimizza $D(S, M)$;
5. applica la traslazione $(\overline{x}, \overline{y})$ a S;
6. ripete le operazioni dal passo due per un certo numero di volte o fino a quando $D(S, M)$ è inferiore a una determinata soglia.

L'algoritmo si basa sull'ipotesi che, quando viene calcolato $D(S, M)$, la posizione stimata della nuova scansione è abbastanza precisa da rendere corrispondenti due segmenti che rappresentano la stessa porzione di ambiente.

L'approccio descritto in [42] presenta un metodo di allineamento globale: l'algoritmo viene eseguito una volta terminata la fase di esplorazione del robot e viene applicato contemporaneamente a tutte le scansioni. L'algoritmo prevede una fase preliminare di filtraggio delle scansioni, la quale comporta l'eliminazione delle scansioni fortemente ridondanti: poiché l'algoritmo viene applicato globalmente, aumentando il numero delle scansioni, aumenta anche la probabilità che il problema diventi computazionalmente intrattabile. La fase di filtraggio si basa sulla definizione di *intersezione* tra due differenti scansioni: l'intersezione tra due scansioni S_1 e S_2 è definita come l'insieme di coppie di segmenti (u_i, v_i), $u_i \in S_1$, $v_i \in S_2$ con u_i simile a v_i nel senso della misura di similarità spiegata nella Sezione 2.7.7. Il problema della selezione delle scansioni viene quindi formulato come un problema di minimizzazione: dato un insieme ordinato di scansioni $S = (S_1, S_2, \cdots, S_n)$, selezionare un sottoinsieme ordinato e minimo di scansioni $S^- = (S_{i1}, S_{i2}, \cdots, S_{im}) \subset S$ tale che, per ogni coppia di scansioni sequenziali $S_j, S_{j+1} \in S^-$, le seguenti due condizioni risultino verificate:

- $|S_j \cap S_{j+1}| \geq 2$, ossia le scansioni devono avere almeno due caratteristiche che corrispondono;
- $S_j \cap S_{j+1} = \{(u_1, v_1), \cdots, (u_k, v_k)\}$, e inoltre ci devono essere almeno due caratteristiche (u_{i1}, u_{i2}) e (v_{i1}, v_{i2}) con u_{i1} non simile a u_{i2} e v_{i1} non simile a v_{i2}.

Dopo la fase di filtraggio, le scansioni rimaste sono pronte per essere allineate. Il metodo di allineamento utilizza un approccio basato sull'ottimizzazione del grafo delle posizioni globali. In tale grafo ogni nodo rappresenta una

posa $X_i = \{x, y, \theta\}$ in cui è stata acquisita una scansione, mentre i collegamenti fra i nodi rappresentano le relazioni tra le posizioni. Per ogni coppia di scansioni X_i, X_j viene calcolato l'errore relativo di posizionamento E_{ij}, definito come:

$$E_{ij} = |X_i - X_j + c_{i,j}|^2 \tag{2.2}$$

dove $c_{i,j}$ è una costante che rappresenta, ad esempio, la stima della differenza tra le pose. L'obiettivo del metodo di allineamento è quello di trovare un insieme ottimo di posizioni che minimizza l'errore globale, così definito:

$$\sum_{i,j} w_{i,j} \cdot E_{i,j} \tag{2.3}$$

dove $w_{i,j}$ rappresenta la forza della corrispondenza tra i e j. Se $E_{i,j}$ è lineare, allora la soluzione $X = (X_1^T, \cdots, X_n^T)^T$ del problema di allineamento è facilmente calcolabile risolvendo il sistema $BX = A$, dove A e B sono definite come:

$$B_{i,i} = \sum_{i,j} w_{i,j} + \sum_{j,i} w_{j,i} \tag{2.4}$$

$$B_{i,j} = -w_{i,j} \tag{2.5}$$

$$A_i = \sum_{i,j} w_{i,j} \cdot c_{i,j} + \sum_{j,i} w_{j,i} \cdot c_{j,i} \tag{2.6}$$

Tuttavia a causa della combinazione di rotazione e traslazione nella definizione di una posa $X_i = (x_i, y_i, \alpha_i)^T$, la differenza $X_i - X_j$ è fortemente non lineare: risulta quindi necessaria una linearizzazione del problema.
Il metodo di allineamento consiste di tre passi:

1. *calcolo delle corrispondenze fra segmenti.* Prima di ottimizzare il grafo, vengono calcolate le corrispondenze tra segmenti per ogni arco i, j. La forza della corrispondenza fra un segmento u della scansione i e un segmento v della scansione j è calcolata utilizzando la misura di distanza, definita nella Sezione 2.7.7, come parametro di una probabilità di corrispondenza gaussiana:

$$w_{u,v}^{i,j} = e^{\frac{-D(u,v,w_u,w_v)}{2\sigma^2}} \tag{2.7}$$

2. *calcolo della rotazione ottima.* Ogni segmento u ha un orientamento α_u^i nel sistema di riferimento locale della scansione i. L'orientamento globale di u è dato dalla somma di α_u^i e di α_i. L'idea dell'allineamento si basa sulla considerazione che due segmenti corrispondenti devono

avere lo stesso orientamento globale. Di conseguenza, l'errore rotazionale fra le scansioni i e j è dato dalla somma pesata delle differenze quadratiche nell'orientamento di tutte le coppie di segmenti:

$$E^R_{i,j} = \sum_{u,v} w^{i,j}_{u,v} |(\alpha^i_u + \alpha_i) - (\alpha^j_v + \alpha_j)|^2 \tag{2.8}$$

La soluzione $X = (\alpha_1, \cdots, \alpha_n)^T$ che minimizza l'errore è ottenuta risolvendo il sistema $BX = A$, dove A e B sono definite come:

$$B_{i,i} = \sum_{i,j} \sum_{u,v} w^{i,j}_{u,v} + \sum_{j,i} \sum_{u,v} w^{j,i}_{u,v} \tag{2.9}$$

$$B_{i,j} = -\sum_{u,v} w^{i,j}_{u,v} \tag{2.10}$$

$$A_i = \sum_{i,j} \sum_{u,v} w^{i,j}_{u,v} \cdot (\alpha^i_u - \alpha^j_v) - \sum_{j,i} \sum_{u,v} w^{j,i}_{u,v} \cdot (\alpha^j_u - \alpha^j_v) \tag{2.11}$$

3. *calcolo della traslazione ottima.* Per ogni coppia di segmenti corrispondenti u e v vengono calcolate le proiezioni dei due estremi di u (s_u e e_u) su v, ottenendo $s'_{u,v}$ e $e'_{u,v}$. Indicati con $S^{i,j}_{u,v}$, $E^{i,j}_{u,v}$ le differenze tra s_u, $s'_{u,v}$ e tra e_u, $e'_{u,v}$, l'obiettivo è quello di minimizzare le differenze pesate su tutti i segmenti:

$$E^T_{i,j} = \sum_{u,v} w^{i,j}_{u,v} (|T_i - T_j + S^{i,j}_{u,v}|^2 + |T_i - T_j + E^{i,j}_{u,v}|^2) \tag{2.12}$$

La soluzione $X = (T_1, \cdots, T_n)^T$ che minimizza l'errore è ottenuta risolvendo il sistema $BX = A$, dove A e B sono definite come:

$$B_{i,i} = \sum_{i,j} \sum_{u,v} w^{i,j}_{u,v} + \sum_{j,i} \sum_{u,v} w^{j,i}_{u,v} \tag{2.13}$$

$$B_{i,j} = -\sum_{u,v} w^{i,j}_{u,v} \tag{2.14}$$

$$A_i = \sum_{i,j} \sum_{u,v} w^{i,j}_{u,v} (S^{i,j}_{u,v} + E^{i,j}_{u,v}) - \sum_{j,i} \sum_{u,v} w^{j,i}_{u,v} \cdot (S^{j,i}_{u,v} + E^{j,i}_{u,v}) \tag{2.15}$$

In [30] è descritto un metodo di allineamento eseguito durante la fase di esplorazione dell'ambiente. Il metodo non si basa su alcuna informazione relativa alla posizione del robot, ma utilizza solamente le informazioni geometriche estratte dalle scansioni. Per scansione si intende un insieme di segmenti, dove ogni segmento è rappresentato dai suoi due estremi espressi in un sistema di riferimento globale. Una mappa parziale è l'integrazione

di due scansioni, di una scansione e di una mappa parziale oppure di due mappe parziali. Per integrare due mappe parziali (e, in particolare, due scansioni) senza fare riferimento ad alcuna informazione relativa alla posizione del robot, il metodo di allineamento considera gli angoli tra coppie di segmenti presenti nella mappa come punti di riferimento dell'ambiente. L'idea di base è quella di trovare corrispondenze fra angoli simili delle due mappe parziali.
Questo metodo di allineamento, già implementato, è stato importato nel nostro lavoro di ricerca ed è descritto in modo dettagliato nella Sezione 3.3.1.

2.6 Il problema della fusione dei segmenti ridondanti

Il processo descritto nella Sezione 2.4 fornisce le indicazioni su come un generico robot mobile può costruire una mappa bidimensionale di ambienti indoor basata su segmenti. Tuttavia questa mappa può contenere segmenti ridondanti, che rappresentano la stessa porzione di ambiente: la dimensione della mappa può pertanto essere ridotta utilizzando diversi metodi di fusione e fondendo tra loro i segmenti ridondanti. Il processo di fusione dei segmenti può avvenire:

- al termine del passo 3 del processo descritto in precedenza. In tal caso si parla di *metodi di fusione dei segmenti on-line*: subito dopo l'ottenimento della mappa aggiornata M^{t+1}, i segmenti ridondanti vengono fusi insieme in modo da mantenere in memoria il minore numero possibile di segmenti. In questo caso il processo di fusione dei segmenti viene ripetuto ad ogni nuova scansione;

- al termine dell'intero processo di acquisizione delle scansioni. In tal caso si parla di *metodi di fusione dei segmenti off-line*: dopo l'acquisizione di n scansioni differenti, i segmenti ridondanti della mappa globale finale vengono fusi insieme in modo da mantenere in memoria il minore numero possibile di segmenti. In questo caso il processo di fusione dei segmenti viene effettuato una sola volta, a mappa globale completa.

I metodi di fusione dei segmenti sono generalmente suddivisi in due fasi:

1. ricerca di un insieme S di segmenti che rappresentano lo stesso oggetto o nell'ambiente. Tale fase è solitamente basata su un insieme di condi-

zioni che i segmenti devono soddisfare per essere inseriti in S: queste condizioni vengono chiamate *condizioni di fusione*;

2. ricerca di un nuovo insieme $\overline{S}$ di segmenti che sostituiscono S nella rappresentazione dell'oggetto o, con $\overline{S}$ che contiene un numero di segmenti minore di S, ossia $|\overline{S}| < |S|$. Questa fase è solitamente basata su un *algoritmo di fusione* che descrive come ottenere $\overline{S}$ da S: i segmenti che costituiscono l'insieme S vengono rimossi dalla mappa globale, mentre i segmenti che costituiscono l'insieme $\overline{S}$ vengono inseriti in essa. Infine, il processo riparte nuovamente dalla fase 1 fino a che non viene trovato nessun nuovo insieme S.

Indicheremo quindi con il termine di *metodo di fusione* l'intero processo che comprende la verifica delle condizioni di fusione e l'applicazione dell'algoritmo di fusione. I metodi di fusione differiscono nelle loro condizioni di fusione, nei loro algoritmi di fusione e possono essere suddivisi in opportune classi:

- *semplice*: classe a cui appartengono i metodi che hanno condizioni e algoritmi di fusione molto semplici;
- $2 \rightarrow 1$: classe a cui appartengono i metodi che fondono coppie di segmenti in un unico nuovo segmento. Per questi metodi si ha quindi $|S| = 2$ e $|\overline{S}| = 1$;
- $m \rightarrow 1$: classe a cui appartengono i metodi che fondono un insieme di segmenti in un unico nuovo segmento. Per questi metodi si ha quindi $|S| \geq 2$ e $|\overline{S}| = 1$;
- $m \rightarrow n$: classe a cui appartengono i metodi che fondono un insieme di segmenti in un nuovo insieme di segmenti. Per questi metodi si ha quindi $|S| \geq 2$ e $|\overline{S}| \geq 1$.

Alcuni metodi di fusione sono caratterizzati da un proprio metodo di estrazione dei segmenti dai punti delle singole scansioni: in alcuni casi è infatti necessaria una particolare rappresentazione dei segmenti, ad esempio statistica, che un comune metodo di estrazione dei segmenti non è in grado di fornire. In questi casi parleremo di metodi di fusione con un *modello sensoriale basato su punti*: tali metodi sono applicabili solo in combinazione col rispettivo metodo di estrazione dei segmenti e non è quindi possibile trascurare il modo in cui i segmenti sono stati estratti dai punti. Per tutti gli altri metodi di fusione, che sono indipendenti dal metodo di estrazione dei segmenti utilizzato, parleremo invece di *modello sensoriale astratto basato*

su segmenti. In base a questa distinzione, i metodi di fusione con modello sensoriale astratto basato su segmenti hanno una rappresentazione dei segmenti indipendente dal metodo di estrazione dei segmenti utilizzato: per ogni singolo segmento è infatti sufficiente memorizzare i suoi estremi, compito che può essere svolto da un qualsiasi metodo di estrazione dei segmenti; tutti gli altri parametri del segmento necessari per applicare lo specifico metodo di fusione, come ad esempio la lunghezza o l'angolo di inclinazione, possono essere calcolati a partire dagli estremi. Un'ultima importante caratteristica dei metodi di fusione è rappresentata dal fatto che alcuni metodi sono *completamente basati sui segmenti*: dopo l'estrazione dei segmenti dai punti delle singole scansioni è infatti possibile ignorare i punti dai quali i segmenti sono stati ricavati. Per altri metodi è invece necessario mantenere in memoria per ogni segmento i punti da cui è stato ricavato: tali metodi non sono completamente basati sui segmenti.

2.7 Panoramica dei metodi per la fusione dei segmenti

2.7.1 Metodi Basic

Alcuni sistemi, ad esempio quello descritto in [4], implementano dei metodi di fusione molto semplici: la maggior parte di questi metodi fonde tra loro coppie di segmenti consecutivi che rispettano determinate condizioni di fusione. Le condizioni sono semplici e nella maggior parte dei casi valutano la distanza angolare e spaziale, intesa come distanza euclidea tra un estremo del primo segmento e uno del secondo.
Questi metodi di fusione possono essere eseguiti sia on-line che off-line, hanno un modello sensoriale astratto basato su segmenti e hanno quindi una rappresentazione dei segmenti indipendente dal metodo di estrazione utilizzato: per ogni segmento è infatti sufficiente memorizzare i suoi estremi.
Sulla base delle caratteristiche appena elencate, abbiamo implementato una nostra versione di metodo Basic, descritta in modo dettagliato nella Sezione 4.1. Di questo metodo sono disponibili due varianti: una on-line e l'altra off-line.

2.7.2 Metodo Hybrid

Il metodo descritto in [43] si basa sull'idea di fondere tra loro coppie di segmenti ridondanti che rispettano alcune condizioni di fusione espresse relativamente allo spazio di Hough.

Questo metodo di fusione viene eseguito on-line, appartiene alla classe $2 \rightarrow 1$ e ha un modello sensoriale astratto basato su segmenti: è infatti sufficiente un metodo di estrazione dei segmenti dai punti delle singole scansioni che sia in grado di calcolare gli estremi di ciascun segmento. Tutti gli altri parametri del segmento, necessari per applicare questo metodo di fusione (ad eccezione della variabile contatore, che viene comunque posta uguale a uno per ogni nuovo segmento estratto), possono essere infatti calcolati in seguito a partire dagli estremi di esso. Ogni segmento viene pertanto rappresentato dal seguente insieme di parametri:

$$s = \{(x_1, y_1), (x_2, y_2), length, \rho, \theta, count\} \tag{2.16}$$

dove:

- (x_1, y_1) e (x_2, y_2) rappresentano le coordinate dei punti estremi del segmento;
- *length* rappresenta la lunghezza del segmento;
- ρ e θ rappresentano la retta su cui giace il segmento nello spazio di Hough;
- *count* rappresenta il peso di un segmento durante la procedura di fusione.

Trattandosi di un processo on-line, il metodo di fusione viene richiamato a ogni nuova scansione. Ogni segmento contenuto nella nuova scansione viene confrontato con i segmenti presenti nella mappa globale corrente al fine di trovare delle corrispondenze: se ciò si verifica, i parametri del segmento presente nella mappa globale vengono aggiornati per tenere conto dell'avvenuta fusione; se nessuna corrispondenza è trovata, il segmento viene aggiunto alla mappa globale, essendo, molto probabilmente, una nuova caratteristica dell'ambiente rilevata.
Questo metodo di fusione è stato oggetto di implementazione in questa tesi ed è descritto in modo dettagliato nella sezione 4.2.

2.7.3 Metodo Weighted Line Fitting

Il metodo descritto in [44] usa una rappresentazione statistica dei segmenti che tiene in considerazione gli errori di misurazione compiuti dal sensore del robot durante ogni singola scansione. Secondo questa rappresentazione, due segmenti corrispondono quando le loro statistiche soddisfano un test chi-quadrato. I segmenti corrispondenti vengono sostituiti da un nuovo segmento utilizzando un approccio di massima verosimiglianza.

Questo metodo di fusione può essere eseguito sia on-line che off-line, appartiene alla classe $2 \rightarrow 1$ e ha un modello sensoriale astratto basato su punti: necessita infatti di un proprio metodo di estrazione dei segmenti dai punti delle singole scansioni, in modo da effettuare una rappresentazione statistica dei segmenti che tenga in considerazione gli errori compiuti dal sensore del robot. Inoltre questo metodo di fusione non opera propriamente su dei segmenti, ma su delle rette: dapprima l'insieme dei punti di ogni scansione viene diviso in sottoinsiemi di punti quasi collineari usando la trasformazione di Hough; in seguito, per ogni sottoinsieme di punti, viene definita una retta candidata L che li approssima. La retta L viene definita in coordinate polari come l'insieme dei punti normali al vettore (R, α). A ogni retta è associata anche una covarianza P_L, che rappresenta l'incertezza della posizione della retta, definita nel modo seguente:

$$P_L = \begin{bmatrix} E\{\delta R(\delta R)^T\} & E\{\delta R(\delta \alpha)^T\} \\ E\{\delta \alpha(\delta R)^T\} & E\{\delta \alpha(\delta \alpha)^T\} \end{bmatrix} = \begin{bmatrix} P_{RR} & P_{R\alpha} \\ P_{\alpha R} & P_{\alpha\alpha} \end{bmatrix} \tag{2.17}$$

dove δR e $\delta \alpha$ sono gli errori di stima dei parametri della retta. L'obiettivo è di trovare la retta $L(R, \alpha)$ che minimizza gli errori δ_k, con δ_k che rappresenta la distanza tra il k-esimo punto e la retta candidata L. In questo modo ogni sottoinsieme di punti quasi collineari di una scansione viene approssimato con una retta: per ricavare il segmento rappresentativo di tali punti è sufficiente proiettare l'insieme dei punti sulla retta e scegliere come estremi del segmento i due punti proiettati più lontani tra loro. Poiché opera su delle rette, questo metodo non è completamente basato sui segmenti: per ogni retta è infatti necessario mantenere in memoria l'insieme dei punti che approssima in modo da poterne ricavare, anche dopo l'applicazione del metodo di fusione, il segmento rappresentativo.
Il metodo riceve quindi in ingresso due rette, verifica se soddisfano le condizioni di fusione e, in caso affermativo, fonde le due rette in un'unica retta mediante uno specifico algoritmo di fusione. L_1^i e L_2^j sono le rette prese in considerazione da scansioni differenti effettuate nelle posizioni i e j:

$$L_1^i = \begin{bmatrix} R_1^i \\ \alpha_1^i \end{bmatrix} \qquad L_2^j = \begin{bmatrix} R_2^j \\ \alpha_2^j \end{bmatrix} \tag{2.18}$$

Si assume di avere anche una stima della posizione j del robot rispetto alla posizione i, definita come $g_{ij} = [x, y, \gamma]$, e l'incertezza di questa misura P_{ij}. Ai fini di poter applicare il metodo, i parametri e la covarianza di L_2^j devono essere espressi nello stesso sistema di riferimento di L_1^i, ossia relativamente

alla posizione i:

$$L_2^i = \begin{bmatrix} R_2^i \\ \alpha_2^i \end{bmatrix} = \begin{bmatrix} R_2^j + x\cos(\alpha_2^j + \gamma) + y\sin(\alpha_2^j + \gamma) \\ \alpha_2^j + \gamma \end{bmatrix} \tag{2.19}$$

$$P_{L_2}^i = BP_{L_2}^j B^T + KP^{ij}K^T \tag{2.20}$$

dove

$$B = \begin{bmatrix} 1 & -x\sin(\alpha_2^i) + y\cos(\alpha_2^i) \\ 0 & 1 \end{bmatrix} \qquad K = \begin{bmatrix} \cos(\alpha_2^i) & \sin(\alpha_2^i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.21}$$

con P^{ij} definita come la covarianza della trasformazione di posizioni e $P_{L_2}^j$ definita come l'incertezza della retta. Al fine di determinare se le due rette sono sufficientemente simili per essere fuse in un'unica retta, si effettua un test chi-quadrato verificando che la differenza tra le due rette sia inferiore a una determinata soglia ϵ definita dalla combinazione delle incertezze delle due rette. La condizione di fusione è:

$$\chi^2 = (\delta L)^T (P_{L_1}^i + P_{L_2}^i)^{-1} \delta L < \epsilon \tag{2.22}$$

dove:

$$\delta L = \begin{bmatrix} R_1^i - R_2^i \\ \alpha_1^i - \alpha_2^i \end{bmatrix} \tag{2.23}$$

Se questa condizione è soddisfatta, le due rette sono sufficientemente simili per essere fuse. La retta finale può essere stimata usando una funzione di massima verosimiglianza e calcolando le coordinate della retta finale L_m^i e la sua incertezza $P_{L_m}^i$ rispetto alla posizione i nel modo seguente:

$$L_m^i = P_{L_m}^i ((P_{L_1}^i)^{-1} L_1^i + (P_{L_2}^i)^{-1} L_2^i) \tag{2.24}$$

$$P_{L_m}^i = ((P_{L_1}^i)^{-1} + (P_{L_2}^i)^{-1})^{-1} \tag{2.25}$$

Il segmento finale viene poi ricavato proiettando sulla retta finale l'insieme dei punti approssimati dalle due rette e scegliendo come estremi del segmento i due punti proiettati più lontani tra loro.

2.7.4 Metodo Fuzzy Set

Il metodo descritto in [29] utilizza una rappresentazione fuzzy[3] dei segmenti per tenere in considerazione gli errori di misurazione. Sulla base di questa

[3]Un insieme fuzzy è caratterizzato da una funzione di grado di appartenenza, che mappa gli elementi di un universo in un intervallo reale continuo $[0; 1]$.

rappresentazione, il metodo seleziona coppie di segmenti per la fusione e li sostituisce con un terzo segmento.
Questo metodo di fusione viene eseguito off-line: rappresenta infatti il passo finale di un sistema di mappatura che punta alla costruzione di una mappa basata su segmenti per ambienti ortogonali, a partire dalle informazioni raccolte da una squadra di robot. Tali informazioni vengono fornite a un server centrale il quale provvede a ricostruire la mappa dell'ambiente. Il metodo appartiene alla classe $2 \rightarrow 1$ e ha un modello sensoriale basato su punti: necessita infatti di un proprio metodo di estrazione dei segmenti dai punti delle singole scansioni, in modo da tenere in considerazione l'incertezza dei segmenti dovuta agli errori di misurazione compiuti dal robot. Per gestire tale incertezza viene introdotta la nozione di *segmento impreciso*. Dato un segmento rilevato $s = \{(x_1, y_1), (x_2, y_2)\}$, il segmento impreciso $F_s : \mathbb{R} \times \mathbb{R} \rightarrow [0, 1]$ è definito come l'insieme fuzzy che assegna ad ogni coppia di punti (x, y) il suo grado di appartenenza all'ambiente reale. La funzione F_s è uguale a uno per i punti che appartengono a s, mentre decresce linearmente fino ad un valore pari a zero per i punti che si trovano sul bordo del rettangolo di errore associato a s. Questo rettangolo, di cui riportiamo un esempio in Figura 2.6, ha lunghezza $e_1 + s + e_2$, dove e_1 ed e_2 sono gli errori commessi nella direzione dello spostamento, e larghezza $2e$ lungo la direzione ortogonale allo spostamento. Una volta calcolati i segmenti imprecisi questo metodo di fusione non necessita più dei punti dai quali sono stati ricavati: si tratta quindi di un metodo completamente basato sui segmenti. Questo metodo di fusione è in grado di operare in condizioni di incertezza. Siccome l'informazione fornita da ogni singolo robot, relativa alla posizione di muri e ostacoli nell'ambiente, è imprecisa a causa di errori nella misurazione, robot differenti (o anche lo stesso robot che esplora la medesima porzione dell'ambiente più volte) possono rilevare parti dello stesso muro o dello stesso ostacolo con differenti gradi di imprecisione. Il metodo permette di stabilire se due segmenti provenienti da differenti scansioni descrivono la stessa porzione di ambiente: se questo si verifica, il metodo procede alla loro fusione in modo da minimizzare l'informazione memorizzata. Data l'ortogonalità dell'ambiente, i segmenti possono avere solo due orientamenti: verticale e orizzontale. Ad esempio, nel caso di un segmento orizzontale $s = \{(x_1, y), (x_2, y)\}$ con errori e_1 ed e_2 nella direzione della traiettoria ed e nella direzione ortogonale alla traiettoria, la funzione F_s viene determinata dal rettangolo di errore, che giustifica la seguente rappresentazione per i segmenti imprecisi:

$$s = \{(x_1, y_1), (x_2, y_2), e, e_1, e_2\} \tag{2.26}$$

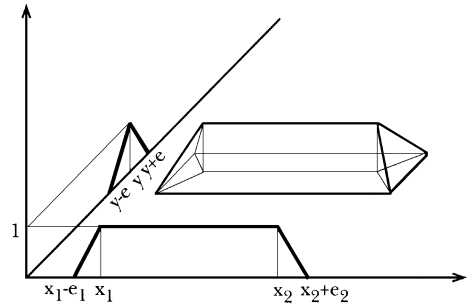

Figura 2.6: Segmento orizzontale impreciso [29].

Il problema principale diventa quindi quello di stabilire quando due segmenti imprecisi rappresentano la stessa porzione di ambiente: questa decisione viene presa sulla base della loro posizione relativa. La posizione relativa viene determinata a partire dai due parametri Δx and Δy, così definiti:

$$F_s = \{(x, y_1), (x, y_2), e, e_1, e_2\}$$

$$F_s^{'} = \{(x^{'}, y_1^{'}), (x^{'}, y_2^{'}), e^{'}, e_1^{'}, e_2^{'}\}$$

$$\Delta x(F_s, F_s^{'}) = |x - x^{'}|$$

$$\Delta y(F_s, F_s^{'}) = \begin{cases} 0 & \text{se } (y_1 - y_2^{'})(y_2 - y_1^{'}) < 0 \\ min(|y_1 - y_2^{'}|, |y_2 - y_1^{'}|) & \text{altrimenti} \end{cases}$$

Il problema della fusione dei segmenti, di cui riportiamo un esempio in Fi-

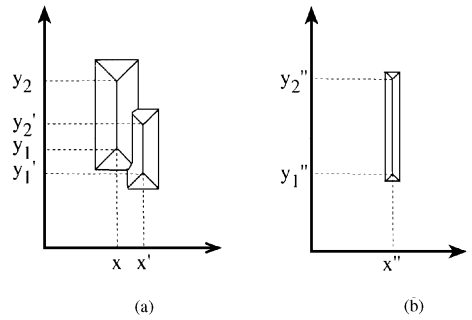

Figura 2.7: I due segmenti verticali in (a) sono fusi nel segmento in (b) [29].

gura 2.7, può quindi essere così definito: data una mappa M, rappresentata come una lista di segmenti ordinati per imprecisione, il metodo calcola una nuova mappa $M^{'}$ in cui alcuni segmenti di M sono stati fusi tra di loro. Per ogni segmento orizzontale (verticale) F_s appartenente alla mappa M, partendo dal segmento più preciso, si calcolano i parametri Δx e Δy rispetto a

tutti gli altri segmenti orizzontali (verticali) in M. Tra i segmenti che hanno Δx e Δy inferiori a un valore di soglia, si sceglie per la fusione la coppia $(F_s, F_s^{'})$ che ha distanza media su entrambi gli assi più piccola rispetto a ogni altra coppia. Successivamente F_s e $F_s^{'}$ sono fusi e sostituiti nella mappa con il nuovo segmento $F_s^{''}$. Il processo continua fino a che non sono possibili ulteriori fusione.

2.7.5 Metodo Viewing Sector

Un altro esempio di metodo che fonde coppie di segmenti è quello descritto in [15].
Questo metodo di fusione viene eseguito on-line, appartiene alla classe $2 \rightarrow 1$ e ha un modello sensoriale basato su punti: necessita infatti di un proprio metodo di estrazione dei segmenti dai punti delle singole scansioni. Ciascun segmento viene rappresentato dal seguente insieme di parametri:

$$s = \{(x_1, y_1), (x_2, y_2), n_k, S_x^k, S_y^k, S_{xx}^k, S_{xy}^k\} \tag{2.27}$$

dove:

- (x_1, y_1) e (x_2, y_2) rappresentano gli estremi del segmento;
- n_k, S_x^k, S_y^k, S_{xx}^k e S_{xy}^k rappresentano cinque parametri di fitting utilizzati per calcolare i parametri della retta su cui giace il segmento nel seguente modo:

$$a_k = \frac{n_k \cdot S_x^k - S_x^k \cdot S_y^k}{n_k \cdot S_{xx}^k - (S_x^k)^2} \tag{2.28}$$

$$b_k = \frac{S_x^k \cdot S_y^k - S_x^k \cdot S_{xy}^k}{n_k \cdot S_{xx}^k - (S_x^k)^2} \tag{2.29}$$

  con $S_x^k = \sum_{i=1}^{n_k} x_i$, $S_y^k = \sum_{i=1}^{n_k} y_i$, $S_{xx}^k = \sum_{i=1}^{n_k} x_i^2$, $S_{xy}^k = \sum_{i=1}^{n_k} x_i \cdot y_i$ e $\{(x_i, y_i)\}, i = 1, \cdots, n_k$ è l'insieme dei punti che ha generato il segmento.

Il metodo, inoltre, è completamente basato sui segmenti.
Questo metodo di fusione si fonda sui concetti di frammentazione di un segmento e di settore di proiezione di un segmento. La frammentazione di un segmento è il processo attraverso cui un segmento viene suddiviso in due o più pezzi, ognuno dei quali può essere considerato come un nuovo segmento della mappa, mentre il segmento originale viene rimosso da essa, come illustrato Figura 2.8. Questo processo permette di trovare quelle che vengono definite corrispondenze parziali fra segmenti. Dato un segmento l_k della mappa globale, il settore di proiezione φ_k è definito come la regione ottenuta

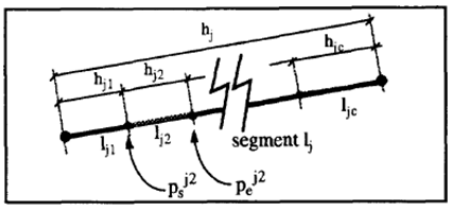

Figura 2.8: Frammentazione di un segmento [15].

da una scansione del sensore tra i punti estremi del segmento (Figura 2.9). Per ogni settore di proiezione i segmenti della mappa locale possono essere

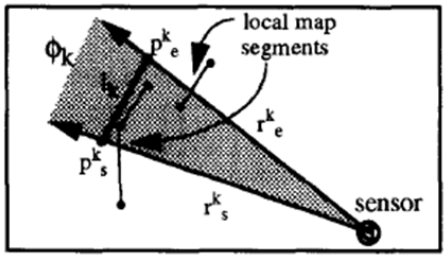

Figura 2.9: Settore di proiezione di un segmento l_k [15].

classificati in quattro modi differenti:

- il segmento locale cade all'esterno del settore di proiezione;
- il segmento locale cade all'interno del settore di proiezione;
- solamente un estremo del segmento locale cade all'interno del settore di proiezione;
- il segmento locale attraversa il settore di proiezione, ma i suoi estremi cadono all'esterno di tale settore.

Solamente i segmenti che cadono all'interno di φ_k vengono considerati come candidati per la corrispondenza. Per la precisione, vengono presi in considerazione i segmenti della mappa locale che cadono almeno in parte all'interno del settore di proiezione φ_k: bisogna quindi procedere a una frammentazione del segmento, in modo da prendere in considerazione solo la parte che cade all'interno del settore di proiezione. Una volta che un sottoinsieme di segmenti della mappa locale viene selezionato, occorre determinare quali di questi segmenti possono essere fusi con l_k. Come riportato in Figura 2.10, esistono quattro possibili situazioni:

- se la distanza di entrambi gli estremi del segmento locale da l_k è minore rispetto a una determinata soglia δ, allora esiste una piena corrispondenza tra il segmento della mappa locale e l_k;

- se la distanza di entrambi gli estremi del segmento locale da l_k è maggiore rispetto a δ e cadono dalla stessa parte di l_k, allora non esiste alcuna corrispondenza;

- se solo un estremo del segmento locale è a distanza inferiore a δ da l_k, allora il segmento della mappa locale è frammentato in due parti, una delle quali ha una corrispondenza con l_k e l'altra no;

- se entrambi gli estremi del segmento locale cadono su lati diversi di l_k e a una distanza maggiore di δ da l_k, il segmento locale è frammentato in tre parti, di cui solamente una parte ha una corrispondenza con l_k.

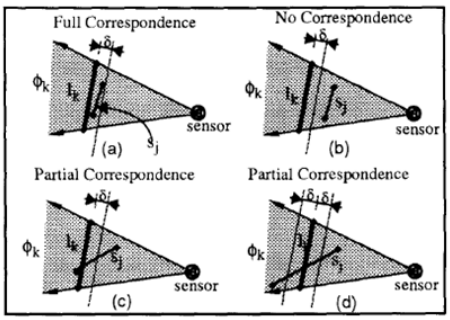

Figura 2.10: Differenti situazioni da considerare quando si analizza la corrispondenza [15].

Negli ultimi due casi si parla di *corrispondenze parziali*. Per aggiornare il segmento l_k della mappa globale, è prima di tutto necessario frammentarlo, proiettando tutti i segmenti della mappa locale che cadono all'interno del settore di proiezione δ_k, come mostrato in Figura 2.11. Ogni nuovo segmento ottenuto a partire da l_{kj} viene modificato sulla base di quattro tipologie di segmenti:

- *occluso*. Tutte le volte che un segmento della mappa locale s_j "blocca" l_k (senza corrispondenza), non esiste alcuna informazione aggiuntiva circa l_{kj} e, di conseguenza, il segmento non viene modificato: s_j potrebbe rappresentare una nuova caratteristica dell'ambiente o potrebbe corrispondere a un altro segmento della mappa globale;

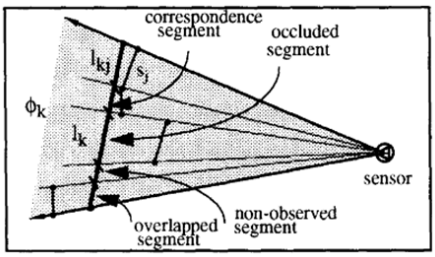

Figura 2.11: La frammentazione di un segmento l_k della mappa globale [15].

- *sovrapposto.* Tutte le volte che l_k "blocca" un segmento della mappa locale s_j ci si riferisce a quest'ultimo come segmento sovrapposto: questo segmento viene interpretato come la scomparsa della caratteristica che rappresenta e di conseguenza viene rimosso dalla mappa globale;
- *non osservato.* A mano a mano che i segmenti della mappa locale che cadono all'interno del settore di proiezione vengono proiettati su l_k, potrebbe succedere che nessuno di loro cada su un particolare pezzo di l_k: ci si riferisce a questo segmento frammentato come a un segmento non osservato. Un segmento non osservato non viene eliminato dalla mappa, in quanto non è causa di particolari conflitti con i segmenti della mappa locale;
- *corrispondente.* Supponiamo che (l_{kj}, s_j) sia una coppia corrispondente, con l_{kj} che rappresenta il segmento ottenuto per frammentazione da l_k mediante la proiezione del segmento della mappa locale s_j. In questo caso l_{kj} e s_j rappresentano la stessa porzione dell'ambiente e, quindi, possono essere fusi insieme. I parametri del segmento risultante vengono calcolati a partire dai cinque parametri di fitting dei due segmenti di partenza.

2.7.6 Metodo Fuzzy Clustering

Il metodo descritto in [33] è basato sull'idea di costruire cluster di segmenti paralleli che giacciono sullo stesso iperpiano e che sono vicini l'uno all'altro. I segmenti appartenenti allo stesso cluster vengono poi fusi tra loro mediante un algoritmo di fusione piuttosto complesso basato sull'utilizzo di una griglia.
Questo metodo di fusione viene eseguito on-line, appartiene alla classe $m \rightarrow 1$ e ha un modello sensoriale basato su punti: necessita infatti di un proprio metodo di estrazione dei segmenti dai punti delle singole scansioni. Ciascun

segmento viene rappresentato dal seguente insieme di parametri:

$$s = \{\phi, \alpha, (x_1, y_1), (x_2, y_2), (x_c, y_c)\} \tag{2.30}$$

dove:

- ϕ rappresenta l'unità autovettoriale del segmento ottenuta mediante algoritmo EAFC (Enhanced Adaptive Fuzzy Clustering) con approccio NC (Noise Clustering);
- α rappresenta il coefficiente di unione;
- (x_1, y_1) e (x_2, y_2) rappresentano gli estremi del segmento ottenuti anch'essi mediante algoritmo EAFC con approccio NC;
- (x_c, y_c) rappresenta il centro del cluster del segmento.

Questo metodo, inoltre, non è completamente basato sui segmenti, in quanto per ogni segmento è necessario memorizzare i punti della scansione che approssima.
Il metodo è sostanzialmente suddiviso in sei fasi:

1. fissa il numero dei cluster e le condizioni di terminazione per ogni cella;
2. ottiene le misurazioni sensoriali dall'ambiente e le trasforma in un sistema di coordinate globali;
3. raggruppa le misurazioni sensoriali nelle relative celle $C_{L_{N\_X}^{i-1,i}}^{L_{N\_Y}^{i-1,i}}$ secondo le seguenti equazioni:

$$L_{N\_X} = \frac{Max\_Length}{N} \tag{2.31}$$

$$L_{N\_Y} = \frac{Max\_Length}{N} \tag{2.32}$$

   dove:

   - L_{N_X} rappresenta la lunghezza della cella lungo l'asse x;
   - L_{N_Y} rappresenta la lunghezza della cella lungo l'asse y;
   - Max_Length rappresenta la massima lunghezza dello spazio di lavoro;
   - N rappresenta il numero di divisioni dello spazio di lavoro.

4. estrae i segmenti in ogni cella mediante l'algoritmo EAFC con approccio NC, e li memorizza secondo la rappresentazione descritta in precedenza;

5. esegue un ''raggruppamento locale". Combina i segmenti di cluster simili in ogni cella mediante una tecnica di fusione tra segmenti compatibili. Memorizza tutte le celle aggiornate nel buffer per il ''raggruppamento globale". Ripete le fasi da due a cinque per il prossimo intervallo di campionamento. Viene utilizzato T_{local} per identificare l'intervallo di campionamento dei sensori;

6. esegue un ''raggruppamento globale". Viene utilizzato T_{global} (con $T_{global} > T_{local}$) per identificare l'intervallo di campionamento del ''raggruppamento globale"'. Le celle aggiornate (ossia quelle memorizzate nel buffer durante la fase cinque) vengono usate come centro per effettuare la fusione nelle celle circostanti. I segmenti fusi e le posizioni delle celle vengono memorizzate nel ''buffer globale".

Una volta che il robot termina la navigazione nello spazio di lavoro, l'informazione della mappa finale è selezionata da ogni cella e dal ''buffer globale". Se le celle sono sovrapposte, vengono selezionati i segmenti nel ''buffer globale" invece che quelli nella cella.

2.7.7 Metodo Segment Similarity Clustering

Il metodo descritto in [42] raggruppa i segmenti sulla base di una particolare misura di distanza ed estrae per ogni gruppo un segmento rappresentante. Il segmento rappresentante è estratto calcolando iterativamente il segmento medio fra le coppie di segmenti presenti nel gruppo.
Questo metodo di fusione viene eseguito off-line, appartiene alla classe $2 \rightarrow 1$ e ha un modello sensoriale astratto basato su segmenti: è infatti sufficiente un metodo di estrazione dei segmenti dai punti delle singole scansioni che sia in grado di calcolare gli estremi di ciascun segmento. Tutti gli altri parametri del segmento, necessari per applicare questo metodo di fusione (ad eccezione della variabile peso, che viene comunque posta inizialmente uguale a uno per ogni nuovo segmento estratto), possono essere infatti calcolati in seguito a partire dagli estremi di esso. Il metodo è inoltre completamente basato sui segmenti.
Il metodo dipende fortemente dalla misura di distanza fra segmenti utilizzata. L'idea alla base della misura di distanza $(m, a, w_a) = D(u, v, w_u, w_v)$ fra due segmenti u e v è quella di fonderli insieme in un segmento intermedio a avente peso w_a. La distanza risultante (Figura 2.12) è il costo della fusione m, che dipende:

- dalla distanza angolare $D_{ang}(u)$ tra u e a (analogamente dalla distanza angolare fra v ed a);

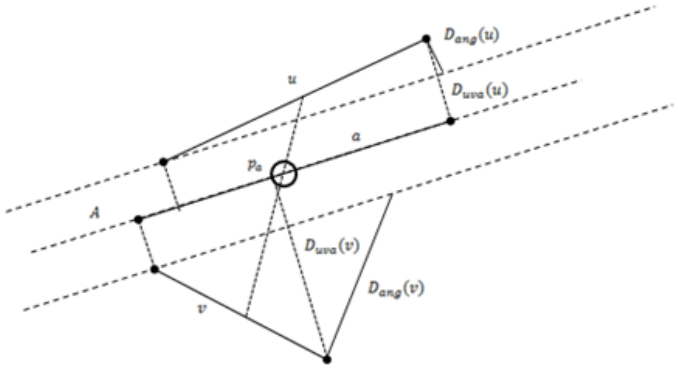

Figura 2.12: Illustrazione della misura di distanza [42].

- dalla distanza spaziale $D_{uva}(u)$ tra u e a (analogamente dalla distanza angolare fra v ed a);
- dalla distanza spaziale D_{uv} tra u e v.

I primi due componenti penalizzano la non collinearità fra i segmenti, mentre l'ultima parte penalizza la distanza spaziale. Dati due segmenti u, v e i rispettivi pesi w_u, w_v, la misura di similarità calcola la direzione d_a del segmento intermedio a come la media pesata delle direzioni di u e di v. Il punto p_a, calcolato come il centro pesato tra i centri di u e di v, identifica univocamente la retta A che contiene il segmento a. Gli estremi di a sono poi calcolati a partire dagli estremi di u e v proiettati su A. Una volta ottenuto il segmento intermedio a è possibile calcolare m, che rappresenta il costo della fusione di u e v nel segmento a, nel seguente modo:

$$m = 0.66 * D_{ang} + 0.09 * D_{uva} + 0.25 * D_{uv} \tag{2.33}$$

dove:

- $$D_{ang} = \frac{w_u * D_{ang}(u) + w_v * D_{ang}(v)}{w_u + w_v} \tag{2.34}$$

  con $D_{ang}(u)$ ($D_{ang}(v)$) uguale alla distanza ortogonale a u (v) tra gli estremi di u (v) e una retta parallela ad A passante per un estremo di u (v);

- $$D_{uva} = max(D_{uva}(u), D_{uva}(v)) \tag{2.35}$$

con $D_{uva}(u)$ ($D_{uva}(v)$)) uguale alla massima distanza tra gli estremi di u (v) e le loro proiezioni su A;

- D_{uv} è la minima distanza tra gli estremi di u e v e le loro rispettive proiezioni su v e u.

2.7.8 Metodo Uncertain Parameters

Il metodo di fusione descritto in [31] viene eseguito on-line, appartiene alla classe $2 \rightarrow 1$ e ha un modello sensoriale basato su punti. Come illustrato in Figura 2.13, ogni segmento viene rappresentato dal seguente insieme di parametri:

$$s = \{\theta, \rho, (x_1, y_1), (x_2, y_2), \sigma_\theta, \sigma_\rho\} \tag{2.36}$$

dove:

- θ e ρ rappresentano la retta su cui giace il segmento nello spazio di Hough;
- (x_1, y_1) e (x_2, y_2) rappresentano gli estremi del segmento;
- σ_θ e σ_ρ rappresentano le incertezze di θ e ρ, calcolate come:

$$\sigma_\theta = \arcsin(\frac{\epsilon}{l/2}) \qquad \sigma_\rho = \epsilon \tag{2.37}$$

  con l che rappresenta la lunghezza del segmento e ϵ la massima distanza tra il segmento e i punti che lo generano.

