Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica

UN ALGORITMO DI GEOLOCALIZZAZIONE INDOORS BASATO SU MAGNETISMO

Relatore:
Chiar.mo Prof.
Luciano Bononi

Presentata da:
Manuel Benedetti

Correlatore:
Dott. Luca Bedogni

III Sessione
Anno Accademico 2015-2016

Alla mia famiglia

Abstract

Negli ultimi anni, il problema della localizzazione indoor tramite dispositivi mobili è stato oggetto di numerose ricerche volte a sperimentare l'impiego di tecnologie diverse e a identificare vantaggi e svantaggi di ciascuna di esse. Nell'ambito di questa tesi, si approfondisce una tecnica di fingerprinting basata su magnetismo che presenta l'importante vantaggio di non necessitare di alcuna infrastruttura di supporto. Viene sviluppata un'applicazione Android che implementa una tecnica di pattern matching frutto di contributi originali, applicando al caso del magnetismo alcune soluzioni software elaborate nel corso di studi precedenti sulla localizzazione indoor relativi a tecnologie differenti. Particolare attenzione è rivolta all'analisi dei risultati ottenuti dal testing del sistema su diversi scenari indoor e al confronto con sistemi esistenti che fanno uso di una simile tecnologia. Vengono discussi possibili sviluppi con l'intento di sfruttare appieno gli aspetti positivi individuati nel corso di questo studio.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

Il crescente interesse verso i sistemi di localizzazione per l'identificazione della posizione di persone all'interno di ambienti eterogenei ha dato origine a numerose ricerche. Fino a pochi anni fa le tecniche di geolocalizzazione conosciute erano per lo più limitate all'ambiente aperto, ma recentemente l'attenzione si è estesa a tecniche applicabili in luoghi chiusi. Grazie all'ampia diffusione di sensori di vario genere sui comuni smartphone, è oggi possibile sviluppare tecniche di localizzazione indoors che si avvalgono di molteplici soluzioni direttamente applicabili in contesti reali.

Nell'ambito di questa tesi, si approfondisce una tecnica di posizionamento indoors basata su magnetismo che sfrutta l'interferenza delle onde magnetiche causata dagli elementi strutturali di un edificio per localizzare l'utente all'interno di grandi spazi indoor come stazioni o centri commerciali.

Tralasciando questa breve introduzione, nei capitoli seguenti la trattazione sarà così articolata.

Nel **secondo capitolo** viene presentata una panoramica dello stato dell'arte sulle metodologie di localizzazione indoor adottate in sistemi reali e documentate nell'ambito di studi recenti, in particolare esponendo il funzionamento di tecniche di fingerprinting e di come queste dinamiche si possano

applicare per lo sviluppo di un sistema basato su rilevazioni del campo magnetico. Vengono illustrati gli aspetti innovativi di una tale tecnologia, prima tra tutti l'immediatezza di applicazione dovuta all'assenza di un'infrastruttura esterna di appoggio.

Nel **terzo capitolo** si tratta dello studio preliminare effettuato per comprendere che tipo di dati magnetici ci si può aspettare di ottenere nel contesto di grandi ambienti indoor, per comprendere come e in che misura essi possano essere poi sfruttati per l'elaborazione di un algoritmo che li utilizzi per la localizzazione. Vengono prese in esame la variabilità, la stabilità e la distribuzione dei valori del magnetismo osservati nel corso di molteplici rilevazioni.

Nel **quarto capitolo** vengono esposte le successive fasi di sviluppo di un algoritmo di posizionamento che fa uso dei dati magnetici. Viene realizzata un'applicazione per smartphone Android che implementa l'algoritmo. Il funzionamento dell'algoritmo viene esposto nel dettaglio. In particolare, vengono descritte le fasi di un'iterazione, che possono essere riassunte in: computazione della probabilità di posizionamento, applicazione di logica fuzzy pattern, scelta della posizione finale con *smoothing* del percorso calcolato, impiego di ottimizzazioni basate sui risultati delle iterazioni precedenti, controllo e potenziale riposizionamento. Infine, vengono svolti alcuni test automatizzati per ottenere una configurazione ottimale dei parametri cruciali dell'algoritmo.

Nel **quinto capitolo** vengono analizzati i risultati ottenuti dal testing dell'applicazione nell'ambito di diversi scenari indoor. Vengono prodotti valori di performance al fine di stimare l'errore medio di posizionamento risultante, discutendo le principali cause di imprecisione. Vengono messi a confronto i livelli di errore medio a partire da diversi livelli di granularità del fingerprint, evidenziando come sia possibile ottenere una precisione maggiore a costo di un mapping più meticoloso. Viene effettuato un confronto con altri sistemi esistenti, in particolare con i risultati di uno studio precedente sull'API svi-

luppata da IndoorAtlas; vengono presentate le differenze essenziali tra i due approcci.

Nel **sesto capitolo** vengono discussi possibili sviluppi futuri alla luce dei risultati ottenuti, con lo scopo di apprendere quali sono i punti di forza e i limiti principali di un sistema come quello sviluppato. Vengono esplorate differenti soluzioni implementative, in particolare inerentemente alla logica di pattern matching adottata, per poi proporre possibili integrazioni con altre tecnologie esistenti al fine di sopperire alle limitazioni riscontrate. Vengono infine analizzate le possibilità applicative in connessione con *IoT* e servizi cloud-based, realizzando a titolo esemplificativo una semplice interfaccia web che raccoglie i dati di posizione da diversi dispositivi che utilizzano il sistema proposto.

Capitolo 2

Localizzazione indoors

L'avvento e l'ormai sempre crescente diffusione degli smartphone hanno introdotto nuove possibilità nella vita di tutti i giorni. Uno smartphone possiede al suo interno sensori hardware di varia natura che offrono all'utente nuove ed inaspettate potenzialità. Una di esse è la possibilità di identificare la propria posizione sulla superficie terrestre in tempo reale, detta anche geolocalizzazione.

Nel corso degli anni, le principali aziende produttrici di software per dispositivi mobili hanno sviluppato servizi location-based e introdotto il supporto a funzioni di geolocalizzazione che hanno trovato diretto utilizzo in applicazioni basate sulla comunicazione tra diversi utenti, quali social network, geomarketing, realtà aumentata, personal tracking, crowdsensing. L'importanza assunta dal sempre crescente scambio di informazioni contestualizzate che ne è seguito è, in parte, motivo del successo dei dispositivi mobili al giorno d'oggi.

Le principali tecniche di localizzazione si basano sul segnale GPS e, in parte, sulla rete di telefonia cellulare. Essenzialmente si tratta però di tecniche di geolocalizzazione outdoor. La localizzazione indoor si differenzia sostanzialmente da quella outdoor per un semplice motivo: il sistema satellitare non è in grado di fornire informazioni utili in caso di attenuazione o perdita del segnale causata da tetti, pareti e altre strutture artificiali con

effetto mascherante. È dunque una sfida quanto mai attuale la ricerca di tecniche alternative da utilizzare efficacemente in un contesto indoor.

2.1 Tecniche di localizzazione indoors

Il problema della localizzazione indoor è stato oggetto di numerose ricerche volte a esplorare l'applicazione di tecniche alternative, prive dei difetti già menzionati che rendono impraticabile l'impiego del sistema satellitare. I sistemi già sviluppati per la localizzazione indoors in real time (indoor RTLS, *real-time locating systems*) sfruttano le diverse proprietà fisiche dell'ambiente circostante, tipicamente appartenenti alla natura dell'ambiente stesso. Lo studio approfondito di tecniche basate su proprietà rilevabili dai sensori di un comune smartphone ha portato alla luce dinamiche diverse a seconda della sorgente di dati utilizzata. Le problematiche di varia natura riscontrate dall'utilizzo di tecniche eterogenee hanno portato a comprendere che l'integrazione di vari sistemi di posizionamento basati su differenti principi fisici può migliorare considerevolmente l'accuratezza del risultato finale [1, 2]. È dunque materia di studi prettamente attuali l'analisi di approcci diversi al fine di ponderare i vantaggi e gli svantaggi di ciascuno di essi, per il loro impiego in un contesto di *sensor fusion* [3].

Gli approcci applicabili possono essere divisi in due macro-categorie; un primo basato su calcoli di trilaterazione o triangolazione a partire dall'entità del segnale ricevuto da molteplici punti di osservazione esterni le cui coordinate sono note a priori, con funzionamento simile al sistema GPS; e un secondo basato su fingerprinting. Nel primo caso, l'individuazione della posizione è realizzata tramite calcoli geometrici a partire dalle locazioni, precedentemente note, di installazioni fisse per la comunicazione con i dispositivi interessati. Nel secondo caso, invece, si hanno due fasi distinte: una prima fase di mappatura dell'ambiente, volta a creare un'impronta, chiamata appunto *fingerprint*; e una seconda che consiste nel confronto tra il fingerprint e le rilevazioni successive ai fini del posizionamento.

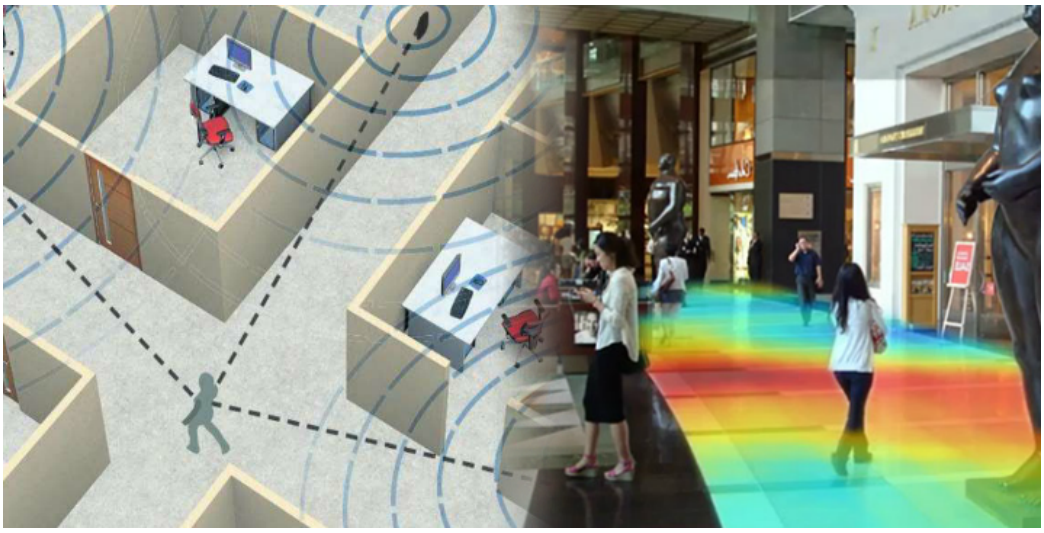


Figura 2.1: Tecniche di localizzazione a confronto: trilaterazione e fingerprinting. *(Fonti: Singapore Land Authority - Straits Times Graphics, IndoorAtlas)*

Si andranno ora a descrivere brevemente vantaggi e svantaggi delle due metodologie.

2.1.1 Tecnologia wireless e trilaterazione

Una tecnica del tutto simile a quella utilizzata dal sistema satellitare è applicabile in contesti indoor tramite sistemi wireless. Ciò presuppone l'esistenza di un'infrastruttura di appoggio costituita da trasmettitori installati in posizioni note a priori, il cui segnale viene captato dai dispositivi mobili; una successiva elaborazione a partire da segnali provenienti da più sorgenti permette di localizzare l'utente. Tuttavia, le stesse problematiche causate dall'interferenza di edifici, pareti e costruzioni vengono spesso riscontrate anche utilizzando onde radio da trasmettitori indoor a ricevitori indoor, anche se in misura minore. In genere, l'accuratezza di posizionamento può essere migliorata a costo di potenziare le infrastrutture wireless.

Grazie all'ormai sempre crescente diffusione degli smartphone e alla con-

seguente possibilità di sfruttare la tecnologia Wi-Fi e Bluetooth per la comunicazione in una varietà di situazioni, negli ultimi anni questa logica ha conosciuto applicazione diretta su scenari reali.

Wi-Fi RSSI

Conseguentemente alla sempre più ampia diffusione di reti Wi-Fi pubbliche, sono state sviluppate tecniche di localizzazione RSSI basate sulla misurazione della potenza del segnale da un dispositivo a diversi access point. I dati ottenuti vengono confrontati con un modello di propagazione per determinare la distanza tra il dispositivo e i diversi access point. Tecniche di trilaterazione possono poi essere impiegate per calcolare la posizione approssimata dell'utente in uno spazio bidimensionale [4, 5].

Bluetooth

Sebbene il Bluetooth sia più adatto a rilevare la prossimità di oggetti piuttosto che a fornire indicazioni di localizzazione, non mancano le sperimentazioni al riguardo. Alcuni studi indicano come sia possibile costruire un sistema di localizzazione basato unicamente su tecnologia Bluetooth utilizzabile efficacemente in un contesto indoors [7]; sono stati progettati sistemi basati su Bluetooth iBeacons con un errore finale pari a 5m [8]. In passato sistemi reali con Bluetooth sono stati realizzati da Apple applicando tecniche di micromapping [9].

2.1.2 Tecniche basate su fingerprinting

Le tecniche di fingerprinting sono basate su un approccio che si differenzia sostanzialmente da quelli precedentemente descritti. Si hanno sempre due fasi distinte. Una prima fase offline consiste nel registrare la potenza del segnale in diversi punti dello spazio, associati alle relative coordinate spaziali, memorizzando i valori ottenuti in un database. In un secondo momento, quando il dispositivo effettua la localizzazione in tempo reale, il valore cor-

rente del segnale viene confrontato con quelli salvati nel database e viene restituita la posizione stimata più probabile corrispondente a un tale valore.

Nella realizzazione di tecniche di fingerprinting su scenari reali è stato principalmente impiegato il segnale Wi-Fi come fonte di dati. Le rilevazioni sono sempre basate su RSSI, ma i risultati di posizionamento sono dati dal confronto tra la potenza del segnale rilevato in tempo reale e i valori precedentemente ottenuti da rilevazioni offline. Sistemi sperimentali hanno implementato un tale sistema con un margine di errore medio di 0.6m e un picco massimo di 1.3m [5, 6]. Oggi i servizi di posizionamento installati su dispositivi Android includono il Wi-Fi fingerprinting come metodo alternativo di localizzazione. Sfruttando dati ottenuti da utenti connessi in rete, vengono inviati periodicamente i dati di posizione ottenuti tramite GPS e cell-ID abbinando ad essi gli identificativi SSID dei vari access point di cui si riceve il segnale alle coordinate attuali, popolando così lo stesso database con il quale vengono poi confrontati i segnali ricevuti in tempo reale per ottenere la posizione stimata.

Magnetismo

Il fingerprinting tramite magnetismo presenta un importante vantaggio: non necessita di un'infrastruttura wireless con cui i device debbano scambiare dati in quanto si basa sulla sola rilevazione del campo magnetico durante la localizzazione.

Il campo magnetico terrestre è originato dalle correnti prodotte dai metalli liquidi che si trovano sul nucleo esterno della terra. Sulla superficie terrestre, il magnetismo può assumere valori diversi a seconda della distanza dal polo magnetico, con variazioni che vanno approssimativamente dai 25 ai 65 micro-Tesla (uT). In natura, molti animali, tra cui pipistrelli, api e diverse specie di uccelli, sono in grado di percepire l'entità del campo magnetico di un qualche luogo per orientarsi nell'ambiente circostante: è la magnetoricezione. Un sistema simile può essere ricreato artificialmente sfruttando il valore rilevato dai comuni smartphone.

Anche i meno recenti dispositivi in commercio dispongono ormai di un sensore capace di rilevare l'entità del magnetismo, il magnetometro. I dati risultano essenzialmente neutri e poco informativi in ambienti aperti; tuttavia all'interno di un edificio gli elementi strutturali quali travi, barre di acciaio e pareti in calcestruzzo creano variazioni localizzate del campo magnetico. Tali variazioni rimangono stabili nel tempo e costituiscono un'impronta magnetica dell'ambiente. Queste interferenze, che deviano, ad esempio, l'orientamento dell'ago di una bussola, possono anche essere sfruttate a vantaggio della localizzazione, in quanto a seconda del valore rilevato è possibile distinguere tra ambienti diversi dello stesso edificio in maniera efficace, come è stato dimostrato da diversi studi [10, 11, 12].

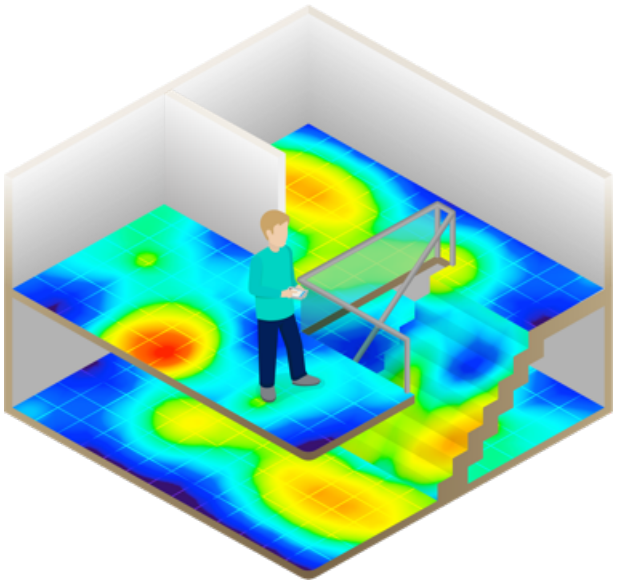


Figura 2.2: Mappa geomagnetica di un edificio, o *fingerprint*.

