UNIVERSITA' DEGLI STUDI DI CAGLIARI

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI

Corso di Laurea Specialistica in Tecnologie Informatiche

Un sistema di posizionamento indoor basato su smartphone

Docente di riferimento

Prof. Andrea Bosin

Candidato:

Alberto Serra

Matr. N. 41860

ANNO ACCADEMICO 2009-2010

Indice Generale

Indice delle figure

1 Introduzione

Il progresso tecnologico ha permesso in questi ultimi anni di realizzare calcolatori sempre più complessi e sempre più piccoli di dimensione. La diffusione su larga scala di telefoni cellulari e palmari ha portato alla nascita di un nuovo tipo di dispositivo: lo **smartphone**. Attraverso lo **smartphone**, che ormai è entrato a far parte della nostra vita quotidiana, è possibile l'utilizzo di programmi in grado di sfruttare contemporaneamente una vasta gamma di sensori integrati nel dispositivo. Una tipologia di questi programmi permettono la navigazione attraverso la localizzazione tramite il ricevitore GPS.

Il sistema **GPS** (*Global Positioning System*) è un sistema di posizionamento *outdoor* (letteralmente "all'aperto") su base satellitare, a copertura globale continua, gestito dal *Dipartimento della Difesa Statunitense*[3]. Il termine GPS, entrato ormai a far parte del linguaggio comune, indica l'intero apparato che permette il funzionamento della maggior parte dei navigatori disponibili in commercio. Questa tipologia di navigatori tuttavia non è utilizzabile in ambito *indoor* ("all'interno"), in quanto la ricezione del segnale proveniente dai satelliti GPS diventa scarsa o pressoché nulla [4].

L'esigenza di creare un sistema di posizionamento in grado di funzionare anche all'interno degli edifici, infatti, deriva direttamente dalle limitazioni dei sistemi di localizzazione satellitare.

Sono questi fattori, uniti a una serie di avanzamenti tecnologici nel campo dei sensori di movimento e sulla potenza di calcolo dei dispositivi mobile, che portano all'idea della creazione di un sistema di localizzazione indoor basato su smartphone.

L'obiettivo di questa tesi è lo studio di un gruppo di tecnologie e architetture legate al posizionamento indoor e inerziale, e lo sviluppo di un'applicazione di **Indoor Navigation** a basso costo infrastrutturale che funzioni esclusivamente sfruttando la sensoristica e le funzionalità presenti in un moderno smartphone come accelerometri, bussole elettroniche, fotocamere e la connessione internet.

Come descritto nei prossimi capitoli, la tecnica scelta differisce dalle classiche soluzioni adottate per la navigazione indoor le quali, nella maggior parte dei casi, sfruttano la comunicazione del terminale con stazioni radio in grado di dare un'indicazione sulla posizione attraverso tecnica della triangolazione [5] [6].

La creazione di un software di posizionamento indoor, pratico e a basso costo, si rivela particolarmente utile non solo per capire in quale luogo di un edificio si trova l'utente in certo istante, ma anche per avere un supporto di navigazione completo, individuando e localizzando gli oggetti all'interno dell'edificio. Sarà possibile una tracciatura, continua e costante, in una mappa nel display dello smartphone, in modo da fornire un riscontro visuale immediato all'utente riguardo la propria posizione e quella di eventuali oggetti precedentemente localizzati nell'edificio.

L'utilizzo del sistema studiato potrà essere associato a varie attività in diversi ambiti come ad esempio:

- **Localizzazione in edifici pubblici complessi:** campus universitari, ospedali, musei.
- **Attività commerciali:** ovvero la ricerca di prodotti in un centro commerciale.
- **Supporto alla sicurezza:** evacuazione di un edificio complesso, supporto per le forze di soccorso pubblico.

Riguardo il lavoro descritto in questa tesi sono stati pubblicati due articoli: il primo dal titolo "***Indoor pedestrian navigation system using a modern smartphone***" per l'evento **Mobile Human Computer Interaction** (**MobileHCI**) del 2010 [1] e il secondo con il titolo "***Inertial Navigation Systems for User-Centric Indoor Applications***" pubblicato in occasione del **Networked & Electronic Media** (**NEM**) sempre del 2010 [2].