Questo metodo è inoltre completamente basato sui segmenti.
Per decidere se due segmenti s e $s^{'}$ rappresentano la stessa porzione dell'ambiente e possono quindi essere fusi tra di loro, devono essere soddisfatte tre condizioni di fusione:

1. la differenza angolare tra s e $s^{'}$ deve essere inferiore a una soglia T_θ;
2. la distanza spaziale tra s e $s^{'}$ deve essere inferiore a una soglia T_D;
3. le proiezioni P_s e $P_{s'}$ di s e $s^{'}$ sulla loro bisettrice (ad esempio la retta che divide a metà l'angolo formato da s e $s^{'}$) si devono sovrapporre.

Se $s = \{\theta, \rho, (x_1, y_1), (x_2, y_2), \sigma_\theta, \sigma_\rho\}$ e $s^{'} = \{\theta^{'}, \rho^{'}, (x_1^{'}, y_1^{'}), (x_2^{'}, y_2^{'}), \sigma_\theta^{'}, \sigma_\rho^{'}\}$ soddisfano le condizioni precedenti, allora possono essere fusi tra loro. I parametri del segmento risultante $s^f = \{\theta^f, \rho^f, (x_1^f, y_1^f), (x_2^f, y_2^f), \sigma_\theta^f, \sigma_\rho^f\}$ sono calcolati nel modo seguente:

$$\theta^f = \frac{\omega_\theta \cdot \theta + \omega_\theta^{'} \cdot \theta^{'}}{\omega_\theta + \omega_\theta^{'}} \qquad \rho^f = \frac{\omega_\rho \cdot \rho + \omega_\rho^{'} \cdot \rho^{'}}{\omega_\rho + \omega_\rho^{'}} \tag{2.38}$$

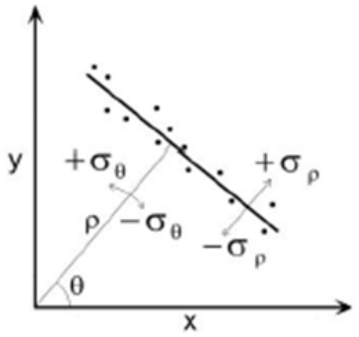

Figura 2.13: I parametri di un segmento [31].

dove i pesi ω_θ e ω_ρ sono definiti come:

$$\omega_\theta = \frac{l}{l+l'} \cdot \frac{1}{\sigma_\theta^2} \qquad \omega_\rho = \frac{l}{l+l'} \cdot \frac{1}{\sigma_\rho^2}$$

con l e $l^{'}$ che indicano rispettivamente la lunghezza dei segmenti s e $s^{'}$. I pesi dei segmenti sono direttamente proporzionali alla loro lunghezza e inversamente proporzionali alla loro incertezza. Di conseguenza i segmenti più lunghi sono considerati più affidabili. Gli estremi del nuovo segmento (x_1^f, y_1^f) e (x_2^f, y_2^f) sono calcolati proiettando gli estremi di s e $s^{'}$ sulla retta con parametri θ^f e ρ^f e selezionando i punti proiettati più lontani tra loro. I due parametri di incertezza σ_θ^f e σ_ρ^f del nuovo segmento sono calcolati come riportato in Figura 2.14. Al fine di ridurre il più possibile il numero di segmenti nella mappa risultante, la precedente procedura di fusione può essere applicata a un insieme di segmenti. Più precisamente, data la mappa corrente e una nuova scansione, per ogni segmento s della scansione viene costruito un insieme C_s che contiene tutti i segmenti della mappa che possono essere fusi con s. I segmenti presenti in $C_s \cup \{s\}$ vengono approssimati con un nuovo segmento, che viene inserito nella mappa risultante (insieme ai segmenti nella mappa corrente che non corrispondono ad alcun segmento della scansione).

2.7.9 Metodo Fusion

Il metodo descritto in [48] è basato sull'idea di determinare un insieme di segmenti ridondanti che rappresentano lo stesso oggetto nell'ambiente e di approssimarli con un unico segmento.
Questo metodo di fusione può essere eseguito sia on-line che off-line, ap-

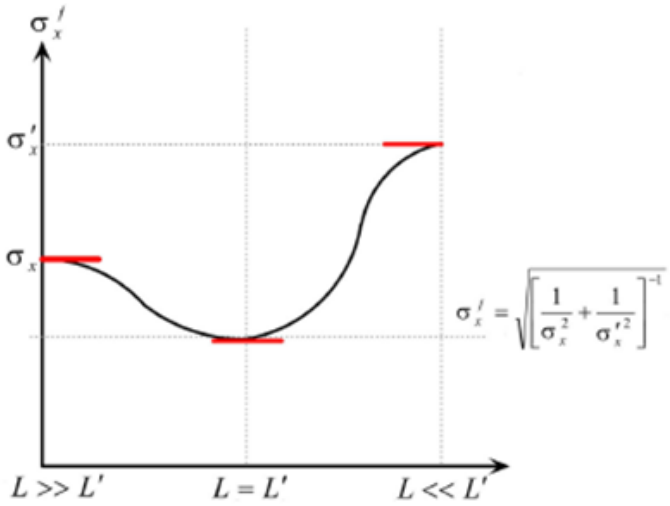

Figura 2.14: La funzione σ_x^f [31].

partiene alla classe $m \to 1$ e ha un modello sensoriale astratto basato su segmenti: è infatti sufficiente un metodo di estrazione dei segmenti dai punti delle singole scansioni che sia in grado di calcolare gli estremi di ciascun segmento. Tutti gli altri parametri del segmento necessari per applicare questo metodo di fusione, come ad esempio la lunghezza, possono essere infatti calcolati in seguito a partire dagli estremi di esso. Questo metodo è inoltre completamente basato sui segmenti.
Il metodo ricerca in maniera iterativa all'interno della mappa il "migliore" insieme di segmenti ridondanti che possono essere approssimati con un unico segmento. Il "migliore" insieme è quello che contiene il segmento più lungo all'interno della mappa corrente. Ad ogni iterazione, l'insieme dei segmenti ridondanti che contengono il segmento più lungo viene rimosso della mappa e approssimato con un unico nuovo segmento che viene aggiunto alla nuova mappa. Il processo termina quando nella vecchia mappa non è più presente alcun segmento.
Questo metodo di fusione è stato oggetto di implementazione in questa tesi ed è descritto in modo dettagliato nella Sezione 4.4.

2.7.10 Metodo Polyline

Il metodo di fusione descritto in [30] è basato sull'idea di fondere un insieme di segmenti in una polilinea, ossia in un insieme ordinato di segmenti orientati consecutivi (cioè tali che il secondo estremo di un segmento coincide con

il primo estremo del successivo) e non adiacenti (cioè tali che un segmento ed il suo successivo non appartengono alla stessa retta).
Questo metodo di fusione viene eseguito on-line, appartiene alla classe $m \rightarrow 1$ e ha un modello sensoriale astratto basato su segmenti.
In base a questo metodo di fusione due segmenti corrispondono quando hanno un valore positivo di corrispondenza. Il valore di corrispondenza tra due segmenti u e v (rispettivamente appartenenti alla nuova scansione S e alla mappa corrente M) è calcolato nel seguente modo. Dapprima il segmento v viene proiettato sulla retta che supporta u, in modo da ottenere il segmento proiettato v_p. Successivamente viene calcolata la lunghezza l_c delle parti comuni a v_p e u. In modo analogo si proietta u su v, ottenendo $l_c^{'}$. Il valore di corrispondenza è calcolato come la media tra l_c e $l_c^{'}$. Quando u e v non si intersecano, il valore di corrispondenza è moltiplicato per $0.95^{D(u,v)/T_p}$ in modo da penalizzare la corrispondenza tra segmenti che sono lontani fra di loro. T_p è la soglia sotto cui due punti sono considerati coincidenti, 0.95 è una costante derivata da prove sperimentali e $D(u,v)$ è la distanza fra segmenti calcolata nel seguente modo:

$$D(u,v) = min(d(u,v), d(v,u)) \tag{2.39}$$

dove $d(u,v) = max_{p' \in v}(min_{p \in u}\|p - p'\|)$ e $\|.\|$ è la distanza Euclidea.
Quando due segmenti hanno un valore positivo di corrispondenza, allora si ritiene che essi rappresentino la stessa porzione di ambiente. Il valore di corrispondenza di u, v è zero quando sono lontani oppure quando si intersecano e sono più lunghi di una soglia. L'algoritmo di fusione di questo metodo è basato sulla nozione di *catena di corrispondenze*. Una catena di corrispondenze è un insieme di coppie di segmenti che hanno un valore di corrispondenza positivo. Formalmente, una catena di corrispondenze è un insieme $C = \{(u,v)|u \in S, v \in M$hanno un valore positivo di corrispondenza$\}$ tale che se $(u,v) \in C$ allora tutti i segmenti che hanno un valore di corrispondenza positivo con u o con v appartengono a C. Ogni catena di corrispondenze viene fusa in una singola polilinea. Questa polilinea viene generata iterativamente costruendo una sequenza di polilinee approssimanti $P_0, P_1, ...$ che convergono alla polilinea P, la quale approssima adeguatamente (e sostituisce nella mappa corrente) i segmenti corrispondenti della catena. In Figura 2.15 è rappresentato il processo di costruzione della polilinea. La polilinea P_0 è composta da un singolo segmento che connette la coppia di punti più lontani (estremi di segmenti) presenti nella catena di corrispondenze. Data la polilinea P_{n-1} e indicato con s il segmento nella catena che si trova alla massima distanza dal suo più vicino segmento $\overline{s}$ in P_{n-1}, se la distanza $D(s,\overline{s})$ è inferiore ad una determinata soglia di errore, allora P_{n-1} è l'ap-

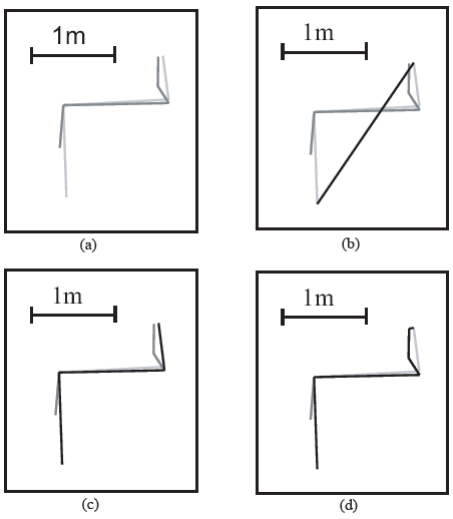

Figura 2.15: Costruzione iterativa di una polilinea per una catena di corrispondenze [30].

prossimazione finale di P. Altrimenti, s è inserito in P_{n-1} per sostituire $\overline{s}$ e s è connesso ai due segmenti più vicini presenti in P_{n-1} per ottenere P_n.

2.7.11 Metodo Mean Shift Clustering

Il metodo descritto in [45] è basato su un classico approccio di clustering, chiamato *Mean Shift Clustering*, il quale sposta simultaneamente e iterativamente tutti i segmenti in uno spazio dei parametri fino a farli convergere verso certi centri di densità. I segmenti che finiscono nello stesso centro di densità appartengono allo stesso cluster e possono quindi essere elaborati in modo da determinare un singolo segmento rappresentativo di essi.
Questo metodo di fusione viene eseguito off-line dopo l'allineamento di tutte le scansioni rilevate dal robot: dalla mappa globale iniziale, costituita da m segmenti, vengono estratti n segmenti (con $n < m$). Il metodo appartiene quindi alla classe $m \to n$, in quanto non esistono delle vere e proprie condizioni di fusione: i segmenti che costituiscono la mappa globale iniziale vengono infatti clusterizzati e conseguentemente fusi tra loro mediante un unico processo globale. Il metodo ha un modello sensoriale astratto basato su segmenti.
Il metodo è sostanzialmente suddiviso in due fasi, le quali utilizzano entrambe il Mean Shift Clustering. La prima fase individua cluster C_α costituiti da segmenti aventi una direzione simile α e caratterizzati da una reciproca

vicinanza spaziale. La seconda fase individua i sotto-cluster di ogni cluster C_α, raggruppando segmenti collineari vicini tra loro. La fase finale è una semplice procedura geometrica: per ogni sotto-cluster viene determinato un segmento rappresentativo di tutti i segmenti appartenenti a quello specifico sotto-cluster.
Il vantaggio di questo metodo è che necessita di un solo parametro, che definisce la scala della mappa o, più intuitivamente, la minima dimensione di un dettaglio significato: non vanno specificati né il numero di cluster, né i passi del gradiente. In realtà ci sono anche altri tre parametri, corrispondenti alla bandwidth h e alle larghezze dei nuclei σ_x e σ_y: tali parametri possono essere tuttavia derivati dal parametro di scala e risultano costanti in tutti gli esperimenti. Lo svantaggio di questo metodo è che il Mean Shift è molto sensibile alla scelta del nucleo, che determina il grado di operazioni locali rispetto alle operazioni globali.
Questo metodo di fusione è stato oggetto di implementazione in questa tesi ed è descritto in modo dettagliato nella Sezione 4.3.

2.8 Riepilogo e scelta dei metodi

In Figura 2.16 è riportata una tabella riassuntiva sui metodi di riduzione di mappe basate su segmenti che sono stati analizzati. Quattro di questi sono stati scelti per le successive fasi di analisi e valutazione sperimentale. Uno per ogni classe di riduzione. Si tratta di metodi caratterizzati da un modello sensoriale astratto basato sui segmenti, quindi non dipendenti dal metodo di estrazione dei segmenti utilizzato, e completamente basati sui segmenti. La seconda caratteristica è motivata dal fatto di voler analizzare metodi che fossero puramente basati sui segmenti e che quindi non utilizzassero altre informazioni nelle procedura di fusione. Infine i metodi sono stati scelti tenendo conto della tipologia del processo, al fine di poter confrontare vantaggi e svantaggi di un'esecuzione on-line/off-line dei metodi di riduzione.

Metodo	Classe	Processo	Modello Astratto del Sensore	Rappresentazione dei Segmenti	Basato sui Segmenti	Data Set
[4]	Semplice	Off-line/On-line	Segmenti	$s = \{(x_{start}, y_{start}), (x_{end}, y_{end})\}$	Si	Non disponibili
[43]	$2 \rightarrow 1$	On-line	Segmenti	$s = \{\rho, \theta, l, (x_{start}, y_{start}), (x_{end}, y_{end}), count\}$	Si	Non disponibili
[44]	$2 \rightarrow 1$	Off-line/On-line	Punti	$s = \{(R, \alpha), P_L\}$	No	Non disponibili
[29]	$2 \rightarrow 1$	Off-line	Punti	$s = \{(x_{start}, y_{start}), (x_{end}, y_{end}), e, e1, e2\}$	Si	Non disponibili
[15]	$2 \rightarrow 1$	On-line	Punti	$s = \{(x_{start}, y_{start}), (x_{end}, y_{end}), n_k, S^k_x, S^k_y, S^k_{xx}, S^k_{xy}\}$	Si	Non disponibili
[33]	$2 \rightarrow 1$	On-line	Punti	$s = \{\phi, \alpha, (x_{start}, y_{start}), (x_{end}, y_{end}), (x_v, y_v)\}$	No	Non disponibili
[42]	$2 \rightarrow 1$	Off-line	Segmenti	$s = \{(x_{start}, y_{start}), (x_{end}, y_{end}), w\}$	Si	Disponibili
[31]	$2 \rightarrow 1$	On-line	Punti	$s = \{\rho, \theta, (x_{start}, y_{start}), (x_{end}, y_{end}), \sigma_\rho, \sigma_\theta\}$	Si	Non disponibili
[48]	$m \rightarrow 1$	Off-line/On-line	Segmenti	$s = \{(x_{start}, y_{start}), (x_{end}, y_{end})\}$	Si	Disponibili
[30]	$m \rightarrow n$	On-line	Segmenti	$s = \{(x_{start}, y_{start}), (x_{end}, y_{end})\}$	Si	Disponibili
[45]	$m \rightarrow n$	Off-line	Segmenti	$s = \{(x_{start}, y_{start}), (x_{end}, y_{end})\}$	Si	Disponibili

Figura 2.16: Tabella riassuntiva dei metodi di fusione. In evidenza i metodi scelti per le prove sperimentali.

Capitolo 3

Costruzione di mappe basate su segmenti

3.1 L'esplorazione dell'ambiente e la costruzione di data set

Nella sezione 2.4 è descritto il processo attraverso cui un robot mobile costruisce una mappa bidimensionale di ambienti indoor basata su segmenti. Durante tale processo il robot fa affidamento a sensori esterocettivi[1]. Tra questi uno dei più utilizzati è il laser scanner bidimensionale, che consente il rilevamento degli oggetti, presenti nell'ambiente circostante il robot, a scale e risoluzioni differenti. Lo strumento permette, a ogni scansione, di acquisire un insieme di punti sparsi nello spazio in modo più o meno regolare: questo insieme viene chiamato *nuvola di punti*. Solitamente una nuvola di punti consiste in una sequenza ordinata (verso antiorario) di 181 misurazioni, eseguite con passo 1° su di un arco di 180°, della distanza a cui si trovano gli ostacoli lungo la direzione del sensore. Le nuvole di punti, acquisite con un sensore laser, sono quindi un insieme di punti espressi in coordinate polari, con origine nel sistema di riferimento del sensore stesso. A partire da tali insiemi di punti vengono estratti i segmenti. Nella prossima sezione descriveremo il metodo di estrazione dei segmenti implementato.

[1] Un sensore si dice esterocettivo quando è in grado di percepire grandezze fisiche esterne al robot.

3.2 Metodo di estrazione dei segmenti

Data una scansione S, il metodo di estrazione si compone di tre passi:

1. il primo passo prevede l'estrazione di un insieme di *cluster*. Un cluster è un sottoinsieme di punti consecutivi $\{p_i, p_{i+1}, \cdots, p_{i+k}\}$ della scansione S, tale che la distanza tra un punto e il successivo è inferiore a una soglia *delta*. Il risultato di questo passo è un insieme di cluster, ciascuno formato da un numero $n \geq 1$ di punti.

2. il secondo passo prevede, per ogni cluster, il calcolo dei parametri della retta, espressa nella forma $ax + by + c = 0$, che meglio approssima l'insieme di punti da cui è composto il cluster. Questo calcolo dei parametri viene eseguito utilizzando il *metodo di regressione ai minimi quadrati*. Il metodo di regressione ai minimi quadrati è una tecnica di ottimizzazione che permette di trovare una funzione, detta curva di regressione, che si avvicina il più possibile a un insieme di dati. Nel nostro caso la curva di regressione è la retta avente parametri a, b, c e l'insieme di dati che deve approssimare è l'insieme dei punti che compongono il cluster. A partire da un insieme di punti $(x, y) \in \mathbb{R}^2$, i parametri della retta vengono così calcolati:

$$a = \sum_{i=1}^{N_p} x_i \sum_{i=1}^{N_p} y_i^2 - \sum_{i=1}^{N_p} y_i \sum_{i=1}^{N_p} x_i y_i \tag{3.1}$$

$$b = \sum_{i=1}^{N_p} y_i \sum_{i=1}^{N_p} x_i^2 - \sum_{i=1}^{N_p} x_i \sum_{i=1}^{N_p} x_i y_i \tag{3.2}$$

$$c = (\sum_{i=1}^{N_p} x_i y_i)^2 - \sum_{i=1}^{N_p} x_i^2 \sum_{i=1}^{N_p} y_i^2 \tag{3.3}$$

3. il terzo passo prevede, a partire da un insieme di punti e dalla retta di equazione $ax + by + c = 0$ che meglio li approssima, l'estrazione del segmento rappresentativo di tali punti. Per estrarre tale segmento è sufficiente proiettare i punti dell'insieme sulla retta che li approssima e scegliere come punti estremi del segmento i punti proiettati più distanti fra di loro.

In questa sezione abbiamo descritto un metodo per l'estrazione dei segmenti a partire da una scansione costituita da punti, al fine di ottenere una mappa bidimensionale basata su segmenti. Tuttavia, come abbiamo già spiegato nella Sezione 2.5, le informazioni fornite dai sensori del robot non sono

affidabili a causa degli errori odometrici che si accumulano durante la navigazione e quindi, per una corretta ricostruzione della mappa, sono necessari dei metodi di allineamento che, dopo l'acquisizione di una nuova scansione, siano in grado di correggere la stima della posizione del robot.

3.3 Metodi di allineamento

Nel nostro lavoro di ricerca abbiamo preso in considerazione tre diversi metodi di allineamento. Il nostro obiettivo era infatti quello di dimostrare quale fosse il metodo di fusione dei segmenti migliore, indipendentemente dal metodo di allineamento utilizzato. I tre metodi che abbiamo considerato sono:

- metodo di allineamento basato sulle corrispondenze tra angoli, già esistente e implementato, e importato nel nostro lavoro di ricerca;
- metodo di allineamento basato sulla struttura portante della mappa, sviluppato e implementato da noi;
- metodo di allineamento basato sul ground truth.

Nelle prossime sezioni descriveremo il funzionamento di questi tre differenti metodi.

3.3.1 Metodo di allineamento basato sulle corrispondenze tra angoli

Come abbiamo già anticipato nella Sezione 2.5.1, l'idea di base del metodo descritto in [30] è quella di trovare corrispondenze fra angoli simili di due mappe parziali, dove una mappa parziale rappresenta l'integrazione di due scansioni, di una scansione e di una mappa parziale oppure di due mappe parziali. Il metodo integra due mappe parziali S_1 e S_2 in una mappa $S_{1,2}$ secondo tre passi:

1. *ricerca delle possibili trasformazioni da S_2 in S_1.* Questo passo prima calcola gli angoli tra coppie di segmenti in S_1 e tra coppie di segmenti in S_2, poi trova le possibili trasformazioni che sovrappongono almeno un angolo α_2 di S_2 a un angolo simile α_1 di S_1. Essendo un problema computazionalmente pesante, sono state sviluppate tre differenti euristiche per ridurne la complessità:
    - si considerano solo gli angoli tra coppie consecutive di segmenti in una scansione;

- si considerano solo gli angoli tra coppie di segmenti scelti in modo casuale all'interno di una scansione;
- si considerano gli angoli tra coppie di segmenti perpendicolari all'interno di una scansione.

2. *valutazione delle trasformazioni.* Questo passo valuta ogni singola trasformazione trovata al passo precedente in modo da individuarne la migliore. Al fine di misurare la qualità di una trasformazione t, la scansione S_2 viene portata nel sistema di riferimento di S_1 sulla base della trasformazione t, ottenendo S_2^t. Viene poi calcolata la lunghezza approssimativa dei segmenti di S_1 che corrispondono a dei segmenti di S_2^t. Il valore della trasformazione è uguale alla lunghezza dei segmenti corrispondenti che la trasformazione produce: più precisamente, il valore della trasformazione è la somma di tutti i valori di corrispondenza calcolati per ogni coppia di segmenti $s_1 \in S_1$ e $s_2^t \in S_2^t$;

3. *applicazione della trasformazione e fusione delle scansioni.* Una volta trovata la trasformazione migliore $\bar{t}$, la scansione S_2 viene espressa nel sistema di riferimento di S_1 utilizzando $\bar{t}$ e ottenendo $S_2^{\bar{t}}$. A causa degli errori che si possono verificare durante la procedura di allineamento, potrebbe accadere che le scansioni non siano allineate correttamente. Al fine di produrre la versione finale della mappa $S_{1,2}$, ogni catena di corrispondenze viene sostituita da una polilinea, la quale viene poi aggiunta all'insieme di segmenti componenti la mappa finale. Una catena di corrispondenze, relativa alla trasformazione $\bar{t}$ per la coppia di scansioni S_1 e $S_2^{\bar{t}}$, è l'insieme $C = \{< s_1, s_2^{\bar{t}} > | s_1 \in S_1, s_2^{\bar{t}} \in S_2^{\bar{t}}$ con un valore di corrispondenza positivo per $\bar{t}\}$ algebricamente chiuso rispetto alla relazione di appartenenza alla retta del segmento.

3.3.2 Metodo di allineamento basato sulla struttura portante della mappa

Il metodo di allineamento descritto nella sezione precedente non funziona sempre: tale metodo può portare infatti a un errato allineamento della mappa nel caso in cui venga trovata una corrispondenza tra angoli sbagliata. Anche tutti gli altri metodi descritti nella Sezione 2.5.1 presentano alcuni problemi:

- gli approcci descritti in [31] e in [41] operano solo su una coppia di scansioni (di cui una nuova e l'altra appartenente alla mappa globale) e ignorano tutte le altre scansioni appartenenti alla mappa globale.

Questo comporta che la posizione corretta del robot nella nuova scansione è univocamente determinata dalla scansione rispetto alla quale è stata allineata. Quindi se vengono fatti errori nel calcolo della posizione corretta, questi errori si propagano di scansione in scansione, determinando un errato allineamento della mappa;

- l'approccio descritto in [42] presenta un metodo di allineamento globale, dove le posizioni corrette del robot nelle singole scansioni vengono calcolate contemporaneamente. Tale metodo è quindi esente dai problemi degli altri metodi descritti in precedenza, tuttavia può essere applicato solo nel caso in cui gli errori dovuti all'odometria siano piuttosto piccoli.

Dalle considerazioni appena fatte emerge che quello dell'allineamento delle scansioni risulta un problema tuttora aperto: non esiste un metodo di allineamento che sia in grado di funzionare in ogni situazione.
Abbiamo così implementato un nostro metodo di allineamento, partendo da alcune osservazioni:

- gli errori odometrici compiuti dal robot sono soprattutto rotazionali. Dall'analisi di diversi data set (insiemi di segmenti che descrivono la porzione di ambiente esplorata dal robot) è infatti emerso che finché il robot si muove senza effettuare rotazioni, gli errori compiuti nella stima della posizione sono trascurabili;

- in ambienti indoor ortogonali i corridoi caratterizzati da una direzione simile sono ragionevolmente paralleli tra di loro;

- il calcolo della posizione corretta del robot nella nuova scansione è più affidabile nel caso in cui siano presenti nell'ambiente alcuni riferimenti sicuri, come ad esempio muri e corridoi dove il robot è già passato;

- le scansioni presenti in un data set sono altamente ridondanti se il robot compie un numero elevato di rilevazioni nella stessa posizione, o spostandosi di pochi centimetri.

Partendo da queste considerazioni, abbiamo quindi implementato un nostro metodo di allineamento basato sui concetti di *rotazione rilevante*, *segmento significativo* e *struttura portante*. Una rotazione rilevante è una rotazione significativa per il robot: introduciamo questo concetto per ignorare tutte le scansioni in cui il robot è nel corso di una rotazione superiore a una determinata soglia, in quanto tali scansioni sono caratterizzate da elevati errori nel rilevamento degli ostacoli presenti nell'ambiente. In questo modo

le uniche scansioni prese in considerazione per l'allineamento sono quelle in cui robot si sta muovendo senza effettuare rotazioni: tali scansioni sono infatti caratterizzate da piccoli errori odometrici e costituiscono quindi delle rappresentazioni affidabili dell'ambiente circostante. Un segmento significativo è invece un segmento caratterizzato da una lunghezza superiore a una determinata soglia: introduciamo questo concetto per ignorare tutti i segmenti corti, i quali potrebbero causare degli errori se presi in considerazione durante l'allineamento. La struttura portante della mappa è infine una rappresentazione semplificata della mappa, costituita da rette. Le rette della struttura portante della mappa vengono calcolate a partire dall'insieme dei segmenti significativi. Utilizziamo delle rette in modo da poter rappresentare corridoi di cui è stata momentaneamente rilevata solo una porzione di muro: a partire dal segmento che rappresenta quella specifica porzione di muro, viene calcolata la retta su cui giace tale segmento, la quale identifica un ipotetico corridoio infinito. In Figura 3.1 viene riportato un esempio

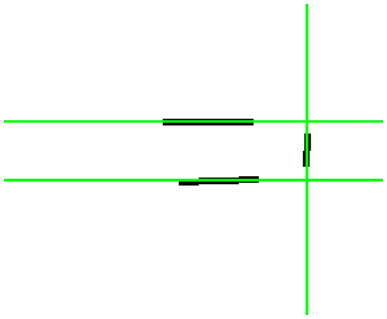

Figura 3.1: Esempio di struttura portante di una mappa: i segmenti significativi sono rappresentati in colore nero; le rette della struttura portante sono rappresentati in colore verde.

di struttura portante di una mappa: dall'insieme dei segmenti significativi (in questo caso tre) vengono estratte le rette su cui giacciono tali segmenti. Queste rette costituiscono la struttura portante della mappa. Ogni segmento delle successive scansioni, valutato come corrispondente a una retta della struttura portante, viene allineato a tale retta, in quanto molto probabilmente costituisce la naturale prosecuzione del corridoio identificato da tale retta.
Dopo aver introdotto questi concetti preliminari, descriviamo ora il funzionamento del nostro metodo di allineamento, costituito dai seguenti passi:

1. memorizza la scansione corrente del robot, effettuata in una determinata posizione;

2. determina la prossima scansione del robot da prendere in considerazione. Tale scansione non deve essere rilevata mentre il robot sta ruotando e deve essere effettuata in una posizione situata a una distanza superiore a una determinata soglia ϵ rispetto alla posizione della precedente scansione considerata. In questo modo vengono ignorate tutte le scansioni caratterizzate da rotazioni, le quali contengono elevati errori di odometria, e tutte le scansioni altamente ridondanti effettuate in una posizione vicina alla posizione della precedente scansione;

3. estrae i segmenti dai punti della nuova scansione mediante il metodo di estrazione dei segmenti descritto nella Sezione 3.2. Tali segmenti vengono filtrati, in modo da prendere in considerazione solo i segmenti che hanno lunghezza superiore a una determinata soglia σ e che rappresentano una caratteristica significativa dell'ambiente;

4. se il robot ha appena finito di effettuare una rotazione rilevante, aggiorna la struttura portante della mappa. L'aggiornamento della struttura portante della mappa viene eseguito a partire dall'insieme dei segmenti significativi rilevati fino alla scansione precedente, escludendo quindi i segmenti filtrati della scansione corrente, i quali devono essere ancora allineati con la mappa globale. Inoltre, nel caso in cui durante l'aggiornamento venga rilevata una nuova retta caratterizzata da una direzione simile a una retta precedentemente rilevata, alla nuova retta viene assegnato lo stesso coefficiente angolare della retta precedente: questo passaggio, che elimina ulteriormente gli errori di calcolo nella stima delle posizioni, è giustificato dalla struttura ortogonale degli ambienti indoor, i quali sono caratterizzati da corridoi ragionevolmente paralleli tra di loro;

5. ricerca le corrispondenze tra i segmenti della scansione corrente e le rette della struttura portante della mappa. Un segmento della scansione e una retta della struttura portante sono considerati corrispondenti se le seguenti due condizioni sono verificate:

   - le distanze tra i punti estremi del segmento e la retta sono inferiori a una determinata soglia δ;
   - la differenza angolare tra la direzione del segmento e la direzione della retta è inferiore a una determinata soglia θ.

6. calcola la rotazione ottima. A partire dalle coppie di segmenti e rette corrispondenti, calcola la rotazione ottima che allinea i segmenti della scansione corrente alle rette della struttura portante della mappa. La

rotazione è calcolata cercando di minimizzare la differenza angolare tra i segmenti e le rette corrispondenti;

7. calcola la traslazione ottima. A partire dalle coppie di segmenti e rette corrispondenti già allineate, calcola la traslazione ottima che sovrappongono i segmenti della scansione corrente alle rette della struttura portante della mappa. La traslazione ottima è calcolata cercando di minimizzare la distanza tra gli estremi dei segmenti e le rette corrispondenti;

8. aggiunge i segmenti filtrati della scansione corrente all'insieme dei segmenti significativi della mappa. Riduce quindi l'insieme dei segmenti significativi, fondendo tra loro i segmenti simili. In questo modo si evita di avere informazione ridondante, in quanto segmenti che rappresentano la stessa porzione di muro vengono fusi tra loro. E' importante mantenere l'insieme dei segmenti significativi più piccolo possibile, in modo da estrarre correttamente, all'iterazione seguente, le rette della struttura portante che rappresentano i corridoi;

9. torna al passo 2 per rilevare una nuova scansione.

Il metodo termina quando il robot si ferma definitivamente e non rileva più alcuna scansione. Vogliamo sottolineare come questo metodo sia applicabile solo in ambienti indoor ortogonali. Inoltre i segmenti creati col metodo di estrazione dei segmenti devono essere delle buone approssimazioni delle porzioni di ambiente reale che rappresentano: per questo motivo ambienti indoor caratterizzati da corridoi molto larghi non sono compatibili con tale metodo, in quanto i punti rilevati nelle singole scansioni si trovano a una elevata distanza dal sensore e sono quindi particolarmente imprecisi. La precisione dei segmenti estratti risulta un vincolo fondamentale per il corretto funzionamento del nostro metodo di allineamento. Tale metodo non si vuole quindi proporre come una soluzione finale al problema dell'allineamento, ma piuttosto come una idea di base per sviluppi futuri.

3.3.3 Metodo di allineamento basato sul ground truth

Alcuni data set vengono forniti con i rispettivi *ground thruth*. Per ground thruth si intende un insieme di informazioni che descrivono l'ambiente reale esplorato dal robot e le traiettorie effettivamente seguite nell'esplorazione dell'ambiente. I sensori utilizzati nella raccolta di questi dati differiscono da quelli montati sul robot. In questo modo la stima delle posizioni dell'ambiente, in cui sono state acquisite delle scansioni, non è più fatta basandosi

sui dati forniti dall'odometria, che, notoriamente, non sono affidabili. L'insieme delle scansioni e le informazioni relative al ground thruth sono fra loro sincronizzate. È così possibile, data una scansione S, risalire alla posizione in cui è stata acquisita.

Capitolo 4

Metodi di riduzione dei segmenti

Analizziamo ora, da un punto di vista teorico, i metodi scelti per le successive fasi di implementazione e valutazione sperimentale. L'obiettivo è quello di far emergere, per ciascun metodo, proprietà e caratteristiche che lo contraddistinguono e che possono essere utilizzate successivamente in fase di analisi dei risultati sperimentali.

4.1 Metodo Basic

Come abbiamo anticipato nel Paragrafo 2.7.1, il metodo Basic da noi realizzato fonde coppie di segmenti ridondanti. È un metodo completamente basato sui segmenti ed è indipendente dal metodo di estrazione dei segmenti utilizzato. Ogni segmento è rappresentato dal seguente insieme di parametri:

$$s = \{(x_1, y_1), (x_2, y_2), length, (x_c, y_c), angle\} \tag{4.1}$$

dove:

- (x_1, y_1) e (x_2, y_2) rappresentano le coordinate dei punti estremi del segmento;
- $length$ rappresenta la lunghezza del segmento;
- (x_c, y_c) rappresentano le coordinate del punto centrale del segmento;
- $angle$ rappresenta l'angolo compreso fra l'asse delle ascisse e il segmento;

Di questo metodo ne abbiamo realizzate due versioni:

- nella versione on-line il metodo Basic viene richiamato subito dopo l'acquisizione e l'allineamento di una nuova scansione S^t con la mappa corrente M^t;

- nella versione off-line il metodo Basic viene richiamato una sola volta al termine dell'intero processo di acquisizione e allineamento delle scansioni.

4.1.1 Condizioni di fusione

Le condizioni di fusione stabiliscono quando due segmenti rappresentano la stessa porzione dell'ambiente e, quindi, possono essere fusi tra loro. In questo metodo due sono le condizioni che devono essere rispettate per considerare una coppia di segmenti a e b corrispondenti:

1. la distanza tra almeno una coppia di punti estremi di a e b è inferiore a una soglia δ;

2. la distanza angolare tra i due segmenti è inferiore a una soglia θ.

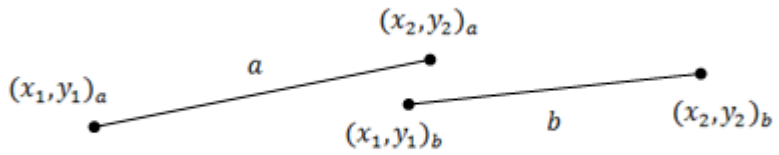

Figura 4.1: I due segmenti soddisfano le condizioni di fusione.

In Figura 4.1 e in Figura 4.2 abbiamo riportato due casi in cui i segmenti a e b soddisfano le condizioni di fusione.
In Figura 4.3 e in Figura 4.4 abbiamo riportato invece due casi in cui i segmenti a e b non soddisfano le condizioni di fusione. Particolarmente interessante è il caso rappresentato in Figura 4.4 in cui i segmenti, nonostante molto probabilmente rappresentino la stessa caratteristica dell'ambiente, non vengono ritenuti corrispondenti da questo semplice metodo.

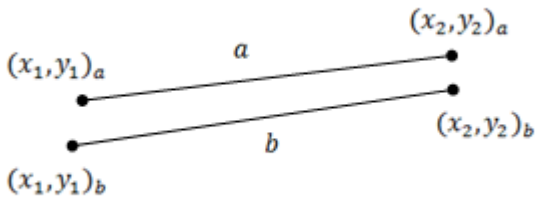

Figura 4.2: I due segmenti soddisfano le condizioni di fusione.

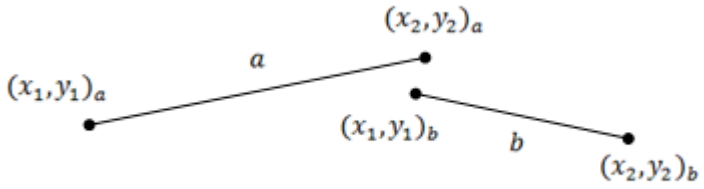

Figura 4.3: I due segmenti non soddisfano le condizioni di fusione.

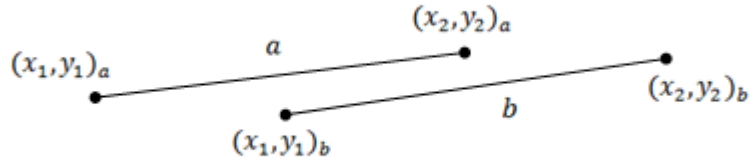

Figura 4.4: I due segmenti non soddisfano le condizioni di fusione.

4.1.2 Algoritmo di fusione

Se a e b soddisfano le condizioni di fusione precedenti, allora possono essere fusi tra di loro. L'algoritmo di fusione non crea un nuovo segmento a partire dai due, ma aggiorna i parametri del segmento a ed elimina il segmento b da quelli presenti nella mappa. L'aggiornamento prevede prima di tutto il calcolo della nuova direzione del segmento come media pesata delle direzioni dei due segmenti:

$$angle_{ab} = \frac{(length_a * angle_a) + (length_b * angle_b)}{length_a + length_b} \tag{4.2}$$

Per determinare in modo univoco la retta che definisce il segmento, oltre alla direzione, calcolata come descritto nell'Equazione 4.2, occorre un punto. Il punto $p = (p_x, p_y)$ viene calcolato come media pesata dei punti centrali dei segmenti:

$$p_x = \frac{(length_a * x_{c_a}) + (length_b * x_{c_b})}{length_a + length_b} \tag{4.3}$$

$$p_y = \frac{(length_a * y_{c_a}) + (length_b * y_{c_b})}{length_a + length_b} \tag{4.4}$$

Una volta determinata la retta su cui giace il nuovo segmento è possibile calcolare le altre informazioni che caratterizzano il segmento:

- i due punti estremi vengono calcolati proiettando i punti estremi di a e di b sulla nuova retta e prendendo i due punti proiettati che si trovano a maggiore distanza fra loro;
- il punto centrale del segmento viene calcolato come la media dei due punti estremi;
- la lunghezza viene calcolata come distanza euclidea fra i due punti estremi.

Infine il segmento b è eliminato dalla mappa.

4.1.3 Versione on-line

Nella versione on-line il metodo Basic viene richiamato subito dopo l'acquisizione e il conseguente allineamento di una nuova scansione S^t con la mappa globale M^t. Il metodo è costituito da tre passi:

1. aggiunge i segmenti presenti nella nuova scansione S^t a quelli della mappa corrente M^t in modo da ottenere la mappa M^{t+1};

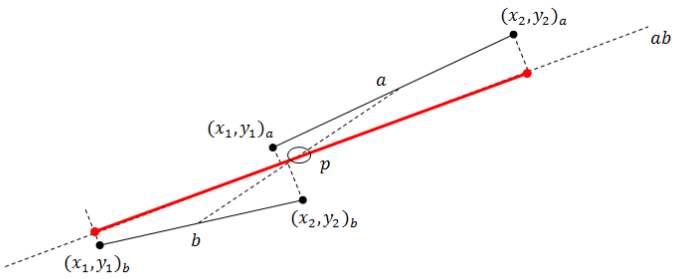

Figura 4.5: Fusione di due segmenti a e b.

2. per ogni segmento $a \in M^{t+1}$ esegue una ricerca esaustiva su tutti i segmenti di M^{t+1}, escluso a, allo scopo di trovare i segmenti corrispondenti ad a. Per ogni segmento b corrispondente, selezionato in base all'ordine con cui è stato trovato, applica l'algoritmo di fusione, il quale aggiorna i parametri del segmento a ed elimina il segmento b;

3. se è stata trovata almeno una coppia di segmenti corrispondenti e, quindi, è stato applicato almeno una volta l'algoritmo di fusione, il metodo ripete il passo 2, altrimenti passa all'acquisizione di una nuova scansione o termina. Il metodo di fusione viene quindi ripetuto iterativamente fino alla sua convergenza, ossia fino a quando non viene più trovata alcuna coppia di segmenti corrispondenti.

4.1.4 Versione off-line

Nella versione off-line il metodo Basic viene richiamato una sola volta al termine dell'acquisizione e dell'allineamento di tutte le scansioni. Il metodo è costituito da due passi:

1. per ogni segmento $a \in M$ esegue una ricerca esaustiva su tutti i segmenti di M, escluso a, allo scopo di trovare i segmenti corrispondenti ad a. Per ogni segmento b corrispondente, selezionato in base all'ordine con cui è stato trovato, applica l'algoritmo di fusione, il quale aggiorna i parametri del segmento a ed elimina il segmento b;

2. se è stata trovata almeno una coppia di segmenti corrispondenti e, quindi, è stato applicato almeno una volta l'algoritmo di fusione, il metodo ripete il passo 1, altrimenti termina. Il metodo di fusione

viene quindi ripetuto iterativamente fino alla sua convergenza, ossia fino a quando non viene più trovata alcuna coppia di segmenti corrispondenti.

4.2 Metodo Hybrid

4.2.1 La trasformata di Hough

La trasformata di Hough è una tecnica di estrazione di *feature* utilizzata nel campo dell'elaborazione digitale delle immagini. Permette di riconoscere particolari configurazioni di punti presenti nell'immagine, come segmenti, curve o altre forme. Il principio fondamentale è che la forma cercata può essere definita a partire da un insieme di parametri.
Nel caso delle mappe basate su segmenti, la trasformata di Hough viene utilizzata per rappresentare in modo compatto ed efficiente le rette su cui giacciono gli stessi segmenti. Infatti, assumendo come rappresentazione della retta la forma $y = mx + q$, qualunque retta viene univocamente determinata dai valori assunti dai parametri m e q. Analogamente, assumendo un tipo di rappresentazione diversa, come ad esempio la forma normale di Hesse $\rho = xcos\theta + ysin\theta$, la retta viene univocamente determinata dalla coppia di parametri (ρ, θ). Fissata quindi la forma di interesse e la sua rappresentazione, è possibile considerare una trasformazione dal piano dell'immagine allo spazio dei parametri. Così facendo, la forma di interesse è rappresentata da un singolo punto nello spazio dei parametri. In Figura 4.6 una retta in forma di Hesse viene rappresentata prima nello spazio dell'immagine, poi nello spazio dei parametri.

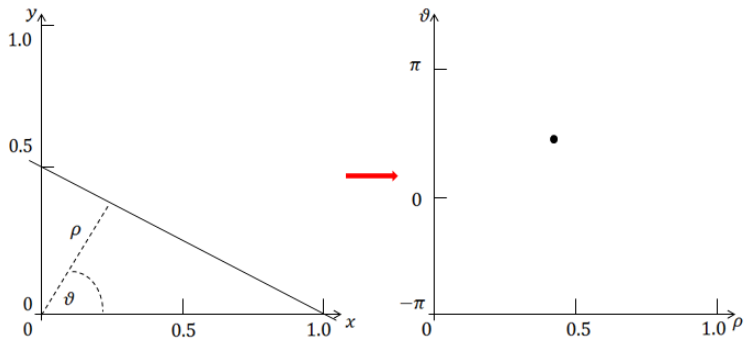

Figura 4.6: La trasformata di Hough permette la rappresentazione di una retta come un singolo punto nello spazio dei parametri.