L'utente dovrà dunque effettuare manualmente una prima mappatura del campo magnetico su posizioni diverse di un edificio durante la fase offline, andando a ricostruire una vera e propria mappa magnetica del luogo. Successivamente, durante la localizzazione si ottiene nuovamente il valore del magnetismo in tempo reale e lo si confronta con quelli precedentemente ottenuti, per poi stabilire la propria posizione all'interno della mappa magnetica.

Se la realizzazione della prima fase non presenta particolari incognite, l'attuazione della seconda in un contesto reale dà luogo a problematiche che possono sfuggire a una prima analisi e che impongono l'impiego di logiche di pattern matching compatibili; in altre parole, non è immediato come determinare la posizione a partire dal semplice confronto dei dati magnetici.

2.2 Fingerprinting tramite magnetismo: pattern matching

Le sperimentazioni a riguardo sono piuttosto recenti. L'idea di base è quella di applicare tecniche già sperimentate con altri sistemi di localizzazione al caso del magnetismo e osservarne i risultati per studiare la possibilità di un loro impiego in un sistema reale.

L'obiettivo è quello di determinare la posizione dell'utente attraverso il confronto dei dati reali con una mappa magnetica ottenuta in una fase precedente. Il criterio con cui effettuare pattern matching non può basarsi unicamente sul confronto del dato ottenuto al momento con i valori presenti nella mappa magnetica, in quanto valori simili del magnetismo possono essere associati a diverse locazioni anche molto distanti tra loro. È necessario prendere in considerazione una storia recente dei valori misurati durante gli spostamenti dell'utente e cercare somiglianze tra la variazione del magnetismo misurata al momento e l'andamento dei valori nella mappa magnetica. Vi sono state diverse sperimentazioni al riguardo, che impiegano modelli matematici anche molto complessi.

2.2.1 Hidden Markov model

Una catena di Markov omogenea è un modello probabilistico che, a partire da un numero finito di stati, permette di definire quale sia la probabilità che a un certo istante ci si trovi in un certo stato, conoscendo unicamente la storia del sistema nell'istante precedente. In un modello *HMM* (*Hid-*

den Markov Model, modello di Markov nascosto), non è dato conoscere la sequenza di stati attraversati; l'aggettivo "nascosto" si riferisce appunto al fatto che gli stati non sono osservabili direttamente, mentre sono visibili le probabilità di transizione tra uno stato e un altro. Lo scopo è quello di individuare una probabile transizione attraverso i diversi stati osservando i valori di probabilità assunti da ogni stato per passi successivi. Il modello *HMM* è applicabile a sistemi reali in cui si vuole ricostruire una sequenza di stati non immediatamente osservabile a partire da dati probabilistici dipendenti dalla sequenza stessa, come ad esempio in ambito di tecniche di speech recognition, criptoanalisi, finanza computazionale.

Studi recenti confermano che algoritmi che fanno uso del modello *HMM*, come la *Monte Carlo localization* (o particle filtering localization) per la stima della posizione di un robot a partire da dati spaziali, sono stati applicati con successo a scenari di localizzazione indoor, sfruttando come fonte di informazione sia il magnetismo, sia la tecnologia wireless [13, 14].

2.2.2 Fuzzy pattern recognition

Quando non è possibile disporre di rilevazioni ad alto livello di precisione, come nel caso della tecnologia wireless, è opportuno considerare la variazione tra valori successivi piuttosto che i valori assoluti, che possono essere soggetti a interferenze temporanee. Può allora essere utile creare un fingerprint che tenga traccia non della sola informazione rilevata, bensì dell'entità delle variazioni registrate in corrispondenza di una certa locazione. Tecniche di fuzzy pattern recognition sono state applicate nell'ambito di studi sulla localizzazione con Wi-Fi fingerprinting con risultati positivi [14, 15, 16].

L'analisi dell'andamento dei valori sulla base di rilevazioni multiple può migliorare l'efficacia del posizionamento anche utilizzando il magnetismo, considerato come piccole variazioni introdotte dalla presenza di materiali metallici nell'ambiente circostante possano dare origine a errori di natura del tutto simile.

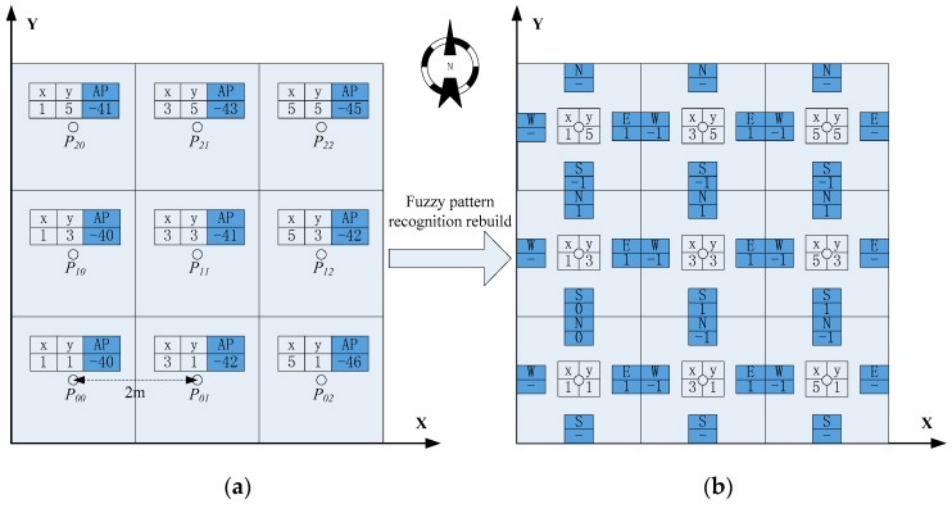


Figura 2.3: Esempio di applicazione di fuzzy pattern recognition ad un fingerprint preesistente: **(a)** fingerprint nel suo stato originario; **(b)** mappa delle variazioni. Per ogni transizione, un valore pari a +1 indica variazione positiva, -1 indica una variazione negativa, mentre uno 0 indica una variazione trascurabile del segnale di input. *Fonte: [14]*

2.2.3 IndoorAtlas

IndoorAtlas è la prima esperienza di localizzazione indoors tramite fingerprinting magnetico applicata con successo ad un sistema reale diffuso sul mercato. La compagnia statunitense, fondata nel 2012 dal professor Janne Haverinen dell'Università di Oulu, ha sviluppato una piattaforma cloud-based che offre un'API utilizzabile da sviluppatori di tutto il mondo per produrre applicazioni per dispositivi mobili dotate di servizi location-based basati su magnetismo [17].

Il sistema realizzato da IndoorAtlas adotta una complessa procedura di creazione del fingerprint magnetico basata sull'individuazione di waypoints successivi e che integra elementi di sensor fusion, sfruttando i dati di accelerometro e Wi-Fi come fonti di dati alternative. L'algoritmo è proprietario e non è dato conoscerne la logica interna.

Durante l'analisi dei risultati di performance dell'algoritmo sviluppato

nell'ambito di questa tesi, si andranno a confrontare i risultati ottenuti con quelli attesi dall'applicazione sviluppata dalla compagnia statunitense.

Capitolo 3

Studio di fattibilità

Si è effettuato uno studio preliminare per compiere una prima raccolta di dati finalizzata a comprendere meglio quali tipi di valori del magnetismo ci si può aspettare di registrare in uno scenario indoors di grandi dimensioni, e se essi siano informativi in misura sufficiente a rendere possibile la realizzazione di un algoritmo che ne fa uso. A partire dai risultati ottenuti è stato possibile formulare una prima previsione relativamente a quali potrebbero essere i risultati sperati e le difficoltà più rilevanti.

3.1 Raccolta dati

Si è sviluppata una semplice applicazione per smartphone Android in grado di raccogliere valori successivi del magnetismo a intervalli di tempo regolari e memorizzarli su un file all'interno del dispositivo per una successiva analisi. L'applicazione offre le funzionalità essenziali: permette di modificare il sampling rate delle rilevazioni del magnetometro, nonché di avviare e, successivamente, terminare la registrazione dei dati magnetici. Durante una registrazione, i valori rilevati vengono scritti su un file *.csv* esterno.

Il magnetismo può essere misurato in weber per metro quadrato (Wb/m^2), tesla (T) o gauss (G). In questo studio si utilizzano i tesla, l'unità di misurazione indicata dal SI e quella con cui sono espresse tutte le misurazioni

restituite dal magnetometro dello smartphone utilizzato; più precisamente, i micro-tesla (uT), in quanto i dati rilevabili sono dell'ordine di grandezza delle decine di uT.

Lo smartphone utilizzato durante i test è un Samsung Galaxy S2 sul quale è installato un sistema operativo Android versione 4.2, versione API 16.

Il sensore magnetico installato all'interno del dispositivo è in grado di rilevare i valori del magnetismo sulle tre componenti x, y e z, i quali vengono restituiti dalla relativa Sensor API Android. Al fine di rendere le misurazioni indipendenti dall'orientamento del dispositivo, si considererà sempre unicamente il valore risultante del magnetismo, equivalente al prodotto vettoriale delle misurazioni sui tre assi, sia in questo studio preliminare, sia durante lo sviluppo dell'applicazione di posizionamento vera e propria:

$$V_r = \sqrt{v_x^2 + v_y^2 + v_z^2}$$

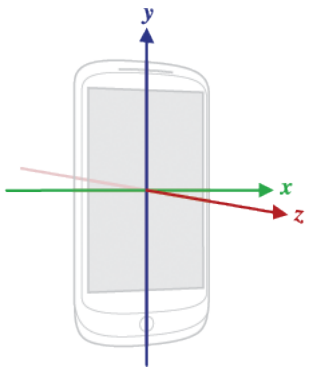


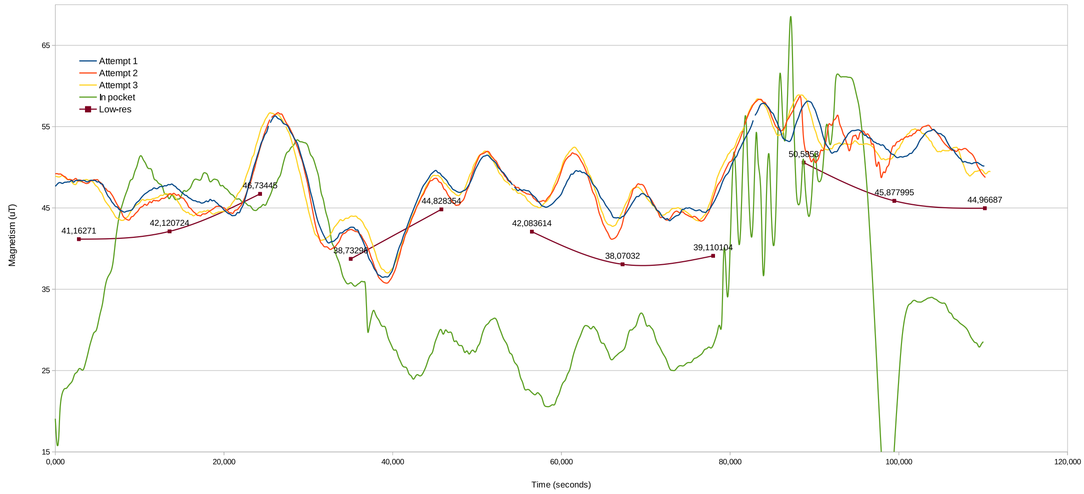
Figura 3.1: Sistema di coordinate del dispositivo.

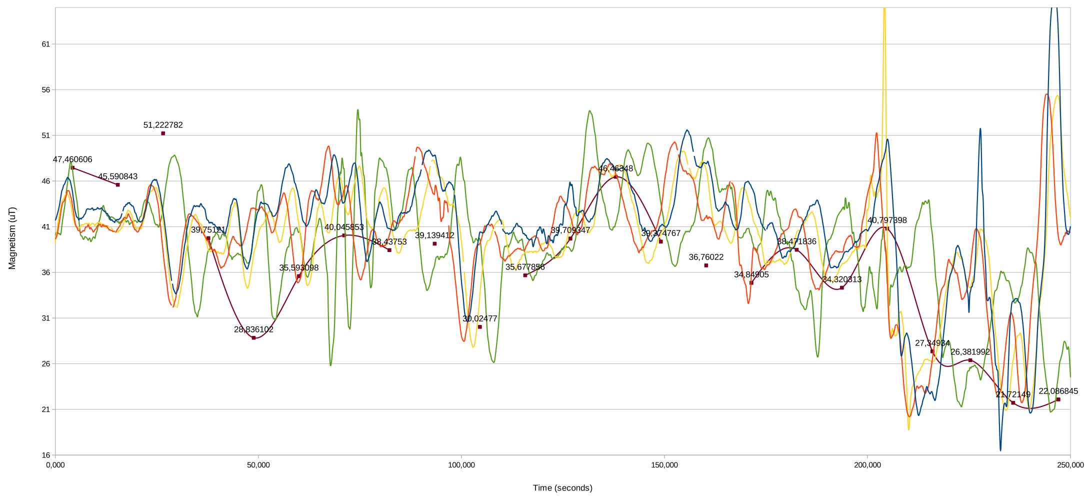
Tramite l'interfaccia fornita dal servizio Android che gestisce i dati del sensore è inoltre possibile ottenere i dati grezzi o, ed è questa la soluzione che è stata adottata, applicare ai valori la calibrazione *hard-iron*, che permette di annullare l'interferenza magnetica generata dalla conformazione dei componenti interni del dispositivo, rendendo così le misurazioni indipendenti dal dispositivo utilizzato per i test.

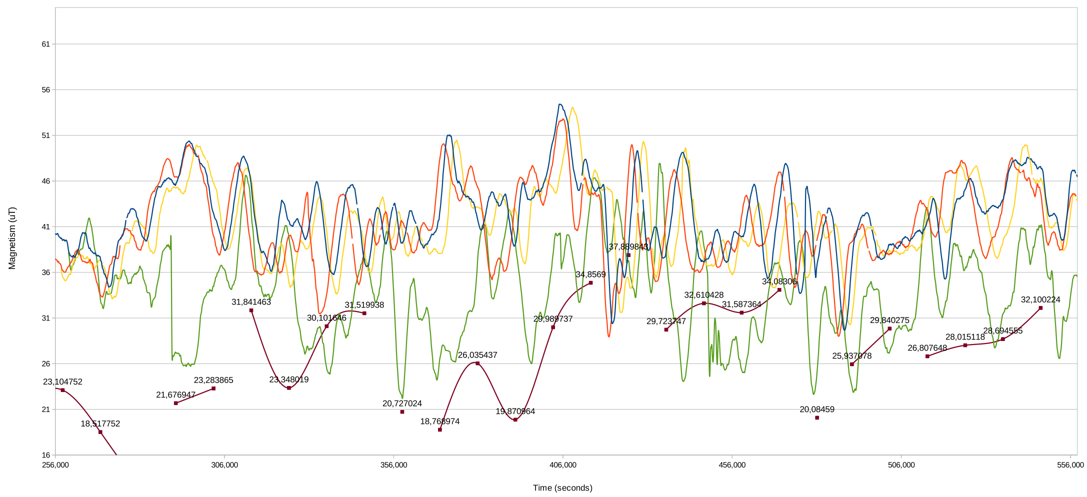
Lo studio è stato effettuato su tre diversi scenari indoor, tre centri commerciali di medie e grandi dimensioni. Si sono registrati i valori del magnetismo utilizzando l'applicazione, percorrendo lo stesso tragitto più volte.

Inizialmente si sono effettuati tre test identici con frequenza di sampling di 20.000 microsecondi, con smartphone in mano. Per ciascuno degli scenari presentati si è poi effettuato un quarto test con smartphone in tasca e un quinto effettuando misurazioni con frequenza di sampling più bassa, pari a 400.000 microsecondi.

Figura 3.2: I tre scenari di test e relativi percorsi sui quali sono state effettuate le rilevazioni.