2 Stato dell'arte

La **localizzazione indoor** si è rivelata da alcuni anni un interessante problema di studio e di ricerca scientifica. Molti approcci sono stati analizzati nel corso degli anni per ottenere una soluzione completa e definitiva.

Per la localizzazione di utenti o oggetti in un ambiente indoor si possono intraprendere due soluzioni: una legata all'utilizzo di un'infrastruttura esterna di supporto continuo e una senza. Vediamo nel dettaglio le due metodologie.

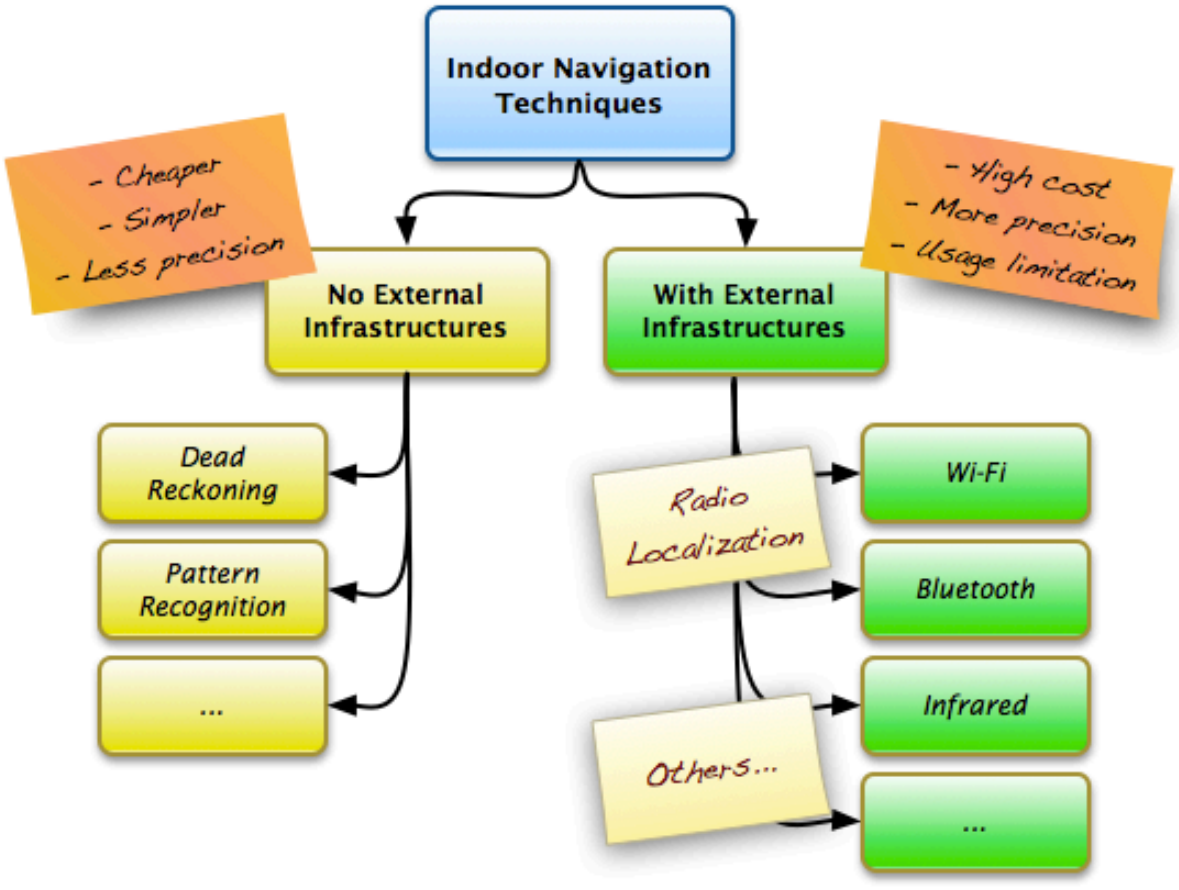

Figura 2.1 – Panoramica degli approcci studiati nella navigazione indoor.