4.2.2 Il metodo

Il metodo descritto in [43] fonde coppie di segmenti ridondanti, è indipendente dal metodo di estrazione dei segmenti utilizzato ed è un metodo completamente basato sui segmenti. Un segmento è rappresentato dal seguente insieme di parametri:

$$s = \{(x_1, y_1), (x_2, y_2), length, \rho, \theta, count\} \tag{4.5}$$

dove:

- (x_1, y_1) e (x_2, y_2) rappresentano le coordinate dei punti estremi del segmento;
- *length* rappresenta la lunghezza del segmento;
- ρ e θ rappresentano la retta su cui giace il segmento nello spazio di Hough;
- *count* rappresenta il peso di un segmento durante la procedura di fusione.

Il metodo Hybrid viene richiamato ogni volta che è acquisita una nuova scansione da parte del robot. Per ogni segmento contenuto nella nuova scansione, il metodo verifica le condizioni di fusione con il sottoinsieme dei segmenti della mappa corrente di cui è ''vicino'' da un punto di vista spaziale: se trova un segmento corrispondente, aggiorna i parametri del segmento della mappa; se non viene trovata alcuna corrispondenza, il segmento della scansione viene aggiunto alla mappa globale, essendo, molto probabilmente, una nuova caratteristica dell'ambiente rilevata.
Per ogni segmento contenuto nella nuova scansione vengono quindi eseguiti tre passi:

1. la ricerca dei segmenti della mappa corrente a cui è vicino;
2. la verifica delle condizioni di fusione;
3. l'eventuale applicazione dell'algoritmo di fusione in caso di corrispondenza.

4.2.3 Misura di vicinanza tra segmenti

In questa sezione introduciamo una *misura di vicinanza* tra coppie di segmenti.
Dati due segmenti a e b, si dice che b è vicino di a se è verificata almeno una delle seguenti condizioni:

- la distanza euclidea tra almeno una coppia dei punti estremi di a e di b è inferiore a una determinata soglia δ;
- la proiezione di almeno un punto estremo di b su a cade all'interno di a e la distanza euclidea tra l'estremo di b e l'estremo di b proiettato è inferiore a una determinata soglia δ.

La prima condizione serve a considerare b vicino di quei segmenti a lui quasi consecutivi o sovrapposti. Nell'esempio riportato in Figura 4.7, per i due segmenti a e b vengono calcolate quattro distanze euclidee:

- $d_1 = distanza\_euclidea((x_{1a}, y_{1a}), (x_{1b}, y_{1b}))$;
- $d_2 = distanza\_euclidea((x_{1a}, y_{1a}), (x_{2b}, y_{2b}))$;
- $d_3 = distanza\_euclidea((x_{2a}, y_{2a}), (x_{1b}, y_{1b}))$;
- $d_4 = distanza\_euclidea((x_{2a}, y_{2a}), (x_{2b}, y_{2b}))$.

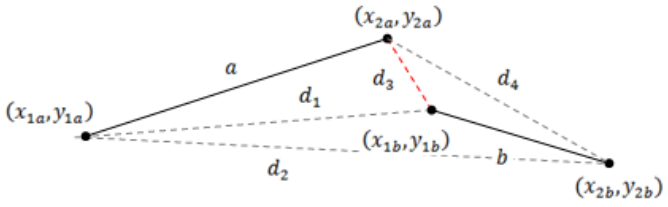

Figura 4.7: Misura di vicinanza.

Se la minima fra queste distanze (nell'esempio in figura d_3) è inferiore a δ, allora b è vicino di a.
La seconda condizione serve invece a considerare b vicino di quei segmenti a lui parzialmente sovrapposti. Nell'esempio riportato in Figura 4.8 il segmento b sarebbe, erroneamente, non considerato vicino di a se la misura di vicinanza fosse composta dalla sola prima condizione. Considerando anche la seconda condizione, invece, i due punti estremi del segmento b vengono proiettati sulla retta su cui giace il segmento a, ottenendo i punti proiettati (x_{1p}, y_{1p}) e (x_{2p}, y_{2p}). Siccome tra questi solo (x_{1p}, y_{1p}) cade all'interno di a, viene calcolata solamente la distanza euclidea tra (x_{1b}, y_{1b}) e (x_{1p}, y_{1p}). Se tale distanza è inferiore a δ, allora b è considerato vicino di a. La misura di vicinanza così definita gode della proprietà riflessiva in quanto ogni segmento è vicino di se stesso. Tale misura non gode però della proprietà

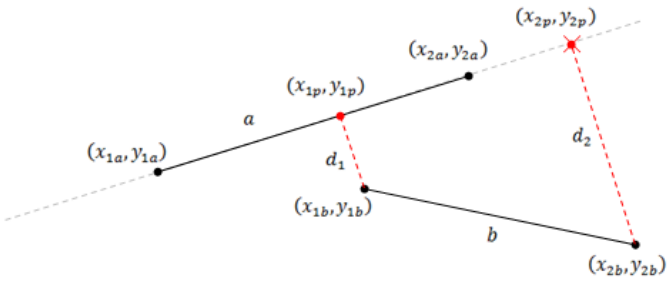

Figura 4.8: Misura di vicinanza.

simmetrica: con la definizione corrente potrebbe benissimo accadere che un segmento b sia vicino di un segmento a, ma non il viceversa.

4.2.4 Ricerca dei segmenti vicini

Per ogni segmento a contenuto nella scansione S, viene eseguita una ricerca esaustiva tra i segmenti della mappa corrente M. L'obiettivo è trovare i segmenti della mappa di cui a è vicino: per fare questo utilizziamo la misura di vicinanza descritta nella precedente sezione. Nella fase successiva la ricerca dei corrispondenti di a viene effettuata su questo sottoinsieme di segmenti, partendo dal più vicino. L'estrazione dei segmenti a cui a è vicino è, quindi, una fase che precede la verifica delle condizioni di fusione. Incorporare il requisito di vicinanza tra quelli delle condizioni di fusione sarebbe sbagliato. La verifica delle condizioni di fusione, anche in questo caso, avverrebbe tra a e un segmento di cui è vicino, tuttavia non è detto che il segmento della mappa, con cui a soddisfa le condizioni di fusione, sia il più vicino.

4.2.5 Condizioni di fusione

Dati due segmenti a e b, con $a \in S$, $b \in M$ e a vicino di b, secondo la misura di vicinanza definita precedentemente, le condizioni di fusione consistono in un confronto tra i parametri ρ e θ dei due segmenti. Se le differenze in valore assoluto tra i parametri ρ e θ del segmento a e del segmento b sono inferiori a una determinata soglia, allora i due segmenti rappresentano molto probabilmente la stessa caratteristica dell'ambiente e possono essere fusi tra di loro. Altrimenti, se le condizioni di fusione non vengono rispettate, il segmento a è aggiunto alla mappa corrente se, e solo se, la sua lunghezza

supera una determinata soglia. È importante notare che la verifica delle condizioni di fusione, per come sono definite, deve avvenire tra coppie di segmenti (a, b) con $a \in S$, $b \in M$ e a vicino di b. Se ciò non avviene potrebbero essere rilevate corrispondenze errate, come ad esempio quella in Figura 4.9.

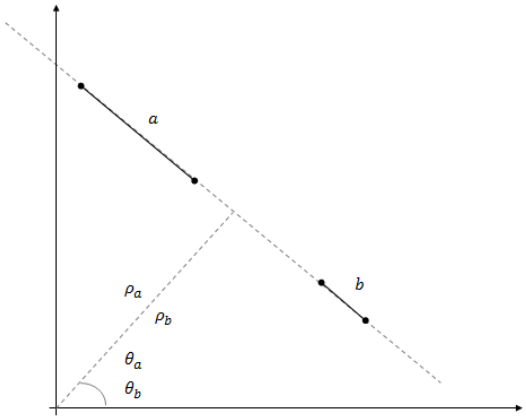

Figura 4.9: Falsa corrispondenza.

4.2.6 Algoritmo di fusione

La fusione di due segmenti è la parte più importante del metodo. Quando due segmenti soddisfano le condizioni di fusione, i due segmenti vengono fusi tra loro aggiornando i parametri ρ e θ del segmento della mappa globale. L'aggiornamento è fatto utilizzando le seguenti formule:

$$\rho_{b,new} = \frac{count * \rho_{b,old} + \rho_a}{count + 1} \tag{4.6}$$

$$\theta_{b,new} = \frac{count * \theta_{b,old} + \theta_a}{count + 1} \tag{4.7}$$

La variabile *count* serve a tenere traccia del numero di volte in cui un segmento della mappa globale è stato fuso con un segmento estratto da una scansione. Tale variabile serve a dare maggior peso, nella procedura di fusione, ai segmenti aventi un contatore elevato. Implicitamente possiamo pensare che un segmento con contatore elevato rappresenti una caratteristica dell'ambiente rilevata più volte e quindi non frutto di eventuali errori di misurazione. I segmenti estratti da una nuova scansione hanno contatore

pari ad uno. Come punti estremi del nuovo segmento vengono presi, all'interno dell'insieme dei punti estremi dei due segmenti di partenza proiettati sulla retta avente parametri $\rho_{b,new}$ e $\theta_{b,new}$, quelli più lontani tra loro.

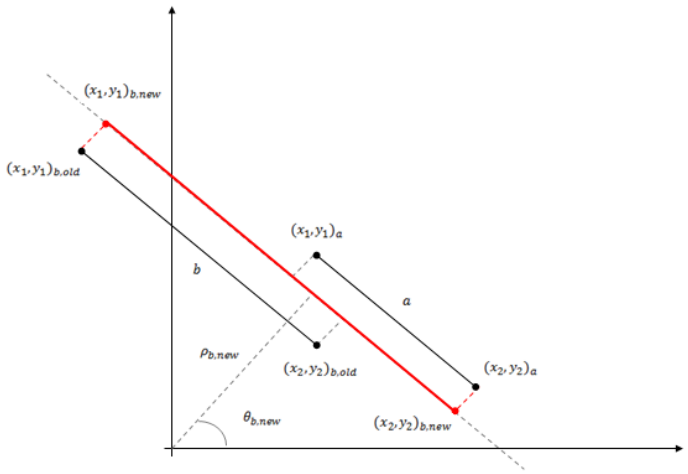

Figura 4.10: Fusione di una coppia di segmenti.

4.3 Metodo Mean Shift Clustering

4.3.1 Mean Shift Clustering

Il mean shift [51] è un processo di discesa del gradiente utilizzato per trovare multipli centri di distribuzioni di densità: esso sposta iterativamente i punti verso i rispettivi centri di densità in una certa vicinanza spaziale. Usando il mean shift come metodo di clustering, i punti che convergono allo stesso centro appartengono allo stesso cluster. Il principale vantaggio del mean shift è che, fatta eccezione per la definizione di vicinanza spaziale, è libero da parametri: non è infatti necessario specificare né la lunghezza del passo del gradiente, né il numero dei cluster. L'unico parametro di cui necessita è quindi quello che definisce la scala della mappa o, più intuitivamente, la minima dimensione di un dettaglio significativo: tutti gli altri parametri necessari per applicare questo processo possono essere derivati dal parametro di scala e risultano costanti in tutti gli esperimenti. Il principale svantaggio è che il mean shift è molto sensibile alla scelta del nucleo, il quale determina il grado di operazioni locali rispetto alle operazioni globali.

Il mean shift denota la differenza $\Delta m = m(x) - x$, con $m(x)$ definito come:

$$m(x) = \frac{\sum_P K(p-x)p}{\sum_P K(p-x)} \tag{4.8}$$

dove $p \in P \subset \mathbb{R}^n$ rappresenta un insieme di punti e K rappresenta il nucleo di uno specifico centro di densità $x \in \mathbb{R}^n$.
Dato un insieme finito di centri cluster $t_i = T \subset \mathbb{R}^n$, l'algoritmo ripete iterativamente il passo $t_i \to m(t_i)$ simultaneamente per tutti i t_i fino a che $max(\Delta m(t_i))$ è sufficientemente piccolo, ossia fino a che non viene raggiunta la convergenza: per tutti i t_i vengono calcolati simultaneamente gli $m(t_i)$ (utilizzando l'equazione precedente con $x = t_i$), prima che tutti i t_i vengano rimpiazzati con gli $m(t_i)$. L'algoritmo viene utilizzato inizializzando l'insieme dei centri cluster a $T^0 = P$, ossia ai punti iniziali: passo dopo passo i punti si avvicinano ai rispettivi centri cluster fino a raggiungere la convergenza.

4.3.2 Il metodo

Il metodo di fusione basato sul mean shift clustering, descritto in [45], è suddiviso in due fasi: la prima fase determina direzioni localmente comuni; la seconda fase raggruppa segmenti collineari. Il clustering è seguito dal processo di fusione, che combina i segmenti appartenenti allo stesso cluster. Dato un insieme $S = \{s_1, \cdots, s_m\}$ costituito da m segmenti, rappresentiamo ogni singolo segmento nel modo seguente:

$$s_i \to p_i = (x, y, \alpha) \in \mathbb{R} \times \mathbb{R} \times [0..\pi] \tag{4.9}$$

dove (x, y) rappresenta il centro del segmento s_i e α rappresenta la sua direzione, espressa come angolo compreso tra s_i e l'asse delle ascisse. Per ogni singolo segmento utilizziamo la seguente notazione: $p^{[1:2]} = (x, y)$ per identificare il centro del segmento; $p^{[3]} = \alpha$ per identificare la direzione del segmento. In questo modo ogni segmento viene rappresentato tramite un punto (il suo centro) dotato di una specifica direzione. I segmenti che eccedono una certa lunghezza vengono suddivisi in segmenti più piccoli: i centri e le direzioni di questi sotto-segmenti formano l'insieme dei punti iniziali $P = T^0$ con cui viene avviato il processo di mean shift clustering. Questo passo garantisce che segmenti corti e segmenti lunghi vengono rappresentati nello stesso modo da punti centrali localmente vicini e uniformemente distribuiti.

4.3.3 Prima fase: rilevamento direzioni localmente comuni

L'obiettivo di questa fase è di raggruppare i segmenti in base alla loro direzione e regione spaziale: l'enfasi è posta quindi sulla separazione dei segmenti aventi una direzione sufficientemente differente e sul raggruppamento di segmenti con una direzione simile. In questa fase utilizziamo un nucleo gaussiano, con $\sigma_n = \sigma \in \mathbb{R}^+$ per tutte le dimensioni n. σ è l'unico parametro dell'intero sistema: dipende dalla scala dei dati e rappresenta la minima dimensione di una caratteristica significativa. Il nucleo gaussiano utilizzato in questa fase è quindi il seguente:

$$K_1(x) = K_1(x^{[1:2]}, x[3]) = \exp^{-\frac{d(x)^2}{2\sigma^2}} \tag{4.10}$$

con la distanza d definita come:

$$d(x) = \sqrt{\|x^{[1:2]}\|^2 + \Big(\frac{min(\|x^{[3]}\|, \pi - \|x^{[3]}\|)}{h}\Big)^2} \tag{4.11}$$

dove la bandwidth h rappresenta un fattore di scala, necessario per convertire la distanza angolare, inizialmente espressa in gradi, in una distanza spaziale. Ad esempio, con $h = 10°/200mm$ indichiamo che a una differenza angolare di 10° corrisponde una distanza spaziale di $200mm$.
Partendo dall'insieme dei punti $T^0 = P$, che rappresentano i segmenti divisi s_i, applichiamo il mean shift clustering con nucleo K_1 fino a raggiungere la convergenza, formando così l'insieme dei punti finali $\overline{T}$. Un cluster C_l in $\overline{T}$ consiste in un insieme di punti che convergono allo stesso centro. Assegniamo quindi un'etichetta $l(s_i)$ ai segmenti corrispondenti, in modo tale che $l(s_i) = l(s_j) \leftrightarrow d(t_i - t_j)$, con d che rappresenta la misura di distanza definita in precedenza.
La Figura 4.11(a) e la Figura 4.11(b) mostrano l'insieme dei punti iniziali, le traiettorie e l'insieme dei punti finali. La Figura 4.11(c) e la Figura 4.11(d) mostrano il risultato di questa prima fase di clustering: i segmenti aventi una direzione simile e caratterizzati da una certa vicinanza spaziale fanno parte dello stesso cluster. Vogliamo far notare che il clustering basato sulla vicinanza spaziale e sulla direzione è differente dal clustering basato sulla sola direzione: permette infatti di distinguere piccole differenze direzionali se i segmenti sono localizzati in differenti vicinanze. Ad esempio, osservando la Figura 4.11(d) possiamo notare che alcuni segmenti dei cluster 1 e 2 condividono approssimativamente la stessa direzione, ma fanno parte di differenti cluster a causa della loro distanza spaziale.

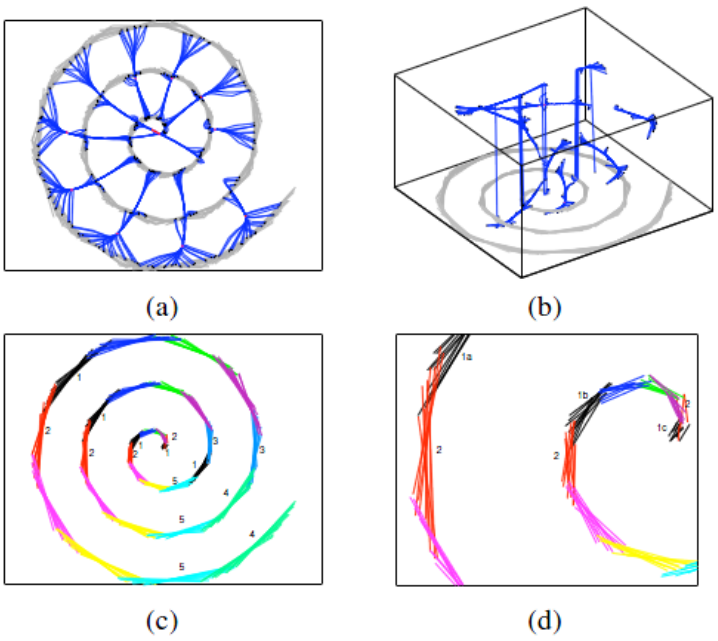

Figura 4.11: Risultato fase 1. a) Vista 2D (x,y). Blu: Traiettorie, punti in nero: T^0, punti in rosso: $\overline{T}$. b) stesso di (a), ma vista 3D. c) Un colore per cluster: segmenti con lo stesso colore fanno parte dello stesso cluster. d) Vista del centro di (c). Segmenti 1a, 1b, 1c appartengono ancora allo stesso cluster. Saranno separati nella fase 2. [45]

4.3.4 Seconda fase: raggruppamento dei segmenti collineari

La prima fase determina direzioni localmente comuni, ma non è in grado di distinguere i segmenti collineari da quelli non-collineari. La seconda fase suddivide dunque ogni cluster C_l della prima fase in sotto-cluster di segmenti collineari, utilizzando l'informazione sulla direzione del cluster. Innanzitutto definiamo la direzione di un cluster come la direzione media $\theta = t_i^{[3]}$ di tutti i punti $t_i \in C_l$ (come risultato della prima fase, tutti i punti hanno comunque approssimativamente la stessa direzione). La direzione del cluster è l'unica informazione che teniamo dalla prima fase: le posizioni $\overline{T}^{[1:2]}$ dei cluster vengono omesse. Omettere l'informazione sulla posizione è il prezzo da pagare per una maggiore robustezza: nella seconda fase, infatti, le posizioni vengono ricalcolate a partire da una determinata direzione. Dato che ogni segmento in un cluster ha una direzione assegnata, è sufficiente rappresentare il segmento s_i in uno spazio bidimensionale: la seconda fase del mean shift clustering opera infatti in uno spazio bidimensionale, definito come la posizione dei punti di un singolo cluster. Anche in questa fase utilizziamo un nucleo gaussiano, con deviazioni standard σ_x, σ_y derivate dal parametro σ definito nella prima fase. Il nucleo gaussiano utilizzato in questa fase è quindi il seguente:

$$K_2(x) = \exp^{-\frac{1}{2}x^T\xi} \tag{4.12}$$

dove ξ rappresenta la matrice di covarianza definita come:

$$\xi = \begin{bmatrix} a & b \\ b & c \end{bmatrix} \tag{4.13}$$

con

$$a = \frac{(\cos\theta)^2}{2\sigma_x^2} + \frac{(\sin\theta)^2}{2\sigma_y^2} \tag{4.14}$$

$$b = -\frac{(\sin 2\theta)}{4\sigma_x^2} + \frac{(\sin 2\theta)}{4\sigma_y^2} \tag{4.15}$$

$$c = \frac{(\sin\theta)^2}{2\sigma_x^2} + \frac{(\cos\theta)^2}{2\sigma_y^2} \tag{4.16}$$

Nella seconda fase inizializziamo il processo con $T^0 = P^{[1:2]}$ e applichiamo, per ogni cluster ottenuto dalla prima fase, il mean shift clustering con nucleo K_2 fino a raggiungere la convergenza: come già detto in precedenza, suddividere il processo in due fasi rende l'approccio più robusto.

4.3.5 Fase finale: fusione

Dopo aver calcolato i sotto-cluster, la fusione è una semplice procedura geometrica. La seconda fase suddivide ogni cluster C_l nei sotto-cluster $C_l^{[1]}, \cdots, C_l^{[k_l]}$. I corrispondenti centri $M_l^{[1]}, \cdots, M_l^{[k_l]}$ sono punti delle singole rette
$R_l^{[1]}, \cdots, R_l^{[k_l]}$ rappresentative dei cluster: ciascuna retta $R_l^{[j]}$ viene identificata attraverso il suo centro $M_l^{[j]}$ e la direzione θ del rispettivo cluster. Per calcolare il risultato finale del processo di fusione, proiettiamo tutti i segmenti s_i di un singolo sotto-cluster $C_l^{[j]}$ su $R_l^{[j]}$ e fondiamo i segmenti sovrapposti in un unico segmento: la Figura 4.12 illustra questo processo.

Figura 4.12: Processo di fusione.

4.4 Metodo basato sulla retta di fusione

4.4.1 Retta di fusione

Il metodo di fusione descritto in [48] si basa sul concetto di *retta di fusione*. Dati una retta r, un segmento s e un valore di tolleranza $\epsilon \geq 0$, r si dice retta di fusione per s rispetto a ϵ se e solo se $max(d(p_1, r), d(p_2, r)) \leq \epsilon$, dove d è la distanza euclidea e (p_1, p_2) sono i punti estremi del segmento s. Analogamente, diciamo che r è retta di fusione rispetto a ϵ per un insieme di segmenti S se e solo se r è retta di fusione rispetto a ϵ per ogni segmento s appartenente a S. Geometricamente, ogni segmento appartenente a S è contenuto nella fascia centrata in r di larghezza 2ϵ e di lunghezza infinita. La retta di fusione gode di alcune proprietà:

1. $\forall \epsilon \geq 0$ e per ogni segmento s, esiste almeno una retta di fusione r per s rispetto a ϵ;

2. dati un insieme S e un valore di tolleranza ϵ, se esiste una retta di fusione per S rispetto a ϵ può non essere unica;

3. dati un insieme S e una retta r, esiste un valore minimo ϵ_0 tale che:

   - $\forall \epsilon \geq \epsilon_0$ r è retta di fusione per S rispetto a ϵ;
   - in particolare $\epsilon_0 = \max_{\overline{s} \in S}(\max(d(\overline{p}_1, r), d(\overline{p}_2, r)))$.

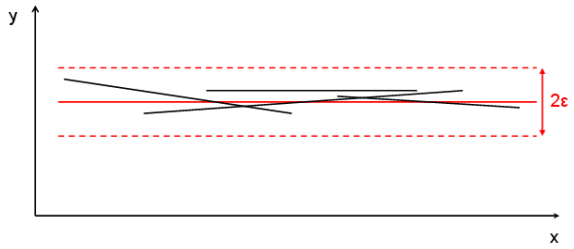

Figura 4.13: r (in rosso) è retta di fusione per l'insieme di segmenti S (in nero) rispetto a ϵ. Ogni segmento di S è contenuto nella fascia centrata in r di larghezza 2ϵ e lunghezza infinita (in rosso tratteggiato).

4.4.2 Condizioni di fusione

Una retta di fusione r, rispetto a ϵ, identifica sulla mappa una fascia centrata in r di larghezza 2ϵ e di lunghezza infinita (Figura 4.13). I segmenti compresi nella fascia possono però rappresentare anche porzioni d'ambiente molto lontane tra loro. La definizione di retta di fusione infatti pone solamente un vincolo sulla distanza tra i segmenti e la retta, non sulla distanza tra i segmenti inclusi (Figura 4.14). Nasce quindi la necessità di introdurre un ulteriore concetto per definire correttamente i segmenti ridondanti: *la catena di segmenti*. Dati una retta r e un insieme di segmenti S, chiamiamo $s_i^{'}$ la proiezione del segmento s_i su r. Definiamo una relazione τ_r tale che: $s_1 \tau_r s_2$ se e solo se $s_1^{'}$ è sovrapposto a $s_2^{'}$ e indichiamo con $\overline{\tau}_r$ la chiusura transitiva di τ_r.
Dati un insieme di segmenti S e una retta r, S si dice catena di segmenti per r se e solo se $\forall s_1, s_2 \in S$ vale che $s_1 \tau_r s_2$.
La definizione di catena di segmenti per una generica retta r indica l'insieme di segmenti la cui proiezione su r individua una parte connessa e limitata di

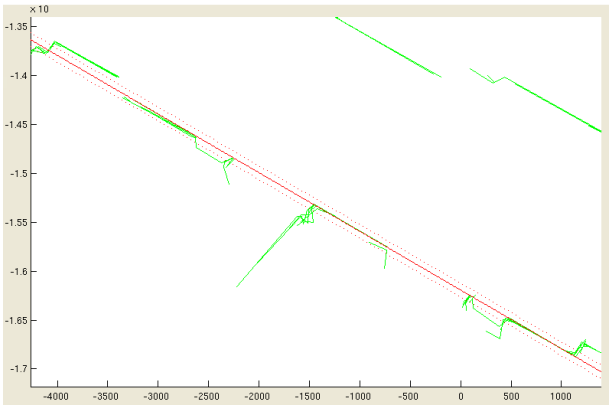

Figura 4.14: La retta di fusione (in rosso) include segmenti che rappresentano porzioni d'ambiente anche molto lontane tra loro.

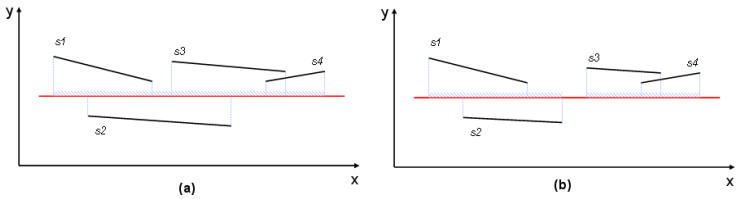

Figura 4.15: (a) l'insieme di segmenti S (in nero) è una catena di segmenti per r (in rosso); (b) l'insieme di segmenti S (in nero) non è una catena di segmenti per r (in rosso)

r: nella Figura 4.15 è illustrato il concetto di catena di segmenti. Utilizzando il concetto di catena di segmenti applicato alla retta di fusione, possiamo specificare le condizioni di fusione per il nostro problema. Dati un insieme di segmenti S e un valore di tolleranza ϵ, S può essere fuso se e solo se sono verificate le seguenti condizioni di fusione:

1. esiste una retta r tale che r è retta di fusione per S rispetto a ϵ;
2. S è una catena di segmenti per la retta di fusione r del punto 1.

Entrambe le condizioni sono indispensabili per individuare correttamente l'insieme di segmenti ridondanti. La prima condizione permette di specificare un valore di tolleranza, ovvero la larghezza della fascia entro la quale i segmenti possono essere fusi. La seconda condizione invece può essere utilizzata per indicare quando i segmenti inclusi si riferiscono allo stesso oggetto nella mappa (Figura 4.16). Una retta di fusione r rappresenta una buona

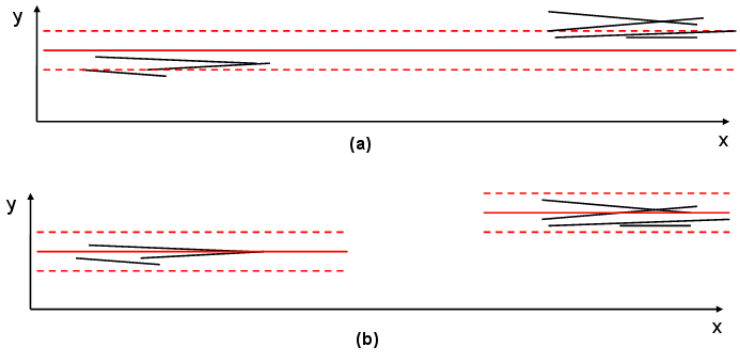

Figura 4.16: (a) l'insieme dei segmenti contenuto nella retta di fusione r (in rosso) non è ridondante in quanto non è catena di segmenti per r; (b) gli insiemi contenuti nelle rette di fusione (in rosso) sono due insiemi ridondanti perché oltre ad essere inclusi in una retta di fusione, sono anche catena di segmenti per tale retta.

approssimazione per l'insieme di segmenti S che include. Infatti tutti i punti appartenenti ai segmenti di S hanno una distanza da r inferiore o uguale a ϵ. Implementiamo dunque un meccanismo di fusione che individua i punti estremi del nuovo segmento sulla retta di fusione.

4.4.3 Algoritmo di fusione

Dati un valore di tolleranza ϵ e un insieme di segmenti S, che verifica le condizioni di fusione per ϵ con la retta r, calcoliamo il segmento risultante dalla fusione $s_f = \{(x_1, y_1), (x_2, y_2)\}$ in questo modo:

1. calcoliamo le proiezioni $s_i^{'}$ su r dei segmenti $s_i \in S$;
2. selezioniamo come punti estremi (x_1, y_1) e (x_2, y_2) del segmento risultante la coppia di punti più lontani appartenenti alle proiezioni $s_i^{'}$ calcolate al passo uno.

In questo modo, ogni catena che verifica le condizioni di fusione è approssimata da un unico segmento sulla retta di fusione che coincide con la proiezione della catena sulla retta, come illustrato in Figura 4.17.

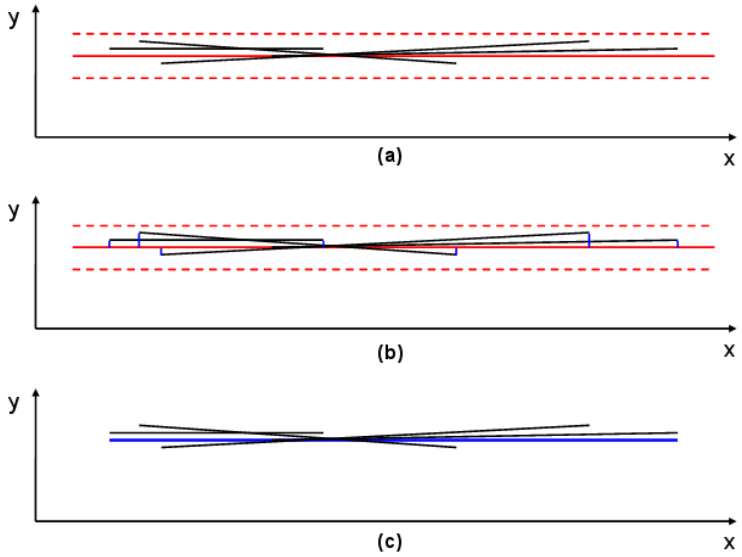

Figura 4.17: (a) l'insieme di segmenti S (in nero) è ridondante in quanto esiste una retta di fusione r (in rosso) per S e S è catena di segmenti per r; (b) i punti estremi dei segmenti di S vengono sono proiettati su r; (c) i punti estremi del nuovo segmento (in blu) che approssima S, sono la coppia di punti proiettati più lontani.

4.4.4 Il metodo

Nella sezione precedente abbiamo introdotto un metodo di fusione che permette di approssimare una catena di segmenti utilizzando il concetto di retta di fusione. Dato un valore di tolleranza ϵ, la fusione di una mappa M diventa un problema di ricerca dell'insieme di rette di fusione rispetto a ϵ che meglio approssimano M. Per affrontare tale problema è possibile implementare un algoritmo che, a ogni iterazione, trova la retta di fusione r^* ottima rispetto a ϵ tale che:

$$\forall r \text{ rispetto a } \epsilon \text{ in } M, \text{ con } r \neq r^*, lungh\_max(r^*) \geq lungh\_max(r)$$

dove $lungh\_max(x) = \max_{S \in C} l(S)$ con C che rappresenta l'insieme delle catene di segmenti incluse nella retta di fusione x e $l(S)$ che rappresenta lunghezza complessiva dei segmenti in S. A ogni passo l'algoritmo fonde, con un unico segmento individuato sulla retta di fusione r^*, la catena di segmenti ridondanti più grande in termini di lunghezza complessiva. La metrica, utilizzata per la scelta della retta di fusione ottima, assegna maggior peso alla lunghezza dei segmenti rispetto alla loro cardinalità. Il rumore nel processo di mappatura infatti produce zone nella mappa costituite da numerosi segmenti corti. Con l'utilizzo della lunghezza complessiva, l'informazione dei segmenti lunghi è considerata più affidabile rispetto a quella rappresentata dai segmenti più corti (Figura 4.18). Dati una mappa M e

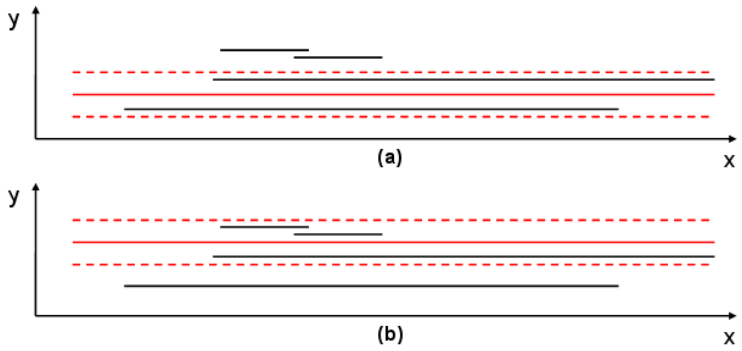

Figura 4.18: (a) la retta di fusione in rosso include la catena con lunghezza complessiva massima; (b) la retta di fusione in rosso include la catena di segmenti con cardinalità massima.

un valore di tolleranza ϵ, il metodo di fusione basato sulla retta di fusione opera nel modo seguente:

1. individua la retta di fusione r^* rispetto a ϵ che include la catena di segmenti ridondanti S con lunghezza complessiva massima;
2. seleziona i punti estremi del nuovo segmento s_f che approssima S come la coppia di punti più lontani appartenenti alle proiezioni $s_i^{'}$ su r dei segmenti $s_i \in S$;
3. aggiunge s_f alla mappa $M^{'}$ e toglie da M l'insieme S;
4. se $M \neq \emptyset$ termina e restituisce la mappa $M^{'}$ che contiene i segmenti ottenuti dalla fusione di M, altrimenti riparte dal passo uno.

Capitolo 5

Realizzazione degli algoritmi

5.1 Algoritmo di estrazione dei segmenti

Nella Sezione 3.2 abbiamo descritto un metodo che permette di approssimare i punti di una scansione S con un insieme di segmenti. La scansione S è costituita da un insieme di punti espressi in coordinate polari, con origine nel sistema di riferimento del sensore. Prima di poter applicare l'algoritmo di estrazione dei segmenti, occorre innanzitutto esprimere i punti in coordinate cartesiane. Data una scansione $S = \{p_1, p_2, \cdots, p_n\}$, dove p_i è il singolo punto espresso in coordinate cartesiane, il metodo estrae un insieme di segmenti compiendo i seguenti tre passi:

1. estrazione di un insieme di cluster. Un cluster è un sottoinsieme di punti consecutivi $\{p_i, p_{i+1}, \cdots, p_{i+k}\}$ della scansione S, tale che la distanza tra un punto e il successivo è inferiore a una soglia δ;

2. filtraggio dei cluster. I cluster composti da un solo punto vengono eliminati;

3. estrazione dei segmenti. Per ogni cluster viene estratto un segmento che approssima l'insieme dei punti del cluster.

Mostriamo ora lo pseudocodice dell'algoritmo che abbiamo implementato relativamente al nostro metodo di estrazione dei segmenti:

```
1 % FUNZIONE: estrai_segmenti
2 %   Estrae, a partire dai punti della scansione S, un insieme di segmenti.
3 % INPUT:
4 %   S: insieme di punti;
5 %   delta: soglia distanza spaziale.
6 % OUTPUT:
7 %   segmenti: insieme di segmenti estratti a partire da S.
```

```
8  function[segmenti] = estrai_segmenti(S, delta)
9
10   segmenti = {};
11
12   % La prima fase raggruppa i punti sulla base della loro distanza.
13   % Il risultato di questa fase è un insieme di cluster, ciascuno
14   % formato da un numero n >= 1 di punti.
15   cluster = estrai_cluster(S, delta);
16
17   % La seconda fase elimina i cluster composti da un solo punto.
18   cluster = filtra_cluster(cluster);
19
20   % La terza fase estrae un segmento rappresentativo per ogni
21   % cluster.
22   for c in cluster
23
24     % Estrae un segmento dal cluster corrente.
25     segmento = estrai_segmento(c);
26
27     % Aggiunge il segmento estratto all'insieme di segmenti
28     % estratti dalla scansione.
29     segmenti = segmenti U {segmento};
30
31   end
32
33   return segmenti;
34
35 end
```

5.1.1 Algoritmo di estrazione dei cluster

Il processo di estrazione dei cluster, a partire da una scansione S, raggruppa i punti di S che possono essere approssimati con un solo segmento. Il clustering viene eseguito sulla base della distanza tra coppie di punti consecutivi: la soglia δ influisce sulla dimensione dei cluster. Esiste un trade off tra la dimensione dei cluster e la qualità del segmento approssimante estratto nella fase seguente. Aumentando *delta*, diminuisce il numero di cluster e di conseguenza il numero di segmenti estratti da S. Tuttavia, i segmenti rischiano di essere un'approssimazione grossolana dell'insieme di punti da cui sono estratti. Diminuendo *delta* invece, il numero dei cluster, e di conseguenza il numero dei segmenti estratti da S, aumenta. In questo caso però, essendo i punti che compongono ogni cluster meno distanti, l'errore che introduciamo, approssimando ogni insieme di punti con un unico segmento, è assai minore.

Mostriamo ora lo pseudocodice dell'algoritmo di estrazione dei cluster:

```
1 % FUNZIONE: estrai_cluster
2 %   Estrae, a partire dai punti della scansione S, un insieme di cluster,
3 %   ciascuno formato da un numero di punti n >= 1.
4 % INPUT:
```

```
5  %   S: insieme di punti;
6  %   delta: soglia di massima distanza fra punti per far parte dello stesso
7  %   cluster.
8  % OUTPUT:
9  %   C: insieme di cluster.
10 function[C] = estrai_cluster(S, delta)
11
12   C = {};
13   c = {};
14
15   while i < dimensione(S) - 1
16     p = S(i);
17     p' = S(i + 1)
18
19     % Crea un nuovo cluster e aggiunge il primo punto p.
20     if c == {}
21       c = {p};
22     end
23
24     % Se la distanza euclidea tra il punto corrente p e quello
25     % successivo p' è inferiore a delta ...
26     if distanza_euclidea(p,p') < delta
27
28       % Aggiunge p' al cluster corrente.
29       c = c U {p'};
30
31       % Elimina p' da S.
32       S = S \ {p'};
33
34     % Altrimenti ...
35     else
36
37       % Salva il cluster corrente.
38       C = C U {c};
39
40       % Crea un nuovo cluster e aggiunge il primo punto p'.
41       c = {p'};
42
43     end
44
45   end
46
47   return C;
48
49 end
```

5.1.2 Algoritmo di estrazione di un segmento

Dato un cluster, l'estrazione del segmento avviene in due passi:

1. calcolo, attraverso il metodo di regressione ai minimi quadrati descritto nella Sezione 3.2, dei parametri della retta che meglio approssima l'insieme di punti;

2. estrazione degli estremi del segmento. L'estrazione avviene proiettando i punti del cluster sulla retta trovata al passo precedente e prendendo tra i punti proiettati quelli che si trovano a maggiore distanza tra loro.

Mostriamo ora lo pseudocodice dell'algoritmo di estrazione di un segmento:

```
% FUNZIONE: estrai_segmento
%   Dato un insieme di punti, distanti meno di una soglia, estrae un segmento
%   che li approssima.
% INPUT:
%   c: insieme di punti.
% OUTPUT:
%   s: segmento che approssima l'insieme di punti.
function[s] = estrai_segmento(c)

  % Estrae i parametri della retta, nella forma ax + by + c = 0,
  % che meglio approssima l'insieme di punti c.
  retta_proiezione = estrai_parametri_retta(c);

  % Proietta i punti di c sulla retta trovata al passo precedente.
  c' = proietta_punti(c, retta_proeizione);

  % Estrae i punti estremi di s a partire dall'insieme dei punti
  % proiettati. I due punti estremi sono i due punti proiettati
  % che si trovano a maggiore distanza tra di loro.
  s = estrai_estremi(c');

end
```

5.2 Algoritmo di allineamento delle scansioni

Nella Sezione 3.3.2 abbiamo descritto un metodo di allineamento basato sulla struttura portante di una mappa. Questo metodo permette di allineare le scansioni effettuate dal robot rispetto a una struttura portante costituita da rette che rappresentano i corridoi dell'ambiente indoor. La struttura portante della mappa viene mantenuta in memoria e aggiornata dopo ogni rotazione rilevante del robot.

Mostriamo ora lo pseudocodice dell'algoritmo che abbiamo implementato relativamente al nostro metodo di allineamento delle scansioni:

```
% FUNZIONE: allinea_scansioni
%   Dato un insieme di scansioni non allineate, le allinea.
% INPUT:
%   S: insieme di scansioni non allineate.
%   gamma: soglia di distanza spaziale utilizzata per l'estrazione dei
%   segmenti;
%   sigma: soglia di lunghezza utilizzata per il filtraggio dei segmenti;
%   eps: soglia di distanza spaziale tra una scansione e la successiva;
```

```
9  %   delta: soglia di distanza spaziale utilizzata per la ricerca di
10 %   corrispondenze e per il calcolo della traslazione ottima.
11 %   theta: soglia di distanza angolare utilizzata per la ricerca di
12 %   corrispondenze e per il calcolo della rotazione ottima.
13 % OUTPUT:
14 %   M: mappa costituita da un insieme di segmenti allineati.
15 function[M] = allinea_scansioni(S, gamma, sigma, eps, delta, theta)
16
17   % Inizializza M all'insieme vuoto.
18   M = {};
19
20   % Estrae la prima scansione, la sua posizione e i suoi segmenti.
21   scan_prec = estrai_prima_scansione(S);
22   pos_prec = estrai_posizione(scan_prec);
23   segmenti = estrai_segmenti(scan_prec, gamma);
24
25   % Aggiunge i segmenti della prima scansione alla mappa M.
26   M = M U {segmenti};
27
28   % Estrae i segmenti filtrati della prima scansione e li aggiunge
29   % all'insieme dei segmenti significativi.
30   segmenti = filtra_segmenti(segmenti, sigma);
31   segmenti_significativi = segmenti;
32
33   % Elimina la prima scansione dall'insieme delle scansioni S.
34   S = S \ {scan_prec};
35
36   % Cicla su tutte le scansioni presenti in S.
37   for s in S
38
39     % Estrae la posizione p della scansione corrente s.
40     p = estrai_posizione(s);
41
42     % Controlla che la scansione corrente s sia esente da rotazioni
43     % e che la posizione nella quale è stata rilevata si trovi a una
44     % distanza superiore a una soglia eps rispetto alla posizione
45     % in cui era stata rilevata la precedente scansione.
46     if distanza_euclidea(p, pos_prec) >= eps and non_sta_ruotando(s)
47
48       % Estrae i segmenti della scansione corrente s.
49       segmenti = estrai_segmenti(s, gamma);
50
51       % Estrae i segmenti filtrati della scansione corrente s, necessari
52       % per effettuare l'allineamento.
53       segmenti_temporanei = filtra_segmenti(segmenti, sigma);
54
55       % Se il robot ha appena terminato una rotazione rilevante ...
56       if rotazione_rilevante(s)
57
58         % Aggiorna la struttura portante della mappa.
59         struttura_portante = aggiorna_struttura_portante(struttura_portante,
60                              segmenti_significativi, delta);
61
62       end
63
64       % Se la struttura portante è costituita da almeno una retta ...
```

```
65        if dimensione(struttura_portante) > 0
66
67          % Calcola le corrispondenze tra le rette della struttura portante
68          % e i segmenti filtrati della scansione corrente. Le corrispondenze
69          % vengono calcolate sulla base di due parametri delta e theta.
70          corrispondenze = cerca_corrispondenze(struttura_portante,
71                           segmenti_temporanei, delta, theta);
72
73          % Se viene trovata almeno una corrispondenza ...
74          if dimensione(corrispondenze) > 0
75
76            % Calcola la rotazione ottima che permette di allineare i
77            % segmenti filtrati della scansione corrente alle rette
78            % della struttura portante della mappa.
79            rotazione = trova_rotazione_ottima(corrispondenze, theta);
80
81            % Calcola la traslazione ottima che permette di sovrapporre
82            % i segmenti filtrati della scansione corrente alle rette
83            % della struttura portante della mappa.
84            traslazione = trova_traslazione_ottima(corrispondenze, delta);
85
86          end
87
88        end
89
90        % Aggiunge i segmenti filtrati della scansione corrente all'insieme
91        % dei segmenti significativi.
92        segmenti_significativi = segmenti_significativi U {segmenti_temporanei};
93
94        % Semplifica l'insieme dei segmenti significativi in modo da
95        % mantenerne in memoria il minore numero possibile.
96        segmenti_significativi = riduci(segmenti_significativi);
97
98        % Applica la rotazione e la traslazione ottima a tutti i segmenti
99        % della scansione corrente.
100       segmenti = applica_trasformazione(segmenti, rotazione, traslazione);
101
102       % Aggiunge i segmenti allineati alla mappa M.
103       M = M U {segmenti};
104
105       % La posizione corrente diventa la posizione precedente nella
106       % prossima iterazione.
107       pos_prec = p;
108
109     end
110
111   end
112
113 end
```

5.3 Algoritmo Basic

Il metodo Basic da noi realizzato, descritto nella sezione 4.1, può essere eseguito sia on-line che off-line. In entrambe le versioni il metodo riceve in ingresso, oltre alla mappa ridondante M (parziale nel caso della versione on-line, globale nel caso della versione off-line), due parametri θ e δ, che influiscono sulla ricerca delle corrispondenze. Ciascun segmento è rappresentato dal seguente insieme di parametri:

$$s = \{(x_1, y_1), (x_2, y_2), length, (x_c, y_c), angle, enabled\} \tag{5.1}$$

dove:

- (x_1, y_1) e (x_2, y_2) rappresentano le coordinate dei punti estremi del segmento;
- *length* rappresenta la lunghezza del segmento;
- (x_c, y_c) rappresentano le coordinate del punto centrale del segmento;
- *angle* rappresenta l'angolo compreso fra l'asse delle ascisse e il segmento;
- *enabled* rappresenta una variabile booleana che indica se il segmento fa parte oppure no della mappa corrente.