Attempt 1
Attempt 2
Attempt 3
In pocket
Low-res
Magnetism (uT)
Time (seconds)
41,16271
42,120724
46,73445
38,73298
44,828354
42,083614
38,07032
39,110104
50,5858
45,877995
44,96687


Magnetism (uT)
Time (seconds)
47,460606
51,222782
45,590843
39,75121
28,836102
35,593098
40,045853
38,43753
39,139412
30,02477
35,677858
39,709347
46,46348
39,274767
36,76022
34,84905
38,471836
34,320313
40,797398
27,34934
26,381992
21,72149
22,086845


Magnetism (uT)
Time (seconds)
23,104752
18,517752
21,676947
23,283865
31,841463
23,348019
30,101646
31,519938
20,727024
18,768974
26,035437
19,870964
29,989737
34,8569
37,889840
29,723747
32,610428
31,587364
34,08306
20,08459
25,937078
29,840275
26,807648
28,015118
28,694555
32,100224

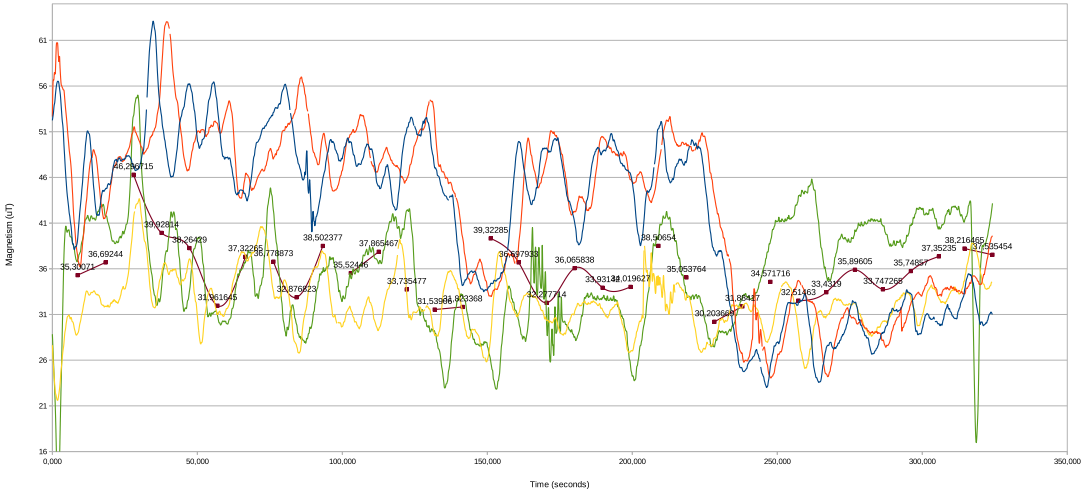


Figura 3.3: Dati del magnetismo per i tre percorsi visti in figura 3.2.

3.2 Analisi dei dati ottenuti

Le rilevazioni dimostrano come le variazioni tra diverse misurazioni dello stesso percorso siano essenzialmente simili. In corrispondenza di elementi strutturali quali pilastri o pareti, vengono registrate interferenze di entità simile le quali si ripetono allo stesso modo per ciascuna delle misurazioni. I dati non presentano variazioni isolate a singole misurazioni; l'errore massimo tra successive registrazioni dello stesso percorso è mediamente di pochi uT. Si osserva tuttavia che non si possono escludere valori isolati con accuratezza ridotta. Le misurazioni con livello di sampling più basso dimostrano come sia effettivamente possibile ricostruire l'andamento generale della variazione di magnetismo anche con un numero inferiore di rilevazioni.

3.2.1 Smartphone in mano e in tasca

È riscontrabile una differenza sostanziale tra i dati misurati con smartphone in mano e quelli con smartphone in tasca, che presentano variazioni a tratti molto più accentuate. Esse sono principalmente causate dalla perdita

di accuratezza del magnetometro in seguito a oscillazioni e rotazioni continue del dispositivo durante la camminata. Tali variazioni suggeriscono la necessità di riconoscere variazioni particolarmente ampie per equipararle a differenze anche meno rilevanti nella mappa magnetica che si andrà a ottenere nell'applicazione da sviluppare in seguito. In caso di errori ripetuti causati da un'iniziale perdita di accuratezza, come nel caso del terzo scenario visto, potrebbe essere necessario indicare all'utente che occorre effettuare la ricalibrazione del dispositivo. La perdita di accuratezza in seguito a rotazioni e movimenti rapidi dipende principalmente dalla qualità del sensore installato.

3.2.2 Risultati

È importante che l'informazione magnetica sia sufficientemente diversificata e con un grado di variabilità sufficientemente alto ai fini dello sviluppo di un algoritmo che possa stimare la posizione a partire da dati di questo tipo. I risultati mettono alla luce come le interferenze del campo magnetico causate dagli elementi strutturali di edifici di grandi dimensioni quali quelli sui quali si è effettuato lo studio presentino variazioni ampie e consistenti, che sono distribuite in modo eterogeneo su un range di 20-30 uT. I dati risultano dunque essere sufficientemente informativi e caratterizzanti; si hanno variazioni costanti e si denota una pressoché totale assenza di zone neutre (con valore del magnetismo stabile).

Le rilevazioni dimostrano come punti diversi all'interno dello stesso percorso possano presentare valori identici del magnetismo. Dunque, un singolo dato non è sufficiente a identificare un punto preciso. Il riconoscimento della posizione dovrà quindi basarsi sul riconoscimento di una traiettoria, confrontando valori successivi rilevati durante lo spostamento dell'utente con una mappa di valori del magnetismo ottenuta durante una fase preliminare. Si procede quindi allo sviluppo di un algoritmo di localizzazione che faccia uso di siffatte rilevazioni per la localizzazione in tempo reale.

Capitolo 4

Algoritmo di posizionamento

In questo capitolo verrà esposta la fase di elaborazione di un algoritmo per la localizzazione che utilizza i dati del magnetismo. Lo scopo primario è quello di definire la struttura essenziale di un potenziale algoritmo che porti a buoni risultati finali e cercare così di comprendere quali siano gli aspetti vantaggiosi e quali le maggiori difficoltà nella realizzazione di un software per un sistema di localizzazione di questo tipo. Si è realizzata un'applicazione Android che realizza le due funzioni essenziali di fingerprinting, già descritte nella loro struttura essenziale nel secondo capitolo: una fase "offline" di mappatura del luogo designato per la localizzazione, che chiameremo *mapping*, e una successiva fase "online" durante la quale l'utente utilizza l'applicazione per l'effettiva localizzazione, spostandosi all'interno dell'edificio, che chiameremo *locating*.

4.1 Mapping e locating

Le due modalità di utilizzo dell'applicazione realizzata implementano dunque due funzioni ben distinte.

Il mapping consiste nella raccolta dei dati del magnetismo su diverse posizioni distribuite lungo tutta la superficie percorribile di un edificio. L'applicazione offre una visualizzazione a griglia; lo spazio è diviso in tasselli

quadrati. L'utente, posizionandosi in corrispondenza di un determinato tassello, registra il valore del magnetismo alla posizione attuale. L'applicazione ricostruisce internamente una mappa dei valori del magnetismo abbinando ad ogni posizione il valore registrato dal sensore. L'applicazione salva poi i dati di mapping, permettendone comunque l'aggiornamento in un secondo momento.

L'operazione di mapping è da effettuare una sola volta, in condizioni ottimali, stazionando su ogni tassello per registrare valori il più possibile accurati e non influenzati da movimenti del dispositivo. In un sistema reale, il mapping dovrà essere effettuato nuovamente in caso di variazioni rilevanti della struttura dell'interno, come la modifica di elementi strutturali o lo spostamento di grandi elementi di arredo.

Per le prime fasi di test si è effettuato il mapping di un centro commerciale di grandi dimensioni (lo scenario è il medesimo utilizzato nel secondo scenario già contemplato durante lo studio preliminare a 3.1). Si è così ottenuta una mappa geomagnetica della struttura.

La fase di locating presenta le maggiori difficoltà di implementazione, in quanto durante questa fase deve essere effettuata la localizzazione vera e propria. Durante lo spostamento dell'utente viene registrato in tempo reale il valore del magnetismo a intervalli di tempo successivi. A partire dal valore ottenuto al momento, nonché da eventuali valori registrati negli ultimi istanti, tramite il confronto con la mappa geomagnetica si deve determinare la posizione finale da mostrare all'utente.

L'intervallo di tempo tra il reperimento di due valori successivi del magnetismo deve essere impostato in accordo con l'ampiezza dei tasselli, in modo che l'utente, muovendosi a velocità media, non possa attraversare un determinato tassello senza che venga effettuata almeno una rilevazione. L'algoritmo sviluppato, tuttavia, contempla casi sporadici in cui vengono oltrepassati tasselli in questo modo, anche se la precisione finale di posizionamento ne è compromessa. Per i test effettuati durante lo sviluppo dell'algoritmo l'intervallo di tempo utilizzato è di 3 secondi.

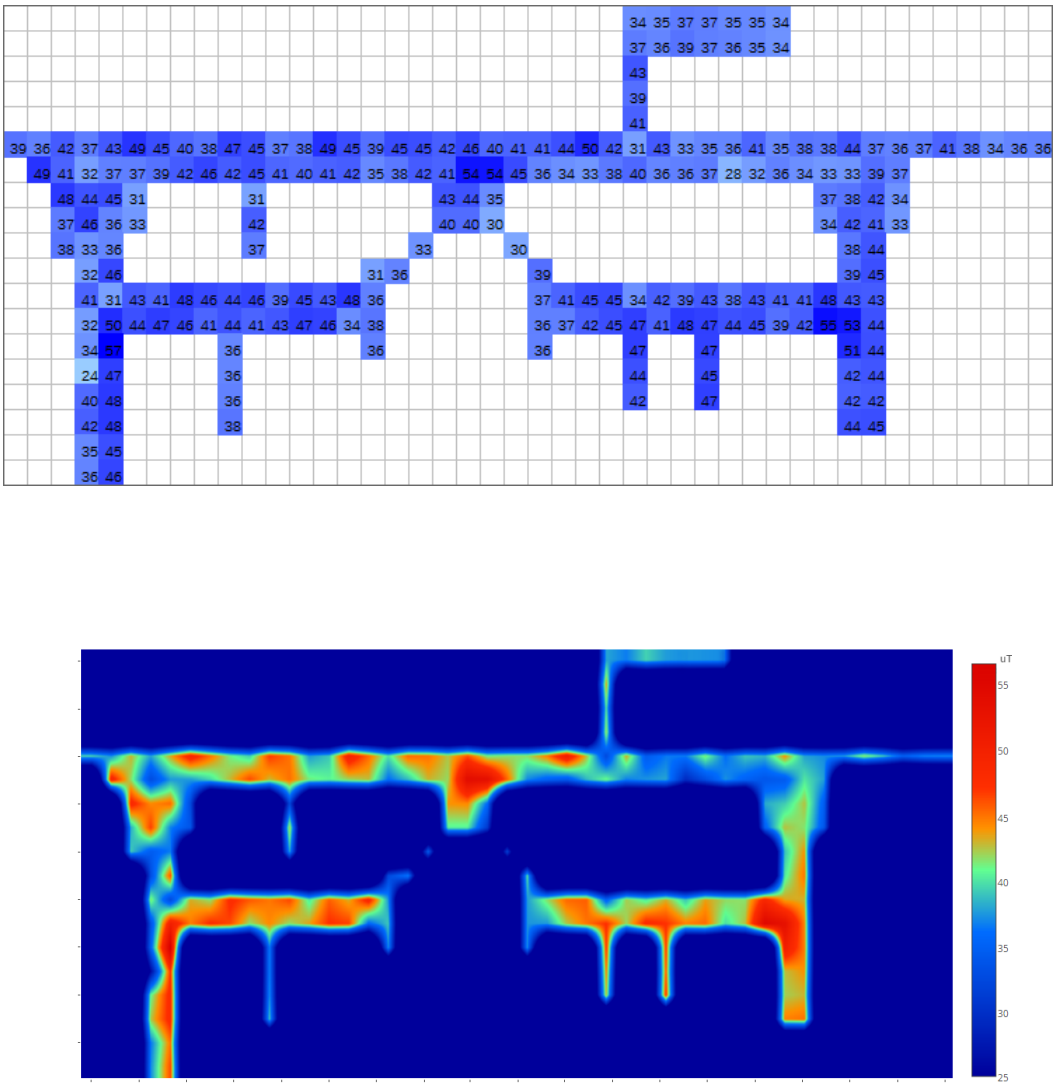


Figura 4.1: Mappa geomagnetica della struttura, ottenuta durante la fase di mapping, e heatmap generata a partire dai dati.

4.2 Un'iterazione

Come precedentemente menzionato, il singolo dato magnetico non è sufficiente a stabilire la posizione corrente. In caso di grandi scenari con mappe magnetiche di dimensioni rilevanti, lo stesso valore può essere assegnato ad

ampie aree dello spazio. Occorre ricostruire la traiettoria reale dell'utente basandosi sul confronto di diversi percorsi probabili che attraversano tasselli successivi.

L'algoritmo lavora su iterazioni successive: per ogni valore del magnetismo rilevato istantaneamente dal magnetometro viene effettuata un'iterazione. Al termine dell'iterazione, la posizione finale calcolata viene mostrata all'utente.

4.2.1 Probabilità di posizionamento

Data la natura incerta delle misurazioni, le quali presentano un certo grado di imprecisione causato dalle microvariazioni del campo magnetico e dalla perdita di accuratezza del sensore a causa di spostamenti rapidi del dispositivo, il riconoscimento della posizione non si può basare su valori esatti, ma si deve considerare un certo errore. A ogni supposizione effettuata a partire dai valori rilevati è dunque opportuno attribuire una certa probabilità.

Si è deciso di sviluppare un modello basato sull'assegnazione di valori di probabilità a ciascun tassello. Per ogni iterazione dell'algoritmo si attribuisce ad ogni posizione una certa probabilità di essere la posizione corretta, basata sulla differenza tra il valore della mappa magnetica e quello misurato al momento. Vengono inoltre contemplate le transizioni tra posizioni adiacenti con probabilità derivanti da iterazioni precedenti, le quali esercitano un'influenza sui nuovi valori. In altre parole, nelle iterazioni successive a un primo posizionamento a ogni tassello viene attribuita probabilità proveniente dai tasselli adiacenti nella configurazione immediatamente precedente. In questo modo, spostamenti successivi porteranno a una concentrazione delle probabilità sulle posizioni più compatibili con la serie di valori del magnetismo rilevati, con progressivo aumento della precisione.

Primi risultati

L'applicazione è stata adattata per testare più agevolmente la performance dell'algoritmo, in modalità "offline": dapprima sono stati registrati i dati

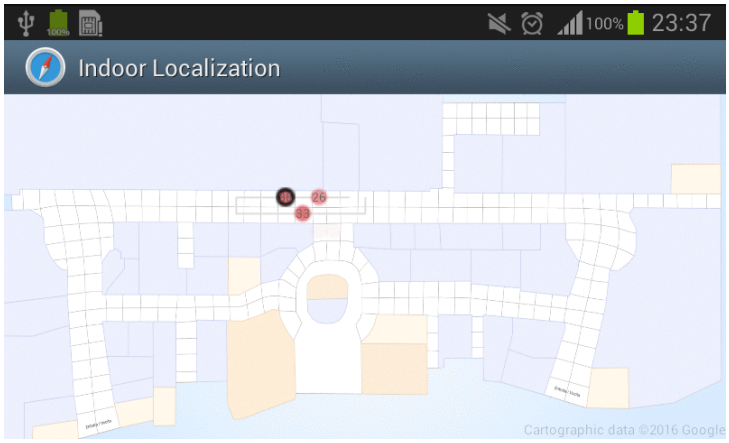


Figura 4.2: Preview in-app dell'attività di locating.

magnetici su alcuni percorsi, spostandosi all'interno dell'edificio di cui abbiamo già ottenuto la mappa magnetica. I dati del magnetismo ottenuti in tempo reale sono stati salvati esternamente all'applicazione. Durante gli sviluppi successivi dell'algoritmo si è poi testata l'applicazione sugli stessi dati simulando un'esecuzione in tempo reale, osservando l'efficacia delle modifiche apportate durante il processo.

In una prima versione dell'algoritmo, la probabilità di ogni tassello è calcolata come una somma bilanciata di due valori percentuali; il primo è influenzato dalla misurazione effettuata al momento in confronto con la mappa magnetica, mentre il secondo viene calcolato in base al valore di probabilità dei tasselli adiacenti dell'iterazione precedente. Questo tipo di approccio si è rivelato subito avere scarsa efficacia. In assenza di singoli valori isolati che determinano una posizione ben determinata nella mappa magnetica, la localizzazione è altamente imprecisa e non viene identificata una traiettoria certa. Vi è troppa dispersione dei valori di probabilità tra tasselli con valori del magnetismo simili. Il problema essenziale è identificabile nel criterio applicato per il trasferimento delle probabilità tra tasselli adiacenti.

Assegnazione delle probabilità

Si rende necessario applicare una logica differente al fine di localizzare la concentrazione di probabilità attorno a una certa locazione e dare maggiore incidenza alle informazioni derivate dalle iterazioni precedenti. Viene adottato un modello di assegnazione della probabilità che ricorda un processo Markoviano: la probabilità di transizione da un tassello a un altro dipende direttamente solo dalla probabilità del tassello nell'istante immediatamente precedente, e la dipendenza è diretta. La probabilità di ogni tassello viene dunque decisa calcolando un prodotto pesato tra due termini, entrambi costituiti da cifre minori di uno. Il primo, come visto in precedenza, è un valore che dipende dalla misurazione attuale, mentre il secondo dipende dalla configurazione precedente delle probabilità.