Dalla figura 2.1 notiamo che nel caso in cui si scelga una soluzione abbinata a un'infrastruttura esterna, solitamente legata alla localizzazione via radio, si ha una precisione accurata (alcuni studi affermano nell'ordine di alcuni centimetri [6]) a discapito però di un alto costo per l'installazione e la manutenzione del sistema.

Senza l'utilizzo di infrastrutture esterne, invece, il costo complessivo dell'intero sistema è minore ma con lo svantaggio di avere una precisione meno accurata del caso precedente.

Gli studi proposti negli ultimi anni, analizzati per il progetto descritto in questa tesi, possono essere raggruppati in tre categorie di soluzioni:

- **Radio localization** (Wi-Fi, Bluetooth, ZigBee, …)
- **Dead reckoning**
- **Pattern recognition e analisi ambientale**

2.1 Radio localization

E' un metodo di posizionamento basato su un'infrastruttura, esterna o interna, costituita da antenne radio, solitamente installate in delle posizioni ben precise conosciute a priori e geolocalizzate.

E' una sorta di emulazione di quello che avviene con la triangolazione tramite i satelliti del sistema GPS. Questa metodologia di posizionamento in alcuni casi prende anche il nome "*GPS-like pseudolites*" [7]. Lo Pseudolite, o pseudo satellite, è un trasmettitore del segnale GPS installato in modo stabile a terra utilizzato per incrementare la diffusione del segnale su ambienti a copertura scarsa o ambienti indoor. Possono essere definiti quindi come dei piccoli satelliti GPS che permettono una precisa localizzazione ma richiedono alti costi di installazione e di calibrazione. In questo sistema l'utente ha la possibilità di vedere la sua posizione, rappresentata in tempo reale, su una mappa presente in un prototipo costruito appositamente.