La rappresentazione differisce da quella definita nella Sezione 4.1 per la presenza del parametro *enabled*. Data una mappa di partenza M fortemente ridondante, il metodo itera più volte sull'insieme di segmenti fino a quando non trova più coppie di segmenti corrispondenti e, di conseguenza, fino a quando la mappa non subisce più modifiche. Per ogni coppia di segmenti il metodo controlla se questi verificano le condizioni di fusione e, se ciò accade, richiama l'algoritmo di fusione, il quale provvede ad aggiornare il primo segmento della coppia (quello per cui si stanno cercando i segmenti corrispondenti) e a eliminare il secondo.

Mostriamo ora lo pseudocodice dell'algoritmo che abbiamo implementato relativamente a questo metodo di fusione:

```
1 % FUNZIONE: fusione_mappa
2 %   Restituisce la mappa M ottenuta dalla fusione di M rispetto ai
3 %   parametri theta e delta.
4 % INPUT:
5 %   M: mappa iniziale;
6 %   theta: soglia distanza angolare;
7 %   delta: soglia distanza spaziale.
```

```
8  % OUTPUT:
9  %   M: mappa ottenuta applicando il metodo di fusione con parametri
10 %   theta e delta sulla mappa di partenza M.
11 function[M] = fusione_mappa(M, theta, delta)
12
13   convergenza = false;
14   corrispondenza_trovata = false;
15
16   % Continua a iterare sui segmenti della mappa fino a quando trova
17   % almeno una corrispondenza, ossia fino a quando esiste ancora una
18   % coppia di segmenti corrispondenti che possono essere fusi; in
19   % caso contrario termina.
20   while not(convergenza)
21
22       for a in M
23
24       for b in M
25
26         % Controlla se i segmenti soddisfano le condizioni di fusione.
27         if segmenti_corrispondenti(a, b, theta, delta)
28
29           % I segmenti soddisfano le condizioni di fusione: applica
30           % l'algoritmo di fusione e fonde i due segmenti.
31           % 1. Aggiorna i parametri del segmento a per tenere conto
32           %    dell'avvenuta fusione con un altro segmento.
33           a = fondi_segmenti(a, b);
34
35           % 2. Elimina il segmento b dai segmenti della mappa.
36           b("enabled") = false;
37           corrispondenza_trovata = true;
38
39         end
40
41       end
42
43     end
44
45     % Controlla se è stata trovata almeno una coppia di segmenti
46     % corrispondenti.
47     if not(corrispondenza_trovata)
48
49       % Nessuna nuova corrispondenza trovata: termina.
50       convergenza = true;
51
52     else
53
54       % Almeno una nuova corrispondenza trovata: reimposta le variabili
55       % per un'ulteriore iterazione del metodo.
56       corrispondenza_trovata = false;
57
58     end
59
60   end
61
62   return M;
63
```

```
64 end
```

5.3.1 Algoritmo di verifica delle condizioni di fusione

Il metodo considera due segmenti a e b corrispondenti quando sono rispettate le seguenti due condizioni:

1. la distanza euclidea tra almeno una coppia di punti estremi di a e di b è inferiore a una soglia δ;
2. la distanza angolare tra i due segmenti è inferiore a una soglia θ.

Come abbiamo spiegato nella Sezione 4.1.1, è importante notare che le condizioni di fusione, definite in questo modo, non riescono a catturare tutte le corrispondenze fra segmenti: ciò influisce sul numero di riduzioni effettuate.

Mostriamo ora lo pseudocodice dell'algoritmo di verifica delle condizioni di fusione:

```
1  % FUNZIONE: segmenti_corrispondenti
2  %   Verifica se due segmenti a e b soddisfano le condizioni di fusione
3  %   rispetto ai parametri theta e delta.
4  % INPUT:
5  %   a: segmento;
6  %   b: segmento;
7  %   theta: soglia distanza angolare;
8  %   delta: soglia distanza spaziale.
9  % OUTPUT:
10 %   corrispondenza: variabile booleana che indica se i segmenti a e b
11 %   sono corrispondenti o no.
12 function[corrispondenza] = segmenti_corrispondenti(a, b, theta, delta)
13
14   % Inizializza corrispondenza a false.
15   corrispondenza = false;
16
17   % Calcola la minima distanza tra un punto estremo del segmento a
18   % e un punto estremo del segmento b.
19   distanza_min = minima_distanza(a,b);
20
21   % I due segmenti sono corrispondenti se:
22   % 1. la differenza angolare tra a e b è inferiore a theta;
23   % 2. la minima distanza tra una coppia di punti estremi di a e
24   %    di b è inferiore a delta.
25   if abs(a("angle") - b("angle")) < theta and distanza_min < delta
26
27     % I segmenti sono corrispondenti: aggiorna corrispondenza a true.
28     corrispondenza = true;
29
30   end
31
32   return corrispondenza;
```

```
33
34 end
```

5.3.2 Algoritmo di fusione

Dati due segmenti a e b corrispondenti, l'algoritmo di fusione aggiorna i parametri del primo segmento della coppia (quello per cui è stata trovata una corrispondenza) ed elimina il secondo. L'aggiornamento prevede:

1. il calcolo della nuova direzione del segmento come media pesata (rispetto alle lunghezze dei segmenti) delle direzioni di a e di b;
2. il calcolo del punto, come media pesata (rispetto alle lunghezze) dei centri di a e di b, che, insieme alla direzione calcolata precedentemente, definisce la retta su cui giace il nuovo segmento;
3. l'estrazione degli estremi del nuovo segmento a partire dalle proiezioni degli estremi di a e b sulla retta definita prima.

Mostriamo ora lo pseudocodice dell'algoritmo di fusione:

```
1  % FUNZIONE: fondi_segmenti
2  %   Data una coppia di segmenti ne estrae uno.
3  % INPUT:
4  %   a: primo segmento corrispondente;
5  %   b: secondo segmento corrispondente;
6  % OUTPUT:
7  %   c: segmento estratto.
8  function[c] = fondi_segmenti(a, b)
9
10   % Dati gli angoli del segmento a e del segmento b, calcola la
11   % direzione della retta su cui giace il segmento c come media
12   % pesata (rispetto alle lunghezze dei segmenti) degli angoli
13   % di a e di b.
14   direzione_retta = (a("length") * a("angle") + b("length") * b("angle")) /
15                     (a("length") + b("length"));
16
17   % Dati i centri dei due segmenti, calcola il punto per cui deve
18   % passare la retta con direzione direzione_retta come media pesata
19   % (rispetto alle lunghezze dei segmenti) dei centri di a e di b.
20   punto_retta("x") = (a("length") * a("xc") + b("length") * b("xc")) /
21                      (a("length") + b("length"));
22   punto_retta("y") = (a("length") * a("yc") + b("length") * b("yc")) /
23                      (a("length") + b("length"));
24
25   % Dati un punto e una direzione estrae la retta r.
26   r = estrai_retta(punto_retta, direzione_retta);
27
28   % Proietta il segmento a sulla retta r.
29   a' = proietta_segmento(a, r);
30
31   % Proietta il segmento b sulla retta r.
```

```
32    b' = proietta_segmento(b, r);
33
34    % Estrae a partire dalle proiezioni gli estremi del segmento c.
35    c = estrai_estremi(a', b');
36
37    return c;
38
39  end
```

5.4 Algoritmo Hybrid

Nella Sezione 4.2 abbiamo descritto un metodo di riduzione dei segmenti online. Il metodo riceve in ingresso, oltre alla mappa corrente dell'ambiente M e alla nuova scansione S, due parametri $\bar{\theta}$ e $\bar{\delta}$, i quali influenzano la verifica delle condizioni di fusione di coppie di segmenti, di cui il primo appartiene alla nuova scansione e l'altro alla mappa. Come già anticipato nella Sezione 4.2.2, ciascun segmento è rappresentato dal seguente insieme di parametri:

$$s = \{(x_1, y_1), (x_2, y_2), length, \rho, \theta, count\} \tag{5.2}$$

dove:

- (x_1, y_1) e (x_2, y_2) rappresentano le coordinate dei punti estremi del segmento;
- $length$ rappresenta la lunghezza del segmento;
- ρ e θ rappresentano la retta su cui giace il segmento nello spazio di Hough;
- $count$ rappresenta il peso di un segmento durante la procedura di fusione.

Il metodo estrae, per ogni segmento a di S, i segmenti della mappa a cui è vicino, secondo la misura di vicinanza descritta nella Sezione 4.2.3, e cerca fra questi, partendo da quello a cui è più vicino fino ad arrivare a quello a cui è meno vicino, un segmento che gli corrisponde, ossia un segmento con cui soddisfa le condizioni di fusione. Se questo si verifica, il metodo applica l'algoritmo di fusione, il quale prevede l'aggiornamento dei parametri del segmento della mappa. Se non viene trovata alcuna corrispondenza, o perché per a non esistono segmenti nella mappa a cui è vicino o perché con nessuno di questi soddisfa le condizioni di fusione, il metodo aggiunge a alla mappa M solo se questo rappresenta una caratteristica nuova e rilevante dell'ambiente.

Mostriamo ora lo pseudocodice dell'algoritmo che abbiamo implementato relativamente a questo metodo di fusione:

```
% FUNZIONE: fusione_scansione_mappa
%   Fonde la nuova scansione S con la mappa corrente M.
% INPUT:
%   M: mappa corrente;
%   S: nuova scansione acquisita;
%   theta: soglia distanza angolare;
%   delta: soglia distanza spaziale.
% OUTPUT:
%   M: mappa ottenuta attraverso la fusione di S con i segmenti di M.
function[M] = fusione_scansione_mappa(M, S, theta, delta)

  for a in S

    corrispondenza_trovata = false;

    if M != {}

      % Trova i segmenti della mappa di cui a è vicino
      segmenti_vicini = trova_segmenti_vicini(a, M, delta);

      if segmenti_vicini != {}

        % Ordina i segmenti della mappa, di cui a è vicino, in base
        % alla vicinanza (in modo decrescente).
        segmenti_vicini = ordina_per_vicinanza(segmenti_vicini);

        for b in segmenti_vicini

          % Verifica se a e b soddisfano le condizioni di fusione.
          if segmenti_corrispondenti(a, b, theta, delta)

            % I segmenti soddisfano le condizioni di fusione: applica
            % l'algoritmo di fusione.
            % 1. Aggiorna i parametri del segmento della mappa.
            b = fusione_segmenti(a, b);
            corrispondenza_trovata = true;

            % 2. Siccome per il segmento a è stato trovato un segmento
            %    b della mappa che gli corrisponde, con cui è stato fuso,
            %    passa a cercare i segmenti corrispondenti per il prossimo
            %    segmento della scansione.
            break;

          end

        end

      end

    end

    % Se a non ha soddisfatto le condizioni di fusione con alcun
```

```
53      % segmento della mappa M, verifica se rappresenta una nuova
54      % caratteristica dell'ambiente e se questa è rilevante per la
55      % mappa. Se questo accade, aggiunge a alla mappa.
56      if not(corrispondenza_trovata) and a("length") >= delta
57
58        % Aggiunge il segmento a M.
59        M = M U {a};
60
61      end
62
63    end
64
65    return M;
66
67  end
```

5.4.1 Algoritmo di estrazione dei segmenti vicini

Come spiegato nella Sezione 4.2.4, la ricerca dei segmenti della mappa M corrispondenti al segmento a della scansione S viene effettuata solo su un sottoinsieme dei segmenti della mappa, formato dai segmenti di cui a è vicino, secondo la misura di vicinanza definita nella Sezione 4.2.3. Tale misura di vicinanza considera a vicino di b se almeno una delle due seguenti condizioni è verificata:

- la distanza euclidea tra almeno una coppia di estremi di a e di b è inferiore a una soglia $\bar{\delta}$;
- la proiezione di almeno un estremo di a su b cade all'interno di b e la distanza euclidea tra l'estremo di a e l'estremo di a proiettato è inferiore a una soglia $\bar{\delta}$.

Mostriamo ora lo pseudocodice dell'algoritmo di estrazione dei segmenti vicini:

```
1  % FUNZIONE: trova_segmenti_vicini
2  %   Trova l'insieme dei segmenti di M di cui a è vicino.
3  % INPUT:
4  %   M: mappa corrente;
5  %   a: segmento della scansione;
6  %   delta: soglia distanza spaziale.
7  % OUTPUT:
8  %   segmenti_vicini: insieme dei segmenti della mappa M che hanno
9  %   a come vicino.
10 function[segmenti_vicini] = trova_segmenti_vicini(a, M, delta)
11
12   segmenti_vicini = {};
13
14   % Per ogni segmento b della mappa verifica se sono rispettate le
15   % condizioni di vicinanza con il segmento a.
16   for b in M
```

```
17
18      % Calcola la minima distanza tra una coppia di punti estremi di a
19      % e di b.
20      minima_distanza_coppia_estremi = minima_distanza(a, b);
21
22      % Estrae la retta su cui giace il segmento b della mappa a partire
23      % da un suo punto estremo e dall'angolo compreso fra il segmento e
24      % l'asse delle ascisse.
25      pendenza = -1 / tan(b("theta"));
26      punto = b("(x1,y1)");
27      retta_b = estrai_retta(punto, pendenza);
28
29      % Proietta il segmento a sulla retta.
30      a' = proietta_segmento(a, retta_b);
31
32      % Calcola le due distanze euclidee tra i punti estremi di a e i punti
33      % estremi di a proiettati.
34      distanza_start = distanza_euclidea(a("(x1,y1)"), a'("(x1,y1)"));
35      distanza_end = distanza_euclidea(a("(x2,y2)"), a'("(x2,y2)"));
36
37      % La minima distanza tra le tre è quella che determina la vicinanza
38      % del segmento b rispetto al segmento a.
39      distanza_minima =
40        min(minima_distanza_coppia_estremi, distanza_start, distanza_end);
41
42      if distanza_minima < delta
43
44        % Aggiunge il segmento b all'insieme dei segmenti vicini.
45        segmenti_vicini = segmenti_vicini U {b};
46
47      end
48
49    end
50
51    return segmenti_vicini;
52
53  end
```

5.4.2 Algoritmo di verifica delle condizioni di fusione

La verifica delle condizioni di fusione per una coppia di segmenti consiste nel confronto tra i parametri ρ e θ delle rette (espresse in forma di Hesse) su cui giacciono i segmenti. Le condizioni sono verificate se:

1. la differenza tra i parametri ρ dei due segmenti è inferiore a una soglia $\bar{\delta}$;

2. la differenza tra i parametri θ dei due segmenti è inferiore a una soglia $\bar{\theta}$.

Mostriamo ora lo pseudocodice dell'algoritmo di verifica delle condizioni di fusione:

```
% FUNZIONE: segmenti_corrispondenti
%   Verifica se due segmenti a e b soddisfano le condizioni di fusione
%   rispetto ai parametri theta e delta.
% INPUT:
%   a: segmento;
%   b: segmento;
%   theta: soglia distanza angolare;
%   delta: soglia distanza spaziale.
% OUTPUT:
%   corrispondenza: variabile booleana che indica se i segmenti a e b
%   sono corrispondenti o no.
function[corrispondenza] = segmenti_corrispondenti(a, b, theta, delta)

  % Inizializza corrispondenza a false.
  corrispondenza = false;

  if abs(a("rho") - b("rho")) < delta and abs(a("theta") - b("theta")) < theta

    % I segmenti sono corrispondenti: aggiorna corrispondenza a true.
    corrispondenza = true;

  end

  return corrispondenza;

end
```

5.4.3 Algoritmo di fusione

L'algoritmo di fusione fonde una coppia di segmenti corrispondenti $a \in S$ e $b \in M$, aggiornando i parametri del segmento b, ossia di quello che appartiene alla mappa. L'aggiornamento del segmento b prevede:

1. l'incremento della variabile *count* di un'unità;
2. il ricalcolo dei parametri ρ e θ come media pesata (rispetto alle variabili *count*) dei parametri ρ e θ dei due segmenti;
3. il ricalcolo dei punti estremi del segmento;

Mostriamo ora lo pseudocodice dell'algoritmo di fusione:

```
% FUNZIONE: fusione_segmenti
%   Data una coppia di segmenti ne estrae uno.
% INPUT:
%   a: segmento della scansione;
%   b: segmento della mappa.
% OUTPUT:
%   c: segmento estratto.
function[c] = fusione_segmenti(a, b)

  % Calcola la variabile count di c a partire dalla variabile count di b
```

```
11    % (il segmento della mappa).
12    c("count") = b("count") + 1;
13
14    % Calcola i parametri rho e theta di c.
15    c("rho") = (a("count") * a("rho") + b("count") * b("rho")) /
16               (a("count") + b("count"));
17    c("theta") = (a("count") * a("theta") + b("count") * b("theta")) /
18                 (a("count") + b("count"));
19
20    % Estrae la retta su cui giace il segmento c.
21    pendenza = -1 / tan(theta);
22    punto = [(c("rho") * cos(c("theta"))) (c("rho") * sin(c("theta")))];
23    retta_c = estrai_retta(punto, pendenza);
24
25    % Proietta il segmento a sulla retta.
26    a' = proietta_segmento(a, retta_c);
27
28    % Proietta il segmento b sulla retta.
29    b' = proietta_segmento(b, retta_c);
30
31    % Estrae a partire dalle proiezioni gli estremi del segmento c.
32    c("(x1,y1)") = estrai_primo_estremo(a'("(x1,y1)"), a'("(x2,y2)"),
33                                        b'("(x1,y1)"), b'("(x2,y2)"));
34    c("(x2,y2)") = estrai_secondo_estremo(a'("(x1,y1)"), a'("(x2,y2)"),
35                                          b'("(x1,y1)"), b'("(x2,y2)"));
36
37  end
```

5.5 Algoritmo Mean Shift Clustering

Nella Sezione 4.3 abbiamo descritto un metodo di fusione off-line basato su un approccio chiamato Mean Shift Clustering, il quale raggruppa segmenti simili nello stesso cluster. Rispetto a quanto anticipato nella Sezione 4.3.2, per ciascun segmento non è sufficiente memorizzare il suo punto centrale e la sua direzione: a livello implementativo è infatti necessario memorizzare delle informazioni aggiuntive, relative ad esempio al cluster di appartenenza e all'angolo medio dei segmenti appartenenti a uno specifico cluster. Ciascun segmento è pertanto rappresentato dal seguente insieme di parametri:

$$s = \{(x_1, y_1), (x_2, y_2), (x_c, y_c), angle, cluster, \overline{angle}, subcluster, (\overline{x}, \overline{y})\} \tag{5.3}$$

dove:

- (x_1, y_1) e (x_2, y_2) rappresentano le coordinate dei punti estremi del segmento;
- (x_c, y_c) rappresentano le coordinate del punto centrale del segmento;

- *angle* rappresenta l'angolo compreso fra l'asse delle ascisse e il segmento;
- *cluster* rappresenta l'identificativo del cluster a cui appartiene il segmento;
- $\overline{angle}$ rappresenta l'angolo medio del cluster;
- *subcluster* rappresenta l'identificativo del sotto-cluster a cui appartiene il segmento;
- $(\overline{x}, \overline{y})$ rappresentano le coordinate medie del punto centrale del sotto-cluster.

Prima di applicare il metodo, i segmenti che eccedono una certa lunghezza vengono suddivisi in segmenti più corti, per le ragioni spiegate nella Sezione 4.3.2. Il metodo è sostanzialmente suddiviso in due fasi, le quali utilizzano entrambe il Mean Shift Clustering: la prima fase individua cluster costituiti da segmenti aventi una direzione simile e caratterizzati da una reciproca vicinanza spaziale; la seconda fase individua i sotto-cluster di ogni cluster, raggruppando segmenti collineari vicini tra loro. La fase finale è una semplice procedura geometrica: per ogni sotto-cluster viene determinato un segmento rappresentativo di tutti i segmenti appartenenti a quello specifico sotto-cluster.

Mostriamo ora lo pseudocodice dell'algoritmo che abbiamo implementato relativamente a questo metodo di fusione:

```
% FUNZIONE: fusione_mappa
%   Restituisce la mappa M' ottenuta dalla fusione di M rispetto al
%   parametro sigma.
% INPUT:
%   M: mappa a segmenti;
%   sigma: valore di minimo dettaglio rilevabile.
% OUTPUT:
%   M': mappa ottenuta applicando il metodo di fusione con parametro
%   sigma sulla mappa di partenza M.
function[M'] = fusione_mappa(M, sigma)

  % Suddivide i segmenti che eccedono una certa lunghezza.
  M = splitta_segmenti(M, sigma);

  % Prima fase: individua cluster costituiti da segmenti aventi
  % una direzione simile e caratterizzati da una reciproca
  % vicinanza spaziale.
  M = rileva_direzioni_comuni(M, sigma);

  % Seconda fase: individua sotto-cluster costituiti da segmenti
  % collineari vicini tra loro.
```

```
22    M = raggruppa_segmenti_collineari(M, sigma);
23
24    % Fase finale: fusione.
25    M' = fondi_segmenti(M);
26
27    return M';
28
29  end
```

Vogliamo far notare che le prime due fasi non fanno altro che aggiornare i parametri dei segmenti appartenenti alla mappa globale M, aggiungendo determinate informazioni: ad esempio, la prima fase assegna a ogni segmento l'identificativo del cluster a cui appartiene e l'angolo medio.

5.5.1 Algoritmo di rilevamento delle direzioni comuni

Come abbiamo già anticipato nella Sezione 4.3.3, l'obiettivo della prima fase è di raggruppare i segmenti in base alla loro direzione e regione spaziale: per fare questo viene utilizzato il Mean Shift Clustering caratterizzato da un opportuno nucleo gaussiano con deviazione standard σ, che rappresenta il minimo dettaglio significativo della mappa. Il parametro relativo alla bandwidth h, che serve a convertire la distanza angolare, inizialmente espressa in gradi, in una distanza spaziale, viene posto uguale a $10°/200mm$: tale parametro risulta costante in tutti gli esperimenti. Applichiamo quindi il Mean Shift Clustering fino a raggiungere la convergenza, assegnando in seguito a ogni segmento l'identificativo del cluster a cui appartiene e l'angolo medio.

Mostriamo ora lo pseudocodice dell'algoritmo di rilevazione delle direzioni comuni:

```
1  % FUNZIONE: rileva_direzioni_comuni
2  %   Restituisce la mappa M aggiornata coi parametri relativi
3  %   all'identificativo del cluster di appartenenza e all'angolo
4  %   medio per ogni segmento.
5  % INPUT:
6  %   M: mappa a segmenti;
7  %   sigma: valore di minimo dettaglio rilevabile.
8  % OUTPUT:
9  %   M: mappa aggiornata.
10 function[M] = rileva_direzioni_comuni(M, sigma)
11
12   % Inizializza bandwidth h, costante in tutti gli esperimenti.
13   h = (pi / 18) / 200;
14
15   % Inizializza l'insieme dei centri di densità T con i segmenti
16   % divisi appartenenti alla mappa rappresentati mediante i
17   % loro punti centrali e direzioni.
18   T = M("(xc,yc),angle");
19
```

```
20    % Inizializza la massima variazione dell'ultima iterazione.
21    lastMax = 0;
22
23    % Ripete fino a che non viene raggiunta la convergenza.
24    while true
25
26      % Resetta la massima variazione per l'iterazione corrente.
27      max = 0;
28
29      for a in T
30
31        % Resetta il valore di Mean Shift per il segmento corrente.
32        numeratore_mean_shift = [0 0 0];
33        denominatore_mean_shift = 0;
34
35        for b in M
36
37          % Calcola la distanza tra il centro di densità che sta
38          % convergendo e il segmento della mappa rappresentato
39          % mediante il suo punto centrale e la sua direzione.
40          distance = sqrt((a("xc") - b("xc"))^2 + (a("yc") - b("yc"))^2 +
41                          (min(abs(a("angle") - b("angle")),
42                                   pi - abs(a("angle") - b("angle"))) / h)^2);
43          nucleo = exp(-distance^2 / (2 * sigma^2));
44          numeratore_mean_shift = numeratore_mean_shift + nucleo * b;
45          denominatore_mean_shift = denominatore_mean_shift + nucleo;
46
47        end
48
49        % Calcola il rapporto tra numeratore e denominatore.
50        rapporto_mean_shift = numeratore_mean_shift / denominatore_mean_shift;
51
52        % Risultato del Mean Shift.
53        mean_shift = abs(rapporto_mean_shift - a);
54
55        % Aggiorna il centro di densità.
56        a = rapporto_mean_shift;
57
58        % Seleziona la massima differenza tra l'angolo di un centro di
59        % densità e l'angolo di un segmento della mappa iniziale.
60        if risultato_mean_shift("angle") > max
61          max = risultato_mean_shift("angle") / h;
62        end
63
64      end
65
66      % Se è stata raggiunta la convergenza ...
67      if abs(max - lastMax) < 10.0
68
69        % Termina il ciclo while e passa all'assegnazione dei cluster.
70        break;
71
72      % Altrimenti ...
73      else
74
75        % Memorizza l'ultima massima variazione e continua il ciclo while.
```

```
76        lastMax = max;
77
78      end
79
80    end
81
82    % T contiene ora tutti i centri di densità che definiscono i cluster.
83    % Assegna a ogni segmento della mappa l'identificativo del cluster a
84    % cui appartiene.
85    M("cluster") = assegna_cluster(T);
86
87    % Assegna a ogni segmento della mappa l'angolo medio anglem dei
88    % segmenti appartenenti allo stesso cluster.
89    M("anglem") = assegna_angolo_medio(T);
90
91    return M;
92
93  end
```

5.5.2 Algoritmo di raggruppamento di segmenti collineari

Come abbiamo già anticipato nella Sezione 4.3.4, la seconda fase suddivide ogni cluster trovato nella prima fase in sotto-cluster di segmenti collineari, utilizzando l'informazione sulla direzione del cluster. Dato che ogni segmento in un cluster ha una direzione assegnata, è sufficiente rappresentare il segmento in uno spazio bidimensionale: la seconda fase del Mean Shift Clustering opera quindi in uno spazio bidimensionale, definito come la posizione dei punti di un singolo cluster. Anche in questa fase utilizziamo un nucleo gaussiano, con deviazioni standard σ_x, σ_y derivate dal parametro σ definito nella prima fase. Per ogni cluster ottenuto dalla prima fase, applichiamo quindi il Mean Shift Clustering fino a raggiungere la convergenza. Una volta raggiunta la convergenza, assegniamo a ogni segmento l'identificativo del sotto-cluster a cui appartiene e le coordinate del punto centrale medio.

Mostriamo ora lo pseudocodice dell'algoritmo di raggruppamento di segmenti collineari:

```
1  % FUNZIONE: raggruppa_segmenti_collineari
2  %   Restituisce la mappa M aggiornata coi parametri relativi
3  %   all'identificativo del sotto-cluster di appartenenza e alle
4  %   coordinate del punto centrale medio.
5  % INPUT:
6  %   M: mappa a segmenti divisa in cluster;
7  %   sigma: valore di minimo dettaglio rilevabile.
8  % OUTPUT:
9  %   M: mappa aggiornata.
10 function[M] = raggruppa_segmenti_collineari(M, sigma)
11
12   % C è l'insieme di tutti i cluster presenti nella mappa.
```

```
13    % Ciascun cluster è costituito da un insieme di segmenti.
14    C = estrai_cluster(M);
15
16    % Per ogni cluster.
17    for cluster in C
18
19      % Estrae l'angolo del cluster.
20      angolo = cluster("anglem");
21
22      % Inizializza sigmax, sigmay.
23      sigmax = 2 * sigma;
24      sigmay = sigma / 10;
25
26      % Calcola la matrice da cui deriva il nucleo gaussiano di
27      % questa seconda fase.
28      a = (cos(angolo))^2 / (2 * sigmax^2) +
29          (sin(angolo))^2 / (2 * sigmay^2);
30      b = -sin(2 * angolo) / (4 * sigmax^2) +
31          sin(2 * angolo) / (4 * sigmay^2);
32      c = (sin(angolo))^2 / (2 * sigmax^2) +
33          (cos(angolo))^2 / (2 * sigmay^2);
34      matrice = [a b; b c];
35
36      % Inizializza l'insieme dei centri di densità con i segmenti
37      % appartenenti allo specifico cluster rappresentati mediante i
38      % i loro punti centrali.
39      T = cluster("(xc,yc)");
40
41      % Inizializza la massima variazione dell'ultima iterazione.
42      lastMax = [0 0];
43
44      % Ripete fino a che non viene raggiunta la convergenza.
45      while true
46
47        % Resetta la massima variazione per l'iterazione corrente.
48        max = [0 0];
49
50        for a in T
51
52          % Resetta il valore di Mean Shift per il segmento corrente.
53          numeratore_mean_shift = [0 0];
54          denominatore_mean_shift = 0;
55
56          for b in cluster
57
58            % Calcola la distanza tra il centro di densità che sta
59            % convergendo e il segmento del cluster rappresentato
60            % mediante il suo punto centrale.
61            diff = a("(xc,yc)") - b("(xc,yc)");
62            nucleo = exp(-(1/2) * diff * matrice * trasposta(diff));
63            numeratore_mean_shift = numeratore_mean_shift + nucleo * b;
64            denominatore_mean_shift = denominatore_mean_shift + nucleo;
65
66          end
67
68          % Calcola il rapporto tra numeratore e denominatore.
```

```
69          rapporto_mean_shift =
70                    numeratore_mean_shift / denominatore_mean_shift;
71
72          % Risultato del Mean Shift.
73          mean_shift = abs(rapporto_mean_shift - a);
74
75          % Aggiorna il centro di densità.
76          a = rapporto_mean_shift;
77
78          % Seleziona la massima differenza tra la coordinata x di
79          % un centro di densità e la coordinata x del punto centrale
80          % di un segmento del cluster.
81          if risultato_mean_shift("xc") > max("xc")
82            max("xc") = risultato_mean_shift("xc");
83          end
84
85          % Seleziona la massima differenza tra la coordinata y di
86          % un centro di densità e la coordinata y del punto centrale
87          % di un segmento del cluster.
88          if risultato_mean_shift("yc") > max("yc")
89            max("yc") = risultato_mean_shift("yc");
90          end
91
92        end
93
94        % Se è stata raggiunta la convergenza ...
95        if abs(max - lastMax) < 10.0
96
97          % Termina il ciclo while e passa all'assegnazione
98          % dei sotto-cluster.
99          break;
100
101       % Altrimenti ...
102       else
103
104         % Memorizza l'ultima massima variazione e continua
105         % il ciclo while.
106         lastMax = max;
107
108       end
109
110     end
111
112     % T contiene ora tutti i centri di densità che definiscono
113     % i sotto-cluster. Assegna a ogni segmento l'identificativo
114     % del sotto-cluster a cui appartiene.
115     M("cluster") = assegna_sottocluster(T, sigma);
116
117     % Assegna a ogni segmento il punto centrale medio (xm,ym)
118     % dei segmenti appartenenti allo stesso sotto-cluster.
119     M("(xm,ym)") = assegna_punto_centrale_medio(T);
120
121   end
122
123   return M;
124
```

```
125 end
```

5.5.3 Algoritmo di fusione

Dopo aver calcolato i sotto-cluster, la fusione è una semplice procedura geometrica. A ogni sotto-cluster corrisponde una retta rappresentativa, avente una direzione derivata dalla prima fase di clustering e passante per il punto centrale derivato dalla seconda fase di clustering. Per calcolare il risultato finale del processo di fusione, proiettiamo tutti i segmenti di un singolo sotto-cluster sulla retta rappresentativa e fondiamo i segmenti sovrapposti in un unico segmento.

Mostriamo ora lo pseudocodice dell'algoritmo di fusione:

```
1  % FUNZIONE: fondi_segmenti
2  %   Estrae per ogni sotto-cluster un segmento rappresentativo di
3  %   tutti i segmenti appartenenti a quello specifico sotto-cluster.
4  % INPUT:
5  %   M: mappa a segmenti divisa in sotto-cluster.
6  % OUTPUT:
7  %   M': mappa ottenuta estraendo per ogni sotto-cluster un segmento
8  %   rappresentativo di tutti i segmenti appartenenti a quello specifico
9  %   sotto-cluster.
10 function[M'] = fondi_segmenti(M)
11
12   % Inizializza M' all'insieme vuoto.
13   M' = {};
14
15   % C è l'insieme di tutti i sotto-cluster presenti nella mappa.
16   % Ciascun sotto-cluster è costituito da un insieme di segmenti.
17   C = estrai_cluster(M);
18
19   for cluster in C
20
21     % Estrae dal singolo sotto-cluster un segmento rappresentativo di
22     % tutti i segmenti appartenenti a quello specifico sotto-cluster.
23     seg = estrai_segmento_rappresentativo(cluster);
24
25     % Aggiunge il segmento seg alla mappa finale M'.
26     M' = M' U {seg};
27
28   end
29
30   return M;
31
32 end
```

5.6 Algoritmo Fusion

Nella Sezione 4.4 abbiamo descritto un metodo che affronta il problema della fusione di una mappa M, rispetto a un valore di tolleranza ϵ, in termini di ricerca dell'insieme di rette di fusione che meglio approssimano M. Questo metodo di fusione, descritto dettagliatamente in [49] e già implementato, è stato importato nel nostro lavoro di ricerca al fine di confrontarne l'efficacia rispetto agli altri metodi di fusione. In questo metodo ciascun segmento è rappresentato dal seguente insieme di parametri:

$$s = \{(x_1, y_1), (x_2, y_2)\} \tag{5.4}$$

dove (x_1, y_1) e (x_2, y_2) rappresentano le coordinate dei punti estremi del segmento. Per ogni segmento è quindi sufficiente memorizzare i suoi punti estremi. Il metodo, a ogni passo di iterazione, cerca la retta di fusione r^* che meglio approssima una catena di segmenti ridondanti S con la massima lunghezza complessiva possibile. Successivamente, individua su r^* il segmento s_f che approssima S, aggiunge s_f alla mappa $M^{'}$ (che conterrà i segmenti ottenuti dalla fusione) e toglie S da M. Il meccanismo termina quando $M \neq \emptyset$ e quindi quando ogni segmento di M è stato approssimato da un segmento s_f individuato su una retta di fusione r^*.

Mostriamo ora lo pseudocodice dell'algoritmo che abbiamo importato relativamente a questo metodo di fusione:

```
% FUNZIONE: fusione_mappa
%   Ritorna la mappa M' ottenuta dalla fusione di M rispetto a eps.
% INPUT:
%   M: mappa a segmenti;
%   eps: valore di tolleranza.
% OUTPUT:
%   M': mappa ottenuta dalla fusione di M rispetto a eps.
function[M'] = fusione_mappa(M, eps)

  % Inizializza M' con l'insieme vuoto.
  M' = {};

  % Continua a ciclare finché esistono ancora segmenti in M
  % da fondere.
  while (M != {})

    % Estrae da M la retta di fusione r_star rispetto a eps
    % che include la catena di segmenti ridondanti catena_seg
    % con la massima lunghezza complessiva.
    [r_star, catena_seg] = estrai_r_star(M, eps);

    % Individua su r_star il segmento s_f che approssima la
    % catena di segmenti ridondanti.
    s_f = fusione_segmenti(catena_seg, r_star);
```

```
25
26      % Aggiunge a M' il segmento s_f
27      M' = M' U {s_f};
28
29      % Elimina catena_seg dall'insieme di segmenti ancora da
30      % fondere.
31      M = M \ catena_seg;
32
33    end
34
35    % Restituisce la mappa M' fusione di M rispetto a eps.
36    return M';
37
38  end
```

L'unico parametro da specificare per il metodo di fusione della mappa è il valore di tolleranza ϵ. Questo parametro è utilizzato esclusivamente per l'estrazione della retta di fusione r^* ottima (riga 20) e determina la larghezza della fascia di ogni retta di fusione. A ogni iterazione, il metodo fonde, in un unico segmento, la catena di segmenti ridondanti con lunghezza complessiva massima. Trovata la retta di fusione ottima r^*, la fusione dei segmenti (riga 24) si riduce nell'individuare i punti estremi del nuovo segmento come la coppia di punti più lontani appartenenti alle proiezioni dei segmenti della catena su r^*.

Mostriamo ora lo pseudocodice dell'algoritmo di fusione:

```
1  % FUNZIONE: fusione_segmenti
2  %   Approssima la catena di segmenti S con un unico segmento.
3  % INPUT:
4  %   S: catena di segmenti inclusa in r;
5  %   r: retta di fusione per S.
6  % OUTPUT:
7  %   seg: segmento che approssima l'insieme S.
8  function[seg] = fusione_segmenti(S, r)
9
10   % Calcola le proiezioni dei segmenti di S su r.
11   S_su_r = proiezione_seg_su_retta(S, r);
12
13   % r("m"): coefficiente angolare della retta r.
14   if r("m") >= 0
15     seg("x1") = min(S_su_r("x"))
16     seg("x2") = max(S_su_r("x"))
17   else if r('m') < 0
18     seg("x1") = max(S_su_r("x"))
19     seg("x2") = min(S_su_r("x"))
20   end
21
22   seg("y1") = min(S_su_r("y"))
23   seg("y2") = max(S_su_r("y"))
24
25   % I punti estremi di seg sono la coppia di punti
26   % più lontani tra i punti di S_su_r.
```

```
27    return seg;
28
29  end
```

5.6.1 Algoritmo di estrazione della retta di fusione ottima

L'estrazione della retta di fusione ottima è un problema complesso in quanto, dati una mappa M e un valore di tolleranza ϵ, potrebbero esistere anche infinite rette di fusione rispetto a ϵ in M. Un metodo esaustivo, che trovi l'insieme di tutte le rette di fusione e che cerchi tra queste la retta di fusione ottima, non è realizzabile. E' stato implementato dunque un meccanismo che, partendo da una soluzione iniziale, sposta questa soluzione sulla mappa in modo da includere la catena di segmenti ridondanti con la massima lunghezza complessiva. Viene scelta come soluzione iniziale la retta di supporto del segmento più lungo nella mappa e vengono imposti due vincoli sugli spostamenti possibili della retta di fusione:

1. sono consentite esclusivamente traslazioni che non ne modificano il coefficiente angolare;

2. il segmento più lungo deve appartenere alla catena di segmenti inclusa nella retta di fusione.

L'algoritmo implementato parte dalla retta di supporto del segmento più lungo $\overline{s}$ e, effettuando esclusivamente traslazioni, trova la retta di fusione r^* tale che:

1. r^* include la catena di segmenti ridondanti S con la massima lunghezza complessiva possibile;

2. $\overline{s} \in S$.

Mostriamo ora lo pseudocodice dell'algoritmo di estrazione della retta di fusione ottima:

```
1  % FUNZIONE: estrai_r_star
2  %   Ritorna la retta di fusione r_star tale che:
3  %   1. r_star ha il coeff angolare della retta di
4  %      supporto del seg più lungo in M;
5  %   2. r_star include la catena di segmenti ridondanti
6  %      con massima lunghezza complessiva possibile;
7  %   3. il segmento più lungo in M appartiene alla
8  %      catena di segmenti catena_seg.
9  % INPUT:
10 %   M: mappa a segmenti;
11 %   eps: valore di tolleranza.
12 % OUTPUT:
```

```
13 %   r_star: retta di fusione;
14 %   catena_seg: catena di segmenti ridondanti inclusa in r_star
15 %   con massima lunghezza complessiva.
16 function[r_star, catena_seg] = estrai_r_star(M, eps)
17
18   % Estrae il segmento più lungo in M.
19   seg_l = estrai_seg_più_lungo(M);
20
21   % Inizializza r_star con la retta di supporto del segmento più lungo.
22   r_seg_l = estrai_retta_di_supporto(seg_l);
23   r_star = r_seg_l;
24
25   % Estrae dalla mappa M la catena di segmenti catena_seg inclusa in
26   % r_star che contiene il segmento più lungo.
27   catena_seg = estrai_catena_seg_inclusa(M, r_star, eps, seg_l);
28
29   % catena_seg è un insieme ridondante di segmenti in quanto verifica
30   % entrambe le condizioni di fusione (catena_seg è inclusa in una retta
31   % di fusione ed è catena per tale retta).
32
33   % Calcola la lunghezza complessiva di catena_seg.
34   lung_max = lunghezza_compl_catena(catena_seg);
35
36   % Per ogni segmento m in M.
37   for m in M
38
39     % Calcola la distanza con segno dei punti estremi
40     % del segmento m da r_star.
41     % distanza_con_segno_punto_retta(p, r) con p=(xp, yp)
42     % e r: y-m*x-q=0 è calcolata in questo modo:
43     %
44     % distanza_con_segno_punto_retta(p, r) =
45     %                   (yp - m*xp - q) / sqrt (1 + m^2)
46     %
47     % La distanza con segno punto-retta si differenzia
48     % dalla distanza classica punto-retta per l'assenza
49     % del modulo al numeratore. In questo modo il
50     % modulo della distanza indica l'entità mentre il
51     % segno indica la direzione dello spostamento
52     % necessario affinché la retta passi per il punto.
53     dist_m_p1 = distanza_con_segno_punto_retta(m("p1"),r_star);
54     dist_m_p2 = distanza_con_segno_punto_retta(m("p2"),r_star);
55     if abs(dist_m_p1) > abs(dist_m_p2)
56       dist_m_max = dist_m_p1;
57     else
58       dist_m_max = dist_m_p2;
59     end
60
61     % dist_m_max contiene la traslazione necessaria affinché la
62     % retta r_star passi per il punto estremo più lontano di m.
63     % Se la distanza massima dei punti estremi di m rispetto a
64     % r_star è minore di eps allora m è già incluso dalla soluzione
65     % r_star e quindi passa al segmento successivo.
66     if abs(dist_m_max) < eps
67       continue;
68     end
```

```
69
70      % Altrimenti calcola la minima traslazione necessaria affinché
71      % il punto estremo più lontano di m sia incluso in r_star.
72      trasl = segno(dist_m_max) * (eps - abs(dist_m_max));
73
74      % Calcola r come traslazione di r_star rispetto a trasl.
75      r = trasla_retta(r_star, trasl);
76
77      % Controlla che seg_l sia incluso in r rispetto ad eps.
78      if (seg_incluso_in_r(seg_l, r, eps))
79
80        % Estrae dalla mappa M la catena di segmenti cs inclusa in r
81        % che contiene il segmento più lungo seg_l.
82        cs = estrai_catena_seg_inclusa(M, r, eps, seg_l);
83
84        % Calcola la lunghezza complessiva di cs.
85        l = lunghezza_compl_catena(cs);
86
87        % Se la nuova retta di fusione r comprende una catena di
88        % segmenti che ha una lunghezza complessiva maggiore rispetto
89        % alla soluzione precedente aggiorna la soluzione ottima.
90        if (l > lung_max)
91          r_star = r;
92          catena_seg = cs;
93          lung_max = l;
94        end
95
96      end
97
98    end
99
100   % Restituisce r_star e catena_seg.
101   return [r_star, catena_seg];
102
103 end
```

Capitolo 6

Realizzazioni sperimentali e valutazione

6.1 Implementazione del sistema

I metodi di fusione dei segmenti ridondanti sono stati implementati in modo indipendente e successivamente integrati all'interno di un sistema realizzato nell'ambiente di calcolo MATLAB®7.12.0.635 R2011a. Il sistema è eseguibile su qualunque piattaforma Windows, Mac OS, GNU Linux in cui è installata una versione di MATLAB®7.X.
Il sistema è stato progettato per essere modulare ed espandibile e si compone di:

- modulo *Gui.m* che fornisce l'interfaccia grafica con cui acquisire i dati in ingresso e visualizzare/manipolare i risultati frutto dell'applicazione di uno dei metodi di fusione;
- un modulo per ogni metodo di riduzione. Ogni modulo riceve dall'interfaccia grafica i parametri con cui impostare il processo di riduzione ed una mappa M su cui lavorare. Come risultato ritorna la mappa ridotta $M^{'}$. Occorre prestare attenzione che i parametri impostati nell'interfaccia grafica, pur avendo lo stesso nome per tutti i metodi, hanno significati differenti. Infatti, i parametri *distanza spaziale* (δ) e *distanza angolare* (θ) assumono i seguenti significati:
  - nel caso dei due metodi Basic e del metodo Hybrid indicano rispettivamente le due soglie, spaziale ed angolare, descritte nelle Sezioni 4.1 e 4.2 utilizzate per stabilire la corrispondenza o meno tra coppie di segmenti;

  - nel caso del metodo Mean Shift, distanza spaziale indica la minima dimensione di una caratteristica significativa, come descritto nella Sezione 4.3.1;
  - nel caso del metodo Fusion, distanza spaziale indica il valore di tolleranza ϵ utilizzato nella ricerca della retta di fusione, come descritto nella Sezione 4.4.