Al primo termine viene assegnato un valore proporzionale alla differenza tra il valore attuale del magnetismo e quello della mappa geomagnetica: minore è l'errore di misurazione, maggiore ne risulta la probabilità. Se l'errore è troppo elevato, la probabilità finale assegnata è nulla.

Il calcolo del secondo termine del prodotto è leggermente più complesso.

Per ogni tassello viene calcolato un valore, che per brevità chiameremo *sum_ in*, equivalente all'ammontare della somma delle probabilità dei tasselli ad esso adiacenti nell'iterazione precedente. I valori di *sum_ in* di ogni tassello, ricalcolati ad ogni iterazione, vengono così influenzati da tutti i possibili spostamenti di probabilità tra tasselli adiacenti.

Infine, il secondo termine del prodotto è dato dal rapporto tra il *sum_ in* del tassello considerato e la somma dei *sum_ in* di tutti i tasselli.

Se si è alla prima misurazione all'avvio dell'attività di locating, in assenza di valori precedenti di probabilità, viene soltanto preso in considerazione il primo termine del prodotto sopra descritto, che dipende dal valore del magnetismo corrente, mentre il secondo è ignorato.

I valori così assegnati ai singoli tasselli sono tutti minori di uno, per cui devono essere normalizzati complessivamente per riportarli a veri e propri valori percentuali di probabilità la cui somma è 100.

Il codice sorgente relativo a questa parte della computazione può essere consultato in appendice A.1.

4.2.2 Elementi di fuzzy pattern recognition

Come visto per i sistemi di localizzazione wireless, un algoritmo di pattern matching applicabile a casi reali deve tenere conto di possibili oscillazioni dei valori rilevati. Nel caso del Wi-Fi fingerprinting, l'instabilità del segnale ricevuto da differenti access point e attenuato da pareti ed elementi strutturali ne è la causa principale. Dunque è opportuno riconoscere l'entità delle variazioni registrate anche avendo una certa variabilità del segnale e lo si può fare applicando logiche di fuzzy pattern recognition [14].

Problematiche del tutto simili sono riscontrabili per il magnetismo. La vicinanza a una struttura portante comporta variazioni anche molto ampie dei valori rilevati; spostamenti di pochi metri possono dunque portare a misurazioni imprecise. Come è stato osservato con i test preliminari, tuttavia, l'andamento delle fluttuazioni rimane consistente e può essere riconosciuto.

Un'ulteriore causa di fluttuazione dei dati è la velocità di spostamento, che agisce a discapito dell'accuratezza delle rilevazioni. Inoltre, se l'utente si muove secondo un pattern circolare o a slalom, i tasselli limitrofi possono essere esclusi dall'algoritmo se la variazione del magnetismo rilevata non corrisponde a quella effettiva nella mappa geomagnetica, contribuendo ad identificare la corretta direzione di spostamento.

Si è dunque deciso di applicare elementi di fuzzy pattern recognition all'algoritmo, introducendo un nuovo elemento di tolleranza dell'errore di misurazione.

Per ogni coppia di tasselli adiacenti della mappa magnetica, viene registrato un valore di transizione positivo ($+1$), negativo (-1) o nullo (0), che dipende dall'entità della variazione di magnetismo nella transizione da un tassello all'altro. Viene così creata una nuova mappa delle transizioni, che assegna un valore ad ogni coppia ordinata di tasselli. Durante il calcolo del sum_in per un certo tassello, alla probabilità ricevuta da un tassello adia-

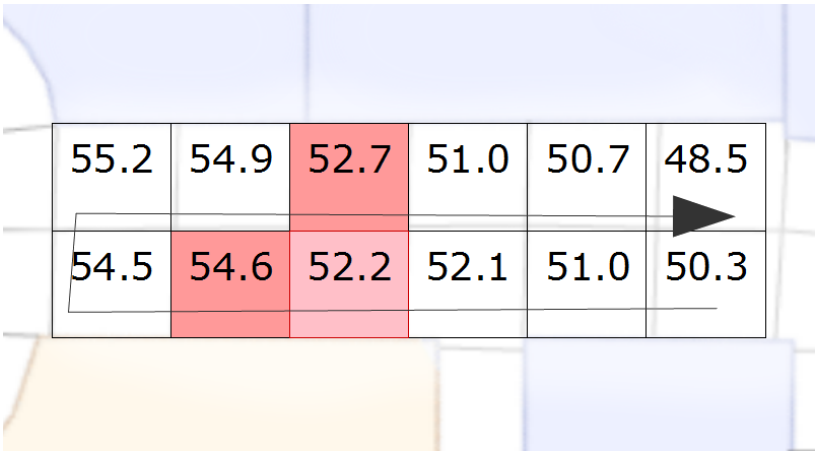


Figura 4.3: Esempio di applicazione di fuzzy pattern. Nella situazione qui presentata, supponendo di trovarsi in corrispondenza della casella evidenziata in rosso, e avendo all'iterazione corrispondente un valore del magnetismo prima pari a 51, poi a 53.5, verrà assegnato un valore di probabilità maggiore alla casella a sinistra rispetto a quella in alto, perché nonostante il valore effettivo sia meno impreciso nel secondo caso, si è tuttavia registrato un aumento sensibile del magnetismo e quindi si predilige un tassello che in fase di mapping ha anch'esso registrato una variazione positiva.

cente è attribuita incidenza minore se l'entità della variazione tra il valore del magnetismo corrente e quello appena misurato non coincide con il valore salvato nella mappa delle transizioni per i due tasselli.

Si verifica dunque che se i valori del magnetismo aumentano consistentemente nell'istante corrente, allora ai tasselli che hanno registrato un consistente aumento del magnetismo anche durante il mapping sarà assegnata maggiore probabilità. In questo modo i valori di probabilità, anche se imprecisi, vengono considerati in relazione alla loro variazione in positivo o negativo, permettendo un'identificazione più mirata del percorso effettivo.

Il codice sorgente relativo a questa parte della computazione può essere consultato in appendice A.2.

4.2.3 Posizioni probabili e posizione calcolata

Dopo avere attribuito valori di probabilità a ogni posizione, occorre un modello che sfrutti tali informazioni per distinguere un'unica posizione finale da mostrare all'utente. Scegliere la posizione con probabilità massima non è sempre la soluzione ottimale; occorre ricostruire un percorso reale, non una sequenza di posizioni sconnesse. La prima parte dell'algoritmo, infatti, elabora molteplici traiettorie possibili, a partire dalle quali è necessario ricostruire una traiettoria realistica, che non sia costituita da una sequenza di improbabili salti tra posizioni anche distanti tra loro.

È necessario sviluppare una procedura di *smoothing* del percorso calcolato che, indipendentemente dalle probabilità assegnate, determini quale sia la posizione finale da mostrare all'utente. Si utilizza una coda in cui vengono memorizzate le ultime posizioni designate come corrette dalle iterazioni precedenti. Ad ogni iterazione, viene designata la posizione da mostrare all'utente, che viene poi posta in cima alla coda. La coda deve sempre contenere posizioni adiacenti fra loro al suo interno. In seguito all'aggiunta di una posizione non adiacente a quella in cima alla coda, essa viene svuotata e viene mantenuta solo l'ultima. Ad ogni iterazione viene designata la posizione da aggiungere in cima alla coda secondo le seguenti modalità.

- Se la coda è piena, viene scelta la più probabile tra le posizioni adiacenti e aggiunta in cima alla coda, cercando così di restare sul percorso individuato. Questo a patto che la probabilità della posizione designata non scenda al di sotto di un certo *threshold* minimo.

- Altrimenti, se la coda non è piena, si considera il tassello più probabile in assoluto: se è adiacente all'ultima posizione predetta, viene scelto come posizione e aggiunto in cima alla coda. Le restanti posizioni nella coda vengono mantenute, dato che la nuova posizione è coerente al percorso calcolato in precedenza.

- Altrimenti si calcola lo scarto di probabilità tra la posizione più probabile in assoluto e quella più probabile adiacente all'ultima posizione:

se è molto piccolo, allora si sceglie quella adiacente da aggiungere alla coda, cercando così di scegliere una posizione coerente al percorso calcolato.

- Altrimenti, nessuna delle posizioni adiacenti all'ultima posizione scelta è risultata abbastanza probabile. Si deve quindi scegliere la posizione più probabile in assoluto, anche se in questo modo si effettua un irrealistico salto rispetto all'ultima posizione. La coda viene svuotata e la posizione più probabile viene aggiunta in cima e designata come posizione attuale.

L'enfasi è posta sul riconoscimento della traiettoria dell'utente, piuttosto che sulle singole posizioni; se la coda è piena, le ultime posizioni predette sono tutte adiacenti, il che è un buon indice del fatto che l'algoritmo abbia predetto la traiettoria reale con margine di errore minimo. In questo caso, possono essere applicati nuovi aggiustamenti per tentare di tenere traccia del percorso individuato durante gli spostamenti successivi dell'utente.

Vengono dunque effettuati alcuni aggiustamenti sulle posizioni più probabili nell'iterazione successiva:

Range di dispersione delle probabilità Se la coda contiene posizioni adiacenti, e dunque non si è verificato un salto di posizione, il range di dispersione delle probabilità può essere ridotto, effettuando una selezione delle traiettorie più probabili; dunque, in questo caso, le posizioni con probabilità più bassa vengono scartate;

Controllo di ridondanza Se la coda è piena e le probabilità vicine all'ultima posizione sono distribuite su direzioni opposte, occorre discriminare tra diverse traiettorie possibili. Si prediligono i tasselli attraversati meno di recente; il nuovo valore di probabilità dei tasselli appena attraversati viene invece diminuito, applicando un controllo di ridondanza. Questo evita che la probabilità residua dalle iterazioni precedenti confonda i posizionamenti successivi. Si potrebbe obiettare che questo va

a discapito del posizionamento nel caso in cui l'utente compia spostamenti particolari, come muoversi alternando tra due caselle. Tuttavia certi spostamenti sono piuttosto inusuali e se avvengono di rado non compromettono la distribuzione delle probabilità sui tasselli in maniera evidente. Naturalmente si suppone che l'utente compia spostamenti realistici durante l'uso dell'applicazione di posizionamento.

E' importante notare che tali aggiustamenti vengono applicati quando i valori di probabilità devono ancora essere normalizzati, per cui moltiplicare alcuni di essi non ha effetti negativi.

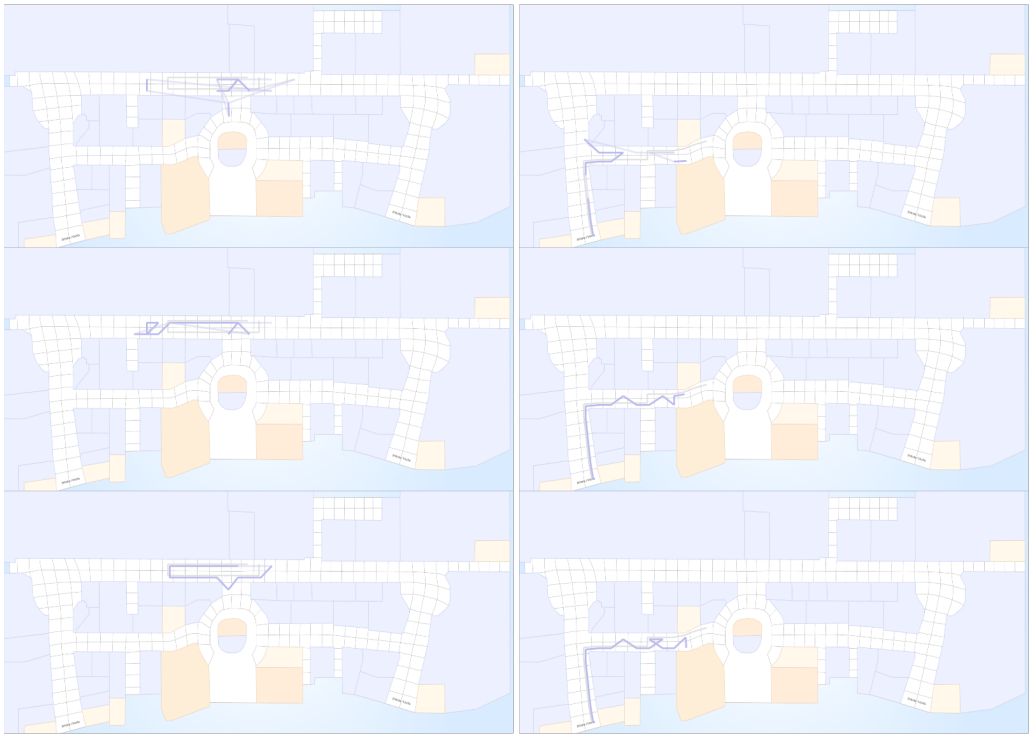


Figura 4.4: Fasi successive di sviluppo dell'algoritmo per due dei percorsi studiati. Fase 1: modello a catena di probabilità con filtro fuzzy pattern; fase 2: aggiunta del controllo di ridondanza; fase 3: aggiustamento del range di dispersione delle probabilità con selezione delle traiettorie probabili.

Il codice sorgente relativo a questa parte della computazione può essere consultato in appendice A.3.

4.2.4 Controllo e riposizionamento

L'applicazione degli accorgimenti descritti ha avuto effetti positivi sulla precisione del posizionamento finale, come si è potuto riscontrare tramite test successivi sui dati dei percorsi ottenuti in precedenza. In caso di grandi incertezze delle rilevazioni dei valori del magnetismo è tuttavia possibile che il posizionamento ne risulti completamente compromesso; si è rivelato necessario allora elaborare una procedura di controllo.

Il controllo viene eseguito una volta ogni 10 iterazioni, in caso di incoerenze evidenti tra i dati di probabilità e il percorso calcolato, al fine di ripristinare uno stato neutro dove un solo tassello ha probabilità massima di volta in volta, finché non ne viene scelto uno con una buona approssimazione rispetto alla posizione reale, da cui l'algoritmo è in grado di recuperare la posizione effettiva.

Per riconoscere una situazione di incoerenza tale da richiedere il ricalcolo della posizione, si esamina la distanza media in tasselli tra le posizioni più probabili e la posizione calcolata nelle ultime 10 iterazioni; se essa supera un limite massimo, allora si effettua il riposizionamento.

Viene nuovamente calcolata la probabilità di ogni tassello come se si trattasse di una misurazione iniziale, allo stesso modo di come visto in precedenza, cioè basandosi unicamente sul confronto tra il valore del magnetismo corrente e quello della mappa magnetica. Si sceglie di dimenticare l'informazione fornita dalle probabilità dell'ultima iterazione appunto perché si è constatato che la differenza tra le probabilità e la posizione designata è divenuta eccessiva.

Basandosi dunque solo sul nuovo valore del magnetismo, si sceglie casualmente uno tra i tasselli il cui valore si avvicina di più a quello appena misurato e lo si sceglie come prossima posizione. Naturalmente questo può portare a imprecisioni, ma una rilocalizzazione approssimata viene comunque

effettuata; successive iterazioni dell'algoritmo sono sufficienti a ripristinare un posizionamento più preciso.

Il codice sorgente relativo a questa parte della computazione può essere consultato in appendice A.4.

4.3 Costanti

Durante l'elaborazione dell'algoritmo si sono introdotti diversi coefficienti numerici che determinano l'incidenza di ogni sottoprocedura dell'algoritmo. Assume importanza attribuire valori ottimali a tali costanti per ottenere la massima precisione possibile. I parametri cruciali sono:

- i pesi applicati ai due termini del prodotto bilanciato di attribuzione della probabilità a un tassello, che determinano il rapporto tra l'incidenza del dato istantaneo e di quello derivante dall'iterazione precedente nel determinare la posizione finale (4.2.1);
- la lunghezza massima della coda delle posizioni successive calcolate, parametro fondamentale che determina quante posizioni adiacenti successive sono necessarie per applicare la scelta di tasselli contigui per il calcolo della posizione, l'aggiustamento del range di dispersione dei valori di probabilità e il controllo di ridondanza, come indicato in precedenza (4.2.3);
- l'incidenza del filtro di fuzzy pattern (4.2.2);
- l'ampiezza di variazione magnetica necessaria a identificare una variazione positiva o negativa nell'applicazione della logica di fuzzy pattern, in microtesla;
- il numero massimo di posizioni mantenute in caso di applicazione dell'operazione di selezione del range di probabilità trattata nella sezione 4.2.3;

- l'incidenza del controllo di ridondanza, anch'esso trattato nella medesima sezione;
- l'incidenza della stessa operazione di selezione sulla probabilità delle posizioni scartate.