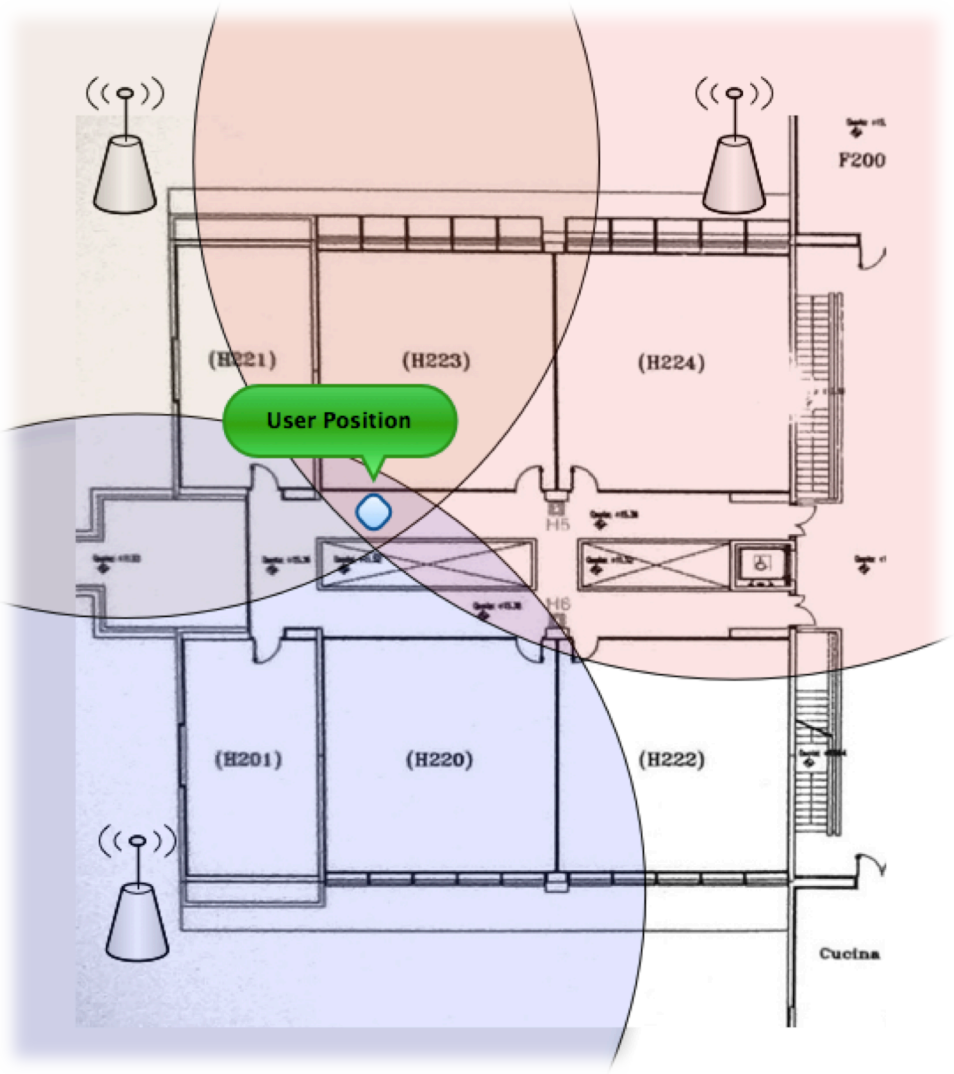

***Figura 2.2 - Esempio di localizzazione radio. In evidenza gli hot-spot* Wi-Fi.**

Molti di questi sistemi di navigazione permettono il tracking indoor a tre dimensioni e si basano su reti radio **UWB** (*Ultra Wide Band*), **Bluetooth**, **ZigBee** o **Wi-Fi** con il supporto di chip **RFID** (*Radio Frequency IDentification* o identificazione a radio frequenza) [6][7][8].

Un'azienda leader nel settore è la finlandese "***Ekahau***" (http://www.ekahau.com/) che fornisce un *Real Time Location System* (RTLS) basato su reti WI-FI, tag RFID e dei particolari device (*Ekahau RTLS Controllers*) costruiti appositamente per il sistema.

Anche Nokia con il suo prototipo *"Nokia Indoor Navigator with High Accuracy Indoor Technology"* utilizza la rete Wi-Fi e gli access point preinstallati nell'edificio e

localizzati in una mappa per fornire, a tutti i dispositivi mobile connessi alla rete, una precisa localizzazione all'interno di un ambiente indoor. [11] .

La navigazione indoor via radio, già disponibile da alcuni anni, presenta l'inconveniente di essere eccessivamente costosa. Essa necessita, infatti, dell'installazione di un'infrastruttura di rete, dell'utilizzo di dispositivi portatili dedicati e tuttavia non è esente da errori di misurazione: il segnale radio (UWB, Bluetooth o WI-FI) da solo non è sufficiente per una perfetta localizzazione continua e costante a causa di interferenze o cadute di segnale.

2.2 Dead reckoning

Una seconda tecnica deriva dal ***dead reckoning*** o navigazione stimata. Nella maggior parte dei casi questo metodo prevede la conoscenza a priori della mappa dell'edificio. Questa è la tecnica che viene descritta (capitolo 3) e utilizzata in questo lavoro di tesi. Il dead reckoning ha come vantaggio principale il basso costo e la richiesta di un'infrastruttura minimale rispetto alle altre tecniche.

Uno dei primi studi che hanno portato alla creazione di sistemi che utilizzano il dead reckoning per la stima della localizzazione indoor viene affiancato alla realtà aumentata. In questo sistema gli utenti sono equipaggiati con degli ingombranti dispositivi che consistono in sensori di accelerazione, videocamera a visuale soggettiva, rilevatore di posizionamento installato in un casco e un display. E' attraverso la "*fusione dei dati*" forniti da questi dispositivi che si misurano la lunghezza del passo e la direzione, e di conseguenza si fa una la stima della posizione corrente dell'utente. [12]

Il *"German Aerospace Centre"* [13] ha realizzato anch'esso un sistema che combina la navigazione satellitare, sensori inerziali montati sulle caviglie, bussole elettroniche, mappe baro-altimetriche e tag RFID attivi. Basato principalmente sul posizionamento inerziale il sistema consiste in un'architettura a due livelli di fusione dei dati provenienti dai sensori che opera utilizzando *Particle Filter* e *Kalman filter* per l'ottimizzazione dei segnali. I tag RFID attivi vengono utilizzati per la correzione della posizione e il miglioramento delle prestazioni generali del sistema.