- modulo *QualityMeasure.m* che, date due mappe M e $M^{'}$, di cui la seconda è ottenuta a partire da M attraverso l'applicazione di uno dei metodi di fusione, calcola la qualità della riduzione come descritto nella sezione 6.2.2. Essendo un modulo indipendente è possibile modificare la misura di qualità utilizzata senza dover modificare il sistema.

- modulo *AlignOffline.m*, infine, che realizza il metodo di allineamento basato sulla struttura portante di una mappa descritto in dettaglio nella sezione 3.3.2.

In Figura 6.1 è descritto lo schema a blocchi dell'architettura del sistema realizzato.

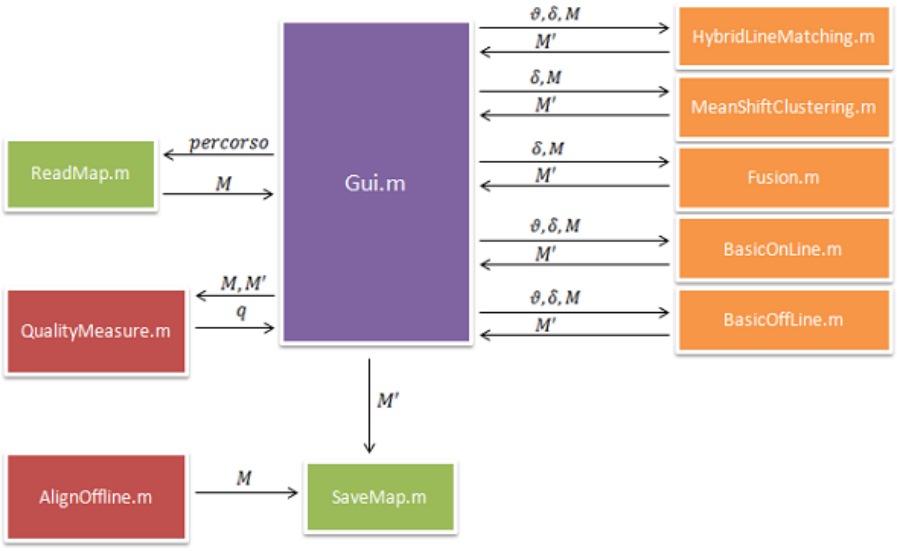

Figura 6.1: Schema a blocchi dei principali moduli che compongono il sistema realizzato.

6.2 Prove sperimentali

Le prove sperimentali di valutazione dei metodi di fusione, descritti nel Capitolo 4, sono state eseguite su dati raccolti nei laboratori del Politecnico di Milano (*data set Polimi*) e su dati disponibili pubblicamente (*data set Stanford*, *data set Bicocca* e *data set USC*). Tali prove sono state condotte su un computer avente le seguenti caratteristiche: processore Intel(R) Core(TM)2 Duo CPU P8400 @ 2.26GHz 2.27GHz, 4,00 GB e sistema operativo Windows 7 a 32 bit.
Un data set è un insieme di scansioni che il robot acquisisce durante la fase di esplorazione dell'ambiente: in ogni scansione sono solitamente riportati la posizione stimata del sensore del robot all'interno dell'ambiente (mantenuta mediante l'odometria) e le posizioni degli ostacoli, memorizzate in coordinate polari ed espresse nel sistema di riferimento del sensore stesso. Come abbiamo già spiegato nel Capitolo 3, sono due i passi che devono essere svolti per ottenere una mappa basata su segmenti a partire da un insieme di scansioni:

1. estrazione dei segmenti dai punti delle scansioni mediante un opportuno metodo di estrazione;

2. allineamento dei segmenti mediante un opportuno metodo di allineamento.

Nel nostro lavoro di ricerca abbiamo preso in considerazione due diversi metodi di estrazione e tre diversi metodi di allineamento. Tutti questi metodi sono stati utilizzati in differenti combinazioni sui data set considerati, in modo da ottenere un insieme di mappe basate su segmenti con caratteristiche differenti.
Per quanto riguarda l'estrazione dei segmenti:

- nei data set Polimi e Stanford i segmenti sono stati estratti dai punti delle scansioni mediante il metodo di estrazione descritto in [4];

- nei data set Bicocca e USC i segmenti sono stati estratti dai punti delle singole scansioni mediante il metodo di estrazione descritto nella Sezione 3.2.

Per quanto riguarda l'allineamento dei segmenti:

- nei data set Polimi e Stanford i segmenti sono stati allineati utilizzando il metodo di allineamento descritto nella Sezione 3.3.1;

- nel data set Bicocca i segmenti sono stati allineati utilizzando il ground truth, che abbiamo introdotto nella Sezione 3.2. L'allineamento basato su ground truth è il migliore che ci sia, in quanto permette di conoscere le posizioni corrette in cui il robot ha effettuato le singole scansioni. Le posizioni non vengono stimate mediante odometria, che è soggetta ad accumulare errori in maniera incrementale durante la navigazione, ma tramite opportuni sensori, come ad esempio telecamere, collocati nell'ambiente;

- nel data set USC i segmenti sono stati allineati utilizzando il metodo di allineamento descritto nella Sezione 3.3.2.

Dai quattro data set iniziali abbiamo quindi ottenuto quattro mappe con caratteristiche eterogenee: per esempio, osservando le Figure 6.2 e 6.3, possiamo notare che le mappe ottenute dai data set Polimi e Stanford sono piuttosto pulite e caratterizzate da uno scarso rumore, mentre le mappe ottenute dai data set Bicocca e USC sono molto più dense e caratterizzate da un maggiore rumore (Figure 6.4 e 6.5), dove con il termine rumore si intende la concentrazione di segmenti in una specifica area della mappa. L'obiettivo del nostro lavoro di ricerca è infatti quello di dimostrare quale sia il metodo di fusione dei segmenti migliore, indipendentemente dalle condizioni (e quindi dal tipo di mappa) in cui esso si trova a dover operare.

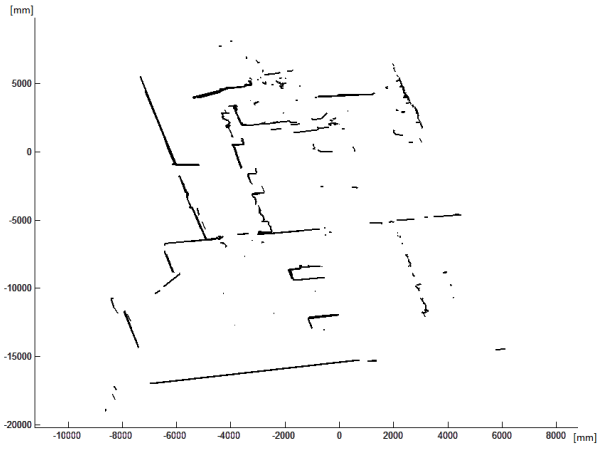

Figura 6.2: Data set Polimi.

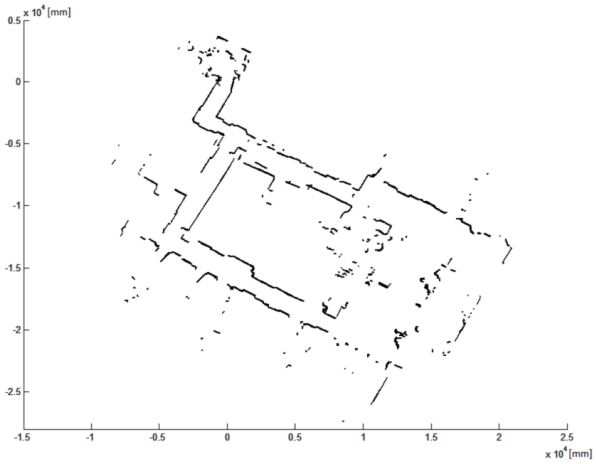

Figura 6.3: Data set Stanford.

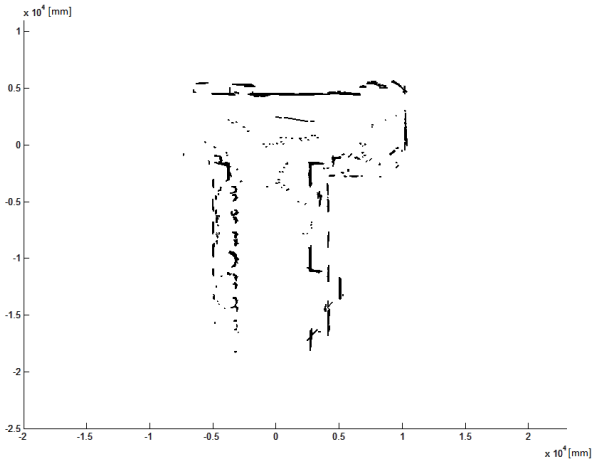

Figura 6.4: Data set Bicocca.

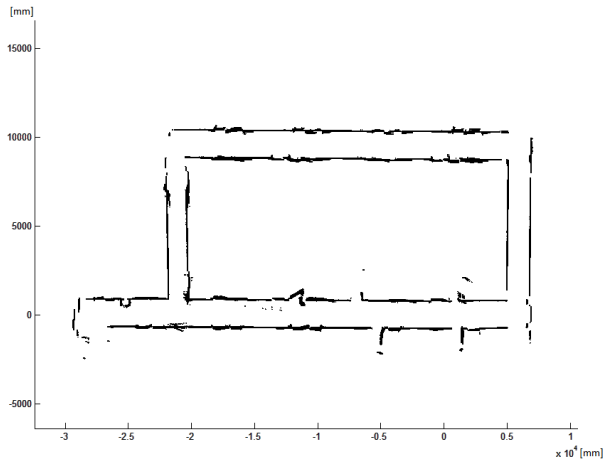

Figura 6.5: Data set USC.

6.2.1 Opzione filtraggio

Il filtraggio è una particolare opzione che abbiamo introdotto nello svolgimento delle prove sperimentali di valutazione. Nella Sezione 4.3.3 abbiamo descritto, per quanto riguarda il metodo Mean Shift Clustering, il concetto di minima dimensione di una caratteristica significativa. Impostando una determinata soglia definiamo quindi il minimo dettaglio rilevabile di un ambiente: il robot non è in grado di riconoscere la presenza di oggetti al di sotto di tale soglia. Ad esempio, se il valore della minima dimensione di una caratteristica significativa viene posto pari a 200 millimetri, qualsiasi oggetto presente nell'ambiente con una dimensione inferiore a 200 millimetri non può essere rilevato dal robot. Prendendo spunto da questo concetto, abbiamo introdotto l'opzione di filtraggio per le mappe basate su segmenti. Attivando l'opzione di filtraggio, tutti i segmenti con una lunghezza inferiore a una determinata soglia vengono cancellati dalla mappa: si assume che il robot non sia in grado di rilevare tali oggetti. Abbiamo introdotto questa opzione per effettuare una "pulizia" delle mappe. Infatti, le mappe contengono, tra gli altri, anche dei segmenti la cui lunghezza è piuttosto corta: tali segmenti potrebbero essere il frutto di errori di rilevazione compiuti dal sensore del robot oppure di errori compiuti durante il processo di estrazione dei segmenti. Per questo motivo risulta interessante effettuare una pulizia iniziale della mappa, rimuovendo tutti i segmenti troppo corti che potrebbero corrispondere a rappresentazioni errate dell'ambiente reale, mantenendone comunque la struttura. Nelle prove sperimentali abbiamo impostato 100 millimetri come soglia per il filtraggio: tale valore è indi-

pendente dai parametri in ingresso di ciascun metodo di fusione, in quanto ciascun parametro ha un proprio significato a seconda del metodo a cui appartiene. Per esempio, nel metodo Fusion la distanza spaziale rappresenta la massima traslazione consentita alla retta di fusione, mentre nel metodo Mean Shift Clustering rappresenta la minima dimensione di una caratteristica significativa. La soglia del filtraggio non può essere quindi scelta come dipendente dai parametri di ciascun metodo, ma va definita in modo univoco al di fuori di essi, in modo da ottenere una comparazione corretta tra i differenti metodi.

6.2.2 Misura di qualità

In questa sezione introduciamo la misura di qualità di una mappa finale, che ci sarà utile nella conduzione delle successive prove sperimentali.
Data una mappa iniziale M, composta da n segmenti, è evidente che non basta confrontare i metodi di riduzione prendendo in considerazione solamente il numero di segmenti $n^{'}$ della mappa ridotta. Se così facessimo, un metodo di riduzione che approssima tutti i segmenti della mappa di partenza con un unico segmento risulterebbe paradossalmente essere il migliore, anche se in realtà non lo è. Ecco perché è importante confrontare i metodi non solo sul numero di riduzioni effettuate, ma anche sulla qualità di queste riduzioni. Essendo la mappa finale una semplificazione della mappa di partenza, ci aspettiamo in generale una diminuzione della qualità. Quanto la qualità diminuisce dipende però fortemente dal metodo di riduzione scelto, poiché è quest'ultimo che determina quali segmenti ridurre e come ridurli.
La misura di qualità utilizza la definizione di distanza definita nella Sezione 2.7.7. L'idea alla base della misura di qualità è che ogni segmento della mappa iniziale deve avere un *rappresentante* nella mappa finale ridotta. Dato un segmento s della mappa iniziale, il rappresentante di s nella mappa finale è definito come il segmento $s^{'}$ che è meno distante da s. La distanza tra il segmento s e il suo rappresentante $s^{'}$ rappresenta una stima dell'errore commesso dal metodo di riduzione nell'approssimare s con $s^{'}$:

$$e_{ss'} = D(s, s^{'}, 1, 1) \tag{6.1}$$

L'errore totale commesso dal metodo di riduzione è definito, di conseguenza, come la somma degli errori introdotti per approssimare ogni segmento s della mappa iniziale con il suo rappresentante $s^{'}$ della mappa finale:

$$E = \sum_{s \in M} D(s, s^{'}, 1, 1) \tag{6.2}$$

dove:

- M è la mappa iniziale su cui è stato applicato il metodo di riduzione;
- $s^{'}$ è il rappresentante di s nella mappa finale.

La qualità q di una mappa è un numero reale compreso nell'intervallo $[0, 1]$ ed è definita a partire dall'errore totale introdotto dal metodo di riduzione nel seguente modo:

$$q = \exp -\frac{(E/n)}{50} \tag{6.3}$$

dove:

- n è il numero di segmenti della mappa iniziale;
- E è l'errore totale commesso dal metodo di riduzione definito in precedenza;
- 50 è un fattore di scala, derivato da prove sperimentali, che serve a normalizzare i valori di qualità.

Nella formula precedente E/n rappresenta quindi l'errore medio introdotto dal metodo di riduzione nell'approssimare ogni segmento della mappa iniziale con un segmento rappresentante della mappa finale. La misura di qualità così definita tende a 1 se l'errore medio tende a 0.

6.2.3 Parametri considerati

Nelle prove sperimentali di valutazione abbiamo preso in considerazione tre diversi parametri al fine di determinare il miglior metodo di fusione dei segmenti:

- *tempo computazionale.* Il tempo computazionale va inteso come il tempo impiegato dal robot per eseguire l'intero metodo di fusione. Nei metodi off-line il tempo computazionale equivale quindi al tempo speso dal robot per una singola esecuzione del metodo, visto che tale metodo viene applicato a mappa globale completa. Nei metodi on-line, invece, si parla di *tempo computazionale totale*, in quanto il tempo impiegato dal robot è dato dalla somma dei tempi spesi dal robot per eseguire il metodo di fusione a ogni nuova scansione presa in considerazione. Nelle prove sperimentali dei metodi on-line abbiamo rilevato, oltre al valore finale del tempo computazionale totale, anche la sua crescita a ogni nuova scansione. Inoltre vogliamo precisare che, nelle successive analisi, la specifica velocità di ciascun metodo è sempre da intendere in relazione agli altri metodi: non abbiamo assegnato dei valori universali per metodo ‘‘lento” o metodo ‘‘veloce”, bensì ogni

volta abbiamo espresso la velocità di un metodo rispetto a quella degli altri metodi eseguiti sullo stesso data set;

- *percentuale di segmenti ridotti.* La percentuale dei segmenti ridotti indica quanti segmenti della mappa iniziale, eventualmente filtrata, sono stati fusi e non sono più presenti nella mappa finale ridotta, secondo la seguente formula:

$$perc_{sr} = \frac{n - n^{'}}{n} \tag{6.4}$$

dove $perc_{sr}$ rappresenta la percentuale dei segmenti ridotti, n rappresenta il numero dei segmenti della mappa iniziale, eventualmente filtrata, e $n^{'}$ rappresenta il numero dei segmenti della mappa finale ridotta. In caso di filtraggio, prendiamo in considerazione i segmenti della mappa iniziale filtrata, e non quelli della mappa iniziale originale, in quanto vogliamo mantenere il filtraggio come una opzione esterna al metodo di fusione. Se utilizzassimo i segmenti della mappa iniziale originale nella precedente formula, il filtraggio diventerebbe un processo interno al metodo, e non è ciò che desideriamo. Il nostro obiettivo è invece quello di misurare la diversa efficacia dei metodi di fusione tra mappe che vengono fornite a priori pulite (con filtraggio) oppure no (senza filtraggio). Inoltre, per le successive analisi, abbiamo assegnato, in maniera indicativa, a ciascuna percentuale di segmenti ridotti, un determinato grado di bontà della riduzione:

  - intorno al 70% o superiore: riduzione ottima;
  - intorno al 60%: riduzione molto buona;
  - intorno al 40%-50%: riduzione buona;
  - intorno al 30%: riduzione discreta;
  - intorno o inferiore al 20%: riduzione scarsa.

- *qualità mappa finale.* La qualità della mappa finale viene calcolata confrontando la mappa finale ridotta con la mappa iniziale, eventualmente filtrata, secondo il processo descritto nella sezione precedente. Per le stesse ragioni spiegate in precedenza, anche per quanto concerne la qualità, in caso di filtraggio, prendiamo in considerazione la mappa iniziale filtrata e non quella originale. Inoltre, per le successive analisi, abbiamo assegnato, in maniera indicativa, a ciascuna percentuale di qualità, un determinato grado di bontà della qualità della mappa:

  - intorno al 90%: qualità ottima;

- intorno all'80%: qualità molto buona;
- intorno all 70%: qualità buona;
- intorno al 60%: qualità discreta;
- intorno al 50%: qualità scarsa;
- intorno o inferiore al 40%: qualità pessima.

Vogliamo ricordare che i parametri di distanza spaziale e angolare hanno un significato che cambia da metodo a metodo: per questo motivo non è possibile effettuare un confronto sperimentale tra i metodi osservando, a parità di parametri in ingresso, quale metodo ha ridotto più segmenti e generato una mappa finale con una qualità migliore. A parità di parametro, infatti, il metodo Fusion produce risultati molto diversi, ad esempio, da quelli del metodo Basic. Un discorso simile vale per le prove sperimentali eseguite con o senza filtraggio: non è possibile confrontare una prova eseguita con filtraggio di un metodo con una prova eseguita senza filtraggio di un altro metodo, nel tentativo di stabilire quale dei due sia il migliore. Infatti, le prove in questo caso sono incomparabili, in quanto condotte su mappe iniziali differenti, con un numero di segmenti diverso.
Alla luce delle considerazioni appena effettuate, l'analisi sperimentale verrà condotta nel seguente modo:

- sceglieremo un metodo su cui concentrare l'analisi. Di questo metodo prenderemo in considerazione le prove più significative (con una qualità superiore all'80%) per determinati parametri in ingresso;
- cercheremo negli altri metodi mappe con una qualità simile, indipendentemente dal valore del parametro in ingresso, in quanto i parametri, come già detto, hanno comunque un significato diverso a seconda del metodo e un'analisi basata sul loro confronto non è conducibile. Di tali mappe guarderemo il valore delle percentuale di segmenti ridotti, in modo da capire, a parità di qualità, quale metodo ha ridotto più segmenti.

La nostra analisi quindi non compara i parametri in ingresso, ma la qualità delle mappe finali: vogliamo individuare il metodo che, a qualità fissate, è in grado di ridurre il maggior numero di segmenti.

6.3 Data set Polimi

Il data set Polimi è costituito da una sequenza di 28 scansioni effettuate utilizzando un robot dotato di un telemetro laser *SICK LMS-200* all'interno

dell'edificio 20 del Dipartimento di Elettronica e Informazione del Politecnico di Milano. Le scansioni sono state acquisite in diversi ambienti formando un circuito chiuso di 40 metri circa. Partendo da un laboratorio, un ambiente composto da numerosi ostacoli sparsi, si è passati attraverso un corridoio stretto, formato da muri rettilinei, fino ad arrivare all'ingresso del dipartimento (uno spazio aperto con lunghi muri perpendicolari) per poi ritornare al laboratorio di partenza. Su questo data set abbiamo eseguito, per ogni metodo di fusione considerato nelle sezioni precedenti, una serie di prove sperimentali di valutazione, i cui risultati completi sono riportati in Appendice A. Abbiamo effettuato due diverse tipologie di prove sperimentali: quelle senza filtraggio e quelle con filtraggio. Nelle prove senza filtraggio la mappa iniziale è costituita da 658 segmenti, mentre nelle prove con filtraggio la mappa iniziale, data in ingresso allo specifico metodo di fusione, è costituita da 434 segmenti. Analizziamo quindi, per ogni metodo, i risultati di queste prove sperimentali, secondo i tre diversi parametri di valutazione definiti nella Sezione 6.2.3:

- il metodo Basic off-line fornisce risultati molto buoni sia per quanto riguarda la percentuale dei segmenti ridotti, che varia tra il 41% e il 66%, sia per quanto riguarda la qualità della mappa finale ridotta, che varia tra il 75% e il 92%. La qualità risulta buona (intorno al 75%) anche per prove sperimentali, sia con filtraggio che senza, eseguite con parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi). Il metodo è quindi molto robusto in quanto, all'aumentare dei parametri in ingresso, genera mappe finali caratterizzate da una buona qualità e da una buona percentuale di segmenti ridotti, come mostrato in Figura 6.6 e 6.7. La percentuale

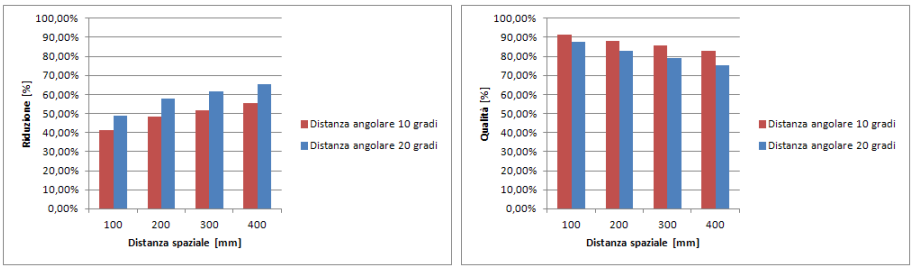

Figura 6.6: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic off-line nelle prove sperimentali eseguite senza filtraggio sul data set Polimi.

dei segmenti ridotti risulta maggiore nelle prove sperimentali eseguite

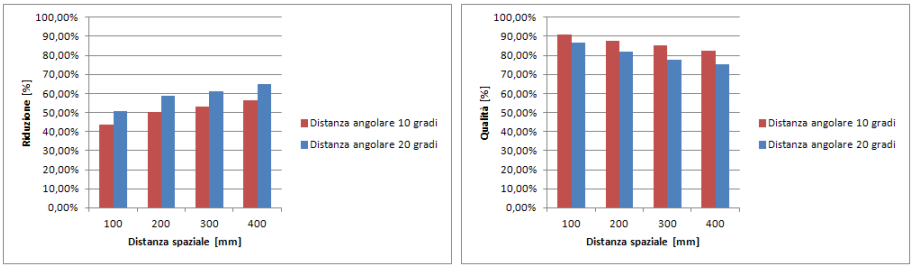

Figura 6.7: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic off-line nelle prove sperimentali eseguite con filtraggio sul data set Polimi.

con filtraggio rispetto a quelle eseguite senza filtraggio: a una maggiore riduzione corrisponde però una peggiore qualità della mappa finale ridotta. Questo risulta evidente confrontando, a parità di parametri in ingresso, le prove eseguite con filtraggio con le corrispondenti prove eseguite senza filtraggio. Il metodo è inoltre abbastanza veloce: il tempo computazionale varia tra i 5 e i 29 secondi;

- anche la versione on-line del metodo Basic produce risultati molto buoni sia per quanto riguarda la percentuale dei segmenti ridotti, che varia tra il 41% e il 65%, sia per quanto riguarda la qualità della mappa finale ridotta, che varia tra il 74% e il 92%. Come per la corrispondente versione off-line, pure il metodo Basic on-line, all'aumentare dei parametri in ingresso, genera mappe finali caratterizzate da una buona qualità e da una buona percentuale di segmenti ridotti (Figura 6.8 e 6.9). Anche in questo caso, la percentuale dei segmenti ridotti risul-

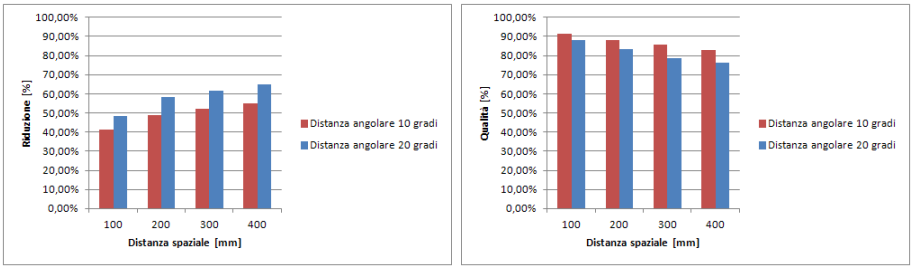

Figura 6.8: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic on-line nelle prove sperimentali eseguite senza filtraggio sul data set Polimi.

ta maggiore nelle prove sperimentali eseguite con filtraggio rispetto a quelle eseguite senza filtraggio. Il metodo è inoltre abbastanza lento:

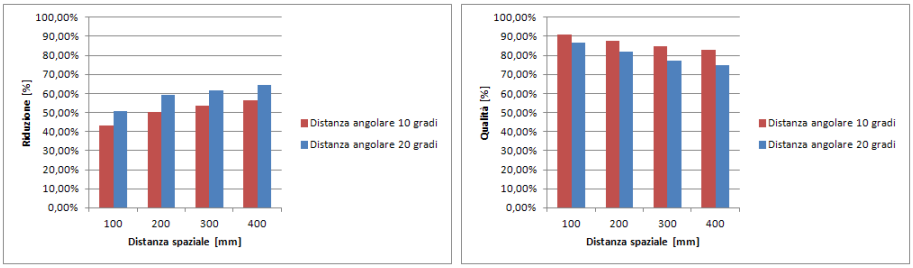

Figura 6.9: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic on-line nelle prove sperimentali eseguite con filtraggio sul data set Polimi.

il tempo computazionale totale varia tra i 31 secondi e i 3 minuti circa. In Figura 6.10 è mostrata, per alcune prove sperimentali, la crescita

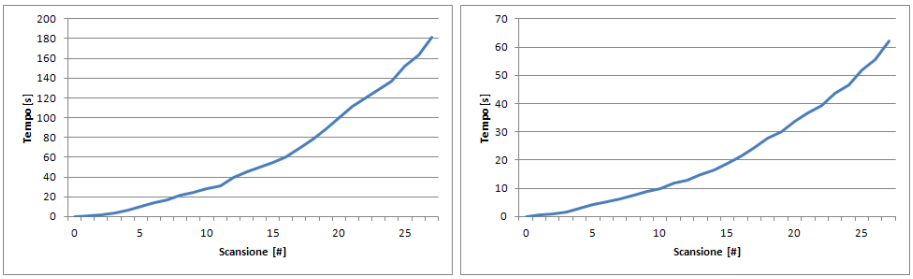

Figura 6.10: Crescita del tempo computazionale totale a ogni nuova scansione rilevata per le prove sperimentali, relative al metodo Basic on-line, eseguite senza filtraggio sul data set Polimi e con parametri di distanza spaziale e angolare pari a 100 millimetri e 10 gradi (figura a sinistra), e 400 millimetri e 20 gradi (figura a destra).

del tempo computazionale totale a ogni nuova scansione rilevata. Si può notare che tale tempo cresce in maniera esponenziale all'aumentare del numero di scansioni: questo accade perché, nella sua versione on-line, il metodo Basic viene chiamato a ogni nuova scansione. A ogni iterazione il metodo confronta tra loro tutte le coppie di segmenti presenti nella mappa corrente e, quindi, più è alto il numero dei segmenti presenti nella mappa, più tempo impiega il robot per eseguire tutti i confronti. Nella versione off-line, invece, il metodo Basic viene chiamato una sola volta, a mappa globale completa, comportando dei tempi computazionali ragionevoli. Osservando la Figura 6.10 si può inoltre notare che il tempo computazionale totale, per il metodo Basic on-line, è maggiore nelle prove eseguite con parametri in ingresso molto bassi piuttosto che in quelle eseguite con parametri elevati: nel

primo caso, infatti, i vincoli sono molto stringenti, vengono effettuate poche riduzioni e, a ogni iterazione, vanno confrontati tanti segmenti; nel secondo caso, invece, a causa della maggiore libertà concessa dai parametri, le riduzioni sono maggiori e, a ogni iterazione, vanno confrontati meno segmenti, in quanto molti sono già stati fusi tra loro e rimossi dalla mappa. Questa considerazione in realtà vale anche per il metodo Basic off-line e, più in generale, per tutti i metodi di fusione considerati: più i parametri sono stringenti, più aumentano i tempi computazionali; meno i parametri sono stringenti, più si riducono i tempi computazionali;

- il metodo Hybrid fornisce, nelle prove eseguite senza filtraggio, risultati molto buoni sia per quanto riguarda la percentuale dei segmenti ridotti, che varia tra il 50% e il 66%, sia per quanto riguarda la qualità della mappa finale ridotta, che varia tra il 74% e l'84%. In queste prove, la qualità risulta buona (intorno al 74%) anche per parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi). Aumentando in maniera considerevole i parametri nelle prove senza filtraggio, il metodo produce comunque mappe finali caratterizzate da una buona qualità e da una buona percentuale di segmenti ridotti, come mostrato in Figura 6.11. Nelle prove eseguite con filtraggio, invece, i risultati prodotti dal metodo sono discreti per quanto riguarda la percentuale dei segmenti ridotti, che varia tra il 28% e il 52%, e ottimi per quanto riguarda la qualità della mappa finale ridotta, che varia tra l'83% e il 96%. In queste prove, la qualità risulta più che buona (intorno all'83%) anche per parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi). All'aumentare dei parametri in ingresso, nelle prove eseguite con filtraggio, il metodo genera comunque mappe finali

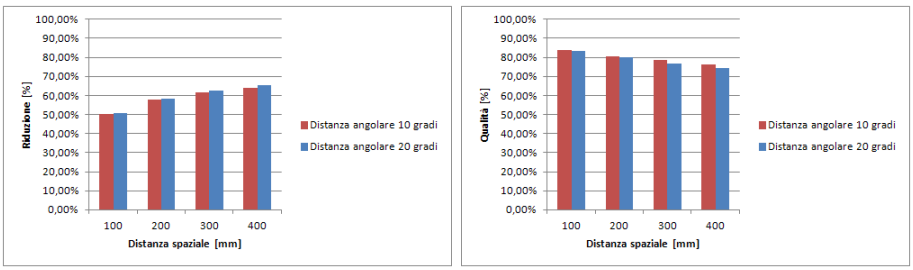

Figura 6.11: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Hybrid nelle prove sperimentali eseguite senza filtraggio sul data set Polimi.

caratterizzate da una buona qualità e da una discreta percentuale di segmenti ridotti, come mostrato in Figura 6.12. Per quanto riguarda

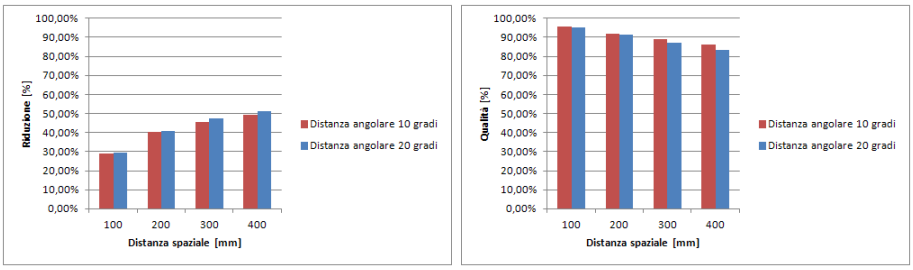

Figura 6.12: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Hybrid nelle prove sperimentali eseguite con filtraggio sul data set Polimi.

il metodo Hybrid, la qualità della mappa finale risulta migliore nelle prove sperimentali eseguite con filtraggio rispetto a quelle eseguite senza filtraggio: a una migliore qualità corrisponde tuttavia una minore percentuale dei segmenti ridotti. In questo metodo si verificano inoltre delle significative differenze tra le prove eseguite con filtraggio e quelle eseguite senza filtraggio: senza filtraggio, il metodo riceve in ingresso una mappa potenzialmente sporca e la riduce ottenendo una mappa finale caratterizzata, nella maggior parte dei casi, da una buona qualità e da una buona percentuale dei segmenti ridotti; in caso di filtraggio, il metodo riceve in ingresso una mappa già pulita e la riduce ottenendo una mappa caratterizzata, nella maggior parte dei casi, da una ottima qualità e da una discreta percentuale dei segmenti ridotti. Questo comportamento viene spiegato col fatto che il metodo Hybrid possiede un proprio meccanismo di filtraggio: per ogni segmento che non corrisponde a nessun altro segmento presente nella mappa, se tale segmento supera una certa soglia, esso viene aggiunto alla mappa finale. Quindi, nel caso in cui il metodo Hybrid riceve in ingresso una mappa già pulita, il filtraggio interno risulta inutile, in quanto la mappa contiene solo segmenti superiori alla soglia del filtraggio: tutti i segmenti che non trovano corrispondenze vengono aggiunti alla mappa finale, le riduzioni effettuate sono poche e la mappa finale ha una qualità ottima rispetto alla mappa ricevuta in ingresso, ma ha anche una bassa percentuale di segmenti ridotti. D'altro canto, in assenza di filtraggio esterno, il metodo Hybrid riceve in ingresso una mappa sporca e opera il proprio filtraggio, rimuovendo dalla mappa tutti i segmenti inferiori a una determinata soglia: in questo caso le riduzioni sono maggiori,

la mappa finale ha una buona qualità e una buona percentuale di segmenti ridotti. Il tempo computazionale totale di questo metodo varia

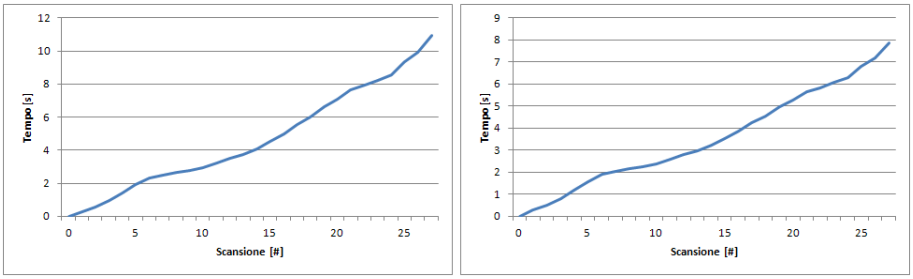

Figura 6.13: Crescita del tempo computazionale totale a ogni nuova scansione rilevata per le prove sperimentali, relative al metodo Hybrid, eseguite senza filtraggio sul data set Polimi e con parametri di distanza spaziale e angolare pari a 100 millimetri e 10 gradi (figura a sinistra), e 400 millimetri e 20 gradi (figura a destra).

tra i 4 e i 10 secondi: si tratta di un metodo particolarmente veloce. In Figura 6.13 viene mostrata, per un paio di prove sperimentali, la crescita del tempo computazionale totale a ogni nuova scansione rilevata. Rispetto all'andamento esponenziale del metodo Basic on-line, si può notare una crescita quasi lineare del tempo all'aumentare del numero delle scansioni. Questo accade perchè, a differenza del metodo Basic on-line, il quale confronta tra loro tutte le coppie di segmenti appartenenti alla mappa, il metodo Hybrid confronta solo alcune coppie di segmenti, di cui il primo appartiene alla nuova scansione e il secondo alla mappa corrente. In questo modo il numero dei confronti effettuati è molto inferiore e, di conseguenza, il tempo computazionale totale impiegato per l'esecuzione del metodo Hybrid è più basso;

- il metodo Mean Shift Clustering fornisce risultati molto buoni solo per valori non troppo elevati del parametro di distanza spaziale. La qualità della mappa finale risulta molto buona (tra l'80% e il 91%) per prove sperimentali, sia con filtraggio che senza, eseguite con un parametro di distanza spaziale non troppo elevato (100 o 200 millimetri): in queste prove si ottiene comunque una buona riduzione (tra il 36% e il 54%). Aumentando tale parametro (fino a 300 o 400 millimetri), il metodo genera mappe caratterizzate da una percentuale dei segmenti ridotti molto alta (tra il 61% e il 79%) e da una qualità solo discreta (tra il 50% e il 68%). Il metodo non è robusto: l'aumento del parametro di distanza spaziale comporta grandi variazioni a livello di riduzione dei segmenti e di qualità della mappa finale, la quale

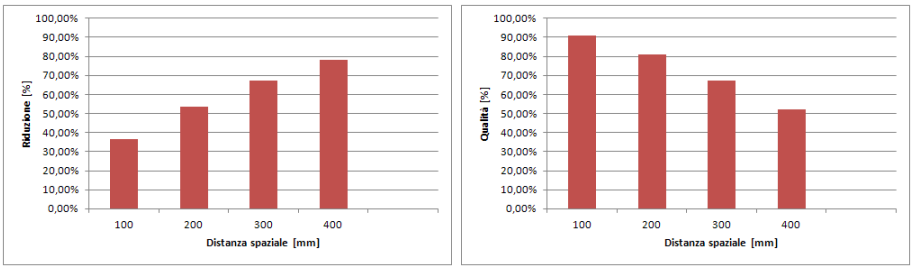

Figura 6.14: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Mean Shift Clustering nelle prove sperimentali eseguite senza filtraggio sul data set Polimi.

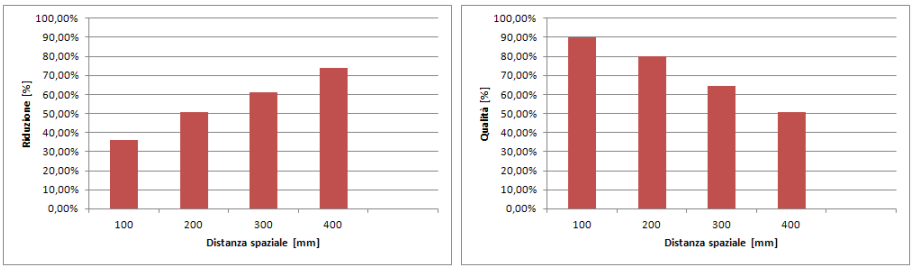

Figura 6.15: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Mean Shift Clustering nelle prove sperimentali eseguite con filtraggio sul data set Polimi.

decresce in maniera notevole all'aumentare di tale parametro (Figura 6.14 e 6.15). Una delle condizioni fondamentali per il corretto funzionamento del metodo Mean Shift Clustering è quella che i segmenti della mappa iniziale, che superano il valore della minima dimensione di una caratteristica rilevabile (e quindi del parametro di distanza spaziale), devono essere inizialmente suddivisi in segmenti più piccoli, in modo da favorire il processo di clustering. Per questo motivo, utilizzando come parametro di distanza spaziale un valore molto elevato, solo pochi segmenti vengono suddivisi: il clustering viene eseguito in una maniera meno precisa e comporta degli errori che si ripercuotono nella successiva fase di fusione, generando una mappa con bassa qualità. Per quanto riguarda il metodo Mean Shift Clustering, sia la percentuale dei segmenti ridotti che la qualità della mappa finale risultano migliori nelle prove sperimentali eseguite senza filtraggio rispetto a quelle eseguite con filtraggio. Il metodo è inoltre abbastanza veloce: il tempo computazionale impiegato per la sua esecuzione varia tra i 9 e i 38 secondi;

- il metodo Fusion produce risultati molto buoni solo per piccoli valori del parametro di distanza spaziale. Nelle prove sperimentali, eseguite con o senza filtraggio e con parametro di distanza spaziale piuttosto basso (20 o 40 millimetri), la qualità della mappa finale risulta molto buona (tra l'80% e il 94%). Tali prove portano comunque a una buona riduzione, compresa tra il 36% e il 62%. Tuttavia, aumentando il parametro di distanza spaziale (fino a 80 o 100 millimetri), il metodo genera mappe caratterizzate da una percentuale dei segmenti ridotti molto alta (tra il 68% e il 75%) e da una qualità solo discreta (tra il 65% e il 70%). Il metodo quindi non è robusto: l'aumento, anche lieve, del parametro di distanza spaziale comporta grandi variazioni a livello di riduzione dei segmenti e di qualità della mappa finale, la quale decresce in maniera notevole all'aumentare di tale parametro (Figura 6.16 e 6.17). Questo comportamento è spiegato col fatto che il metodo Fusion approssima i segmenti, che appartengono alla catena di corrispondenze, con un nuovo segmento la cui direzione è univocamente determinata dalla direzione della retta di fusione su cui giace il segmento più lungo appartenente alla catena. In questo modo, segmenti che appartengono alla catena di corrispondenze, caratterizzati da direzioni anche molto differenti da quelli della retta di fusione, vengono approssimati con un unico nuovo segmento la cui direzione è molto diversa dalla loro, provocando un peggioramento della qualità della mappa fi-

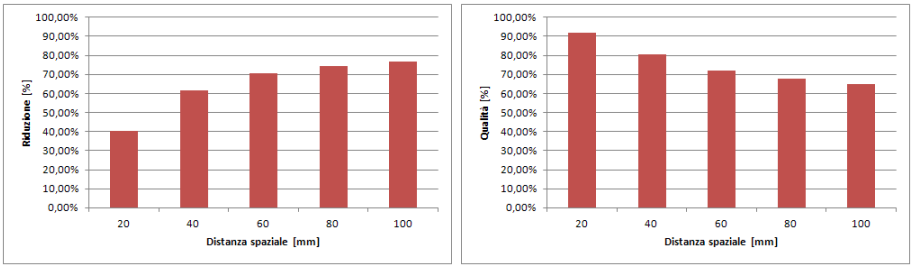

Figura 6.16: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Fusion nelle prove sperimentali eseguite senza filtraggio sul data set Polimi.

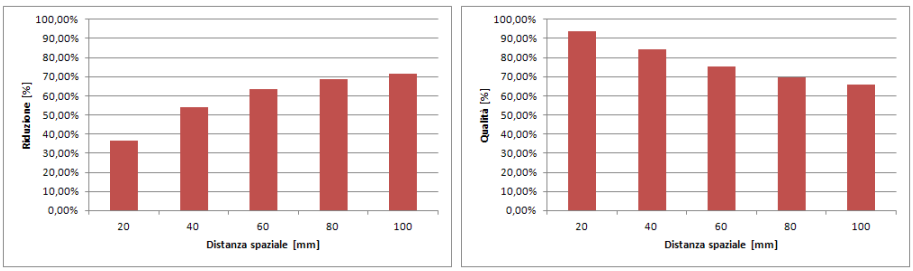

Figura 6.17: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Fusion nelle prove sperimentali eseguite con filtraggio sul data set Polimi.

nale. La probabilità di raggruppare segmenti con direzioni differenti nella stessa catena di corrispondenze cresce con l'aumentare del valore del parametro di distanza spaziale: per questo motivo il metodo Fusion offre prestazioni ottimali solo per valori molto bassi di tale parametro. Come nel caso del metodo Hybrid, anche per il metodo Fusion la qualità della mappa finale risulta migliore nelle prove sperimentali eseguite con filtraggio rispetto a quelle eseguite senza filtraggio. Il metodo è inoltre molto veloce, infatti il tempo computazionale varia tra i 3 e i 10 secondi.