Implementazione in Python

L'intero algoritmo è stato implementato nuovamente in Python per permetterne il testing al di fuori di un dispositivo mobile, che possiede risorse di calcolo pur sempre limitate e rende difficoltoso effettuare esecuzioni ripetute ed estrarre risultati complessivi. Grazie allo script così sviluppato è stata attuata una serie di test automatizzati sui dati di posizione precedentemente ottenuti (4.2.1), al fine di sondare la performance dell'algoritmo applicando valori diversi dei parametri descritti per comprendere quale incidenza sia opportuno attribuire a ciascuno di essi. Esecuzioni ripetute dell'algoritmo con configurazioni diverse dei parametri hanno portato a stabilire, ad esempio, che una sequenza di tre posizioni adiacenti successive calcolate è ottimale, per lo meno nei casi di test contemplati, per considerare il posizionamento affidabile e applicare le correzioni già descritte.

Capitolo 5

Analisi di performance

I potenziali futuri scenari di applicazione pratica di un sistema di localizzazione indoor quale quello realizzato sono tipicamente strutture pubbliche quali stazioni, centri commerciali, biblioteche, aeroporti. Si tratta dunque di ambienti eterogenei, potenzialmente molto diversi tra loro, i quali possono presentare caratteristiche particolari, come la presenza di strutture portanti più o meno spesse o l'impiego di particolari materiali di costruzione, che possono compromettere l'utilizzo di una particolare tecnica di posizionamento o renderla largamente inefficace. Nel caso del sistema studiato, che dipende dal fingerprint magnetico di uno specifico edificio, ciò è quanto mai vero; è essenziale dunque testare il funzionamento dell'applicazione sviluppata in ambienti diversi, di medie e grandi dimensioni. Effettuando un'analisi del posizionamento su scenari reali e confrontando la traiettoria reale con quella calcolata, si otterranno valori di performance che diano un'indicazione sia del risultato generale raggiungibile con l'impiego di una tecnica di questo tipo a grandi ambienti indoor, sia dell'effettivo livello di precisione dell'algoritmo qui specificatamente elaborato, in modo da poterne quantificare l'efficacia, la scalabilità e la precisione complessiva, nonché applicare un confronto con altri sistemi esistenti dello stesso tipo già studiati.

Dopo avere ultimato lo sviluppo dell'applicazione Android al fine di operare su dati raccolti in tempo reale e mostrare direttamente la posizione

calcolata all'utente, si modifica la stessa in modo da poter estrarre i dati di posizione ed effettuare un confronto con la traiettoria realmente percorsa. Si procede quindi al testing dell'applicazione in contesti differenti.

5.1 Test su scenari indoor

Vengono effettuati test di performance in tre specifici scenari indoor. Si tratta di tre ambienti di dimensioni diverse e con peculiari caratteristiche strutturali:

1. un grande centro commerciale; l'ambiente è costituito da ampi corridoi, alti soffitti e imponenti strutture portanti e si estende su un'area di 24.000 m^2;

2. una stazione ferroviaria di medie-piccole dimensioni; si hanno ambienti eterogenei, sia ampi come la sala d'aspetto, sia più ristretti come i corridoi di comunicazione, con variazioni geomagnetiche più localizzate; approssimativamente 3.000 m^2;

3. i laboratori del dipartimento di Informatica dell'Università di Bologna; ambiente di piccole dimensioni; sono presenti numerosi elementi di arredo in grado di causare microvariazioni del campo magnetico, quali postazioni pc e stampanti; approssimativamente $300m^2$.

5.1.1 Granularità delle misurazioni

Un parametro fondamentale nel determinare la precisione finale di posizionamento è la granularità delle misurazioni durante la fase di mapping. In ambienti interni, spostamenti di pochi metri possono determinare variazioni ingenti del magnetismo, soprattutto in prossimità di strutture portanti e elementi in ferro. È dunque importante che ad aree con variazioni localizzate siano associate più rilevazioni nella mappa magnetica.

Figura 5.1: Diversi scenari in cui si è effettuato il testing dell'applicazione: un centro commerciale; una stazione ferroviaria; i laboratori del dipartimento di Informatica di Bologna.

Per i primi due scenari studiati, sono state effettuate misurazioni ogni 5 metri, in modo da avere una densità sufficientemente elevata delle rilevazioni nella mappa geomagnetica da poter individuare variazioni del magnetismo efficientemente. Ciò implica che una localizzazione di un tassello di distanza dalla posizione reale corrisponderebbe, appunto, a un errore di posizionamento di 5 metri.

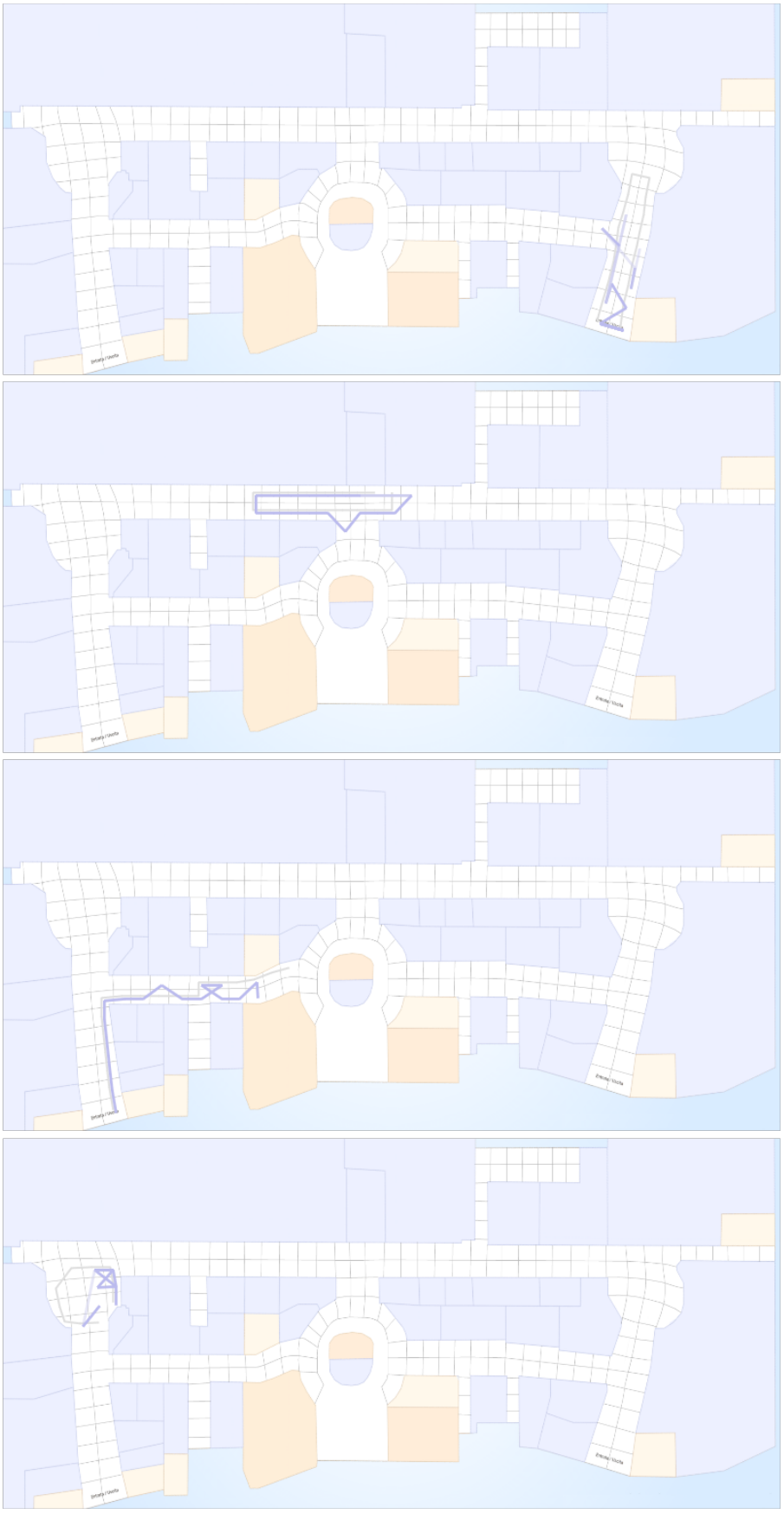
Per il terzo caso di test si è invece deciso di studiare il comportamento dell'applicazione con mappature di granularità differente. Sono state dunque effettuate tre mappature successive, con misurazioni ogni 1, 3 e 5 metri, per poi osservare la traiettoria calcolata su due percorsi differenti a ogni livello.

5.1.2 Test e risultati

Sono stati individuati alcuni percorsi di natura differente all'interno delle tre strutture interessate dai test. Vengono poi registrati i risultati della computazione dell'applicazione in tempo reale durante l'attraversamento di ognuno di essi, avendo preventivamente effettuato un primo posizionamento in corrispondenza della rispettiva posizione iniziale. Si analizzano infine i risultati ottenuti.

Per il primo scenario, quello la cui mappa geomagnetica presenta la maggiore quantità di dati, e dunque il più indicativo del livello di variabilità del magnetismo che è possibile riscontrare, sono state eseguite rilevazioni su quattro percorsi differenti di media lunghezza; i primi tre essenzialmente lineari, il quarto avente un pattern circolare.

È interessante notare come nel primo caso di test del primo scenario la scarsa informazione data dalla conformazione del campo magnetico localmente all'area interessata abbia compromesso il posizionamento in maniera considerevole. È purtroppo un requisito fondamentale per un uso efficiente dell'applicazione la presenza di variazioni rilevanti nei valori del magnetismo nella mappa magnetica. Se le variazioni sono minime, il fingerprint potrebbe non rivelarsi adatto e il posizionamento ne potrebbe risultare compromesso. Lo stesso avverrebbe se si utilizzasse l'applicazione all'aperto. Nel test menzionato si può notare che l'attraversamento di un'area con valori neutri e troppo simili tra loro nell'area interessata (si veda l'area corrispondente della mappa magnetica, figura 4.1) causa incertezze nel determinare la traiettoria corretta. Il fenomeno, isolato a questo unico caso osservato, dipende evidentemente dalla scarsa concentrazione di variazioni magnetiche localmente all'area interessata. Non si tratta dunque di un difetto nell'algoritmo di localizzazione, bensì di una conformazione non ideale dell'ambiente sul quale è stato effettuato il test.



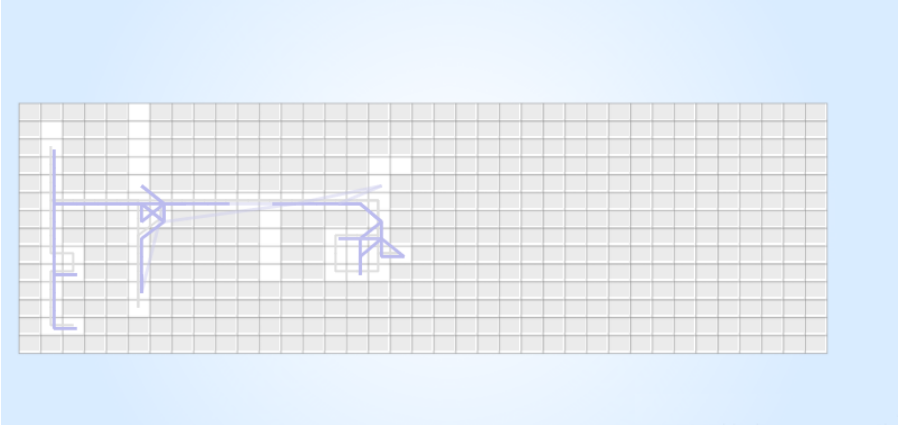


Figura 5.2: *(continua dalla pag. precedente)* Test effettuati nel primo e nel secondo scenario visti. Tasselli di 5m di lato; il percorso reale (in grigio) viene qui mostrato sotto a quello calcolato (in blu). Si noti la presenza di alcuni salti di posizione, in particolare in corrispondenza del primo e del quarto percorso ricostruiti. La planimetria del secondo scenario è stata riprodotta dall'applicazione a partire dal fingerprint magnetico.

Nel caso del secondo e del terzo percorso, la ricostruzione della traiettoria reale è nettamente migliore, con errori minimi di al più un tassello.

Nel quarto caso di test, la traiettoria corretta viene momentaneamente abbandonata, avvengono due salti di posizione e infine ci si ricongiunge al percorso reale.

Per il secondo scenario, di medie dimensioni, dopo avere effettuato il mapping è stato studiato un unico percorso di maggiore lunghezza, che interessa l'intera struttura attraversando più volte gli stessi ambienti. Tralasciando alcuni errori isolati, il percorso reale viene approssimato con sufficiente precisione.

Nel terzo scenario, i percorsi studiati sono due, entrambi di media lunghezza. In entrambi i casi si è seguita la stessa traiettoria per i tre livelli di precisione considerati.

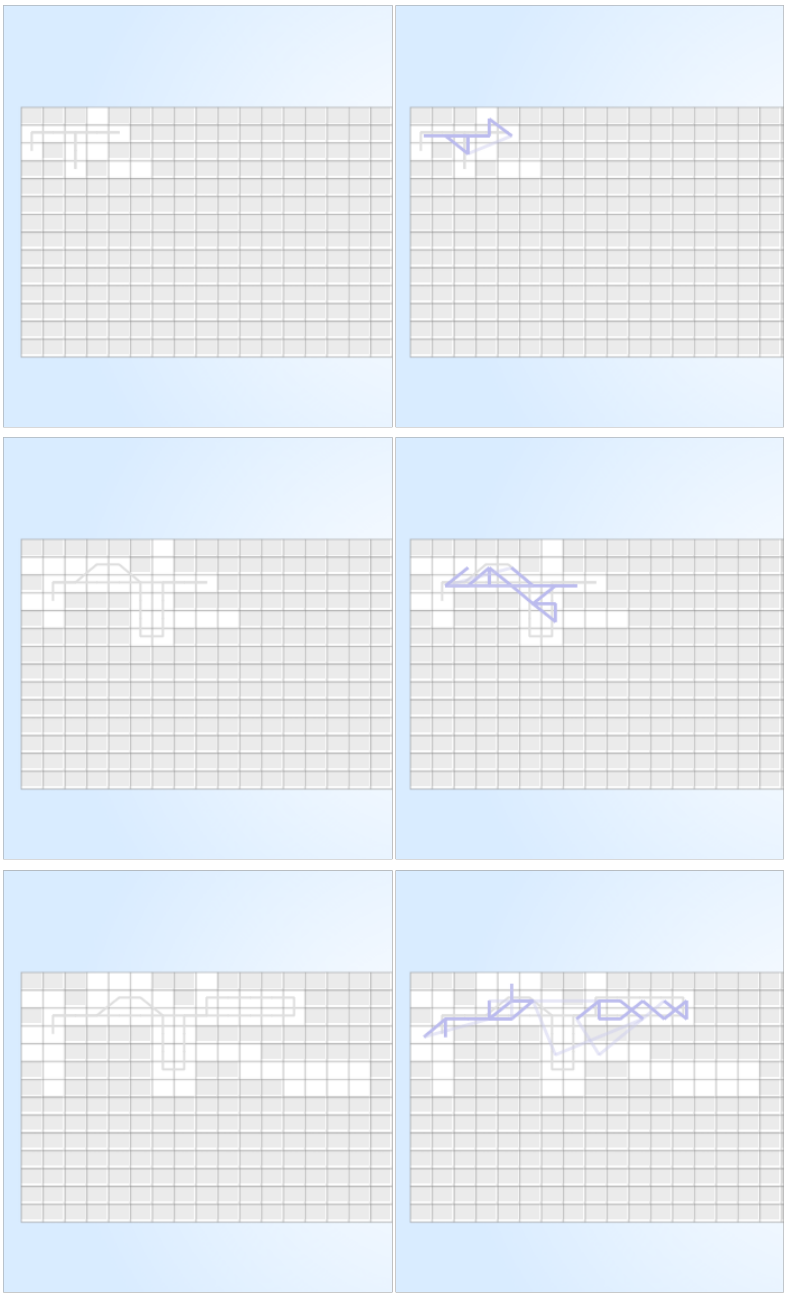
5.1.3 Valori di performance

Per ognuno dei percorsi studiati sono stati estratti alcuni valori di performance:

- numero di posizioni correttamente individuate;
- distanza media dell'errore di posizionamento (calcolata tenendo conto dell'ampiezza dei tasselli);
- numero massimo di posizioni corrette individuate di seguito.

Indipendentemente dal numero di posizioni esatte individuate, è interessante notare come l'errore medio è mediamente di pochi metri. Numerosi sono i casi di *near-miss* nel quinto percorso, che non pregiudicano comunque il contenimento dell'errore a livelli soddisfacenti. Complessivamente, si denota un livello di precisione media di posizionamento che soddisfa le aspettative.