Un altro prototipo per la stima del posizionamento che sfrutta il dead reckoning, senza nessuna infrastruttura esterna, è quello sviluppato dal *"German Research Center for Artificial Intelligence"*. L'obiettivo di questo sistema è la localizzazione dell'utente in ambienti estremi come ad esempio i percorsi di climbing [14]. Partendo da una posizione iniziale nota vengono utilizzati una mappa e uno smartphone dotato di accelerometro e bussola. Per rendere l'immagine della mappa utilizzabile durante la navigazione, il sistema necessita di una serie di dati riguardanti l'utente (ad esempio la lunghezza delle gambe e dei piedi). Non viene fatta la stima costante e adattiva della lunghezza del passo o del singolo movimento relativo all'arrampicata, ma semplicemente si suppone che l'utente conservi lo stesso andamento per tutto il tempo. I punti di riferimento all'interno dello scenario indoor dovranno essere comunque identificati in partenza dall'utente nella mappa.

2.3 Pattern recognition e analisi ambientale

Si tratta di un metodo di posizionamento che impiega le informazioni, visive e non, relative all'ambiente circostante.

Un tipico esempio di questo sistema è quello che sfrutta le immagini catturate da una videocamera per riconoscere il luogo preciso in cui la persona si sta muovendo[15]. Utilizza un procedimento che consiste nel confrontare le foto acquisite, suddivise in una serie di pattern rettangolari, con delle immagini conosciute e localizzate in una mappa.

Un altro prototipo di localizzazione è stato costruito sfruttando una serie di modelli sociali di descrizione dello spazio circostante. Grazie alla videocamera montata all'interno di un comune **PDA** (*Personal Digital Assistant*), l'utente scatta delle foto definendo la direzione ogni volta che raggiunge un punto predefinito in un percorso pianificato. Una funzione di tracciatura della posizione è integrata nel sistema e registra un codice identificativo della persona, l'instante di tempo alla posizione visitata, il tempo trascorso dall'ultima posizione e il tempo di arrivo stimato alla posizione successiva.

Il sistema che propone la ***Microsoft*** [16] è un navigatore basato sulla registrazione di una serie di attività umane in un contesto indoor e outdoor. Il modello costruito consiste in un'applicazione chiamata ***Greenfield*** e un prototipo di smartphone "***Menlo***" con *Microsoft Windows Embedded CE 6.0 R2* che contiene fotocamera, accelerometro a tre assi Bosch BMA150, un supporto esterno dotato di magnetometro e un barometro digitale Bosh BMP085 che misura la pressione atmosferica, attraverso la quale si stima l'altitudine e si riconosce il piano corrispondente nell'edificio.

Figura 2.2 – Il prototipo "Menlo" con l'applicazione "Greenfield" della Microsoft (Per gentile concessione della Microsoft Research).

L'applicazione del "Menlo" registra i passi e la direzione, le foto scattate durante il percorso ed eventualmente il segnale GPS se disponibile. Vengono così raccolti una serie di dati relativi all'ambiente circostante, che in seguito si sfruttano per riconoscere la posizione di un nuovo utente.

Un sistema a basso costo di posizionamento indoor viene presentato dalla *University of Technology* di Graz [17]. Questo sistema utilizza la fotocamera di un cellulare per determinare la posizione dell'utente attraverso dei piccoli codici a barre bidimensionali. La fotocamera esegue la scansione dell'ambiente alla ricerca di questi codici, posizionati in dei punti strategici, in modo da ottenere la localizzazione dell'utente.