Esponiamo ora ulteriori considerazioni di carattere generale sul tempo computazionale:

- a parità di parametri in ingresso, il tempo computazionale totale impiegato per l'esecuzione della versione on-line del metodo Basic è superiore al tempo computazionale impiegato per l'esecuzione della corrispondente versione off-line: questo accade perchè il metodo Basic on-line, come già detto, viene chiamato a ogni nuova scansione, mentre la corrispondente versione off-line viene chiamata una sola volta a mappa globale completa;

- il tempo computazionale impiegato per l'esecuzione di ogni metodo di fusione è maggiore nelle prove sperimentali eseguite senza filtraggio rispetto a quelle eseguite con filtraggio. Questo è ovvio, in quanto le mappe filtrate hanno un numero di segmenti inferiore rispetto a quelle non filtrate.

Il metodo Mean Shift Clustering è quello che fornisce le prestazioni peggiori sul data set Polimi: questo risulta evidente da un confronto delle prove sperimentali.
Per quanto riguarda le prove sperimentali eseguite senza filtraggio:

- utilizzando un valore di parametro di distanza spaziale pari a 100 millimetri, si ottiene, eseguendo il metodo Mean Shift Clustering, una mappa finale caratterizzata da una qualità pari a circa il 91% e da una percentuale di segmenti ridotti pari a circa il 36%. Come mostrato in Figura 6.18, le due versioni del metodo Basic e il metodo Fusion producono mappe, con una qualità superiore al 91%, caratterizzate da una maggiore percentuale di segmenti ridotti: questi metodi forniscono dunque prestazioni migliori rispetto a quelle del metodo Mean Shift Clustering;

- con un valore di parametro di distanza spaziale pari a 200 millimetri, si ottiene, applicando il metodo Mean Shift Clustering, una mappa finale caratterizzata da una qualità pari a circa l'81% e da una percentuale dei segmenti ridotti pari a circa il 54%. Come mostrato in Figura 6.19, gli altri metodi di fusione generano mappe, con una qualità superiore o leggermente inferiore all'81%, caratterizzate da una maggiore percentuale di segmenti ridotti.

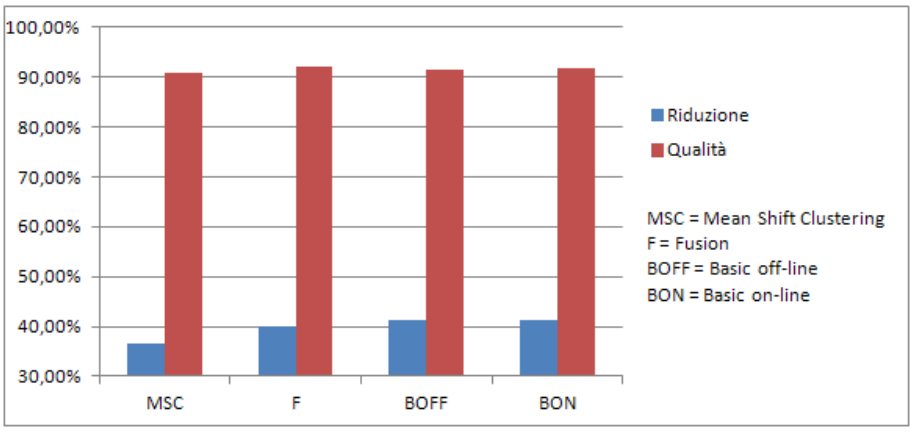

Figura 6.18: Nelle prove sperimentali eseguite senza filtraggio sul data set Polimi, in mappe finali caratterizzate da una qualità pari a circa il 91%, il metodo Mean Shift Clustering riduce meno segmenti rispetto al metodo Fusion e alle due versioni del metodo Basic.

Per quanto riguarda le prove sperimentali eseguite con filtraggio:

- con un valore di parametro di distanza spaziale pari a 100 millimetri, si ottiene, eseguendo il metodo Mean Shift Clustering, una mappa finale caratterizzata da una qualità pari a circa il 90% e da una percentuale di segmenti ridotti pari a circa il 36%. Come mostrato in Figura 6.20, le due versioni del metodo Basic e il metodo Fusion generano mappe, con una qualità superiore al 90%, caratterizzate da una maggiore percentuale di segmenti ridotti;

- utilizzando un valore di parametro di distanza spaziale pari a 200 millimetri, si ottiene, applicando il metodo Mean Shift Clustering, una mappa finale caratterizzata da una qualità pari a circa l'80% e da una percentuale dei segmenti ridotti pari a circa il 51%. Come mostrato in Figura 6.21, gli altri metodi di fusione producono mappe, con una qualità superiore all'80%, caratterizzate da una maggiore riduzione:

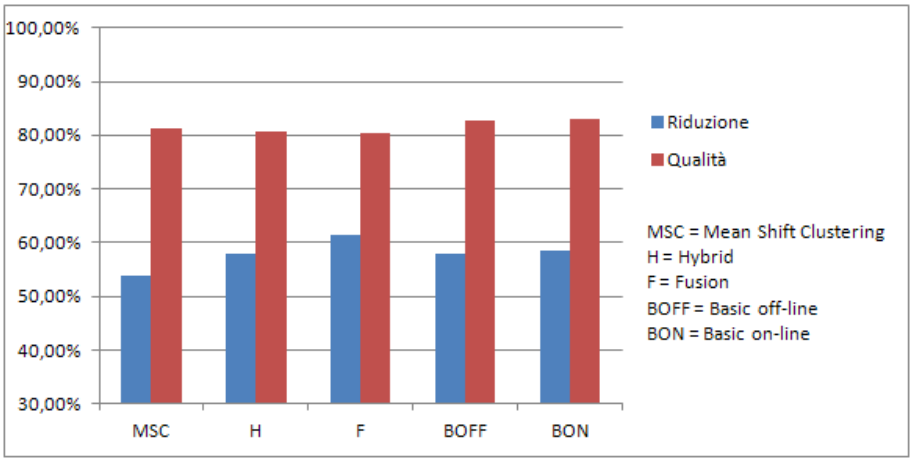

Figura 6.19: Nelle prove sperimentali eseguite senza filtraggio sul data set Polimi, in mappe finali caratterizzate da una qualità pari a circa l'81%, il metodo Mean Shift Clustering riduce meno segmenti rispetto a tutti gli altri metodi.

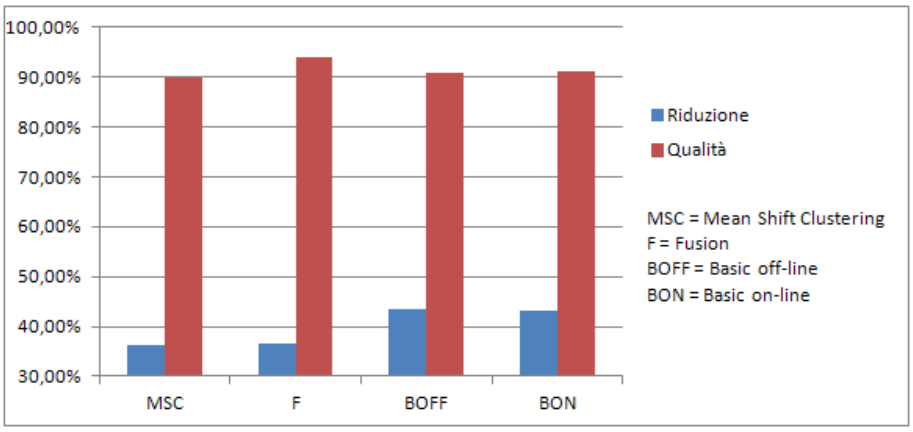

Figura 6.20: Nelle prove sperimentali eseguite con filtraggio sul data set Polimi, in mappe finali caratterizzate da una qualità pari a circa il 90%, il metodo Mean Shift Clustering riduce meno segmenti rispetto al metodo Fusion e alle due versioni del metodo Basic. Nell'esempio riportato in figura, il metodo Mean Shift Clustering ha una percentuale di riduzione molto vicina a quella del metodo Fusion, tuttavia la sua qualità è parecchio inferiore e, di conseguenza, anche in questo caso il metodo Mean Shift Clustering offre prestazioni inferiori rispetto al metodo Fusion.

l'unica eccezione è rappresentata dal metodo Hybrid, il quale genera una mappa con una percentuale di segmenti ridotti leggermente inferiore a quella del metodo Mean Shift Clustering, offrendo tuttavia una qualità più elevata.

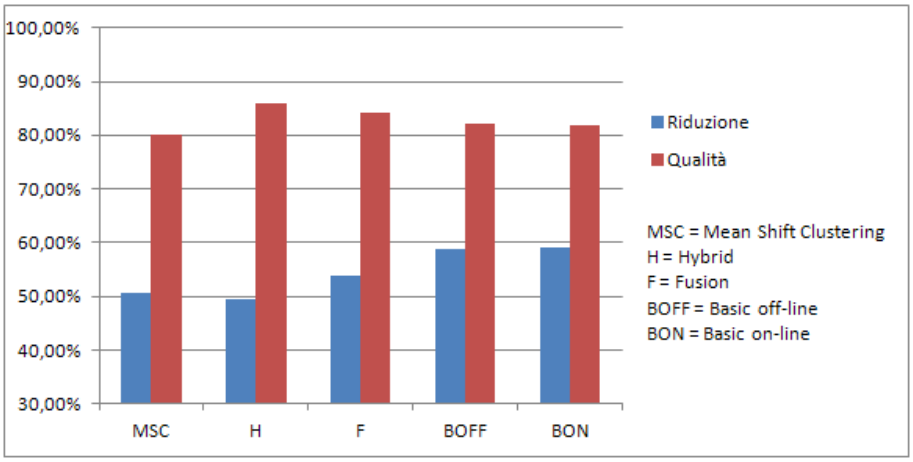

Figura 6.21: Nelle prove sperimentali eseguite con filtraggio sul data set Polimi, in mappe finali caratterizzate da una qualità pari a circa l'80%, il metodo Mean Shift Clustering riduce meno segmenti rispetto a tutti gli altri metodi, eccetto il metodo Hybrid. Nell'esempio riportato in figura, infatti, il metodo Mean Shift Clustering ha una percentuale di riduzione leggermente superiore rispetto a quella del metodo Hybrid, tuttavia la sua qualità è abbastanza inferiore e, di conseguenza, anche in questo caso il metodo Mean Shift Clustering offre prestazioni inferiori rispetto al metodo Hybrid.

Da tutte queste osservazioni possiamo concludere che sul data set Polimi il metodo Mean Shift Clustering fornisce, in generale, prestazioni inferiori rispetto a tutti gli altri metodi. Dal confronto tra le prove sperimentali eseguite sui restanti metodi non emerge alcun risultato significativo, in quanto tutti garantiscono prestazioni simili, impedendoci quindi di stabilire quale sia il metodo di fusione migliore su questo specifico data set. Tuttavia, il metodo Hybrid e il metodo Fusion si fanno preferire rispetto alle due versioni del metodo Basic a causa del minore tempo computazionale richiesto per la loro esecuzione.
Inoltre, osserviamo come le due versioni del metodo Basic producono, a parità di parametri in ingresso, risultati molto simili tra loro sia per quanto riguarda la percentuale dei segmenti ridotti, sia per quanto riguarda la qualità della mappa finale ridotta. Il funzionamento del metodo Basic on-line è infatti il medesimo della corrispondente versione off-line: a ogni nuova scansione viene chiamato il metodo di fusione, il quale verifica le condizioni di

fusione e applica, eventualmente, l'algoritmo di fusione. L'unica differenza tra le due versioni consiste nell'ordine in cui i segmenti vengono considerati e, eventualmente, fusi: nella versione off-line ciascun segmento viene confrontato con tutti gli altri segmenti appartenenti alla mappa globale finale; nella versione on-line, invece, ciascun segmento viene confrontato con tutti gli altri segmenti appartenenti alla mappa parziale corrente. In questo modo, ad esempio, nella versione off-line due segmenti possono venire fusi in unico nuovo segmento, che sarà la causa di ulteriori fusioni, mentre nella versione on-line, a causa del differente ordine con cui i segmenti vengono confrontati, quel nuovo segmento potrebbe non generarsi, ma potrebbe generarsene un altro che sarebbe la causa, in futuro, di differenti fusioni. Tutte queste considerazioni giustificano le piccole differenze che si verificano nelle prestazioni fornite dalle due versioni dei metodi Basic: queste differenze non si ripetono con una logica precisa, in quanto sono appunto determinate dall'ordine con cui i segmenti vengono confrontati, che è casuale e varia da mappa a mappa in base al modo in cui i segmenti vengono memorizzati nelle rispettive strutture dati.

6.4 Data set Stanford

Il data set Stanford è costituito da una sequenza di scansioni disponibili in rete nel *Robotics Data Set Repository (Radish)* [35], sotto la voce *data set stanford gates-1*. L'insieme di dati è ottenuto attraverso un tour di 30 minuti nel primo piano dell'edificio Gates Computer Science di Stanford. Il robot impiegato per la raccolta delle informazioni è un Pioneer 2DX con un telemetro laser *SICK LMS-200*. Su questo data set abbiamo eseguito, per ogni metodo di fusione considerato nelle sezioni precedenti, una serie di prove sperimentali di valutazione, i cui risultati completi sono riportati in Appendice A. Di questo data set sono state prese in considerazione solo alcune scansioni, precisamente una ogni 4 secondi. Abbiamo effettuato due diverse tipologie di prove sperimentali: quelle senza filtraggio e quelle con filtraggio. Nelle prove senza filtraggio la mappa iniziale è costituita da 1507 segmenti, mentre nelle prove con filtraggio la mappa iniziale è costituita da 1117 segmenti. Analizziamo quindi, anche per il data set Stanford, i risultati delle relative prove sperimentali:

- il metodo Basic off-line fornisce risultati molto buoni sia per quanto riguarda la percentuale dei segmenti ridotti, che varia tra il 42% e il 68%, sia per quanto riguarda la qualità della mappa finale, che varia tra il 73% e il 93%. Nelle prove sperimentali, eseguite con o senza

filtraggio e con parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi), la qualità della mappa finale risulta buona (tra il 73% e il 76%). Quindi, come avveniva per il data set Polimi, anche per il data set Stanford il metodo Basic off-line si mostra robusto: all'aumentare dei parametri il metodo produce comunque mappe caratterizzate da una buona qualità e da una buona percentuale di segmenti ridotti, come mostrato in Figura 6.22 e 6.23. Le mappe finali, inoltre, sono caratterizzate da una percentuale di segmenti ridotti maggiore nelle prove sperimentali eseguite con filtraggio rispetto a quelle eseguite senza filtraggio. Il metodo è abbastanza

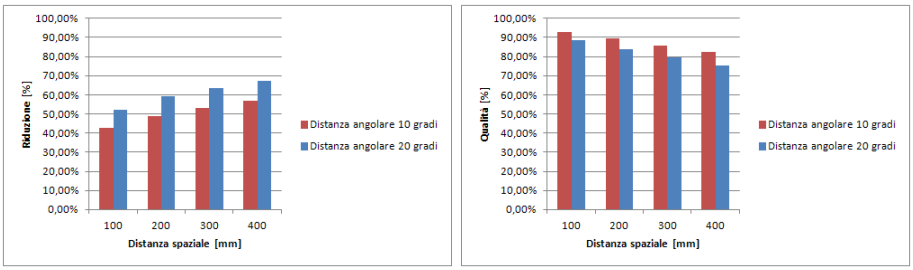

Figura 6.22: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic off-line nelle prove sperimentali eseguite senza filtraggio sul data set Stanford.

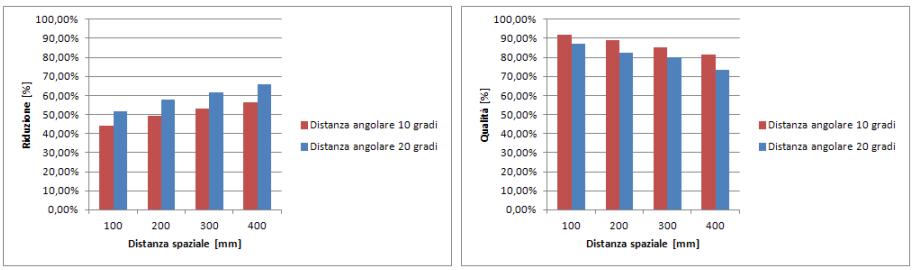

Figura 6.23: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic off-line nelle prove sperimentali eseguite con filtraggio sul data set Stanford.

veloce: il tempo computazionale varia tra i 37 secondi e i 3 minuti circa;

- anche i risultati prodotti dal metodo Basic on-line sono molto buoni sia per quanto riguarda la percentuale dei segmenti ridotti, che varia tra il 43% e il 68%, sia per quanto riguarda la qualità della mappa

finale ridotta, che varia tra il 73% e il 93%. Da tali valori si può

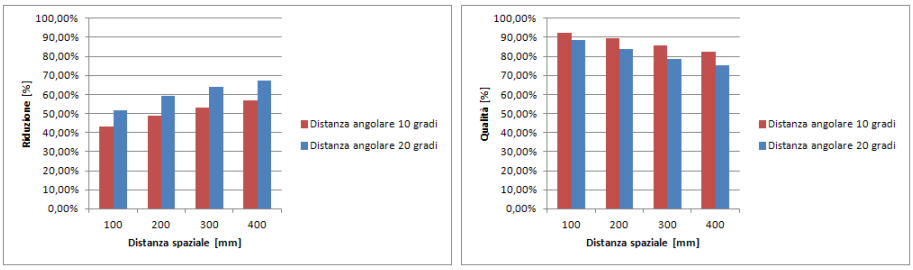

Figura 6.24: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic on-line nelle prove sperimentali eseguite senza filtraggio sul data set Stanford.

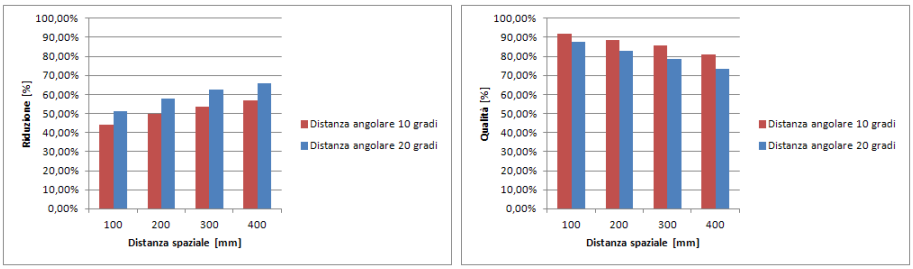

Figura 6.25: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic on-line nelle prove sperimentali eseguite con filtraggio sul data set Stanford.

concludere che pure la versione on-line del metodo Basic si mostra robusta su questo data set (Figura 6.24 e 6.25): infatti, per le stesse ragioni descritte nella sezione precedente, le prestazioni fornite dai due metodi Basic sono molto simili. Anche in questo caso, la percentuale dei segmenti ridotti risulta maggiore nelle prove eseguite con filtraggio rispetto a quelle eseguite senza filtraggio. Il metodo è inoltre piuttosto lento: il tempo computazionale totale impiegato per la sua esecuzione varia tra i 4 e i 22 minuti circa. In Figura 6.26 è mostrata la crescita del tempo computazionale totale, a ogni nuova scansione rilevata, in un paio di prove sperimentali. Osservando la figura, si può concludere che la crescita esponenziale di tale tempo conferma le considerazioni effettuate nella sezione precedente;

- il metodo Hybrid evidenzia, nelle prove eseguite senza filtraggio, risultati buoni per quanto riguarda la percentuale dei segmenti ridotti, che

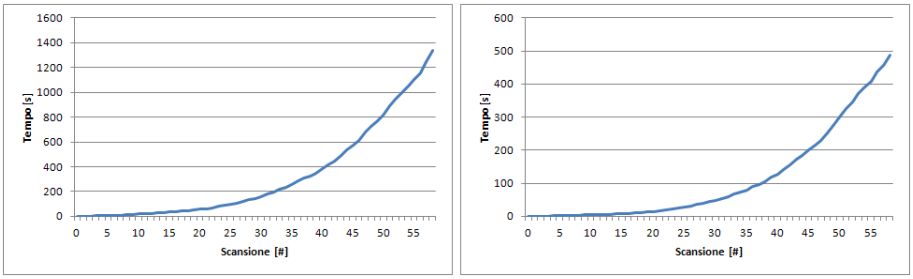

Figura 6.26: Crescita del tempo computazionale totale a ogni nuova scansione rilevata per le prove sperimentali, relative al metodo Basic on-line, eseguite senza filtraggio sul data set Stanford e con parametri di distanza spaziale e angolare pari a 100 millimetri e 10 gradi (figura a sinistra), e 400 millimetri e 20 gradi (figura a destra).

varia tra il 39% e il 56%, e risultati molto buoni per quanto riguarda la qualità della mappa finale ridotta, che varia tra l'81% e il 90%. In queste prove la qualità risulta buona (intorno all'82%) anche per parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi). Aumentando in maniera considerevole i parametri nelle prove senza filtraggio, il metodo genera comunque mappe finali caratterizzate da una buona qualità e da una buona percentuale di segmenti ridotti, come mostrato in Figura 6.27. Nelle prove spe-

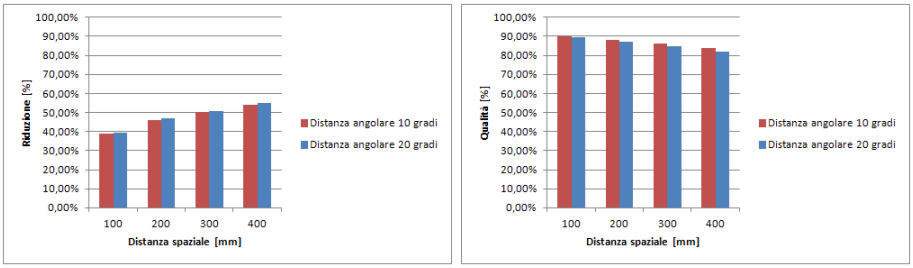

Figura 6.27: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Hybrid nelle prove sperimentali eseguite senza filtraggio sul data set Stanford.

rimentali eseguite con filtraggio, invece, il metodo produce risultati scarsi per quanto riguarda la percentuale dei segmenti ridotti, che varia tra il 18% e il 40%, e risultati ottimi per quanto riguarda la qualità della mappa finale ridotta, che varia tra l'89% e il 99%. In queste prove la qualità della mappa finale ridotta risulta ottima (intorno all'89%) anche per parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi). Aumentando in maniera

considerevole i parametri nelle prove eseguite con filtraggio, il metodo produce comunque mappe finali caratterizzate da una ottima qualità e da una scarsa percentuale di segmenti ridotti, come mostrato in Figura 6.28. La qualità risulta migliore nelle prove sperimentali eseguite con

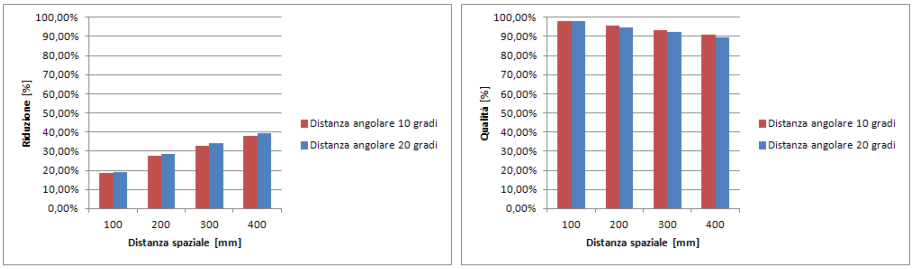

Figura 6.28: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Hybrid nelle prove sperimentali eseguite con filtraggio sul data set Stanford.

filtraggio rispetto a quelle eseguite senza filtraggio. Per questo data set, inoltre, le differenze tra le prove eseguite senza filtraggio e quelle eseguite con filtraggio sono ancora più evidenti: tali differenze sono giustificate, come già spiegato nella sezione precedente, dal filtraggio interno compiuto dal metodo. Infine, dato che il tempo computazionale totale varia tra i 35 secondi e il minuto circa, il metodo risulta molto veloce. In Figura 6.29 è rappresentata, per alcune prove spe-

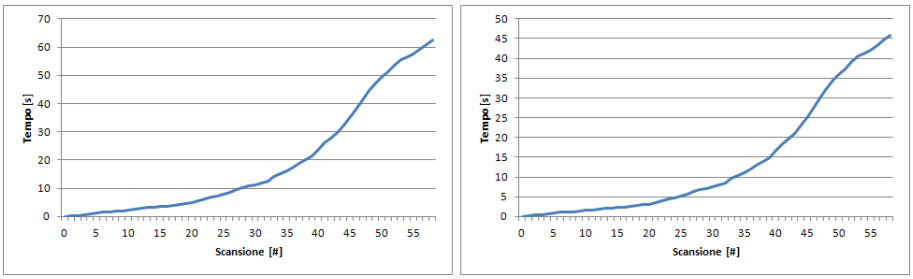

Figura 6.29: Crescita del tempo computazionale totale a ogni nuova scansione rilevata per le prove sperimentali, relative al metodo Hybrid, eseguite senza filtraggio sul data set Stanford e con parametri di distanza spaziale e angolare pari a 100 millimetri e 10 gradi (figura a sinistra), e 400 millimetri e 20 gradi (figura a destra).

rimentali, la crescita del tempo computazionale totale a ogni nuova scansione rilevata. A differenza di quanto avveniva per il data set Polimi, in questo caso il tempo non fa registrare una crescita lineare, ma un andamento molto particolare. Questo comportamento viene

giustificato dal fatto che il tempo computazionale speso dal robot a ogni singola iterazione del metodo è strettamente legato al numero dei segmenti presenti nella nuova scansione: il metodo confronta infatti tutti i segmenti della scansione con tutti i segmenti della mappa e, di conseguenza, un aumento dei segmenti della scansione provoca un aumento notevole dei confronti. Ad esempio, se nella mappa corrente sono presenti 140 segmenti e nella nuova scansione ne sono presenti 2, il numero dei confronti è 280 (140×2), mentre se ne sono presenti 10, il numero dei confronti sale fino a 1400 (140×10), ossia 5 volte tanto;

- il metodo Mean Shift Clustering fornisce risultati molto buoni solo per valori non troppo elevati del parametro di distanza spaziale. Nelle prove sperimentali, eseguite con o senza filtraggio e con un parametro di distanza spaziale non troppo elevato (100 o 200 millimetri), la qualità della mappa finale ridotta risulta ottima (tra l'83% e il 92%) e la riduzione buona (tra il 43% e il 54%). Aumentando tale parametro

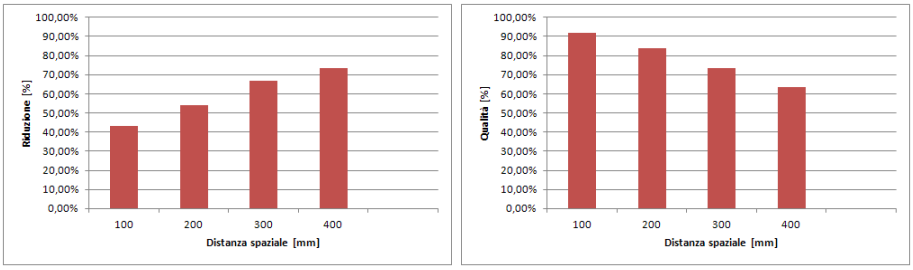

Figura 6.30: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Mean Shift Clustering nelle prove sperimentali eseguite senza filtraggio sul data set Stanford.

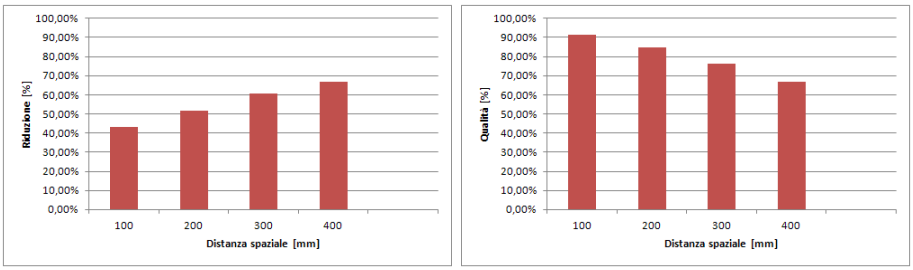

Figura 6.31: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Mean Shift Clustering nelle prove sperimentali eseguite con filtraggio sul data set Stanford.

(fino a 400 millimetri), il metodo genera mappe caratterizzate da una percentuale dei segmenti ridotti molto alta (tra il 66% e il 74%) e da una qualità solo discreta (tra il 63% e il 67%). Come avveniva per il data set Polimi, e per le stesse ragioni, anche per il data set Stanford l'aumento del parametro di distanza spaziale comporta grandi variazioni a livello di riduzione dei segmenti e di qualità, come mostrato in Figura 6.30 e 6.31. La qualità, inoltre, risulta generalmente migliore nelle prove sperimentali eseguite con filtraggio rispetto a quelle eseguite senza filtraggio. Il metodo è infine molto veloce: il tempo computazionale varia tra gli 8 secondi e il minuto circa;

- i risultati prodotti dal metodo Fusion sono molto buoni solo per piccoli valori del parametro di distanza spaziale. La qualità della mappa finale ridotta risulta ottima (tra l'83% e il 94%) per prove sperimentali, sia con filtraggio che senza, eseguite con un parametro di distanza spaziale piuttosto basso (20 o 40 millimetri): in queste prove si ottiene comunque una riduzione molto buona (tra il 42% e il 61%). All'au-

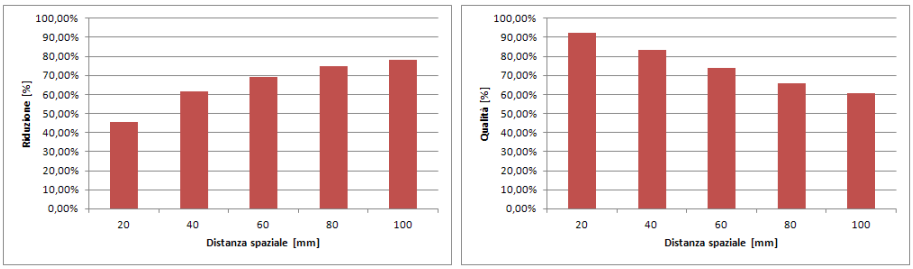

Figura 6.32: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Fusion nelle prove sperimentali eseguite senza filtraggio sul data set Stanford.

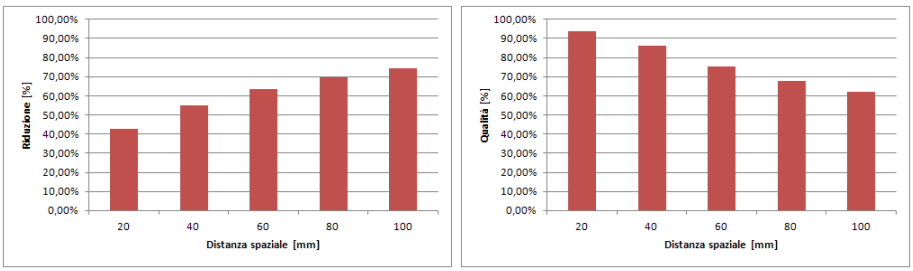

Figura 6.33: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Fusion nelle prove sperimentali eseguite con filtraggio sul data set Stanford.

mentare del parametro di distanza spaziale (fino a 80 o 100 millimetri),

il metodo riduce molti segmenti (tra il 69% e il 79%), generando però mappe con una qualità solo discreta (tra il 60% e il 68%). Anche per questo data set, quindi, un aumento lieve del parametro di distanza spaziale comporta grandi variazioni a livello di riduzione dei segmenti e di qualità (Figura 6.32 e 6.33): questa è una conferma dei risultati ottenuti per il data set Polimi. La qualità risulta migliore nelle prove sperimentali eseguite con filtraggio rispetto a quelle eseguite senza filtraggio. Il metodo è inoltre molto veloce: il tempo computazionale varia tra i 24 e i 50 secondi.

Sul data set Stanford valgono, relativamente al tempo computazionale, le stesse considerazioni generali effettuate per il data set Polimi.
Il metodo Fusion è quello che fornisce le prestazioni migliori sul data set Stanford: questo risulta evidente da un confronto delle prove sperimentali.
Per quanto riguarda le prove sperimentali eseguite senza filtraggio:

- con un valore di parametro di distanza spaziale pari a 20 millimetri, si ottiene, eseguendo il metodo Fusion, una mappa finale caratterizzata da una qualità pari a circa il 92% e da una percentuale di segmenti ridotti pari a circa il 45%. Come mostrato in Figura 6.34, gli altri metodi di fusione generano mappe, con una qualità inferiore o leggermente superiore, caratterizzate da una minore percentuale di segmenti ridotti;

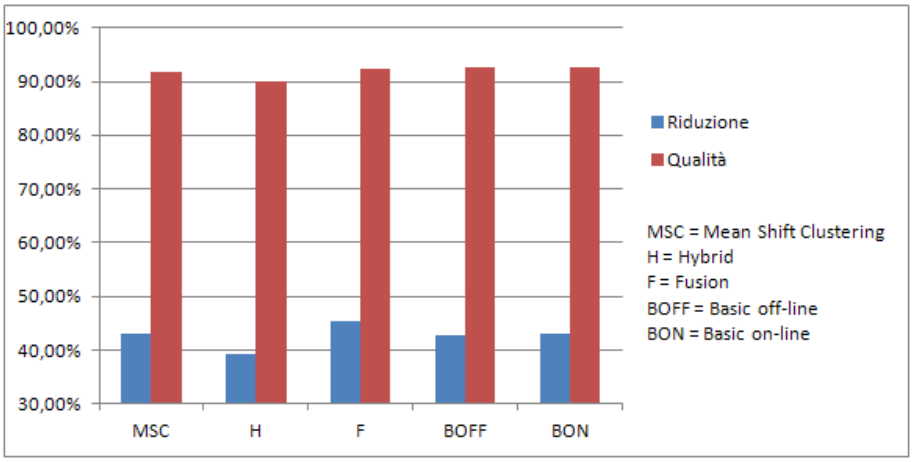

Figura 6.34: Nelle prove sperimentali eseguite senza filtraggio sul data set Stanford, in mappe finali caratterizzate da una qualità pari a circa il 92%, il metodo Fusion riduce più segmenti rispetto a tutti gli altri metodi.

- utilizzando un valore di parametro di distanza spaziale pari a 40 millimetri, si ottiene, applicando il metodo Fusion, una mappa finale caratterizzata da una qualità pari a circa l'83% e da una percentuale dei segmenti ridotti pari a circa il 62%. Come mostrato in Figura 6.35, gli altri metodi di fusione producono mappe, con una qualità inferiore o leggermente superiore all'83%, caratterizzate da una minore percentuale di segmenti ridotti.

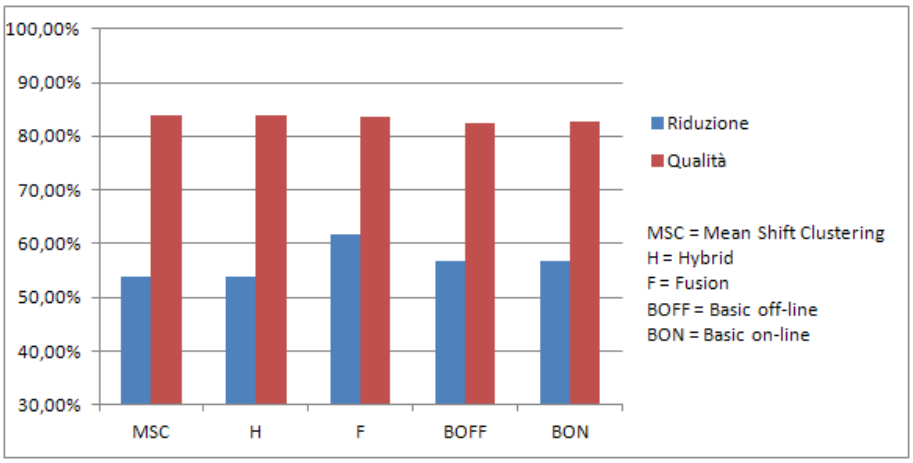

Figura 6.35: Nelle prove sperimentali eseguite senza filtraggio sul data set Stanford, in mappe finali caratterizzate da una qualità pari a circa l'83%, il metodo Fusion riduce più segmenti rispetto a tutti gli altri metodi.

Per quanto riguarda le prove sperimentali eseguite con filtraggio:

- utilizzando un valore di parametro di distanza spaziale pari a 20 millimetri, si ottiene, applicando il metodo Fusion, una mappa finale caratterizzata da una qualità pari a circa il 94% e da una percentuale di segmenti ridotti pari a circa il 43%. Come mostrato in Figura 6.36, il metodo Hybrid genera una mappa, con qualità inferiore al 94%, caratterizzata da una minore percentuale di segmenti ridotti. Il metodo Mean Shift Clustering, invece, produce una mappa caratterizzata da una percentuale di segmenti ridotti leggermente superiore a quella del metodo Fusion, tuttavia la sua qualità è abbastanza inferiore e, di conseguenza, anche in questo caso il metodo Fusion offre prestazioni migliori;

- con un valore di parametro di distanza spaziale pari a 40 millimetri, si ottiene, eseguendo il metodo Fusion, una mappa finale caratterizzata

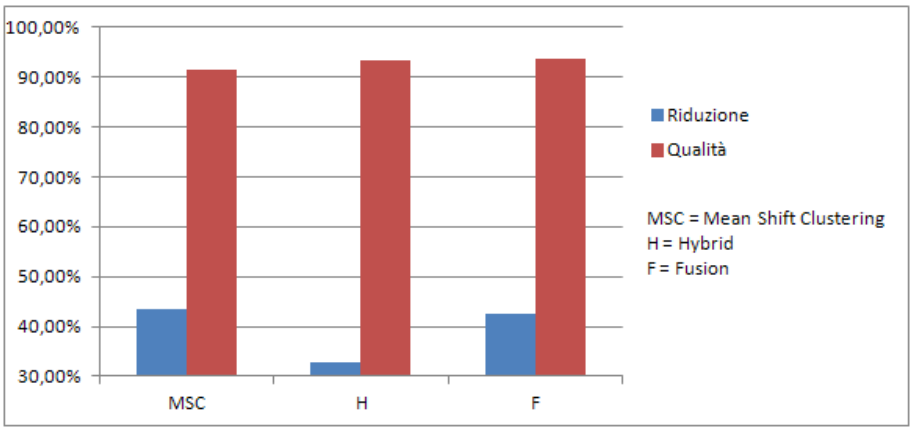

Figura 6.36: Nelle prove sperimentali eseguite con filtraggio sul data set Stanford, in mappe finali caratterizzate da una qualità pari a circa il 94%, il metodo Fusion offre prestazioni migliori rispetto al metodo Hybrid e al metodo Mean Shift Clustering.

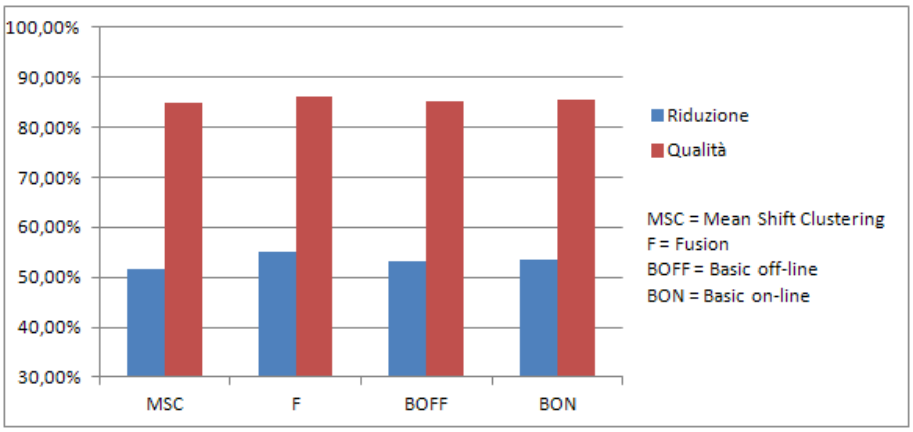

Figura 6.37: Nelle prove sperimentali eseguite con filtraggio sul data set Stanford, in mappe finali caratterizzate da una pari a circa l'86%, il metodo Fusion riduce più segmenti rispetto al metodo Mean Shift Clustering e alle due versioni del metodo Basic.

da una qualità pari a circa l'86% e da una percentuale dei segmenti ridotti pari a circa il 55%. Come mostrato in Figura 6.37, il metodo Mean Shift Clustering e le due versioni del metodo Basic generano mappe, con una qualità inferiore all'86%, caratterizzate da una minore percentuale di segmenti ridotti.

Da tutte queste osservazioni possiamo concludere che sul data set Stanford il metodo Fusion fornisce, in generale, prestazioni superiori rispetto a tutti gli altri metodi. Inoltre, osserviamo come le due versioni del metodo Basic producono ancora, a parità di parametri in ingresso, risultati molto simili sia per quanto riguarda la percentuale dei segmenti ridotti, sia per quanto riguarda la qualità della mappa finale ridotta.

6.5 Data set Bicocca

Il data set Bicocca è costituito da una sequenza di scansioni, accompagnate dal rispettivo ground thruth, disponibili in rete nel *Rawseeds project* [50]. Il ground thruth fornito, relativo alla traiettoria seguita dal robot durante l'esplorazione dell'ambiente, è stato ricostruito utilizzando due sistemi tra loro indipendenti. Un sistema basato su telecamere, *tag* visivi montati sul robot e software ad hoc, ed un sistema basato su di un insieme di telemetri laser e software ad hoc utilizzato per localizzare uno scafo rettangolare montato sul robot. Su questo data set abbiamo eseguito, per ogni metodo di fusione considerato nelle sezioni precedenti, una serie di prove sperimentali di valutazione, i cui risultati completi sono riportati in Appendice A. Abbiamo effettuato due diverse tipologie di prove sperimentali: quelle senza filtraggio e quelle con filtraggio. Nelle prove senza filtraggio la mappa iniziale è costituita da 2220 segmenti, mentre nelle prove con filtraggio la mappa iniziale è costituita da 1320 segmenti. Analizziamo quindi, per ogni metodo, i risultati di queste prove sperimentali, secondo i tre diversi parametri di valutazione definiti nella Sezione 6.2.3:

- i risultati forniti dal metodo Basic off-line sono ottimi per quanto riguarda la percentuale dei segmenti ridotti, che oscilla tra il 61% e l'86%, e molto variabili per quanto riguarda la qualità della mappa finale ridotta, che oscilla tra il 61% e l'87%. La qualità risulta scarsa (tra il 61% e il 65%) per prove sperimentali, sia con filtraggio che senza, eseguite con parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi). A differenza di quanto avveniva per i precedenti data set, il metodo non si mostra robusto se applicato al data set Bicocca: l'aumento dei parametri di

distanza spaziale e angolare comporta infatti, in questo caso, grandi variazioni a livello di riduzione dei segmenti e di qualità, come mostrato in Figura 6.38 e 6.39. Questo comportamento è motivato dalla

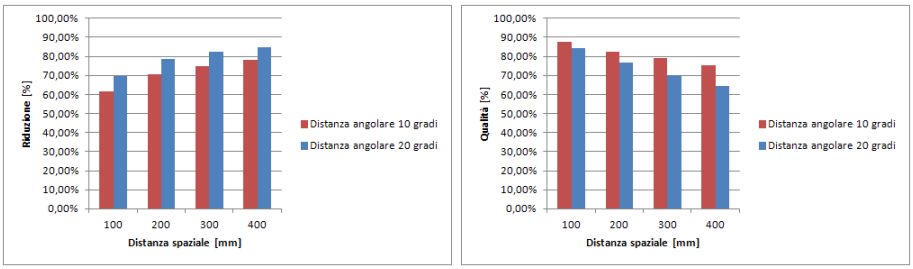

Figura 6.38: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic off-line nelle prove sperimentali eseguite senza filtraggio sul data set Bicocca.