La scarsa precisione ottenuta nel primo caso di test, causata dalla poca informatività dei dati derivati dal mapping, è indice del fatto che un



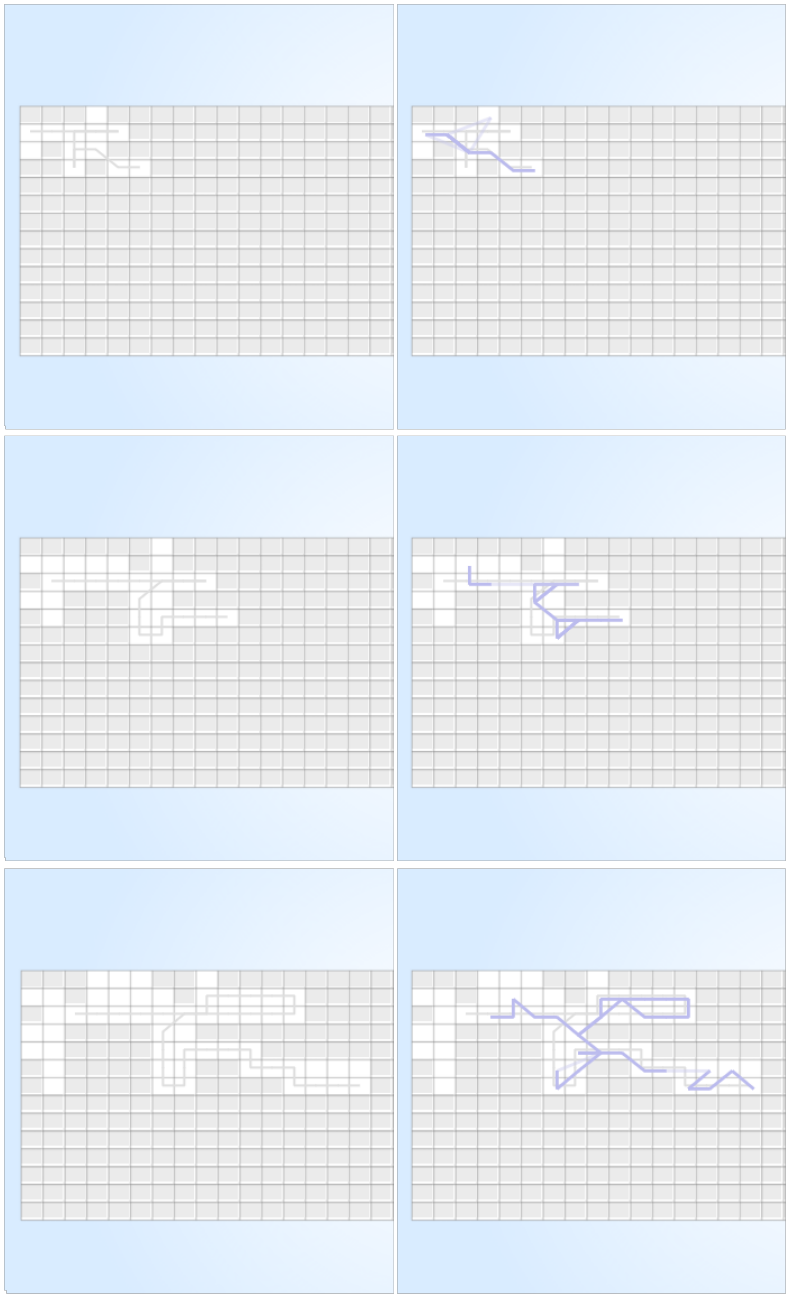


Figura 5.3: *(continua dalla pag. precedente)* Test effettuati nel terzo scenario considerato; vengono ricostruiti due percorsi con granularità crescente della mappa magnetica per tasselli rispettivamente di 5m, 3m e 1m di lato. Percorso reale a sinistra, percorso calcolato a destra.

Scenari di test		N. posizioni corrette	Distanza media dell'errore	N. posizioni corrette di seguito
Scenario 1	Percorso 1	3/18	16,93m	1
	Percorso 2	15/18	1,11m	12
	Percorso 3	11/18	2,52m	9
	Percorso 4	3/10	7,22m	2
Scenario 2		22/67	6,83m	9
Scenario 3	Percorso 1 (5m)	6/14	4,31m	4
	Percorso 1 (3m)	8/22	4,68m	3
	Percorso 1 (1m)	13/32	1,38m	5
	Percorso 2 (5m)	5/13	4,96m	4
	Percorso 2 (3m)	7/18	3,46m	5
	Percorso 2 (1m)	17/35	0,60m	9

Tabella 5.1: Valori di performance ottenuti per i tre scenari visti.

sistema di localizzazione di questo tipo potrebbe rivelarsi non adatto a tutti i tipi di ambiente; potrebbe infatti essere soggetto a imprecisioni troppo elevate all'interno di strutture in cui si rileva una variabilità troppo bassa dell'informazione magnetica.

Si nota come vi siano miglioramenti sostanziali utilizzando mappe magnetiche con misurazioni più ravvicinate. Stando ai test effettuati nel terzo scenario studiato, l'accuratezza dell'applicazione migliora in maniera evidente con tasselli di lato 1m, raggiungendo valori di errore irrisori. Si vedano la tabella 5.1 e i grafici dell'errore per iterazioni successive (figura 5.4) e dell'errore medio (figura 5.5). Ciò è un buon indicatore del fatto che si possa migliorare notevolmente l'efficienza di un sistema di posizionamento di questo tipo al costo di effettuare un mapping più meticoloso, impiegando naturalmente più tempo nel processo.

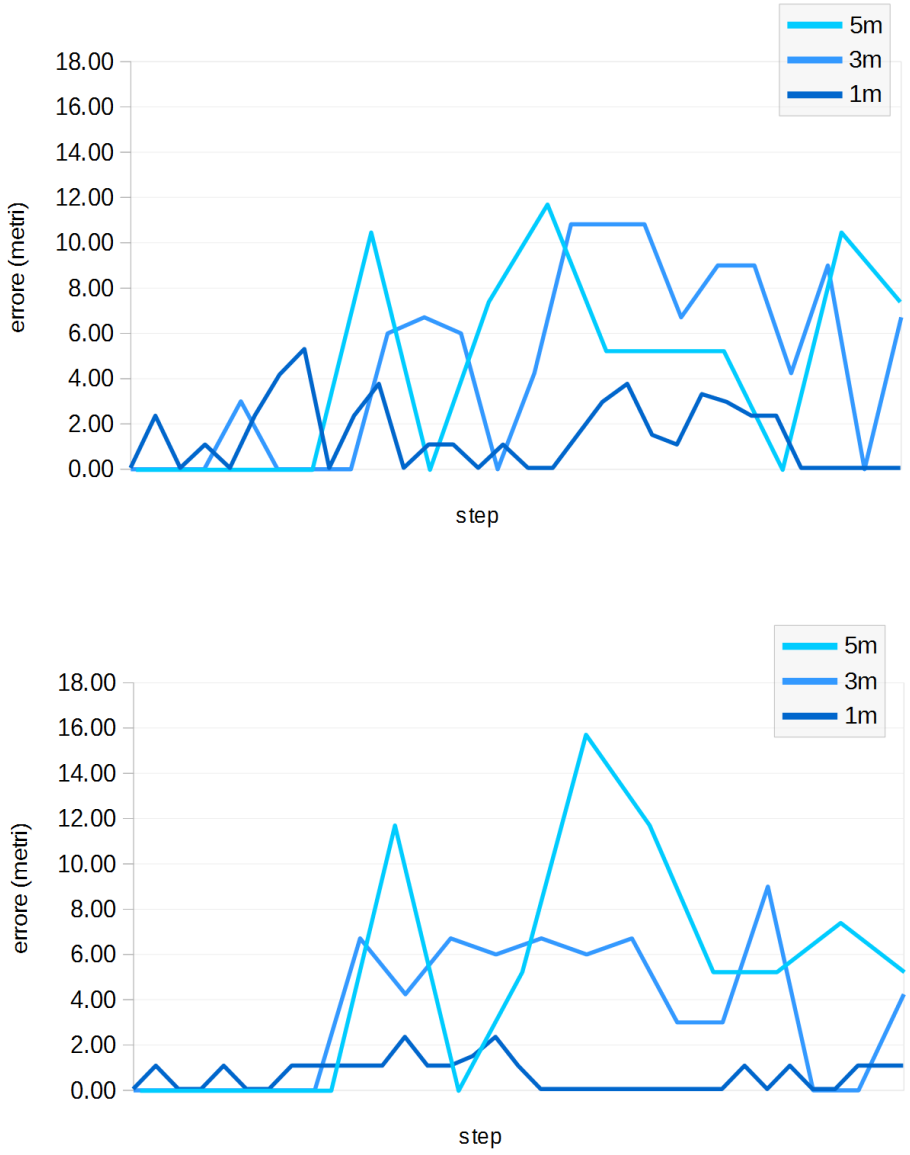


Figura 5.4: Errore in metri per iterazioni successive per i due percorsi visti (terzo scenario di test). L'errore medio viene contenuto in maniera particolarmente efficace per mappature di tasselli di lato 1m.

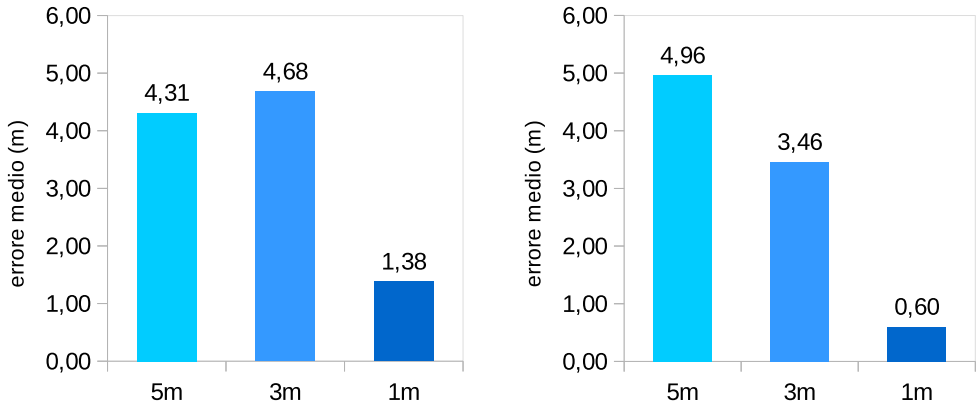


Figura 5.5: Errore medio di posizionamento a diversi livelli di precisione del fingerprint per i due percorsi visti (terzo scenario di test).

5.2 Confronto con IndoorAtlas

Si è deciso di mettere a confronto i risultati di performance del prototipo realizzato con quelli ottenibili da un sistema di localizzazione già esistente basato su magnetismo, al fine di valutarne l'efficacia complessiva rispetto allo stato dell'arte in materia. La scelta è ricaduta sull'API fornita da IndoorAtlas (2.2.3), il primo esempio di realizzazione di un simile sistema disponibile globalmente sul mercato e utilizzabile da sviluppatori di tutto il mondo. Si è preso in esame uno studio dell'API di IndoorAtlas [18] effettuato nell'ambito del dipartimento di Informatica dell'Università di Bologna; si tratta dello stesso scenario in cui è stata effettuata la terza serie di test sopra descritta, a diversi livelli di granularità di mapping.

L'applicazione adotta una metodologia di mappatura basata su principi di funzionamento diversi, più complessi della semplice mappatura di posizioni equidistanti. La procedura di mapping consiste nell'individuazione di *waypoint*, punti cardine della struttura presso i quali effettuare le rilevazioni; successivamente, l'utente è invitato a muoversi in maniera naturale lungo le aree percorribili della struttura interessata, mentre l'applicazione registra i valori del magnetismo, in congiunzione con i dati ottenuti da ac-

celerometro e giroscopio e opzionalmente il segnale Wi-fi, qualora venga attraversato uno di questi waypoint; viene poi automaticamente generata una mappa geomagnetica a partire dalle informazioni rilevate nei diversi punti di osservazione.

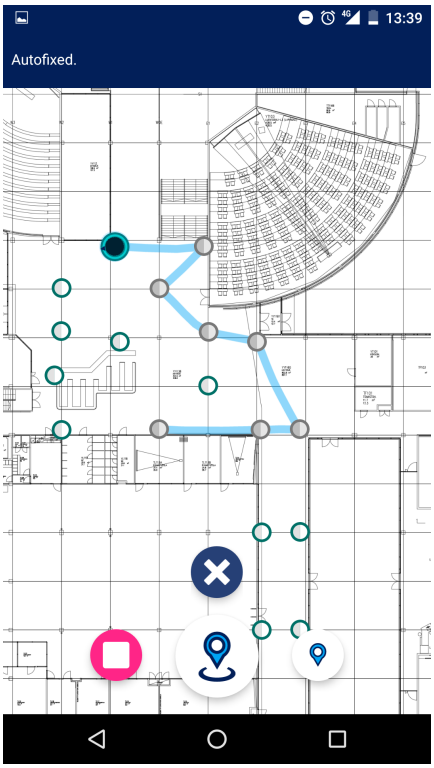


Figura 5.6: Procedura di mapping dell'applicazione *MapCreator* di IndoorAtlas. La fase di mapping consiste nell'individuazione di diversi waypoint e nel loro successivo attraversamento in ordine casuale.

Ciononostante, a livello informativo le due metodologie di mapping sono comunque paragonabili; infatti, nello studio considerato sono state prese in esame diverse mappature, una delle quali presenta esattamente lo stesso numero di *waypoint* pari al numero di tasselli del fingerprint realizzato per i test a più bassa risoluzione, con tasselli di 5m di lato, tra quelli riportati nella sezione 5.1.1; in entrambi i casi, 12 sono le rilevazioni effettuate. È dunque

lecito supporre che le due mappature presentino una quantità di informazione del tutto comparabile.

Andiamo dunque a paragonare i risultati ottenuti dall'utilizzo di entrambi i sistemi.

	6 rilevazioni	9 rilevazioni	**12 rilevazioni**	27 rilevazioni	54 rilevazioni
IndoorAtlas	4.87m	4.78m	**3.76m**	–	–
App realizzata	–	–	**4.63m**	4.07m	0.99m

Tabella 5.2: Confronto con IndoorAtlas: errore medio di localizzazione a confronto per i test effettuati nel contesto del dipartimento di Informatica di Bologna (dati su IndoorAtlas ottenuti da uno studio precedente [18]).

Stando ai risultati dello studio su IndoorAtlas, l'entità dell'errore di localizzazione in metri è migliore di quella ottenuta tramite il prototipo qui realizzato, ma tutto sommato paragonabile. Considerando l'apporto di diverse fonti di dati per la localizzazione finale nel caso di IndoorAtlas, nonché il più elaborato metodo di mappatura, è del tutto giustificato il maggiore livello di precisione ottenuto; ci è però possibile notare come tramite un approccio basato unicamente sull'utilizzo dell'informazione magnetica l'errore medio è comunque contenibile entro livelli accettabili.

Relativamente alla fase di mapping è inoltre importante notare che, nel caso dell'API di IndoorAtlas, i molteplici accorgimenti adottati, come l'applicazione di meccaniche di interpolazione, l'accesso a fonti di dati alternative, la possibilità di integrare informazioni provenienti da molteplici dispositivi e la continua interazione con l'utente per individuare quali aree è necessario attraversare, facilitano e velocizzano il processo di creazione della mappa geomagnetica, permettendo di raggiungere una buona precisione di posizionamento anche a partire da poche rilevazioni. Il sistema implementato nell'ambito di questa tesi, più essenziale e conseguentemente di più bassa scalabilità,

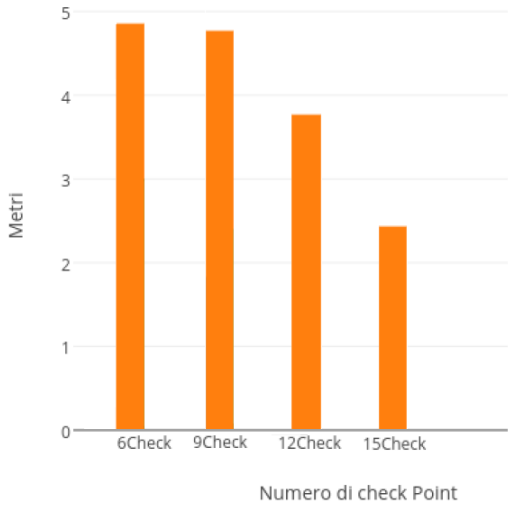


Figura 5.7: Errore medio di localizzazione dell'API IndoorAtlas sul terzo scenario visto in questo capitolo (da uno studio realizzato precedentemente [18]).

necessita di un'attività di mapping più estesa. La creazione della mappa magnetica di un interno di grandi dimensioni, dell'ordine delle decine di migliaia di m^2, può infatti richiedere diverse ore, come si è potuto constatare durante il testing. Un simile approccio, pur sempre consentendo di ridurre l'errore a livelli più bassi al costo di uno sforzo maggiore iniziale, va irrimediabilmente a discapito dell'usabilità, aumentando i tempi di *maintenance*. In uno scenario di applicazione reale, infatti, in seguito a modifiche strutturali o di arredo potrebbe essere necessario ottenere un nuovo fingerprint dell'ambiente interessato; nel caso di strutture pubbliche come aeroporti o stazioni potrebbe non essere sempre possibile effettuare attività di mapping prolungate.