3 Il dead reckoning

Il **dead reckoning** (dall'inglese *deduced reckoning*, letteralmente "stima derivata") è il processo utilizzato per la stima della posizione corrente a partire da una posizione precedente nota. Essa viene misurata tramite la stima della velocità, del tempo e della direzione. Moderni sistemi inerziali di navigazione dipendono dal dead reckoning e sono applicati specialmente nel campo dei veicoli automatizzati [18].

3.1 La navigazione stimata

Se consideriamo la navigazione in un sistema bidimensionale è sufficiente conoscere l'orientamento del mezzo, ossia l'angolo di imbardata, mentre se consideriamo un sistema a tre dimensioni, ad esempio un velivolo o un sottomarino, è necessario conoscere anche il beccheggio e il rollio del mezzo. I tre movimenti rappresentano le modalità di spostamento di un oggetto rispetto ai suoi assi di riferimento nello spazio:

- L'**imbardata** è l'oscillazione rispetto all'asse verticale passante per il baricentro del mezzo.
- Il **beccheggio** è il movimento rispetto al proprio asse trasversale.
- Il **rollio** è l'andamento del mezzo rispetto al proprio asse longitudinale.

In generale gli apparati che utilizzano la navigazione stimata sono basati sulla conoscenza della velocità del mezzo. Da questa grandezza, infatti, è possibile ricavare la distanza percorsa in un certo intervallo di tempo e ottenere la posizione del mezzo rispetto ad un'altra posizione nota.

Su tale principio è fondata, ad esempio, la tecnica di navigazione marittima mediante il compasso nautico: l'apertura dello strumento rappresenta la distanza percorsa in un certo periodo di tempo dedotta dalla velocità dell'imbarcazione. Se è nota la rotta sulla carta nautica, è allora possibile avere una stima della posizione attuale.

I metodi tradizionali per il calcolo della velocità e della distanza percorsa erano basati in passato sul conteggio dei passi o sullo srotolamento fuori dalla nave di una corda annodata da cui l'usanza, ancora oggi in vigore, di misurare le velocità nautiche in nodi. Nei giorni nostri, tali sistemi si sono evoluti in contapassi meccanici e in odometri,

ovvero in strumenti che misurano il numero di giri di una ruota e dunque la distanza percorsa, ed in turbine sottomarine in grado di ruotare ad una velocità proporzionale con quella della nave.

Durante la Seconda Guerra Mondiale, inoltre, si sono diffuse tecniche per il calcolo della velocità mediante le osservazioni via radar dell'effetto *Doppler* e con l'utilizzo di accelerometri e giroscopi all'interno di un sistema di posizionamento inerziale. Proprio questi ultimi rappresentano ormai gli strumenti di navigazione più diffusi per scopi civili e militari basati sulla tecnica del dead reckoning.

3.2 La stima di una nuova posizione

Il dead reckoning ha come svantaggio l'accumulo degli errori di calcolo relativi al posizionamento. La ragione deriva dal fatto che la nuova posizione viene stimata solo dalla conoscenza che si ha della posizione precedente, in questo modo la probabilità di commettere un errore di stima della posizione cresce durante tutto il percorso.