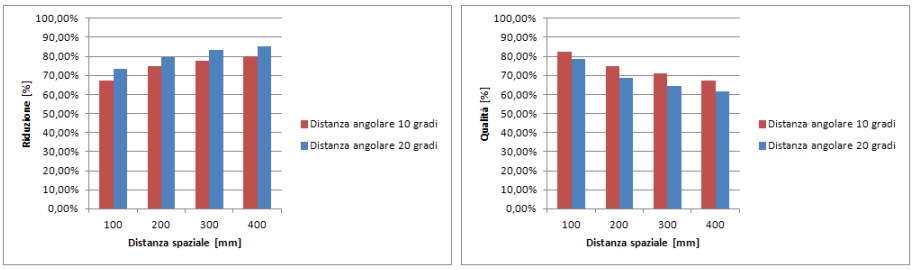

Figura 6.39: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic off-line nelle prove sperimentali eseguite con filtraggio sul data set Bicocca.

differente composizione delle mappe: i data set Polimi e Stanford corrispondono a mappe pulite e caratterizzate da uno scarso rumore; il data set Bicocca corrisponde invece a una mappa densa e rumorosa. Nel primo caso il metodo, come già spiegato nelle sezioni precedenti, è robusto ed in grado di generare mappe finali caratterizzate da una buona qualità e da una buona percentuale dei segmenti ridotti, anche per valori molto elevati dei parametri di distanza spaziale e angolare in ingresso. Nel secondo caso, invece, il metodo non è in grado di fornire prestazioni ottimali, per via della complessità della mappa, e, per parametri in ingresso elevati, genera mappe finali caratterizzate da una scarsa qualità. La percentuale dei segmenti ridotti risulta maggiore nelle prove sperimentali eseguite con filtraggio rispetto a quelle

eseguite senza filtraggio. Dato che il tempo computazionale varia tra i 17 secondi e i 3 minuti circa, il metodo si mostra abbastanza veloce.

- il metodo Basic on-line produce risultati ottimi per quanto riguarda la percentuale dei segmenti ridotti, che oscilla tra il 61% e l'85%, e risultati molto variabili per quanto riguarda la qualità della mappa finale ridotta, che oscilla tra il 61% e l'89%. Le prestazioni fornite dal metodo sono, anche per questo data set, simili a quelle della corrispondente versione off-line: il metodo non è quindi robusto (Figura 6.40 e 6.41) e, all'aumentare dei parametri di distanza spaziale e angolare, non è in grado di generare mappe caratterizzate da una buona qualità. Anche

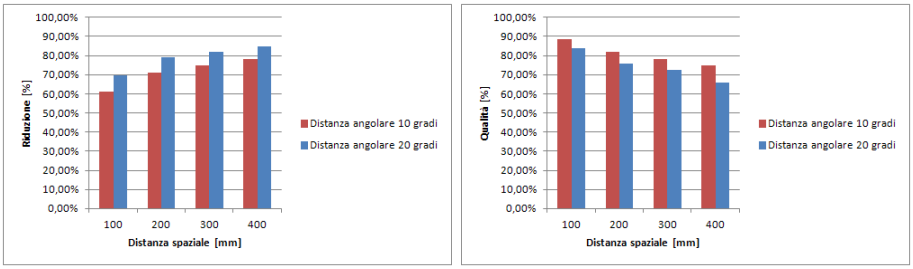

Figura 6.40: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic on-line nelle prove sperimentali eseguite senza filtraggio sul data set Bicocca.

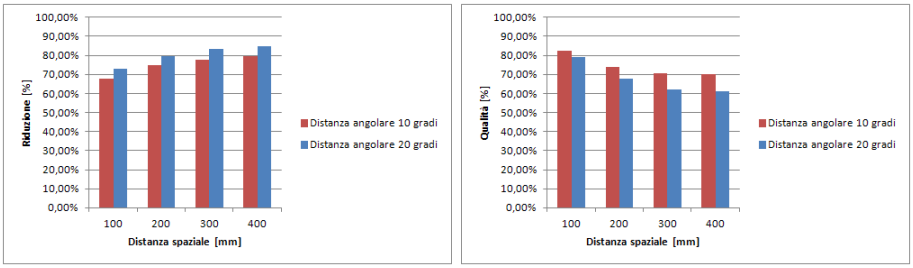

Figura 6.41: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic on-line nelle prove sperimentali eseguite con filtraggio sul data set Bicocca.

in questo caso, la percentuale dei segmenti ridotti risulta maggiore nelle prove sperimentali eseguite con filtraggio rispetto a quelle eseguite senza filtraggio. Il metodo è molto lento: il tempo computazionale totale varia tra i 6 minuti e le 2 ore circa. In Figura 6.42 è mostrata la crescita del tempo computazionale totale, a ogni nuova scansione

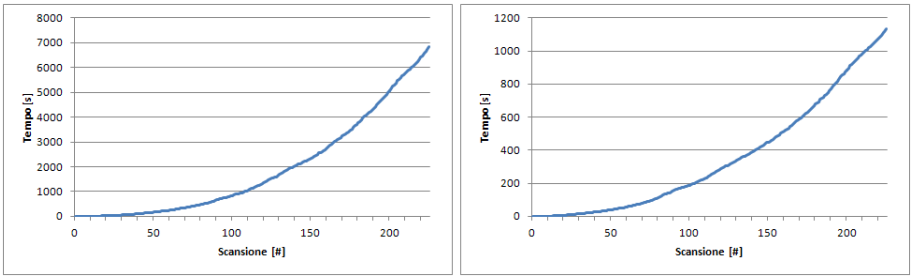

Figura 6.42: Crescita del tempo computazionale totale a ogni nuova scansione rilevata per le prove sperimentali, relative al metodo Basic on-line, eseguite senza filtraggio sul data set Bicocca e con parametri di distanza spaziale e angolare pari a 100 millimetri e 10 gradi (figura a sinistra), e 400 millimetri e 20 gradi (figura a destra).

rilevata, per un paio di prove sperimentali: anche per questo data set si registra un andamento esponenziale;

- nelle prove sperimentali eseguite senza filtraggio, il metodo Hybrid fa registrare risultati ottimi per quanto riguarda la percentuale dei segmenti ridotti, che varia tra il 67% e l'84%, e risultati buoni per quanto riguarda la qualità della mappa finale ridotta, che varia tra il 70% e l'82%. In queste prove, tuttavia, la qualità risulta solo discreta (intorno al 71%) per parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi). Il metodo quindi produce, nelle prove eseguite senza filtraggio e con parametri elevati, mappe finali caratterizzate da una discreta qualità e da una ottima percentuale di segmenti ridotti, come mostrato in Figura 6.43. Nelle

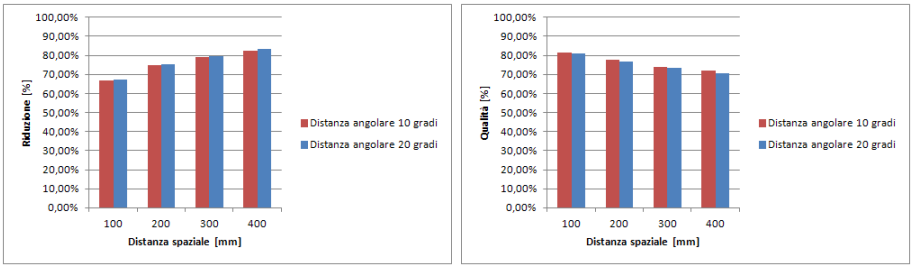

Figura 6.43: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Hybrid nelle prove sperimentali eseguite senza filtraggio sul data set Bicocca.

prove sperimentali eseguite con filtraggio, invece, il metodo fornisce risultati molto buoni per quanto riguarda la percentuale dei segmenti ridotti, che varia tra il 45% e il 72%, e risultati ottimi per quanto

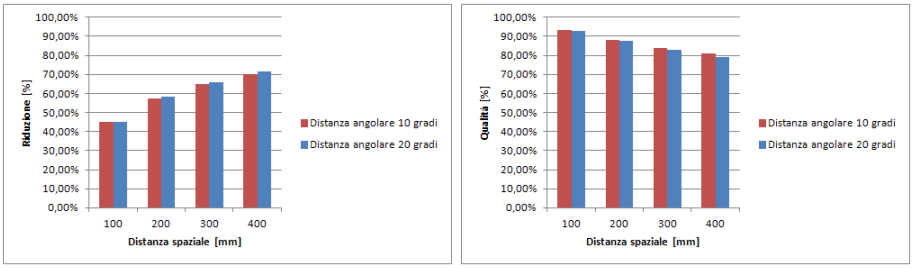

Figura 6.44: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Hybrid nelle prove sperimentali eseguite con filtraggio sul data set Bicocca.

riguarda la qualità della mappa finale ridotta, che varia tra il 79% e il 94%. In queste prove la qualità risulta buona (intorno al 79%) anche per parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi). All'aumentare del valore dei parametri in ingresso, il metodo genera quindi mappe finali caratterizzate da una buona qualità e da una ottima percentuale di segmenti ridotti, come mostrato in Figura 6.44. La qualità risulta migliore nelle prove sperimentali eseguite con filtraggio rispetto a quelle eseguite senza filtraggio. Anche per questo data set, inoltre, le differenze tra le prove eseguite senza filtraggio e quelle eseguite con filtraggio sono molto evidenti: ancora una volta, queste differenze sono causate dal filtraggio interno eseguito dal metodo. Infine, dato che il tempo computazionale varia tra i 25 secondi e il minuto circa, il metodo è molto veloce. In Figura 6.45 è mostrata, per alcune prove sperimentali, la

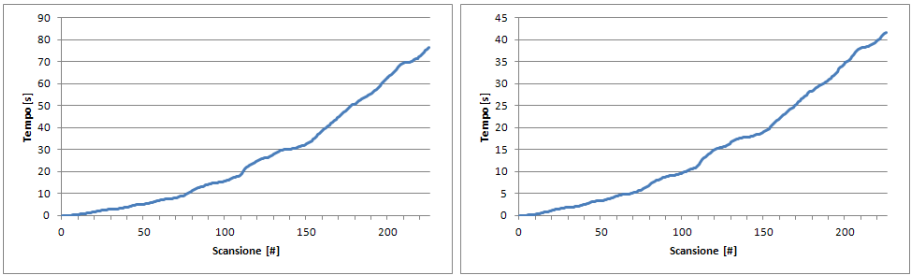

Figura 6.45: Crescita del tempo computazionale totale a ogni nuova scansione rilevata per le prove sperimentali, relative al metodo Hybrid, eseguite senza filtraggio sul data set Bicocca e con parametri di distanza spaziale e angolare pari a 100 millimetri e 10 gradi (figura a sinistra), e 400 millimetri e 20 gradi (figura a destra).

crescita del tempo computazionale totale a ogni nuova scansione ri-

levata. Anche per questo data set si può notare un andamento quasi lineare: le variazioni del tempo computazionale tra le singole iterazioni sono giustificabili, come già spiegato nella sezione precedente, con il diverso numero di segmenti presenti nella scansione;

- il metodo Mean Shift Clustering produce risultati buoni solo per valori molto bassi del parametro di distanza spaziale. La qualità della mappa finale ridotta risulta buona (tra il 77% e l'81%) per prove sperimentali, sia con filtraggio che senza, eseguite con un parametro di distanza spaziale molto piccolo (100 millimetri): in queste prove si ottiene una ottima riduzione (tra il 65% e il 71%). Con l'aumento

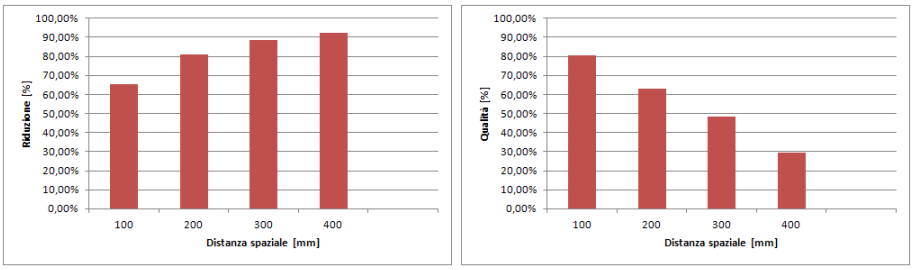

Figura 6.46: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Mean Shift Clustering nelle prove sperimentali eseguite senza filtraggio sul data set Bicocca.

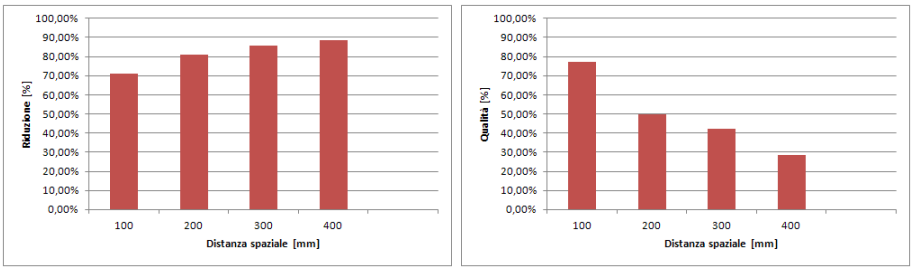

Figura 6.47: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Mean Shift Clustering nelle prove sperimentali eseguite con filtraggio sul data set Bicocca.

di tale parametro (fino a 400 millimetri), il metodo produce mappe finali caratterizzate da una percentuale dei segmenti ridotti molto alta (tra l'88% e il 93%) e da una qualità pessima (tra il 28% e il 30%). Rispetto ai precedenti data set, il metodo si mostra ancora meno robusto: le mappe generate hanno una qualità che, all'aumentare del

parametro in ingresso, decresce in maniera novetole, fino a diventare pessima (Figura 6.46 e 6.47). Il metodo Mean Shift Clustering, quindi, offre prestazioni ancora più basse nel caso in cui viene applicato a mappa dense e molto rumorose: data la maggiore concentrazione dei segmenti, il clustering avviene in una maniera molto più difficoltosa e comporta una serie di errori che si ripercuotono nella fase finale di fusione. Sia la percentuale dei segmenti ridotti che la qualità risultano generalmente migliori nelle prove sperimentali eseguite senza filtraggio rispetto a quelle eseguite con filtraggio: l'unica eccezione, in questo caso, è rappresentata dalla prova sperimentale eseguita con parametro di distanza spaziale pari a 100 millimetri. Il metodo è moderatamente veloce: il tempo computazionale impiegato varia tra i 55 secondi e i 14 minuti circa;

- il metodo Fusion evidenzia risultati molto buoni solo per piccoli valori del parametro di distanza spaziale. Infatti, nelle prove sperimentali,

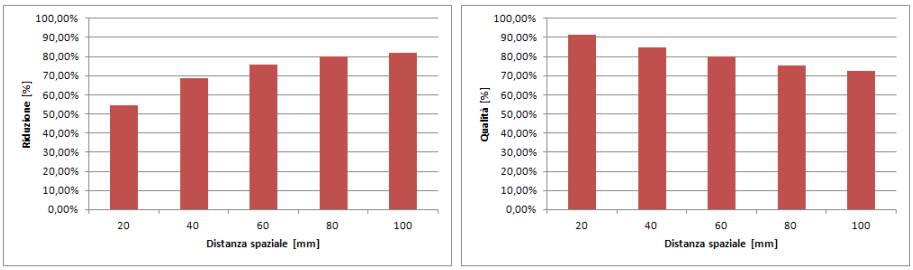

Figura 6.48: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Fusion nelle prove sperimentali eseguite senza filtraggio sul data set Bicocca.

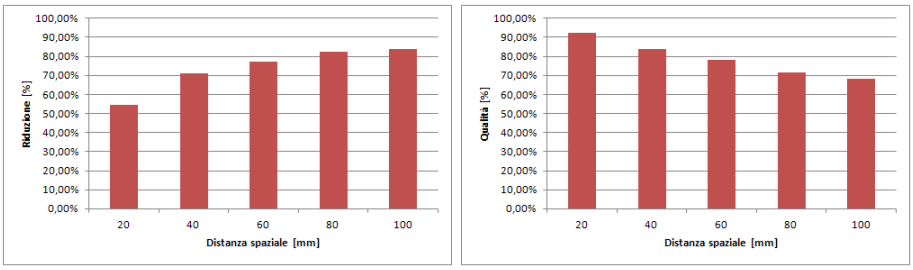

Figura 6.49: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Fusion nelle prove sperimentali eseguite con filtraggio sul data set Bicocca.

eseguite con o senza filtraggio e con un parametro di distanza spaziale piuttosto basso (20 o 40 millimetri), la qualità della mappa finale

ridotta risulta ottima (tra l'83% e il 93%) e la riduzione molto buona (tra il 54% e il 72%). Aumentando il parametro di distanza spaziale (fino a 80 o 100 millimetri), il metodo produce mappe finali caratterizzate da una percentuale dei segmenti ridotti molto alta (tra l'80% e l'84%) e da una qualità solo discreta (tra il 68% e il 76%). Anche per il data set Bicocca, l'aumento, seppur lieve, del parametro di distanza spaziale comporta grandi variazioni a livello di riduzione dei segmenti e di qualità, come mostrato in Figura 6.48 e 6.49. La qualità risulta migliore nelle prove sperimentali eseguite senza filtraggio rispetto a quelle eseguite con filtraggio. Il tempo computazionale varia tra i 34 secondi e i 2 minuti circa, quindi si tratta di un metodo molto veloce.

Sul data set Bicocca valgono, relativamente al tempo computazionale, le stesse considerazioni fatte per i data set Polimi e Stanford.
Il metodo Fusion è quello che fornisce le prestazioni migliori sul data set Bicocca: questo risulta evidente da un confronto delle prove sperimentali.
Per quanto riguarda le prove sperimentali eseguite senza filtraggio:

- utilizzando un valore di parametro di distanza spaziale pari a 40 millimetri, si ottiene, eseguendo il metodo Fusion, una mappa finale caratterizzata da una qualità pari a circa l'85% e da una percentuale di segmenti ridotti pari a circa il 69%. Come mostrato in Figura 6.50, il metodo Hybrid e il metodo Mean Shift Clustering producono mappe, con una qualità inferiore all'85%, caratterizzate da una minore percentuale di segmenti ridotti;

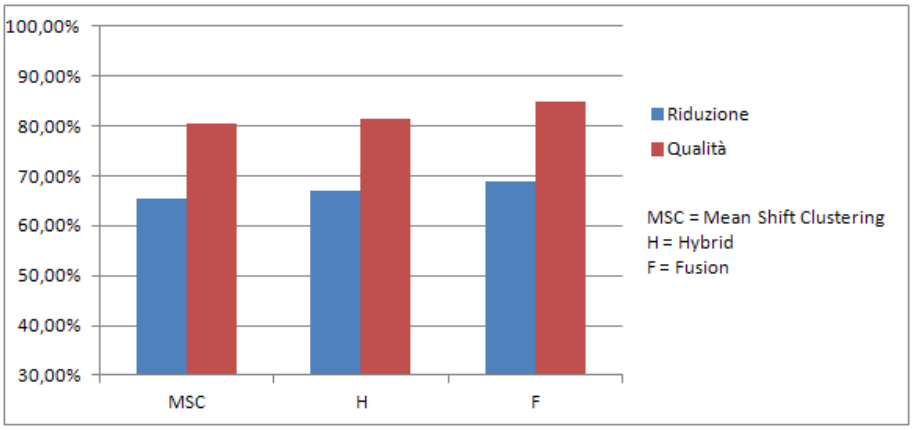

Figura 6.50: Nelle prove sperimentali eseguite senza filtraggio sul data set Bicocca, in mappe finali caratterizzate da una qualità pari a circa l'85%, il metodo Fusion riduce più segmenti rispetto al metodo Hybrid e al metodo Mean Shift Clustering.

- con un valore di parametro di distanza spaziale pari a 60 millimetri, si ottiene, applicando il metodo Fusion, una mappa finale caratterizzata da una qualità pari a circa l'80% e da una percentuale dei segmenti ridotti pari a circa il 76%. Come mostrato in Figura 6.51, il metodo Hybrid e le due versioni del metodo Basic generano mappe, con una qualità inferiore o leggermente superiore all'80%, caratterizzate da una minore percentuale di segmenti ridotti.

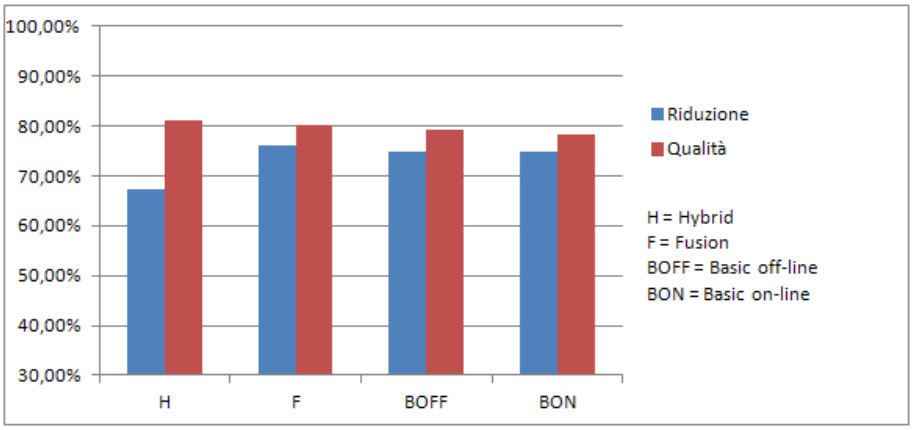

Figura 6.51: Nelle prove sperimentali eseguite senza filtraggio sul data set Bicocca, in mappe finali caratterizzate da una qualità pari a circa l'80%, il metodo Fusion riduce più segmenti rispetto al metodo Hybrid e alle due versioni del metodo Basic.

Per quanto riguarda le prove sperimentali eseguite con filtraggio:

- utilizzando un valore di parametro di distanza spaziale pari a 40 millimetri, si ottiene, applicando il metodo Fusion, una mappa finale caratterizzata da una qualità pari a circa l'84% e da una percentuale di segmenti ridotti pari a circa il 71%. Come mostrato in Figura 6.52, gli altri metodi generano una mappa, con una qualità inferiore o leggermente superiore all'84%, caratterizzata da una minore percentuale di segmenti ridotti.

- con un valore di parametro di distanza spaziale pari a 60 millimetri, si ottiene, eseguendo il metodo Fusion, una mappa finale caratterizzata da una qualità pari a circa il 78% e da una percentuale dei segmenti ridotti pari a circa il 77%. Come mostrato in Figura 6.53, il metodo Hybrid e le due versioni del metodo Basic producono mappe, con una qualità leggermente superiore al 78%, caratterizzate da una minore percentuale di segmenti ridotti.

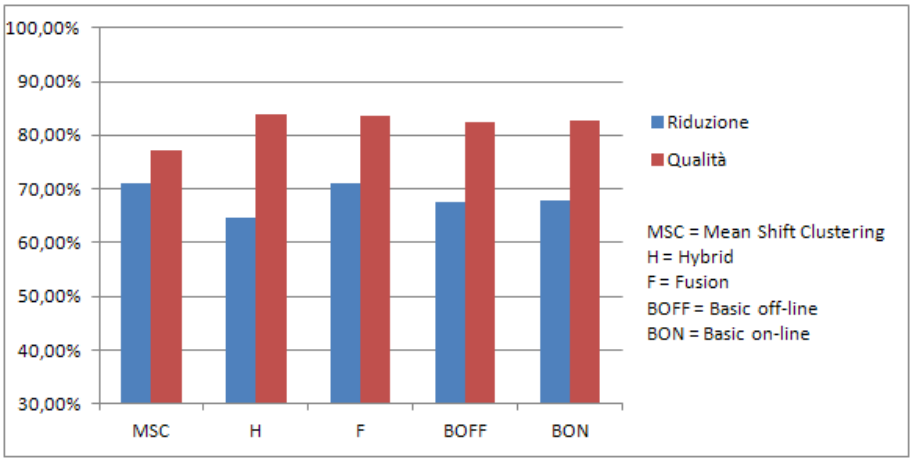

Figura 6.52: Nelle prove sperimentali eseguite con filtraggio sul data set Bicocca, in mappe finali caratterizzate da una qualità pari a circa l'84%, il metodo Fusion riduce più segmenti rispetto a tutti gli altri metodi. Nell'esempio riportato in figura, il metodo Mean Shift Clustering ha una percentuale di riduzione molto vicina a quella del metodo Fusion, tuttavia la sua qualità è parecchio inferiore e, di conseguenza, anche in questo caso il metodo Fusion offre prestazioni superiori rispetto al metodo Mean Shift Clustering.

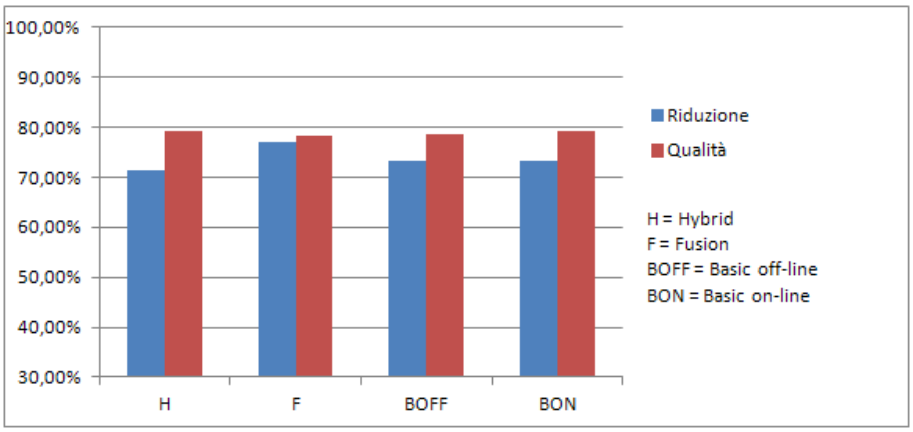

Figura 6.53: Nelle prove sperimentali eseguite con filtraggio sul data set Bicocca, in mappe finali caratterizzate da una qualità pari a circa il 78%, il metodo Fusion riduce più segmenti rispetto al metodo Hybrid e alle due versioni del metodo Basic.

Da tutte queste osservazioni possiamo concludere che sul data set Bicocca il metodo Fusion fornisce, in generale, prestazioni superiori rispetto a tutti gli altri metodi. Inoltre, osserviamo come le due versioni del metodo Basic producono, a parità di parametri in ingresso, risultati molto simili sia per quanto riguarda la percentuale dei segmenti ridotti, sia per quanto riguarda la qualità della mappa finale ridotta.

6.6 Data set USC

Il data set USC è costituito da una sequenza di scansioni disponibili in rete nel *Robotics Data Set Repository* (*Radish*) [35], sotto la voce *data set usc-sal200-021120*. L'insieme di dati è ottenuto attraverso due tour separati attraverso il secondo piano dell'edificio USC SAL. Il robot impiegato per la raccolta delle informazioni è un Pioneer 2DX con un telemetro laser *SICK LMS-200*. Per ogni metodo abbiamo eseguito una serie di prove sperimentali. I risultati completi sono riportati in Appendice A. Abbiamo effettuato due diverse tipologie di prove: quelle senza filtraggio e quelle con filtraggio. Nelle prove senza filtraggio la mappa iniziale è costituita da 3604 segmenti, mentre nelle prove con filtraggio la mappa iniziale è costituita da 2206 segmenti. Analizziamo quindi, per ogni metodo, i risultati ottenuti, secondo i tre diversi parametri di valutazione definiti nella Sezione 6.2.3:

- il metodo Basic off-line fornisce risultati ottimi per quanto riguarda la percentuale dei segmenti ridotti, che varia tra il 78% e il 94%, e risultati molto buoni per quanto riguarda la qualità della mappa ridotta, che varia tra il 74% e il 90%. In questo caso, a differenza di quanto accadeva con il data set Bicocca, la qualità della mappa finale risulta buona (intorno al 75%) anche per prove sperimentali, sia con filtraggio che senza, eseguite con parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi). Le prestazioni del metodo peggiorano, quindi, meno rapidamente all'aumentare dei parametri, tuttavia rimangono inferiori a quelle ottenute sui data set Polimi e Stanford. In Figura 6.54 e 6.55 sono riportati i risultati ottenuti. In questo caso non ci sono grosse differenze tra prove eseguite con e senza filtraggio per quanto riguarda la percentuale di segmenti ridotti, tuttavia la qualità delle mappe ottenute dalle prove con filtraggio risulta essere leggermente inferiore. Il metodo, come sui data set precedenti, è abbastanza veloce: il tempo varia tra i 23 secondi e i 4 minuti circa;

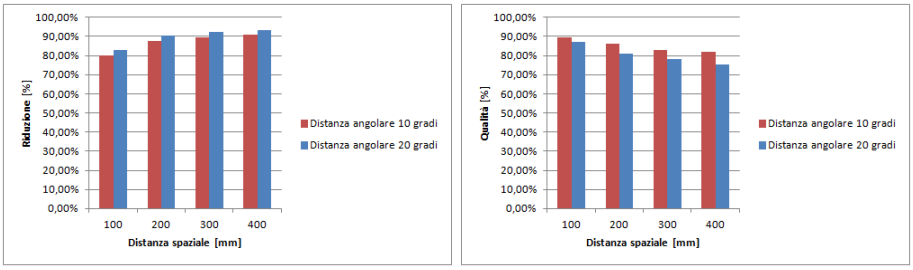

Figura 6.54: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic off-line nelle prove sperimentali eseguite senza filtraggio sul data set USC.

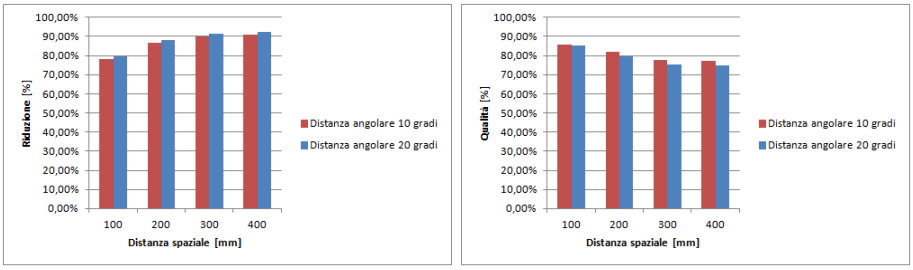

Figura 6.55: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic off-line nelle prove sperimentali eseguite con filtraggio sul data set USC.

- il metodo Basic on-line mantiene prestazioni simili a quelle ottenute sul data set Bicocca. Produce, anche in questo caso, risultati ottimi per quanto riguarda la percentuale dei segmenti ridotti, che oscilla tra il 79% e il 95%, e risultati molto variabili per quanto riguarda la qualità della mappa, che oscilla tra il 65% e il 90%. La qualità della mappa finale risulta discreta (tra il 65% e il 74%) per prove sperimentali, sia con filtraggio che senza, eseguite con parametri di distanza spaziale e angolare piuttosto elevati (rispettivamente 400 millimetri e 20 gradi). Le motivazioni di tale comportamento sono sempre da ricercare nella tipologia di data set che si sta utilizzando. Essendo il data set USC simile per caratteristiche al data set Bicocca, i risultati ottenuti non ci sorprendono ma, anzi, confermano le supposizioni fatte prima. In Figura 6.56 e 6.57 sono riportati i risultati ottenuti dal metodo all'aumentare dei parametri di distanza spaziale e angolare. La percentuale

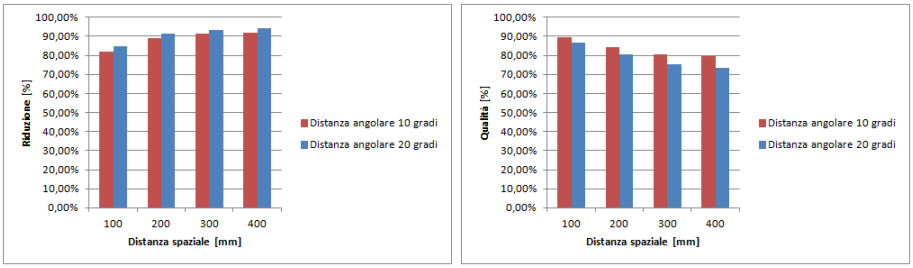

Figura 6.56: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic on-line nelle prove sperimentali eseguite senza filtraggio sul data set USC.

dei segmenti ridotti e la qualità delle mappe ottenute risulta maggiore nelle prove eseguite senza filtraggio. Per quanto riguarda il tempo

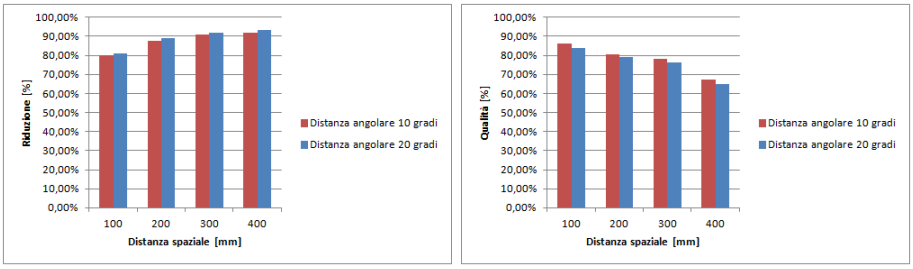

Figura 6.57: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Basic on-line nelle prove sperimentali eseguite con filtraggio sul data set USC.

computazionale, il metodo ottiene su questo data set le prestazioni peggiori. Infatti il tempo totale impiegato per la sua esecuzione varia tra gli 8 minuti e le 3 ore circa. Ciò è dovuto all'alto numero di segmenti che compone il data set. In Figura 6.58 è mostrata, per alcune

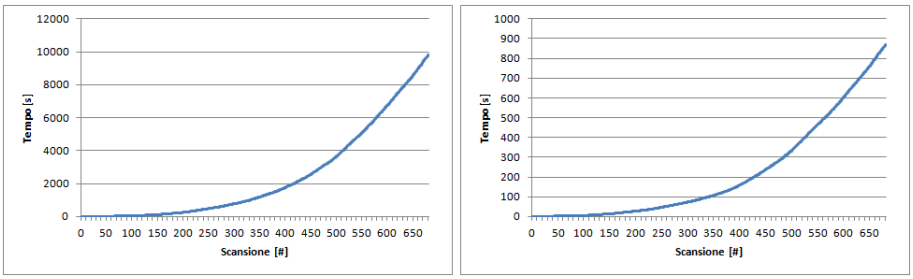

Figura 6.58: Crescita del tempo computazionale totale a ogni nuova scansione rilevata per le prove sperimentali, relative al metodo Basic on-line, eseguite senza filtraggio sul data set USC e con parametri di distanza spaziale e angolare pari a 100 millimetri e 10 gradi (figura a sinistra), e 400 millimetri e 20 gradi (figura a destra).

prove sperimentali, la crescita del tempo computazionale a ogni nuova scansione; l'andamento si mantiene anche in questo caso esponenziale.

- il metodo Hybrid produce risultati ottimi per quanto riguarda la percentuale dei segmenti ridotti e molto buoni per quanto riguarda la qualità della mappa finale sia nelle prove sperimentali eseguite senza filtraggio sia in quelle eseguite con filtraggio. Nel primo caso notiamo che la percentuale dei segmenti ridotti varia tra l'80% e il 92%, mentre la qualità varia tra il 77% e l'89%. Nel secondo caso la percentuale dei segmenti ridotti varia tra il 67% e l'87%, mentre la qualità tra il 79% e il 92%. In entrambi i casi, inoltre, la qualità della mappa finale risulta buona anche per parametri di distanza spaziale e angolare elevati (rispettivamente 400 millimetri e 20 gradi). Aumentando, quindi, i parametri in entrambe le tipologie di prove, il metodo produce mappe finali caratterizzate da una buona qualità e da una ottima percentuale di segmenti ridotti, come mostrato in Figura 6.59 e Figura 6.60. La qualità della mappa finale ridotta risulta migliore nelle prove sperimentali eseguite con filtraggio. Infatti, anche in questo caso, le differenze tra le due tipologie di prove sono molto evidenti e sono causate dal filtraggio interno utilizzato dal metodo. Il metodo è molto veloce, nonostante l'elevato numero di segmenti che compone il data set: il tempo computazionale varia tra i 32 secondi e i 2 minuti circa. In Figura 6.61 è mostrata, per alcune prove sperimentali, la crescita

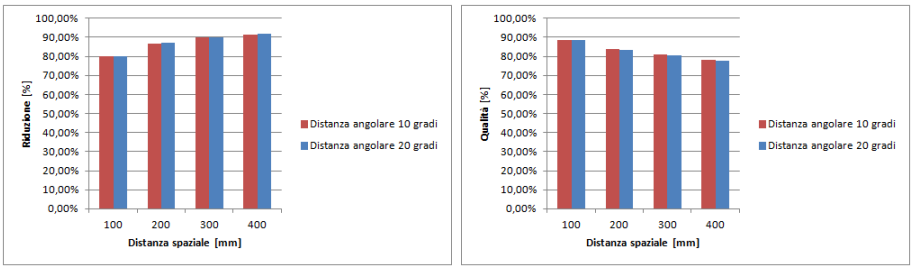

Figura 6.59: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Hybrid nelle prove sperimentali eseguite senza filtraggio sul data set USC.

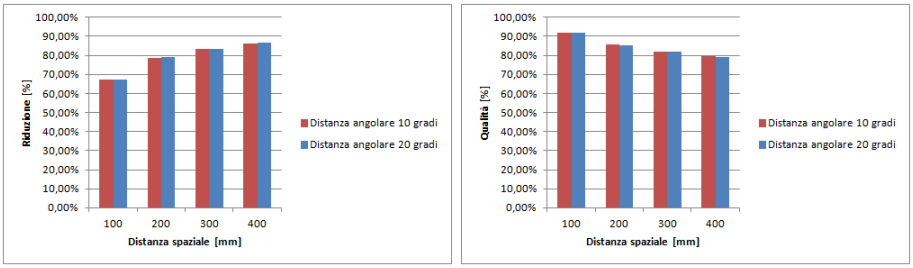

Figura 6.60: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Hybrid nelle prove sperimentali eseguite con filtraggio sul data set USC.

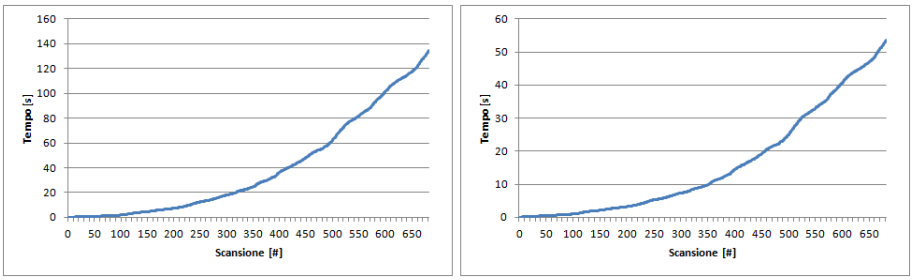

Figura 6.61: Crescita del tempo computazionale totale a ogni nuova scansione rilevata per le prove sperimentali, relative al metodo Hybrid, eseguite senza filtraggio sul data set USC e con parametri di distanza spaziale e angolare pari a 100 millimetri e 10 gradi (figura a sinistra), e 400 millimetri e 20 gradi (figura a destra).

del tempo computazionale totale a ogni nuova scansione rilevata. L'andamento quasi lineare che si verificava nei data set precedenti è ora meno accentuato in favore di un andamento quasi esponenziale. Ciò si verifica, in modo meno accentuato, anche nel data set Bicocca ed è dovuto all'elevato numero di segmenti che compone il data set. Infine notiamo le classiche variazioni del tempo computazionale tra le singole iterazioni tipiche del metodo Hybrid e dovute al numero differente di segmenti presenti in ogni scansione;

- il metodo Mean Shift Clustering produce risultati buoni solo per valori molto bassi del parametro di distanza spaziale. Ad esempio, impostando come distanza spaziale un valore molto piccolo (100 millimetri), otteniamo una buona qualità della mappa finale (tra il 69% e il 78%) ed una percentuale di riduzione ottima (tra l'82% e l'85%). Tuttavia, come si è verificato per il data set Bicocca, aumentando il parametro di distanza spaziale i risultati ottenuti sono caratterizzati da una percentuale di segmenti ridotti molto alta (tra il 93% e il 96%), ma da una qualità pessima (tra il 16% e il 22%). In questo metodo quindi l'aumento del parametro di distanza spaziale comporta un crollo delle prestazioni, come mostrato in Figura 6.62 e 6.63. Sia la percentuale

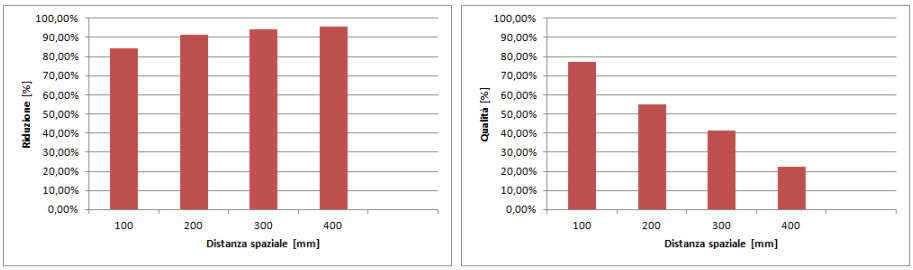

Figura 6.62: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Mean Shift Clustering nelle prove sperimentali eseguite senza filtraggio sul data set USC.

dei segmenti ridotti che la qualità della mappa finale risultano migliori nelle prove sperimentali eseguite senza filtraggio rispetto a quelle eseguite con filtraggio. Il Mean Shift Clustering, come il Basic on-line, è molto lento su questo data set: il tempo impiegato per la sua esecuzione varia tra i 12 minuti e le 2 ore circa. L'alto numero di segmenti rende infatti computazionalmente pesante la convergenza delle due fasi di clustering di cui si compone;

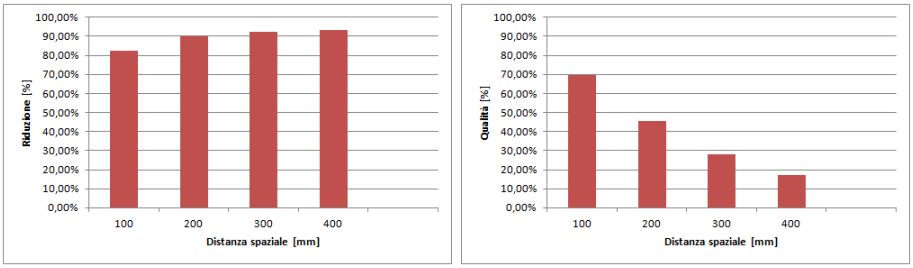

Figura 6.63: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Mean Shift Clustering nelle prove sperimentali eseguite con filtraggio sul data set USC.

- il metodo Fusion fornisce, nelle prove eseguite, prestazioni ottime, ma solo per piccoli valori del parametro di distanza spaziale. Ad esempio, impostando come valore per la distanza spaziale 20 o 40 millimetri, la qualità della mappa finale (tra l'86% e il 93%), cosi come la percentuale di riduzione (tra il 76% e l'88%), risulta ottima. Aumentando il valore del parametro (fino a 80 o 100 millimetri), il metodo produce mappe finali caratterizzate da una percentuale dei segmenti ridotti molto alta (tra il 92% e il 95%) a scapito di una minore qualità (tra il 65% e il 77%). Ciò è dovuto al fatto che il metodo cerca di approssimare, con un unico segmento, segmenti anche molto distanti tra loro, introducendo cosi errori sempre maggiori. In Figura 6.64 e 6.65 sono riportati i risultati delle prove eseguite. La qualità della mappa

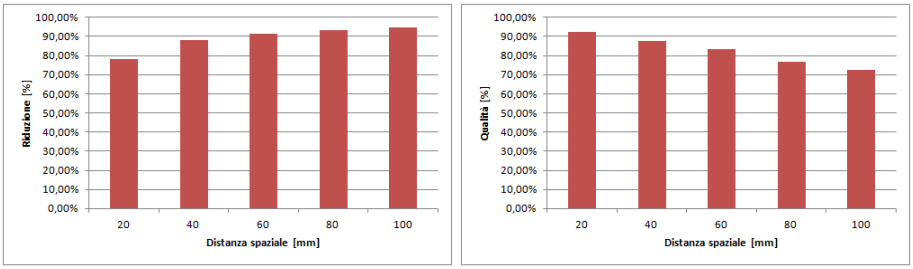

Figura 6.64: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Fusion nelle prove sperimentali eseguite senza filtraggio sul data set USC.

finale ridotta risulta migliore nelle prove sperimentali eseguite senza filtraggio rispetto a quelle eseguite con filtraggio: a una migliore qualità corrisponde però una minore percentuale dei segmenti ridotti. Il metodo si mantiene sempre molto veloce, nonostante l'elevato numero

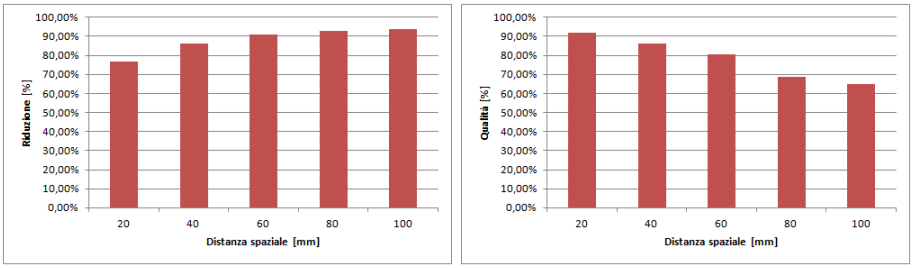

Figura 6.65: Percentuale dei segmenti ridotti e qualità della mappa finale ottenute con il metodo Fusion nelle prove sperimentali eseguite con filtraggio sul data set USC.

di segmenti di partenza. Infatti il tempo computazionale impiegato per la sua esecuzione varia tra i 34 secondi e i 2 minuti circa;

Sul data set USC valgono, relativamente al tempo computazionale, le stesse considerazioni fatte per gli altri data set.
Il metodo Fusion, anche in questo caso, è quello che fornisce le prestazioni migliori, e ciò risulta evidente confrontando le prove sperimentali.
Per quanto riguarda le prove sperimentali eseguite senza filtraggio:

- utilizzando un valore di parametro di distanza spaziale pari a 40 millimetri, si ottiene, con il metodo Fusion, una mappa finale caratterizzata da una qualità pari a circa l'87% e da una percentuale di segmenti ridotti pari a circa l'88%. Come mostrato in Figura 6.66, il metodo

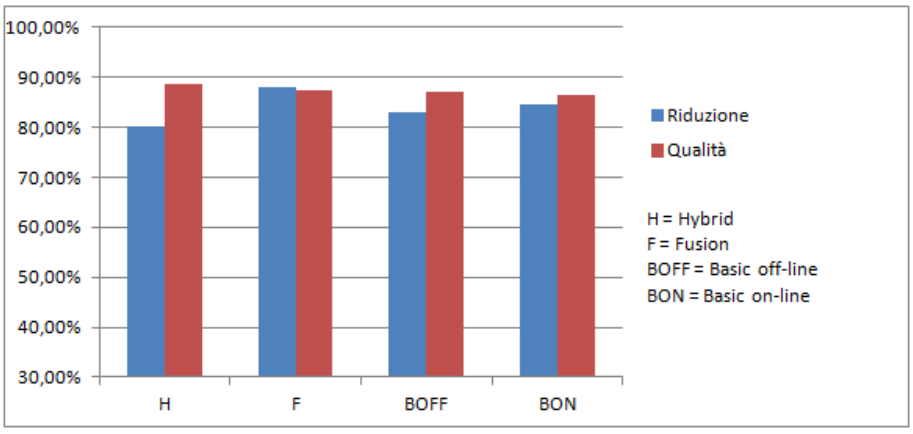

Figura 6.66: Nelle prove sperimentali eseguite senza filtraggio sul data set USC, in mappe finali caratterizzate da una qualità pari a circa l'87%, il metodo Fusion riduce più segmenti rispetto al metodo Hybrid e alle due versioni del metodo Basic.