D'altra parte, l'impiego di un sistema più essenziale come quello realizzato presenta almeno due importanti vantaggi.

In primo luogo, l'applicazione sviluppata offre funzioni di mapping semplici, non condizionate dalla fonte di dati di riferimento, mentre è la computazione attuata durante il locating ad adottare criteri di complessità maggiore. Le soluzioni di pattern matching individuate potrebbero dunque essere applicate a tecnologie diverse dal magnetismo in maniera trasversale. Al contrario, il meccanismo di raccolta dei dati non necessita di un apporto originale da parte dell'utente, a tal punto da essere potenzialmente automatizzabile ottenendo comunque risultati coerenti. Le dinamiche di localizzazione, insomma, intervengono solo in un secondo momento, e, come è stato osservato, la precisione finale non è compromessa. Perché allora optare per una soluzione di questo tipo? Ad esempio ai fini dell'integrazione con tecniche di Simultaneous Location and Mapping, di cui si parlerà più approfonditamente in seguito (6.2), e in generale qualora si abbia la necessità di separare le due fasi della localizzazione per agire su ciascuna di esse in maniera indipendente.

Inoltre, l'impiego del segnale Wi-Fi come fonte di dati alternativa da parte

	IndoorAtlas	**Applicazione realizzata**
Metodo di mapping	Elaborato e specifico; ha grande influenza sulla precisione finale	Più generico; indipendente dalle dinamiche di localizzazione
Scalabilità	Indefinitamente scalabile	Indefinitamente scalabile, ma il mapping può richiedere tempi elevati
Tempi di maintenance	Ridotti	Fortemente dipendenti dalla dimensione dell'interno
Costi di deployment	Minimi; non trascurabili se si utilizza il Wi-Fi	Nulli

Tabella 5.3: Un confronto finale con IndoorAtlas che mette in luce le differenze principali nei due approcci al fingerprinting.

di IndoorAtlas presuppone comunque una copertura sufficiente dell'interno interessato tramite diversi access point, i cui costi di installazione non sono sempre trascurabili. Al contrario, i valori di performance risultanti dai test del prototipo qui realizzato sono il risultato di un semplice confronto tra dati rilevati in momenti successivi da dispositivo mobile a dispositivo mobile, e quindi in maniera totalmente indipendente da un qualsiasi tipo di infrastruttura fissa o rete di comunicazione. E, come già menzionato, sono molteplici i vantaggi derivanti dall'uso di una tecnologia dai costi di deployment essenzialmente nulli.

Capitolo 6

Sviluppi futuri

I risultati ottenuti durante il testing del prototipo sviluppato indicano come la realizzazione di un sistema per la localizzazione indoors che sfrutti la tecnologia studiata sia del tutto possibile, ma non esente da difficoltà. Alcuni aspetti implementativi sono indubbiamente migliorabili; possono inoltre essere sperimentate altre tecniche di pattern matching più sofisticate o di natura differente da quelle qui esplorate.

Si andrà ora a descrivere quali aspetti presentano maggiore possibilità di sviluppi futuri per il raggiungimento di risultati più precisi o a fini di integrazione di questa tecnologia all'interno di un contesto più ampio.

6.1 Pattern matching: aspetti migliorabili

L'algoritmo che si è elaborato nell'ambito di questa tesi presenta indubbiamente aspetti migliorabili. Tra questi elenchiamo quelli che sono risultati più evidenti durante la fase implementativa.

La procedura di controllo e repositioning descritta alla sezione 4.2.4 potrebbe essere effettuata con modalità differenti. Ad esempio, si potrebbe individuare un'area attorno alla quale si ha concentrazione massima dei valori di probabilità, per poi applicare una ridistribuzione dei valori su un raggio

più ampio. Diversi sono i criteri applicabili e la loro esplorazione esaustiva può risultare complessa, ma di sicuro interesse.

In alternativa, è possibile tenere traccia di ipotetici percorsi alternativi durante iterazioni successive per un eventuale riposizionamento. Più in generale, un simile sistema potrebbe essere esteso all'intero algoritmo: sarebbe infatti possibile considerare una storia recente dei valori delle probabilità dei singoli tasselli, dunque non solo considerando lo stato immediatamente precedente, ma anche le diverse iterazioni passate, individuando le sequenze alternative di posizioni successive per le quali il matching con i valori della mappa geomagnetica è più efficace; risulterebbe poi semplice, in caso di un posizionamento errato, risalire ai percorsi alternativi di cui si è tenuta traccia e selezionarne un altro. A livello teorico, una tale logica, non più descrivibile con un modello di Markov, ma che ricalca modelli probabilistici più complessi, è sicuramente applicabile, a patto di definire criteri validi per il riconoscimento delle traiettorie possibili, il che richiederebbe sperimentazioni più approfondite.

6.2 Sensor fusion

Alla luce dei test effettuati, si può affermare che il maggior punto debole della localizzazione tramite magnetismo risiede nel fatto che a partire da rilevazioni isolate si può ottenere informazione di posizionamento insufficiente o comunque altamente imprecisa. Più grande è l'ambiente, meno rilevanza assume il singolo dato. Se ad esempio non si ha alcun tipo di informazione iniziale di posizione, una prima localizzazione risulta piuttosto ardua. Sperimentalmente si è osservato che occorrono diverse iterazioni dell'algoritmo perché la posizione calcolata si avvicini a quella reale; la performance dell'algoritmo è notevolmente migliore quando invece si tratta di mantenere la posizione corretta durante gli spostamenti successivi a un primo posizionamento sufficientemente preciso. Per ovviare al problema si potrebbe per esempio dedurre la posizione iniziale se si suppone di sapere quali tasselli

costituiscono gli ingressi dell'interno dove si utilizza l'applicazione e confrontando i valori del magnetismo per tali punti, approccio che è però troppo limitante in quanto l'utente potrebbe avviare l'applicazione di localizzazione in un punto qualsiasi dell'edificio.

Si è anche visto come l'attraversamento di zone "neutre" con bassa variabilità dei valori del magnetismo può irrimediabilmente compromettere il posizionamento e di come gli ambienti studiati non siano esenti da aree di questo tipo. In questi casi, la possibilità di usufruire di metodi alternativi di posizionamento può rivelarsi determinante.

L'utilizzo di un approccio che fa uso di sensor fusion può aiutare a superare queste e altre limitazioni. Sarebbe interessante sperimentare l'efficacia dell'algoritmo elaborato in congiunzione con altri metodi di geolocalizzazione basati su fingerprinting. Una prima localizzazione approssimata potrebbe avvenire ad esempio tramite segnale Wi-Fi, per poi impiegare i dati del magnetometro in un secondo momento. Questo approccio, mirato a sfruttare i punti di forza di diverse tecniche di fingerprinting, presenta l'ulteriore vantaggio di necessitare di un'unica attività di mapping durante la quale possono essere compiute molteplici rilevazioni contemporaneamente per uno stesso punto, come ad esempio, appunto, il segnale Wi-Fi e il campo magnetico.

Un'altra possibile integrazione potrebbe essere costituita dall'individuazione dell'orientamento del dispositivo tramite giroscopio durante lo spostamento, come ulteriore metodologia di riconoscimento della traiettoria compiuta.

Molteplici sono poi le opzioni esplorabili considerando rilevazioni da altri tipi di sensori, come ad esempio la possibilità di effettuare il mapping del geomagnetismo tramite dati ottenuti da diversi utenti i cui dispositivi dispongono di metodologie di localizzazione alternative. Essendo in grado di conoscere la posizione grazie ad altre tecniche, sarebbe sufficiente inviare i dati di posizione abbinati alla rilevazione del campo magnetico a un server centralizzato, potendo così ricostruire una mappa geomagnetica grazie all'apporto di diversi utenti, piuttosto che in seguito a una raccolta manua-

le dei dati che, come è stato riscontrato durante il testing dell'applicazione, può comportare grande dispendio di tempo se si vuole mantenere l'errore di posizionamento a livelli minimi. Tecniche di Simultaneous Location and Mapping (*SLAM*), che permettono di aggiornare le mappe magnetiche pre-esistenti durante la localizzazione di comuni utenti, sono già state esplorate per altri sistemi di posizionamento e in futuro potrebbero essere impiegate anche in questo ambito [19]. Si tratta di un ulteriore aspetto che potrebbe garantire una rapida diffusione di questa tecnologia.

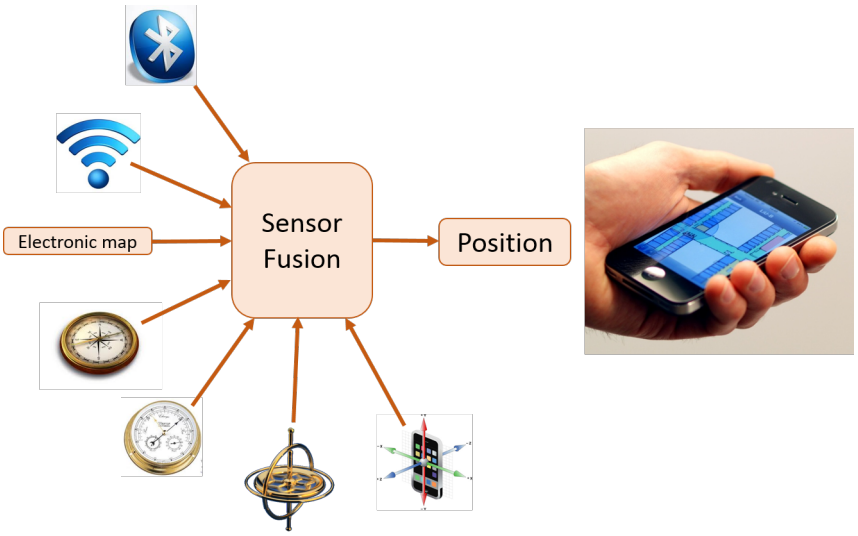


Figura 6.1: Un approccio alla localizzazione basato su *sensor fusion* può fare uso di diverse fonti di dati, quali segnale Bluetooth, Wi-Fi, orientamento del dispositivo (tramite giroscopio e accelerometro), livello di illuminazione, dati inerziali.

6.3 Cloud

Alle tecniche di localizzazione indoor già esistenti o in via di sviluppo corrispondono numerosi ambiti di immediata applicazione nella vita di tutti i giorni, nonché di integrazione con le reti di informazione già esistenti, per

esempio con finalità di crowdsensing, geo-advertising/geo-marketing, servizi di emergenza.

Molteplici sono le possibilità applicative in connessione con il mondo dell'Internet of Things. La possibilità di localizzare con precisione gli smartphone presenti all'interno di un edificio può consentire, ad esempio, di contare le persone presenti in stanze diverse, attivando conseguentemente servizi di diverso tipo quali l'attivazione dell'illuminazione o dell'impianto di riscaldamento, o disponendo l'apertura o la chiusura di porte automatizzate, o ancora di offrire una password Wi-Fi solo alle persone effettivamente presenti all'interno di un locale. Questi sono solo alcuni dei tanti esempi di impiego pratico delle informazioni di posizione ottenibili da diversi utenti in ambito *IoT*.

Si tratta fondamentalmente di campi applicativi inerentemente legati al web e che prevedono la presenza di una piattaforma che raccoglie i dati da diversi utenti e li utilizza collettivamente.

Web server

L'avanzato livello di sviluppo delle web technologies e il raggiungimento di standard adeguati per lo scambio di dati nell'ambito della comunicazione internet tra dispositivi diversi ci offre la possibilità di raccogliere i dati di posizione da più utenti con minimo sforzo implementativo.

In seguito all'installazione del prototipo di applicazione realizzato nell'ambito di questa tesi su diversi dispositivi, si è deciso di creare una semplice applicazione web che raccolga i dati di posizione da diversi utenti per mostrarli in un'interfaccia unificata. Si è realizzato un web server che comunica con l'applicazione realizzata e che tiene traccia di più dispositivi, mostrandone gli spostamenti successivi.

La comunicazione dai dispositivi al server di appoggio è basata su semplici chiamate POST HTTP per lo scambio dei dati di posizione, in abbinamento a un ID unico di dispositivo. I dati vengono scambiati in formato JSON e vengono ricevuti e salvati all'interno di un file locale al server, insieme a

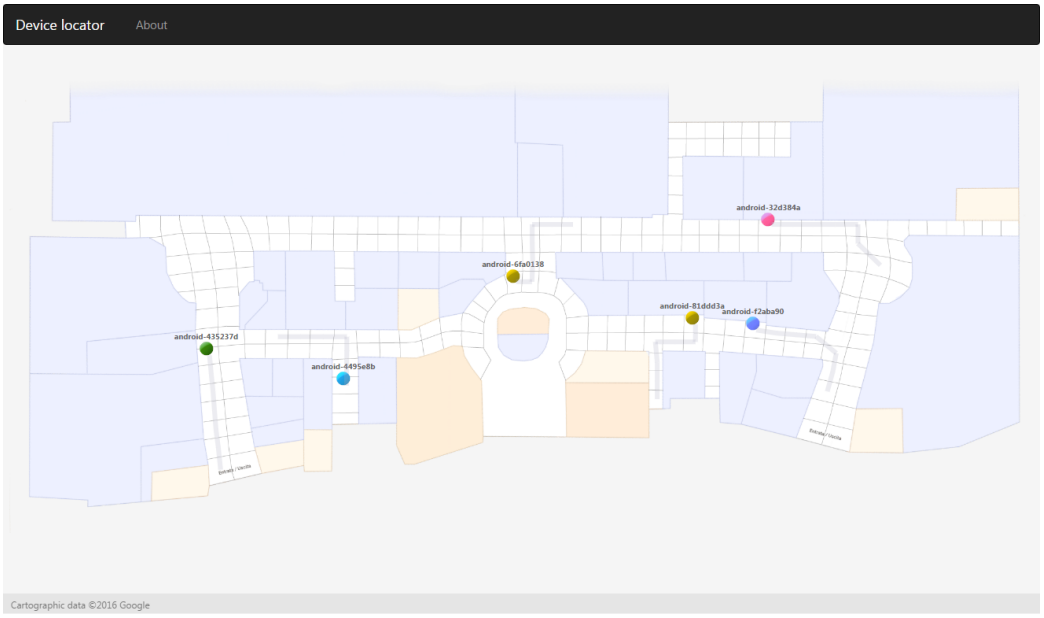


Figura 6.2: Il web server realizzato raccoglie le informazioni di posizione da diversi utenti e le mostra in un'interfaccia unificata.

un timestamp che permette di gestire i dati provenienti da diversi utenti per sincronizzarne l'invio al client ed aggiornare o eliminare le informazioni marcate come *outdated*. Su lato client viene poi effettuato il polling dei dati a intervalli regolari e le posizioni vengono infine mostrate all'utente, avendo cura di riconoscere spostamenti dello stesso dispositivo e di ricostruirne la traiettoria.

La relativa semplicità di implementazione di un meccanismo per disporre dei dati provenienti da una molteplicità di utenti è un'ulteriore conferma dell'immediatezza applicativa di un sistema di posizionamento come quello realizzato.

Conclusioni

La natura della localizzazione indoor è notevolmente più complessa di quella outdoor e non riconducibile a uno standard predefinito né ad un'unica tecnologia. Studi recenti hanno dimostrato l'efficacia di un approccio basato sulla fusione di tecniche alternative che fanno uso dei dati rilevabili dai sensori di un comune smartphone. Il raggiungimento di uno standard per la localizzazione indoor tramite dispositivi mobili interesserebbe un ampio settore di marketing con infinite possibilità di applicazione immediata: geoadvertising, personal tracking e crowdsensing sono solo alcuni esempi. Le più recenti sperimentazioni di tecniche di wayfinding in luoghi chiusi fanno presagire uno sviluppo importante in questo senso in un prossimo futuro.

La localizzazione tramite magnetismo presenta alcuni notevoli vantaggi rispetto alle altre metodologie applicate in questo ambito. Non è necessaria alcuna infrastruttura esterna, e dal momento che praticamente ogni nuovo modello di smartphone prodotto ai giorni nostri è dotato di magnetometro, il sistema è di diretta applicazione a scenari reali, senza costi di installazione e, oltretutto, senza la necessità di una rete internet di supporto.

Nell'ambito di questa tesi, si è realizzata un'applicazione per smartphone Android per la localizzazione tramite magnetismo in grado di ricostruire la traiettoria di un dispositivo tramite una tecnica di fingerprinting. Particolare attenzione è stata data alla fase implementativa dell'algoritmo di localizzazione, che fa uso di tecniche di pattern matching e di approssimazione della traiettoria reale la cui applicazione ha contribuito al miglioramento della performance complessiva del sistema.