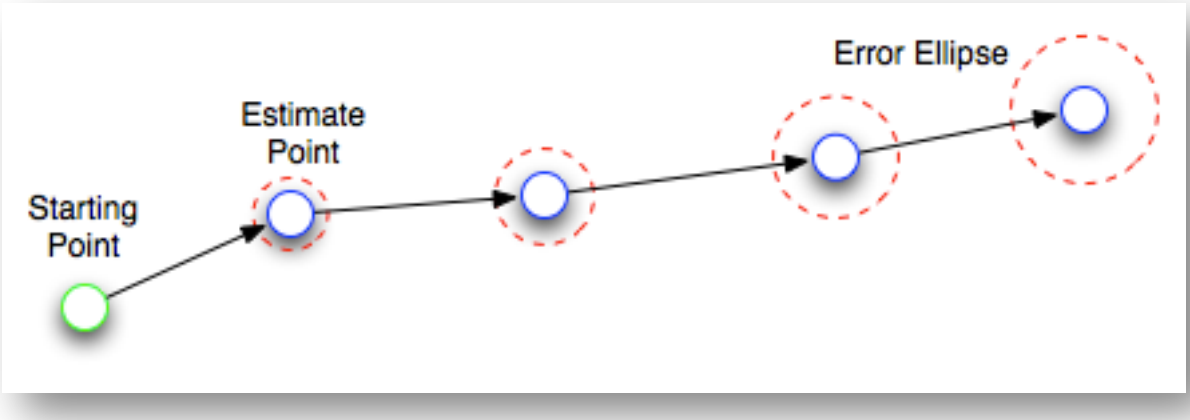

Figura 3.1 – La navigazione con la tecnica del dead reckoning.

Nella figura 3.1 viene rappresentato in verde il punto di partenza con un errore di posizionamento pari a zero. Nelle posizioni successive l'errore, indicato dall'ellisse tratteggiata in rosso, cresce a causa della dipendenza con la posizione precedente. Questa ellisse di errore, o ellisse di confidenza, rappresenta la posizione stimata e aumenta di dimensione passo dopo passo. Nella figura successiva viene rappresentato nel dettaglio il singolo passo e l'ellisse corrispondente.

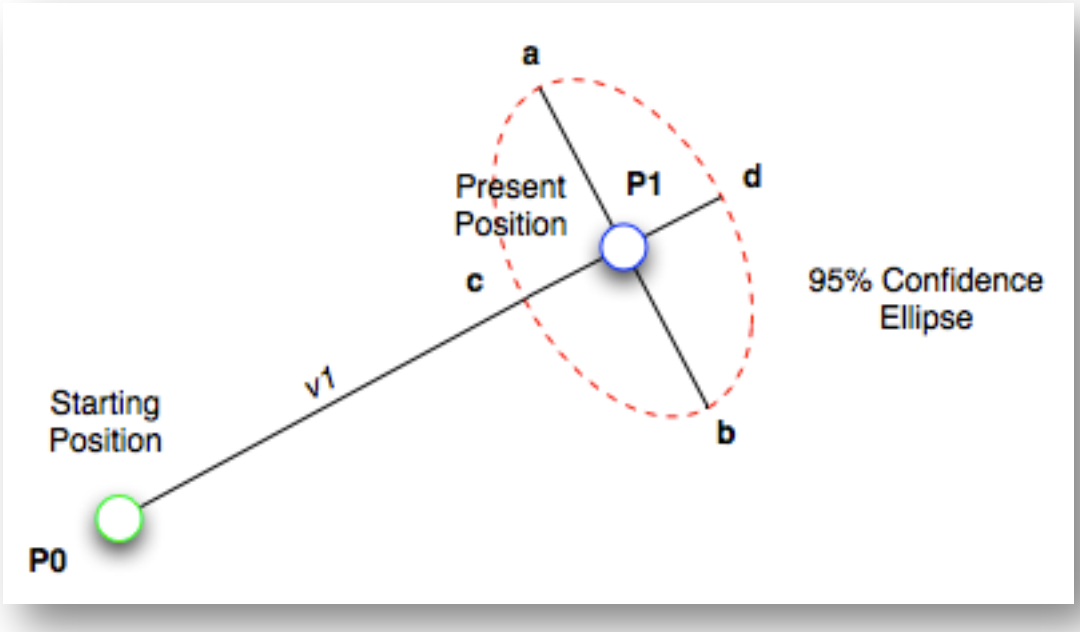

Figura 3.2 – Dettaglio del singolo passo e dell'ellisse di errore associata.

La direzione e la velocità sono unite in un vettore di movimento che rappresenta il cambiamento di posizione da un punto noto $\boldsymbol{P_0}$ a una posizione stimata $\boldsymbol{P_1}$. La precisione di questa stima viene rappresentata come una ellisse di confidenza, e gli assi dell'ellisse vengono determinati dall'accuratezza della direzione stimata e dalla misurazione della velocità. Questo viene rappresentato nella figura 3.2. Un utente si muove dal punto $\boldsymbol{P_0}$ al punto $\boldsymbol{P_1}$ e l'ellisse rappresenta la posizione probabile calcolata al 95% con l'asse ***ab***, determinato dall'accuratezza del sensore di direzione e ***cd***, determinato dall'accuratezza del calcolo della velocità.