Hybrid e le due versioni del metodo Basic producono mappe, con qua-

lità inferiore o leggermente superiore all'87%, caratterizzate da una minore percentuale di segmenti ridotti;

- utilizzando invece un valore di parametro di distanza spaziale pari a 60 millimetri, si ottiene, sempre con il metodo Fusion, una mappa finale con una qualità pari a circa l'83% e con una percentuale dei segmenti ridotti pari a circa il 91%. Come mostrato in Figura 6.67, gli altri

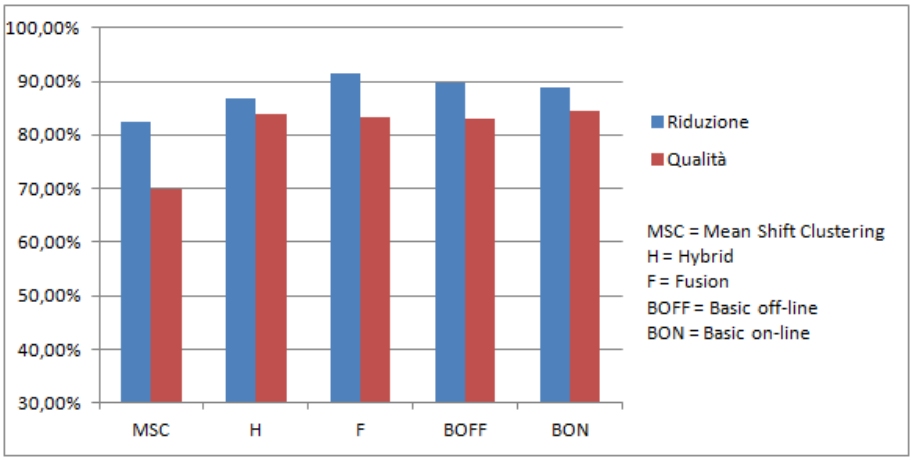

Figura 6.67: Nelle prove sperimentali eseguite senza filtraggio sul data set USC, in mappe finali caratterizzate da una qualità pari a circa l'83%, il metodo Fusion riduce più segmenti rispetto a tutti gli altri metodi.

metodi di fusione producono mappe, con una qualità inferiore o leggermente superiore all'83%, caratterizzate da una minore percentuale di segmenti ridotti.

Per quanto riguarda le prove eseguite con filtraggio:

- utilizzando un valore pari a 20 millimetri per la distanza spaziale, si ottiene, eseguendo il metodo Fusion, una mappa finale con una qualità pari a circa il 92% e con una percentuale di segmenti ridotti pari a circa il 77%. Come mostrato in Figura 6.68, il metodo Hybrid produce una mappa, con una qualità leggermente inferiore al 92%, caratterizzata da una minore percentuale di segmenti ridotti.

- utilizzando un valore di parametro di distanza spaziale pari a 40 millimetri, si ottiene, sempre con il metodo Fusion, una mappa caratterizzata da una qualità e da una percentuale dei segmenti ridotti pari a circa l'86%. Come mostrato in Figura 6.69, gli altri metodi di fusione

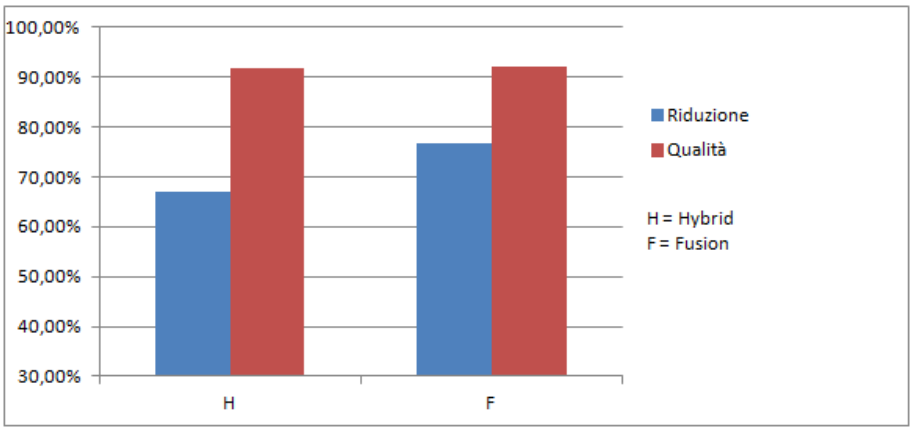

Figura 6.68: Nelle prove sperimentali eseguite con filtraggio sul data set USC, in mappe finali caratterizzate da una qualità pari a circa il 92%, il metodo Fusion riduce più segmenti rispetto al metodo Hybrid.

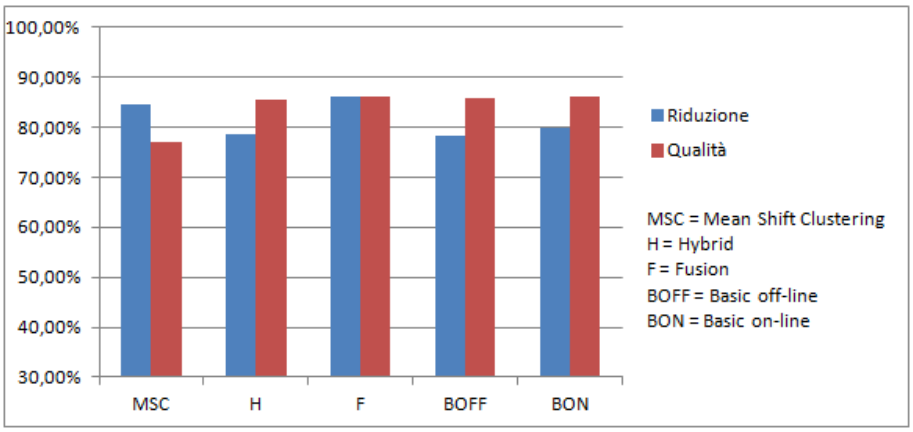

Figura 6.69: Nelle prove sperimentali eseguite con filtraggio sul data set USC, in mappe finali caratterizzate da una qualità pari a circa l'86%, il metodo Fusion riduce più segmenti rispetto a tutti gli altri metodi.

producono mappe, con una qualità inferiore o leggermente superiore all'86%, caratterizzate da una minore percentuale di segmenti ridotti.

Da tutte queste osservazioni possiamo concludere che sul data set USC il metodo Fusion fornisce, in generale, prestazioni superiori rispetto a tutti gli altri metodi.

6.7 Considerazioni finali

Riassumiamo ora i principali risultati ottenuti:

- le due versioni del metodo Basic (off-line e on-line) sono molto robuste se eseguite su mappe pulite e caratterizzate da uno scarso rumore, come nel caso dei data set Polimi e Stanford;
- le due versioni del metodo Basic forniscono prestazioni molto simili tra loro per quanto riguarda sia la qualità delle mappe finali, sia la percentuale dei segmenti ridotti;
- la versione off-line del metodo Basic è nettamente preferibile alla corrispondente versione on-line per quanto riguarda i tempi computazionali;
- il metodo Hybrid è l'unico metodo per cui si sono registrate significative differenze tra le prove sperimentali eseguite con filtraggio e quelle eseguite senza filtraggio;
- tra i due metodi di riduzione on-line considerati, il metodo Hybrid è quello che fornisce le prestazioni migliori in termini di tempi computazionali;
- il metodo Fusion è il metodo che fornisce le prestazioni migliori: è molto veloce e, a parità di qualità delle mappe finali, riduce un numero maggiore di segmenti rispetto a tutti gli altri metodi. Tuttavia, funziona bene solo per piccoli valori del parametro di distanza spaziale in ingresso;
- il metodo Mean Shift Clustering è in grado di fornire buone prestazioni solo se applicato a data set che corrispondono a mappe poco rumorose;
- il metodo Mean Shift Clustering produce mappe caratterizzate da una qualità che decresce in maniera notevole all'aumentare del valore del parametro di distanza spaziale in ingresso;

- il metodo Mean Shift Clustering, dal punto di vista del tempo computazionale, è quello che fornisce le prestazioni peggiori tra i metodi off-line.

Capitolo 7

Direzioni future di ricerca e conclusioni

Le mappe basate su segmenti sono un modo efficiente per rappresentare gli ambienti indoor in cui i robot mobili devono operare. Rispetto alle mappe basate su griglia di occupazione, le mappe basate su segmenti permettono un utilizzo efficiente delle risorse di memoria e, generalmente, operazioni più semplici per i processi di gestione ed aggiornamento della mappa. Scarsa attenzione è stata finora prestata ai metodi che permettono di ridurre il numero dei segmenti ridondanti, cioè che rappresentano gli stessi oggetti nell'ambiente. Senza l'applicazione di questi metodi, molti potenziali vantaggi delle mappe a segmenti potrebbero essere limitati. Spesso questi metodi consistono in soluzioni ad hoc, sviluppate per la particolare tipologia di sensore utilizzato.

In questo lavoro di tesi abbiamo confrontato metodi differenti per la riduzione del numero di segmenti ridondanti utilizzati nella rappresentazione di ambienti indoor. I risultati ottenuti possono essere utilizzati come linee guida da seguire per la definizione di un approccio generale alla risoluzione di tale problema.

Il nostro lavoro è stato così strutturato. Per prima cosa abbiamo presentato una panoramica dei metodi esistenti in letteratura classificandoli per: classe di riduzione, processo di registrazione delle scansioni, rappresentazione dei segmenti utilizzata, indipendenza o meno dal metodo di estrazione dei segmenti utilizzato, indipendenza o meno dall'insieme di dati (ad esempio, punti) acquisiti dal sensore dopo aver estratto i segmenti. Tra questi ne sono stati scelti cinque per le successive fasi di analisi, implementazione e valutazione sperimentale. La scelta dei metodi è stata fatta cercando di se-

lezionare un metodo rappresentativo per ogni classe di riduzione, scegliendo inoltre metodi le cui procedure di riduzione fossero completamente basate sui segmenti e indipendenti dal metodo di estrazione dei segmenti utilizzato. La fase di analisi ha messo in luce, da un punto di vista teorico, potenzialità e limiti di ogni metodo selezionato: tali supposizioni sono state poi in gran parte verificate nelle successive fasi di implementazione e valutazione sperimentale. I metodi sono stati infatti implementati e applicati ad alcuni data set provenienti da acquisizioni in ambienti reali e pubblicamente disponibili in rete. Le prove sperimentali sono state quindi effettuate su diverse mappe basate su segmenti, derivate da questi insiemi di dati mediante l'applicazione di diversi metodi di estrazione dei segmenti e di allineamento delle scansioni. L'eterogeneità delle mappe prese in considerazione ha permesso di confrontare i metodi applicandoli in differenti condizioni, evidenziandone così vantaggi e svantaggi. Il lavoro svolto estende quello descritto in [52]. I risultati ottenuti ci mostrano come il confronto tra metodi non è un compito banale. Diversi sono gli aspetti che devono essere tenuti in considerazione per avere un paragone quanto più attendibile. La percentuale di segmenti ridotti certamente non basta e i risultati lo confermano. Spesso, infatti, metodi che hanno un'alta percentuale di riduzione sono caratterizzati anche da una bassa qualità della mappa risultante oppure da tempi computazionali molto elevati. Inoltre, un altro aspetto fondamentale, non considerato in [52], è che il testing dei metodi è stato fatto su data set eterogenei, allineati utilizzando metodi differenti, al fine di ottenere risultati non influenzati dalla particolare tipologia di mappa o dal particolare metodo di allineamento utilizzato.

Le ricerche future potranno concentrarsi sul testing dei metodi anche in ambienti outdoor strutturati (per esempio, urbani), oppure su dati ottenuti attraverso l'utilizzo di sensori differenti. Infatti, i data set utilizzati per le prove sperimentali provengono tutti da scansioni ottenute attraverso l'utilizzo di telemetri laser. Un ulteriore indirizzo di ricerca riguarda l'utilizzo congiunto di più metodi di riduzione, nel tentativo di combinarne i vantaggi. Un esempio potrebbe essere l'utilizzo congiunto dei metodi Hybrid e Fusion. Ad esempio, Hybrid potrebbe essere utilizzato come metodo di riferimento, mentre Fusion potrebbe essere richiamato ogni n scansioni, oppure una volta sola al termine dell'acquisizione di tutte le scansioni e andrebbe a ridurre insiemi di segmenti la cui corrispondenza non è stata rilevata da Hybrid. In questo modo, si potrebbe ottenere un nuovo metodo on-line caratterizzato da una percentuale maggiore di segmenti ridotti, da una buona qualità delle mappe generate e da tempi computazionali ragionevoli per un esecuzione

on-line.

Bibliografia

[1] S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*, pp. 1-35. Morgan Kaufmann, 2003.

[2] J. Borenstein, B. Everett, e L. Feng. Navigating Mobile Robots: Systems and Techniques. Publisher: A. K. Peters, Ltd., Wellesley, 1996.

[3] D. Kortenkamp, R.P. Bonasso e R. Murphy. AI-based Mobile Robots: Case studies of successful robot systems. Cambridge, MA, 1998. MIT Press.

[4] Hector H. Gonzalez e Jean-Claude Latombe. Navigation Strategies for Exploring Indoor Environments. *INT J ROBOT RES*, vol. 21, no. 10-11, 2002, pp. 829-848.

[5] T.S. Levitt, B.J. Kuipers. Navigation and mapping in large scale space. In *AI Magazine*, vol. 9, 1998, pp. 25-43.

[6] U.R. Zimmer. Robust world-modeling and navigation in a real world. In *Neurocomputing*, vol. 13, 1996, pp. 2-4.

[7] J. Buhmann, W. Burgard, A.B. Cremers, D. Fox, T. Hofmann, F. Schneider, J. Strikos, and S. Thrun. The mobile robot Rhino. In *AI Magazine*, 16(1), 1995, pp. 31-38.

[8] Wolfgang Wagner. Simulation of Obstacle Avoiding Autonomous Robots Controlled by Neuro-Evolution. 1999.

[9] Amalia F. Foka e Panos E. Trahanias. Predictive Control of Robot Velocity to Avoid Obstacles in Dynamic Environments. In *Conference on Intelligent Robots and Systems*, October 2003, pp. 370-375.

[10] J. Borenstein and Y. Koren. The vector field histogram - fast obstacle avoidance for mobile robots. *IEEE Transaction of Robotics and Automation*, June 1991, 7(3):278-288.

[11] T. Weimouth, D. Kortnekamp. Topological mapping for mobile robots using a combination of sonar and vision sensing. In *Proceedings of the twelfth national conference on artificial intelligence*, 1994, pp. 979-984.

[12] S. Engelson e D. McDermott. Error correction in mobile robot map learning. In *Proc. ICRA*, May 1992, pp. 2555-2560.

[13] Kristopher R. Beevers. Topological mapping and map merging with sensing-limited robots. In *Master's thesis, Rensselaer Polytechnic Institute*, Troy, NY, April 2004.

[14] M.J. Mataric. A distributed model for mobile robot environment-learning and navigation. In *Master's thesis, MIT*, 1990.

[15] Javier Gonzalez, Anibal Ollero e Antonio Reina. Map Building for a Mobile Robot equipped with a 2D Laser Rangefinder. In *Proc. ICRA*, 1994, pp. 1904-1909.

[16] Rolf Lakaemper, Xinyu Sun, Diedrich Wolter e Longin Jan Latecki. Building Polygonal Maps from Laser Range Data. In *Proc. CogRob*, 2004, pp.56-62.

[17] D. Austin e B. McCarragher. Geometric constraint identification and mapping for mobile robots. *ROBOT AUTON SYST*, vol. 35, 2001, pp. 59-76.

[18] P.E. Debevec, C.J. Taylor e J. Malik. Modeling and rendering architecture from photographs. In *Proc. of the 23rd International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1996, pp. 11-20.

[19] E. Zalama, G. Candela, J. Gomez, e S. Thrun. Concurrent mapping and localization for mobile robots with segmented local maps. In *Proc. IROS*, 2002, pp. 546-551.

[20] Emma Brunskill e Nicholas Roy. SLAM using Incremental Probabilistic PCA and Dimensionality Reductionx. In *Proc. ICRA*, 2005, pp. 342-347.

[21] Jason Meltzer, Rakesh Gupta, Ming-Hsuan Yang e Stefano Soatto. Simultaneous Localization and Mapping using Multiple View Feature Descriptors. In *Proc. IROS*, vol. 2, 2004, pp. 1550-1555.

[22] J. Vandorpe, H. Van Brussel, H. Xu. Exact Dynamic Map Building for a Mobile Robot using Geometrical Primitives Produced by a 2D Range Finder. In *Proc. ICRA*, April 1996, pp. 901-908.

[23] A. Elfes. Sonar-based real-world mapping and navigation. In *IEEE Journal of Robotics and Automation*, RA-3(3), 1987, pp. 249-265.

[24] A. Elfes. Occupancy Grids: A Probabilistic Framework for Robot Perception and Navigation. *PhD thesis, Carnegie Mellon University*, 1989.

[25] W. Burgard, A.B. Cremers, D. Fox, D. Hahnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an a interactive museum tour-guide robot. *Artificial Intelligence archive*, vol. 114, 1999, pp. 3-55.

[26] H. P. Moravec. Sensor fusion in certainty grids for mobile robots. In *AI Magazine*, 9(2), 1988, pp. 61-74.

[27] K.S. Chong e L. Kleeman. Sonar based map building for a mobile robot. In *Proc. ICRA*, vol. 2, 1997, pp. 1700-1705.

[28] P.J. McKerrow. Echolocation from range to outline segments. *Journal of Robotics and Autonomus System*, vol. 11 no. 4, 1993, pp 205-211.

[29] M. Lopez Sanchez, F. Esteva, R. Lopez de Mantaras e C. Sierra. Map Generation by Cooperative Low-Cost Robots in Structured Unknown Environments. *Journal of Robotics and Autonomus System*, vol. 5, 1998, pp. 97-111.

[30] Francesco Amigoni, Simone Gasparini e Maria Gini. Building Segment-Based Maps Without Pose Information. In *P IEEE*, vol 94, no. 7, pp. 1340-1359.

[31] Francesco Amigoni, Giulio Fontana e Fabio Garigiola. A Method for Building Small-Size Segment-Based Maps. In *Proc DARS*, 2006, pp. 11-20.

[32] Li Zhang e Bijoy K. Ghosh. Line Segment Based Map Building and Localization Using 2D Laser Rangefinder. In *Proc. ICRA*, vol. 3, 2000, pp. 2538-2543.

[33] Y. L. Ip, A. B. Rad, K. M. Chow e Y. K. Wong. Segment-Based Map Building Using Enhanced Adaptive Fuzzy Clustering Algorithm for Mo-

bile Robot Applications. *Journal of Intelligent and Robotic Systems*, vol. 35, 2002, pp. 221-245.

[34] Roman Mazl e Libor Preucil. Building a 2D Environment Map from Laser Range-Finder Data. In *Proc. of the IEEE Intelligent Vehicles Symposium*, vol. 1, 2000, pp. 290-295.

[35] A. Howard and N. Roy. The robotics data set repository (radish). http://radish.sourceforge.net

[36] J.L. Crowley. Navigation for an intelligent mobile robot. *IEEE Journal of Robotics and Automation*, vol. 1, 1985, pp. 31-41.

[37] S. Scheding, E.M. Nebot, M. Stevens, H. Durrant-Whyte. Experiments in autonomous under-ground guidance. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, Albuquerque, NM, 1997, pp. 1898-1903.

[38] E.G. Arajuo e R.A. Grupen. Feature detection and identification using a sonar array. In *Proc. ICRA*, vol. 2, 1998, pp. 1584-1589.

[39] J.A. Janet, S.M. Scoggins, M.W. White, J.C. Sutton, E. Grant, W.E. Snyder. Self-organising geometric certainty maps: a compact and multi-functional approach to place recognition and motion planning. In *Proc. ICRA*, Albuquerque, NM, vol. 4, 1997, pp. 3421-3426.

[40] V. Nguyen, S. Gächter, A. Martinelli, N. Tomatis, and R. Siegwart. A comparison of line extraction algorithms using 2D range data for indoor mobile robotics. *AUTON ROBOT*, vol. 23, no. 2, pp. 97-111, 2007.

[41] F. Lu and E. Milios. Robot pose estimation in unknown environment by matching 2D range scans. *J INTELL ROBOT SYST*, 18(3):249-275, 1998.

[42] Jan Elseberg, Ross T. Creed, and Rolf Lakaemper. A Line Segment Based System for 2D Global Mapping. In *Proc. ICRA*, 2010.

[43] Wei-Jen Kuo, Shih-Huan Tseng, Jia-Yuan Yu, and Li-Chen Fu. A Hybrid Approach to RBPF Based SLAM with Grid Mapping Enhanced by Line Matching. In *Proc. IROS*, 2009.

[44] Samuel T. Pfister, Stergios I. Roumeliotis, and Joel W. Burdick. Weighted Line Fitting Algorithms for Mobile Robot Map Building and Efficient Data Representation. In *Proc. ICRA*, pp. 1304-1311, 2003.

[45] Rolf Lakaemper. Simultaneous Multi-Line-Segment Merging for Robot Mapping Using Mean Shift Clustering. In *Proc. IROS*, 2009.

[46] Marco Mattavelli, Vincent Noel e Edoardo Amaldi. Fast Line Detection Algorithms Based on Combinatorial Optimization. *Computer Science*, vol. 2059, 2001, pp. 410-419.

[47] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *AUTON ROBOT*, vol. 4, no. 4, 1997, pp. 333-349.

[48] F. Amigoni and M. Vailati. A Method for Reducing Redundant Line Segments in Maps. In *Proc. ECMR*, Croatia, September 23-25 2009, pp. 61-66.

[49] M. Vailati. Sviluppo di un metodo per la riduzione dei segmenti ridondanti in mappe costruite da robot mobili. *Master's thesis, Politecnico di Milano*, 2008.

[50] The Rawseeds project. http://www.rawseeds.org/

[51] Y. Cheng. Mean shift, mode seeking, and clustering. *Pattern Analysis and Machine Intelligence*, 17(8):790-799, August 1995.

[52] F. Amigoni and S. Gasparini. Analysis of Methods for Reducing Line Segments in Maps: Towards a General Approach. In *Proc. IROS*, 2008, pp. 2896-2901.

[53] S. Scheding, E.M. Nebot, M. Stevens, H. Durrant-Whyte. Experiments in autonomous underground guidance. In *Proc. ICRA*, Albuquerque, NM, 1997, pp. 1898-1903.

Appendice A

Prove sperimentali

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		658	386	41,34%	0h 0m 29s	91,59%
10	100	X	434	245	43,55%	0h 0m 9s	90,94%
10	200		658	340	48,33%	0h 0m 24s	88,25%
10	200	X	434	215	50,46%	0h 0m 10s	87,45%
10	300		658	317	51,82%	0h 0m 17s	85,65%
10	300	X	434	203	53,23%	0h 0m 7s	85,17%
10	400		658	294	55,32%	0h 0m 19s	82,82%
10	400	X	434	190	56,22%	0h 0m 8s	82,49%
20	100		658	336	48,94%	0h 0m 18s	87,84%
20	100	X	434	214	50,69%	0h 0m 8s	86,46%
20	200		658	276	58,05%	0h 0m 14s	82,84%
20	200	X	434	179	58,76%	0h 0m 6s	82,05%
20	300		658	254	61,40%	0h 0m 16s	79,19%
20	300	X	434	168	61,29%	0h 0m 5s	77,64%
20	400		658	229	65,20%	0h 0m 13s	75,42%
20	400	X	434	153	64,75%	0h 0m 6s	75,10%

Figura A.1: Risultati prove sperimentali metodo Basic off-line sul data set Polimi.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		658	387	41,19%	0h 3m 1s	91,62%
10	100	X	434	247	43,09%	0h 1m 7s	91,14%
10	200		658	338	48,63%	0h 2m 21s	88,14%
10	200	X	434	216	50,23%	0h 0m 49s	87,57%
10	300		658	316	51,98%	0h 1m 58s	85,62%
10	300	X	434	202	53,46%	0h 0m 43s	85,00%
10	400		658	295	55,17%	0h 1m 40s	83,08%
10	400	X	434	190	56,22%	0h 0m 41s	82,83%
20	100		658	339	48,48%	0h 2m 11s	88,20%
20	100	X	434	214	50,69%	0h 0m 49s	86,50%
20	200		658	273	58,51%	0h 1m 29s	83,15%
20	200	X	434	177	59,22%	0h 0m 36s	81,81%
20	300		658	252	61,70%	0h 1m 21s	78,85%
20	300	X	434	167	61,52%	0h 0m 33s	77,35%
20	400		658	232	64,74%	0h 1m 2s	76,05%
20	400	X	434	155	64,29%	0h 0m 31s	74,81%

Figura A.2: Risultati prove sperimentali metodo Basic on-line sul data set Polimi.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		658	327	50,30%	0h 0m 10s	83,65%
10	100	X	434	309	28,80%	0h 0m 6s	95,57%
10	200		658	277	57,90%	0h 0m 9s	80,64%
10	200	X	434	259	40,32%	0h 0m 5s	91,65%
10	300		658	252	61,70%	0h 0m 8s	78,55%
10	300	X	434	236	45,62%	0h 0m 5s	89,14%
10	400		658	236	64,13%	0h 0m 8s	76,48%
10	400	X	434	219	49,54%	0h 0m 4s	86,03%
20	100		658	325	50,61%	0h 0m 10s	83,33%
20	100	X	434	307	29,26%	0h 0m 6s	95,16%
20	200		658	273	58,51%	0h 0m 9s	80,11%
20	200	X	434	256	41,01%	0h 0m 5s	91,18%
20	300		658	245	62,77%	0h 0m 8s	76,59%
20	300	X	434	229	47,24%	0h 0m 5s	87,12%
20	400		658	227	65,50%	0h 0m 7s	74,18%
20	400	X	434	211	51,38%	0h 0m 4s	83,25%

Figura A.3: Risultati prove sperimentali metodo Hybrid sul data set Polimi.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
/	20		658	394	40,12%	0h 0m 10s	92,05%
/	20	X	434	275	36,64%	0h 0m 4s	93,88%
/	40		658	253	61,55%	0h 0m 9s	80,32%
/	40	X	434	200	53,92%	0h 0m 4s	84,21%
/	60		658	192	70,82%	0h 0m 8s	71,95%
/	60	X	434	158	63,59%	0h 0m 4s	75,51%
/	80		658	168	74,47%	0h 0m 8s	67,75%
/	80	X	434	136	68,66%	0h 0m 3s	69,70%
/	100		658	152	76,90%	0h 0m 8s	65,01%
/	100	X	434	123	71,66%	0h 0m 3s	65,95%

Figura A.4: Risultati prove sperimentali metodo Fusion sul data set Polimi.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
/	100		658	418	36,47%	0h 0m 26s	90,74%
/	100	X	434	277	36,18%	0h 0m 38s	89,78%
/	200		658	304	53,80%	0h 0m 38s	81,13%
/	200	X	434	214	50,69%	0h 0m 15s	80,06%
/	300		658	214	67,48%	0h 0m 20s	67,49%
/	300	X	434	169	61,06%	0h 0m 15s	64,27%
/	400		658	144	78,12%	0h 0m 14s	51,98%
/	400	X	434	113	73,96%	0h 0m 9s	50,93%

Figura A.5: Risultati prove sperimentali metodo Mean Shift sul data set Polimi.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		1507	863	42,73%	0h 2m 26s	92,69%
10	100	X	1117	625	44,05%	0h 1m 2s	91,89%
10	200		1507	771	48,84%	0h 1m 38s	89,41%
10	200	X	1117	565	49,42%	0h 0m 55s	88,83%
10	300		1507	706	53,15%	0h 1m 30s	85,62%
10	300	X	1117	523	53,18%	0h 0m 49s	85,08%
10	400		1507	651	56,80%	0h 1m 37s	82,30%
10	400	X	1117	487	56,40%	0h 0m 56s	81,56%
20	100		1507	724	51,96%	0h 1m 52s	88,34%
20	100	X	1117	541	51,57%	0h 0m 50s	87,34%
20	200		1507	615	59,19%	0h 1m 12s	83,78%
20	200	X	1117	469	58,01%	0h 0m 53s	82,45%
20	300		1507	550	63,50%	0h 1m 18s	79,66%
20	300	X	1117	429	61,59%	0h 0m 37s	80,00%
20	400		1507	495	67,15%	0h 1m 7s	75,35%
20	400	X	1117	381	65,89%	0h 0m 39s	73,58%

Figura A.6: Risultati prove sperimentali metodo Basic off-line sul data set Stanford.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		1507	859	43,00%	0h 22m 13s	92,53%
10	100	X	1117	623	44,23%	0h 10m 45s	91,77%
10	200		1507	768	49,04%	0h 20m 3s	89,30%
10	200	X	1117	559	49,96%	0h 9m 41s	88,43%
10	300		1507	705	53,22%	0h 16m 9s	85,87%
10	300	X	1117	521	53,36%	0h 8m 10s	85,55%
10	400		1507	653	56,67%	0h 14m 0s	82,62%
10	400	X	1117	483	56,76%	0h 7m 35s	81,07%
20	100		1507	725	51,89%	0h 15m 42s	88,56%
20	100	X	1117	544	51,30%	0h 8m 41s	87,61%
20	200		1507	617	59,06%	0h 12m 32s	83,71%
20	200	X	1117	470	57,92%	0h 6m 31s	82,81%
20	300		1507	540	64,17%	0h 10m 4s	78,51%
20	300	X	1117	420	62,40%	0h 5m 28s	78,66%
20	400		1507	493	67,29%	0h 8m 7s	75,14%
20	400	X	1117	379	66,07%	0h 4m 36s	73,66%

Figura A.7: Risultati prove sperimentali metodo Basic on-line sul data set Stanford.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		1507	918	39,08%	0h 1m 2s	90,00%
10	100	X	1117	910	18,53%	0h 0m 47s	98,15%
10	200		1507	811	46,18%	0h 0m 54s	88,03%
10	200	X	1117	807	27,75%	0h 0m 41s	95,65%
10	300		1507	752	50,10%	0h 0m 51s	85,98%
10	300	X	1117	749	32,95%	0h 0m 38s	93,46%
10	400		1507	696	53,82%	0h 0m 47s	83,82%
10	400	X	1117	695	37,78%	0h 0m 36s	91,11%
20	100		1507	910	39,62%	0h 1m 2s	89,51%
20	100	X	1117	903	19,16%	0h 0m 47s	97,82%
20	200		1507	801	46,85%	0h 0m 54s	87,00%
20	200	X	1117	797	28,65%	0h 0m 40s	94,93%
20	300		1507	740	50,90%	0h 0m 49s	84,58%
20	300	X	1117	737	34,02%	0h 0m 37s	92,31%
20	400		1507	677	55,08%	0h 0m 45s	81,93%
20	400	X	1117	678	39,30%	0h 0m 35s	89,66%

Figura A.8: Risultati prove sperimentali metodo Hybrid sul data set Stanford.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
/	20		1507	822	45,45%	0h 0m 50s	92,34%
/	20	X	1117	642	42,52%	0h 0m 28s	93,53%
/	40		1507	578	61,65%	0h 0m 47s	83,49%
/	40	X	1117	502	55,06%	0h 0m 27s	86,28%
/	60		1507	464	69,21%	0h 0m 45s	73,79%
/	60	X	1117	405	63,74%	0h 0m 25s	75,24%
/	80		1507	380	74,78%	0h 0m 44s	65,96%
/	80	X	1117	339	69,65%	0h 0m 24s	67,65%
/	100		1507	326	78,37%	0h 0m 44s	60,81%
/	100	X	1117	285	74,49%	0h 0m 24s	61,90%

Figura A.9: Risultati prove sperimentali metodo Fusion sul data set Stanford.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
/	100		1507	858	43,07%	0h 1m 8s	91,85%
/	100	X	1117	633	43,33%	0h 0m 56s	91,35%
/	200		1507	695	53,88%	0h 0m 22s	83,93%
/	200	X	1117	539	51,75%	0h 0m 17s	84,81%
/	300		1507	497	67,02%	0h 0m 29s	73,50%
/	300	X	1117	438	60,79%	0h 0m 10s	76,04%
/	400		1507	399	73,52%	0h 0m 24s	63,71%
/	400	X	1117	372	66,70%	0h 0m 8s	66,78%

Figura A.10: Risultati prove sperimentali metodo Mean Shift sul data set Stanford.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		2220	849	61,76%	0h 2m 55s	87,39%
10	100	X	1320	429	67,50%	0h 0m 39s	82,32%
10	200		2220	652	70,63%	0h 1m 57s	82,23%
10	200	X	1320	334	74,70%	0h 0m 33s	74,95%
10	300		2220	559	74,82%	0h 1m 35s	79,17%
10	300	X	1320	295	77,65%	0h 0m 23s	70,95%
10	400		2220	487	78,06%	0h 1m 19s	75,22%
10	400	X	1320	266	79,85%	0h 0m 25s	67,22%
20	100		2220	678	69,46%	0h 2m 11s	84,37%
20	100	X	1320	353	73,26%	0h 0m 31s	78,52%
20	200		2220	471	78,78%	0h 1m 17s	76,68%
20	200	X	1320	268	79,70%	0h 0m 25s	68,62%
20	300		2220	387	82,57%	0h 1m 0s	70,28%
20	300	X	1320	219	83,41%	0h 0m 23s	64,32%
20	400		2220	333	85,00%	0h 0m 50s	64,61%
20	400	X	1320	195	85,23%	0h 0m 17s	61,42%

Figura A.11: Risultati prove sperimentali metodo Basic off-line sul data set Bicocca.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		2220	860	61,26%	1h 55m 13s	88,36%
10	100	X	1320	424	67,88%	0h 25m 39s	82,63%
10	200		2220	647	70,86%	1h 6m 50s	81,93%
10	200	X	1320	332	74,85%	0h 15m 48s	73,88%
10	300		2220	561	74,73%	0h 50m 11s	78,29%
10	300	X	1320	294	77,73%	0h 12m 27s	70,65%
10	400		2220	484	78,20%	0h 40m 14s	74,62%
10	400	X	1320	268	79,70%	0h 10m 37s	70,04%
20	100		2220	674	69,64%	1h 14m 11s	83,69%
20	100	X	1320	354	73,18%	0h 18m 37s	79,17%
20	200		2220	467	78,96%	0h 35m 50s	75,75%
20	200	X	1320	269	79,62%	0h 10m 17s	67,91%
20	300		2220	396	82,16%	0h 25m 12s	72,34%
20	300	X	1320	217	83,56%	0h 7m 6s	62,28%
20	400		2220	343	84,55%	0h 19m 1s	65,77%
20	400	X	1320	199	84,92%	0h 6m 5s	61,10%

Figura A.12: Risultati prove sperimentali metodo Basic on-line sul data set Bicocca.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		2220	732	67,03%	0h 1m 16s	81,36%
10	100	X	1320	726	45,00%	0h 0m 46s	93,19%
10	200		2220	559	74,82%	0h 0m 59s	77,47%
10	200	X	1320	561	57,50%	0h 0m 36s	88,03%
10	300		2220	465	79,05%	0h 0m 50s	74,10%
10	300	X	1320	466	64,70%	0h 0m 30s	84,03%
10	400		2220	391	82,39%	0h 0m 43s	72,07%
10	400	X	1320	394	70,15%	0h 0m 26s	81,00%
20	100		2220	729	67,16%	0h 1m 16s	81,14%
20	100	X	1320	723	45,23%	0h 0m 46s	93,03%
20	200		2220	551	75,18%	0h 0m 59s	76,97%
20	200	X	1320	551	58,26%	0h 0m 35s	87,44%
20	300		2220	452	79,64%	0h 0m 50s	73,37%
20	300	X	1320	451	65,83%	0h 0m 30s	83,08%
20	400		2220	374	83,15%	0h 0m 41s	70,70%
20	400	X	1320	377	71,44%	0h 0m 25s	79,20%

Figura A.13: Risultati prove sperimentali metodo Hybrid sul data set Bicocca.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
/	20		2220	1011	54,46%	0h 1m 43s	91,63%
/	20	X	1320	599	54,62%	0h 0m 37s	92,48%
/	40		2220	693	68,78%	0h 1m 36s	84,84%
/	40	X	1320	382	71,06%	0h 0m 34s	83,67%
/	60		2220	534	75,95%	0h 1m 33s	80,12%
/	60	X	1320	303	77,05%	0h 0m 33s	78,34%
/	80		2220	441	80,14%	0h 1m 32s	75,10%
/	80	X	1320	235	82,20%	0h 0m 34s	71,46%
/	100		2220	400	81,98%	0h 1m 32s	72,42%
/	100	X	1320	216	83,64%	0h 0m 34s	68,39%

Figura A.14: Risultati prove sperimentali metodo Fusion sul data set Bicocca.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
/	100		2220	766	65,50%	0h 13m 39s	80,51%
/	100	X	1320	383	70,98%	0h 9m 20s	77,14%
/	200		2220	427	80,77%	0h 3m 39s	63,12%
/	200	X	1320	252	80,91%	0h 2m 2s	49,88%
/	300		2220	254	88,56%	0h 5m 3s	48,35%
/	300	X	1320	189	85,68%	0h 1m 2s	42,31%
/	400		2220	172	92,25%	0h 3m 34s	29,39%
/	400	X	1320	149	88,71%	0h 0m 55s	28,49%

Figura A.15: Risultati prove sperimentali metodo Mean Shift sul data set Bicocca.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		3604	714	80,19%	0h 3m 27s	89,59%
10	100	X	2206	480	78,24%	0h 1m 29s	85,77%
10	200		3604	454	87,40%	0h 1m 54s	86,14%
10	200	X	2206	298	86,49%	0h 0m 48s	81,75%
10	300		3604	371	89,71%	0h 1m 31s	82,97%
10	300	X	2206	220	90,03%	0h 0m 34s	77,80%
10	400		3604	332	90,79%	0h 1m 8s	81,87%
10	400	X	2206	199	90,98%	0h 0m 27s	77,44%
20	100		3604	611	83,05%	0h 2m 53s	87,07%
20	100	X	2206	454	79,42%	0h 1m 22s	85,41%
20	200		3604	345	90,43%	0h 1m 24s	81,15%
20	200	X	2206	264	88,03%	0h 0m 42s	79,70%
20	300		3604	277	92,31%	0h 1m 3s	77,99%
20	300	X	2206	187	91,52%	0h 0m 28s	75,21%
20	400		3604	241	93,31%	0h 0m 46s	75,13%
20	400	X	2206	169	92,34%	0h 0m 23s	74,84%

Figura A.16: Risultati prove sperimentali metodo Basic off-line sul data set USC.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		3604	655	81,83%	2h 43m 39s	89,31%
10	100	X	2206	445	79,83%	1h 10m 35s	86,26%
10	200		3604	401	88,87%	0h 56m 42s	84,53%
10	200	X	2206	269	87,81%	0h 25m 51s	80,65%
10	300		3604	315	91,26%	0h 35m 2s	80,46%
10	300	X	2206	203	90,80%	0h 14m 58s	77,99%
10	400		3604	286	92,06%	0h 28m 14s	79,56%
10	400	X	2206	178	91,93%	0h 11m 27s	67,24%
20	100		3604	552	84,68%	1h 57m 27s	86,46%
20	100	X	2206	422	80,87%	1h 4m 12s	83,63%
20	200		3604	302	91,62%	0h 32m 41s	80,65%
20	200	X	2206	238	89,21%	0h 20m 57s	78,92%
20	300		3604	236	93,45%	0h 19m 37s	75,39%
20	300	X	2206	176	92,02%	0h 11m 27s	76,05%
20	400		3604	209	94,20%	0h 14m 30s	73,63%
20	400	X	2206	151	93,16%	0h 8m 29s	65,05%

Figura A.17: Risultati prove sperimentali metodo Basic on-line sul data set USC.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
10	100		3604	720	80,02%	0h 2m 14s	88,62%
10	100	X	2206	726	67,09%	0h 1m 20s	91,89%
10	200		3604	473	86,88%	0h 1m 28s	83,84%
10	200	X	2206	471	78,65%	0h 0m 54s	85,57%
10	300		3604	368	89,79%	0h 1m 9s	81,12%
10	300	X	2206	369	83,27%	0h 0m 39s	82,17%
10	400		3604	304	91,56%	0h 0m 55s	78,00%
10	400	X	2206	307	86,08%	0h 0m 33s	79,41%
20	100		3604	718	80,08%	0h 2m 9s	88,50%
20	100	X	2206	721	67,32%	0h 1m 17s	91,77%
20	200		3604	467	87,04%	0h 1m 23s	83,48%
20	200	X	2206	464	78,97%	0h 0m 49s	85,14%
20	300		3604	361	89,98%	0h 1m 5s	80,67%
20	300	X	2206	362	83,59%	0h 0m 38s	81,82%
20	400		3604	296	91,79%	0h 0m 54s	77,47%
20	400	X	2206	298	86,49%	0h 0m 32s	79,04%

Figura A.18: Risultati prove sperimentali metodo Hybrid sul data set USC.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
/	20		3604	784	78,25%	0h 4m 6s	92,56%
/	20	X	2206	511	76,84%	0h 1m 34s	92,00%
/	40		3604	435	87,93%	0h 4m 0s	87,42%
/	40	X	2206	307	86,08%	0h 1m 31s	86,27%
/	60		3604	309	91,43%	0h 4m 0s	83,25%
/	60	X	2206	204	90,75%	0h 1m 30s	80,53%
/	80		3604	246	93,17%	0h 3m 51s	76,68%
/	80	X	2206	163	92,61%	0h 1m 29s	68,49%
/	100		3604	196	94,56%	0h 3m 55s	72,71%
/	100	X	2206	138	93,74%	0h 1m 27s	65,13%

Figura A.19: Risultati prove sperimentali metodo Fusion sul data set USC.

Distanza angolare (°)	Distanza spaziale (mm)	Filtro	#seg originale	#seg fusione	Riduzione (%)	Tempo (h m s)	Qualità (%)
/	100		3604	558	84,52%	1h 48m 39s	77,02%
/	100	X	2206	390	82,32%	2h 15m 18s	69,81%
/	200		3604	309	91,43%	1h 0m 58s	55,10%
/	200	X	2206	225	89,80%	1h 1m 25s	45,69%
/	300		3604	214	94,06%	0h 22m 36s	41,42%
/	300	X	2206	171	92,25%	0h 19m 16s	28,15%
/	400		3604	149	95,87%	0h 12m 16s	22,56%
/	400	X	2206	144	93,47%	0h 12m 42s	16,97%

Figura A.20: Risultati prove sperimentali metodo Mean Shift sul data set USC.