L'applicazione è stata testata in diversi tipi di ambienti quali stazioni e centri commerciali. I risultati si sono rivelati incoraggianti; l'algoritmo elaborato è in grado di approssimare il percorso dell'utente con buona efficacia, mantenendo l'errore medio a livelli contenuti. Gli esiti ottenuti sono comparabili a quelli di sistemi reali già studiati, come osservato effettuando il confronto con ricerche precedenti. Il sistema è facilmente integrabile con altre tecnologie per l'utilizzo centralizzato dei dati di posizione da più dispositivi in un contesto più ampio. Tuttavia il sistema non è esente da problemi di scalabilità, di affidabilità del risultato finale e di adattabilità a un qualsiasi tipo di ambiente. Si sono compresi gli aspetti più vantaggiosi e i limiti principali di una tale tecnica, esplorando possibili sviluppi che sfruttano approcci differenti per ovviare alle problematiche presentate. Se le future soluzioni software saranno in grado di integrare questa ed altre tecnologie in maniera efficace, il magnetismo potrebbe presto emergere tra le tecniche primariamente applicabili in un contesto indoor.

Appendice

Codice dell'applicazione

A.1 Assegnazione delle probabilità

```
1     public void locate(float curr_mag) {
2         if (totalProbability()==0f) { first_measurement = true; }
3         probability_old = probability;
4         probability = initProbability(tileAmount);
5
6         // Find what the current measured variation is
7         if (old_mag == 0f) old_mag = curr_mag;
8         int variation = 0;
9         if (curr_mag >= old_mag + fpr_step)
10            variation = 1;
11        else if (curr_mag <= old_mag - fpr_step)
12            variation = -1;
13        else
14            variation = 0;
15
16        // For each tile, compute the sum of probabilities from adjacent
              tiles with the same variation
17        float sum_in_all = 0f;
18        float[] sum_in = new float[tileAmount];
19        for (int i = 0; i < tileAmount; i++) {
20            sum_in[i] =  sum_in(i, variation);
21            sum_in_all += sum_in[i];
22        }
23
24        // Assign relative probability
25        float total_probability = 0f;
26        for (int i = 0; i < tileAmount; i++) {
27            if (Math.abs(curr_mag - tileMagnitude(i)) > maxbias) continue;
28            if (first_measurement)
29                probability[i] = ( (maxbias - Math.abs(curr_mag -
                      tileMagnitude(i))) / maxbias ) ;
30            else {
31                // take into account older measurements
32                float in_tiles = (sum_in[i] / sum_in_all); // Value within
                       0-1 range
33                probability[i] =
```

```
34                      (float) (Math.pow(( (maxbias - Math.abs(curr_mag -
                            tileMagnitude(i))) / maxbias ),(1f/
                        newval_amount))) *
35                          (float) (Math.pow(in_tiles,(1f/
                                oldval_amount)));
36              }
37              total_probability += probability[i];
38          }
39
40          // Find the tiles with maximum probability
41          int maxtile = 0;
42          float maxprob = 0f;
43          ArrayList maxtiles = new ArrayList(); // in descending order
44          maxtiles.add(0.00f);
45          for (int i = 0; i < tileAmount; i++) {
46              // maximum
47              if (probability[i] > maxprob) {
48                  maxtile = i; maxprob = probability[i];
49              }
50              // other tiles with the highest probability
51              for (int j = 0; j < maxtiles.size(); j++) {
52                  if (probability[i] > (float) maxtiles.get(j)) {
53                      maxtiles.add(j,probability[i]);
54                      Collections.sort(maxtiles, Collections.reverseOrder())
                            ;
55                      if (maxtiles.size() > historySize) maxtiles.remove(
                            historySize);
56                      break;
57                  }
58              }
59          }
60
61          // Kill the tiles whose probability is just too low
62          int pos = queue.get(0);
63          if (maxtiles.size() == historySize)
64              for (int i = 0; i < tileAmount; i++) {
65                  float min = (float) Collections.min(maxtiles);
66                  if (probability[i] < min) {
67                      if (just_left_path && is_adjacent(i, pos)) {
68                          // yet, tolerant with casual wrong readings: keep
                                the tiles near last predicted position
69                          total_probability -= probability[i] * (1f-
                                tolerance);
70                          probability[i] *= tolerance;
71                          // ...unless FPR rules them out
72                          if (transition[pos][i] != variation) {
73                              total_probability -= probability[i] * (1-(1f/
                                    fpr_incidence));
74                              probability[i] *= (1f/fpr_incidence);
75                          }
76                      } else {
77                          total_probability -= probability[i];
78                          probability[i] = 0f;
79                      }
80                  }
81              }
82
83          // Decide whether we're in need to discriminate between multiple
                probable paths
84          boolean consistent_path = (queue.size() == queuesize); // we're
                consistently following a path if there've been enough
                consecutive tiles
```

```
85          boolean multiple_ways_up_ahead = false;
86          int count_paths = 0; // we're going to count how many probable
                paths are up ahead
87          ArrayList<Integer> list_of_paths = new ArrayList<Integer>();
88          for (int i = 0; i < tileAmount; i++) {
89              if ((i != pos) && (is_adjacent(i,pos)) && probability[i] != 0f
                    ) {
90                  count_paths++;
91                  list_of_paths.add(i);
92              }
93          }
94          if (count_paths > 2) multiple_ways_up_ahead = true;
95          else if (count_paths == 2) { // find out whether they're going in
                opposite directions
96              int first = list_of_paths.remove(0);
97              int second = list_of_paths.remove(0);
98              if (!is_adjacent(first, second)) // then they're opposite
99                  multiple_ways_up_ahead = true;
100         }
101
102         // When in doubt between paths, favour those tiles whose
                measurement is the first in a while, discouraging redundancy
                between tiles
103         if (consistent_path && multiple_ways_up_ahead) {
104             // Calculate the redundancy of nearby tiles for potential
                    tiles
105             int redundancy_sum_in[] = new int[tileAmount]; Arrays.fill(
                    redundancy_sum_in,0);
106             for (int tile1 = 0; tile1 < tileAmount; tile1++) {
107                 if (is_adjacent(tile1, pos) && (probability[tile1] != 0f))
                        { // this is a potential tile...
108                     // Calculate the redundancy of its nearby tiles
109                     for (int tile2 = 0; tile2 < tileAmount; tile2++) {
110                         if ((tile1 != tile2) && is_adjacent(tile1, tile2))
                                {
111                             if (redundancy[tile2] == 0)
112                                 redundancy_sum_in[tile1] += 5;
113                             else
114                                 redundancy_sum_in[tile1] += redundancy[
                                        tile2];
115                         }
116                     }
117                     // Apply this influence on its probability: less often
                            hit, more probable
118                     total_probability += probability[tile1] * (float) Math
                            .pow(redundancy_sum_in[tile1],redundancy_incidence
                            ) - probability[tile1];
119                     probability[tile1] = probability[tile1] * (float) Math
                            .pow(redundancy_sum_in[tile1],redundancy_incidence
                            );
120                 }
121             }
122         }
123
124         // Multiply each probability value so that the maximum probability
                is assigned
125         float ratio = 100f / total_probability;
126         for (int i = 0; i < tileAmount; i++) {
127             probability[i] = probability[i] * ratio;
128         }
129         if (first_measurement) first_measurement = false;
130     }
```

A.2 Applicazione del filtro di fuzzy pattern

```
1       // Create the transition matrix
2       private void populateTransitionMatrix() {
3           transition = new int[tileAmount][tileAmount];
4           for (int tile1 = 0; tile1 < tileAmount; tile1++){
5               int tile1x = tileX(tile1);
6               int tile1y = tileY(tile1);
7               for (int tile2 = 0; tile2 < tileAmount; tile2++) {
8                   int tile2x = tileX(tile2);
9                   int tile2y = tileY(tile2);
10                  if (is_adj(tile1x, tile1y, tile2x, tile2y)) {
11                      if (tileMagnitude[tile1x][tile1y] >= tileMagnitude[
                            tile2x][tile2y] + fpr_step)
12                          transition[tile1][tile2] = -1;
13                      else if (tileMagnitude[tile1x][tile1y] <=
                            tileMagnitude[tile2x][tile2y] - fpr_step)
14                          transition[tile1][tile2] = +1;
15                      else
16                          transition[tile1][tile2] = 0;
17                  }
18              }
19          }
20      }
21
22      // Determine the influence of each adjacent tile based on transition
            values
23      private float sum_in(int tile, float variation) {
24          float sum = 0;
25          int tile1x = tileX(tile);
26          int tile1y = tileY(tile);
27          for (int tile2 = 0; tile2 < probability.length; tile2++) {
28              int tile2x = tileX(tile2);
29              int tile2y = tileY(tile2);
30              if ((tile1x == tile2x) && (tile1y == tile2y)) continue;
31              if ((tile2x == -1) || (tile2y == -1)) continue;
32              if (tileIndex[tile2x][tile2y] == 0) continue;
33              if (is_adj(tile1x, tile1y, tile2x, tile2y)) {
34                  if (transition[tile2][tile] == variation)
35                      sum += probability_old[tile2];
36                  else sum += probability_old[tile2]/fpr_incidence;
37              }
38          }
39          return sum;
40      }
```

A.3 Scelta della posizione

```
1     public void predictPosition() {
2         int pos = queue.get(0); // last predicted position
3         // We're consistently following a path if there've been enough
              consecutive tiles
4         boolean consistent_path = (queue.size() == queuesize);
5         // Select the most probable tile
6         int best = -1;
7         float max_prob = 0f;
8         for (int i = 0; i < tileAmount; i++) {
9             if (probability[i] > max_prob) {
10                best = i;
11                max_prob = probability[i];
12            }
13        }
14        // Select the most probable tile adjacent to the current position,
               if any
15        int near = -1;
16        float max_prob_near = 0f;
17        for (int i = 0; i < tileAmount; i++) {
18            if ((i != pos) && (is_adjacent(i,pos)) && (probability[i] >
                  max_prob_near)) {
19                near = i;
20                max_prob_near = probability[i];
21            }
22        }
23        // Update the queue holding the last few positions
24        if (just_left_path) just_left_path = false;
25        if (consistent_path && (near != -1) && (probability[near] >
              prob_threshold)) {
26            // Follow the path and pick the most probable adjacent tile
27            queue.add(0,near);
28            if (queue.size() > queuesize) queue.remove(queuesize);
29        } else if ((pos != best) && is_adjacent(pos,best)) {
30            // The most probable tile is adjacent to the previous position
31            queue.add(0,best);
32            if (queue.size() > queuesize) queue.remove(queuesize);
33        }
34        else if ((near != -1) && (distance(near,best) > 2) && (Math.abs(
              probability[near]-probability[best]) < 1f)) {
35            // If the most probable tile and the nearest probable are
                  really close, choose the nearest
36            queue.add(0,near);
37            if (queue.size() > queuesize) queue.remove(queuesize);
38        }
39        else if (best != -1) {
40            // Abandon this path and pick the most probable tile
41            queue.clear();
42            queue.add(best);
43            if (!just_left_path) just_left_path = true;
44        } else {
45            // There is no probability left, due to extremely bad readings
46            queue.clear();
47            return;
48        }
49        // Lastly, update the distanceValues array with the newest
              position value
50        updateDistanceValues(queue.get(0));
51    }
```

A.4 Controllo e riposizionamento

```
1     public boolean evaluatePosition(float curr_mag) {
2         /* This function is only ran once every 10 steps in case extremely
               wrong readings occurred */
3         if (!( (distanceValues.size() == 10 && mediumDistanceValue() > 17f
              ) || (queue.size() == 0) ))
4             return true;
5
6         // Else, recalculate the predicted position
7         // Recalculate probability array, only based on new readings
8         float[] recalculatedProb = new float[probability.length];
9         float total_probability = 0f;
10        for (int i = 0; i < recalculatedProb.length; i++) {
11            if (Math.abs(curr_mag - tileMagnitude(i)) > maxbias) continue;
12            recalculatedProb[i] = ( (maxbias - Math.abs(curr_mag -
                  tileMagnitude(i))) / maxbias ) ;
13            total_probability += recalculatedProb[i];
14        }
15        float ratio = 100f / total_probability;
16        for (int i = 0; i < recalculatedProb.length; i++) {
17            recalculatedProb[i] = recalculatedProb[i] * ratio;
18        }
19
20        // Now, find the 10 tiles with most probability
21        int[] max10tiles = new int[10];
22        for (int nn = 0; nn < 10; nn++) {
23            int max = 0; float maxprob = 0f;
24            for (int i = 0; i < recalculatedProb.length; i++) {
25                if (recalculatedProb[i] > maxprob) {
26                    maxprob = recalculatedProb[i]; max = i;
27                }
28            }
29            max10tiles[nn] = max;
30            recalculatedProb[max] = 0f; // just so that it won't be
                  selected next time
31        }
32
33        // Make a random one of the 10 tiles the predicted position
34        Random r = new Random(); int rr = r.nextInt(10);
35        int tile = max10tiles[rr];
36        Arrays.fill(probability,1f);
37        probability[tile] = 50f;
38        queue.add(tile); // since it's the predicted position
39
40        return false;
```

Bibliografia

[1] Vladimir Maximov, Oleg Tabarovsky. *Survey of Accuracy Improvement Approaches for Tightly Coupled ToA/IMU Personal Indoor Navigation System.* Proceedings of International Conference on Indoor Positioning and Indoor Navigation, ottobre 2013, Montbeliard, France.

[2] Mahtab Hossain, Hien Nguyen Van, Yunye Jin, Wee-Seng Soh. *Indoor Localization using Multiple Wireless Technologies.* Department of Electrical Computer Engineering, National University of Singapore, 2007.

[3] L. Bedogni, M. Di Felice, L. Bononi. *Context-Aware Android Applications through Transportation Mode Detection Techniques*, Wiley's Wireless Communications and Mobile Computing Journal (WCMC), 2016.

[4] Jie Yang, Yingying Chen. *Indoor Localization Using Improved RSS-Based Lateration Methods.* IEEE Global Telecommunications Conference, 2009. GLOBECOM 2009: 1–6.

[5] Manikanta Kotaru, Kiran Joshi, Dinesh Bharadia, Sachin Katti. *SpotFi: Decimeter Level Localization Using WiFi.* Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication. SIGCOMM '15. New York, NY, USA: ACM: 269–282.

[6] Moustafa Youssef, Ashok Agrawala. *The Horus location determination system.* Wireless Networks. 14 (3): 357–374. 2007-01-04.

[7] Raffaele Bruno, Franca Delmastro. *Design and Analysis of a Bluetooth-based Indoor Localization System.* IIT Institute CNR 2003.

[8] Marcel Estel, Laura Fischer. *Feasibility of Bluetooth iBeacons for Indoor Localization.* Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2015.

[9] *Music City Center Unveil Wayfinding App.* http://www.nashville.gov/News-Media/News-Article/ID/3477/Mayor-Music-City-Center-Unveil-Wayfinding-App. Ultima modifica 2014/11/28.

[10] W.Storms. *Magnetic field aided indoor navigation.* BiblioScholar, novembre 2012.

[11] B. Li, T. Gallagher, A.G. Dempster, C. Rizos. *How feasible is the use of magnetic field alone for indoor positioning.* International Conference on Indoor Positioning and Indoor Navigation (IPIN), pp.1-9, novembre 2012.

[12] Dries Vandermeulen, Charles Vercauteren, Maarten Weyn. *Indoor localization Using a Magnetic Flux Density Map of a Building. Feasibility study of geomagnetic indoor localization.* Faculty of Applied Engineering-CoSys-Lab, University of Antwerp, Ambient 2013 : The Third International Conference on Ambient Computing, Applications, Services and Technologies.

[13] Janne Haverinen, Anssi Kemppainen. *Global indoor self-localization based on the ambient magnetic field.* 5th International Conference on Computational Intelligence, Robotics and Autonomous Systems, 2009.

[14] Yepeng Ni, Jianbo Liu, Shan Liu, Yaxin Bai. *An Indoor Pedestrian Positioning Method Using HMM with a Fuzzy Pattern Recognition Algorithm in a WLAN Fingerprint System.* Sensors (Basel, Switzerland), 2016.

[15] Andreas Teuber, Bernd Eissfeller. *WLAN Indoor Positioning Based on Euclidean Distances and Fuzzy Logic*. Institute of Geodesy and Navigation, University FAF, Munich, Germany, 2006.

[16] Hung-Yuan Chung, Chun-Cheng Hou, Yu-Shan Chen. *Indoor Intelligent Mobile Robot Localization Using Fuzzy Compensation and Kalman Filter to Fuse the Data of Gyroscope and Magnetometer*. IEEE Transactions on Industrial Electronics, 2015.

[17] Liat Clark. *Finnish startup can locate you indoors using magnetic field anomalies*. Wired UK, 9 luglio 2012.

[18] Davide Boldrin. *Analisi e sviluppo di un'applicazione per la localizzazione indoor*. Università di Bologna, Corso di Studio in Informatica, 2016.

[19] Greg Sterling. *Magnetic Positioning - The Arrival of 'Indoor GPS'*. Opus Research, Inc, 2014.

Ringraziamenti

I miei più sentiti ringraziamenti vanno al professor Luciano Bononi per essere stato relatore di questa tesi e al dottor Luca Bedogni per il prezioso supporto e la massima disponibilità durante la realizzazione della stessa.

Ringrazio inoltre la mia famiglia per il sostegno ricevuto durante il percorso che mi hanno permesso di intraprendere, e per tutto il resto.