Mentre il calcolo e l'incertezza di una sola lettura possono essere descritti in questo modo, l'incertezza delle letture multiple viene calcolata come somma cumulativa delle incertezze su tutte le letture a partire dall'ultima locazione precisa conosciuta. Questo si esprime con la seguente formula:

$$P_n = P_0 + \sum_{i=1}^{n} (v_i + v_e)$$

dove n è il numero dei calcoli dead reckoning a partire dal punto P_0; P_n è la posizione corrente; v_e è il vettore errore per ogni calcolo effettuato.

Supponendo un percorso rettilineo, l'ellisse di confidenza risultante dopo *n* iterazioni ha assi di dimensione ***n*ab*** e ***n*cd*** e, nel peggiore dei casi, queste ellissi crescono in maniera linearmente dipendente alla distanza percorsa. La precisione dei calcoli dei sensori di misurazione del movimento è fondamentale per avere la massima certezza possibile nella stima della posizione.

L'utilizzo del dead reckoning nei sistemi di navigazione inerziale implementati sui dispositivi portatili è ostacolato dalla lettura sporca dovuta alla scarsa qualità dei sensori di posizionamento presenti. Il rumore del sensore potrebbe offuscare il segnale e incrementare l'errore potenziale, è d'obbligo quindi una precisa lettura del valore, e ove possibile, l'utilizzo di tecniche per la correzione dei dati o la ricalibrazione del sistema attraverso l'utilizzo di altre posizioni note. Le due tecniche più importanti sulla correzione dei dati, utilizzate soprattutto nel campo della navigazione indoor, sono il Kalman Filter e il Particle Filter [19] [20] [30].

Il Kalman Filter consiste in un insieme equazioni matematiche ed è utilizzato su sistemi lineari in cui il rumore è gaussiano a media nulla. Si può usare indipendentemente dal fatto che si conosca o no la natura del sistema. Utilizza più sorgenti di misurazione dei dati e stima in tempo reale il nuovo stato del sistema, e nel caso dei navigatori inerziali, la nuova posizione probabile. Il Particle filter è usato invece per sistemi non lineari. Nella navigazione indoor si impiega per stimare la posizione corrente attraverso l'utilizzo di "*particelle*" che indicano una posizione, alle quali si associa un peso specifico. Maggiore è il peso della particella, maggiore è la probabilità che la essa indichi la posizione corretta. Il Particle filter e il Kalman filter costituiscono degli ottimi modelli per il filtro del segnale ma richiedono un elevato costo computazionale.

4 Architettura del sistema di navigazione indoor

A differenza delle tecniche descritte nei capitoli precedenti, la soluzione presentata in questa tesi è esclusivamente basata su delle particolari funzionalità di un comune smartphone moderno unite alla tecnica del dead reckoning. Queste funzionalità sono:

- **Sensore di accelerazione (accelerometro)**
- **Sensore di orientamento (bussola elettronica)**
- **Connessione internet**
- **Fotocamera**

I dati letti dai sensori del dispositivo, combinati con una mappa e una postazione iniziale nota, forniscono la posizione attuale dell'utente. Pertanto non c'è nessuna necessità di rimanere connessi con nessun sistema di posizionamento esterno come il GPS, il quale in un contesto indoor sarebbe in ogni caso inaccessibile, o utilizzare sistemi di trilaterazione radio; la localizzazione viene fatta esclusivamente attraverso il dead reckoning, lo smartphone e la mappa dell'edificio (figura 4.1).

L'utilizzo della tecnica del dead reckoning può portare a un accumulo di errore come descritto nel capitolo 3. Nel lavoro di questa tesi è stato deciso di correggere l'errore ricalibrando la posizione dell'utente tramite delle postazioni prelocalizzate nell'edificio, in modo da stimare la localizzazione successiva a partire da delle posizioni ben note. Queste posizioni, inclusa anche la posizione di partenza, vengono indicate attraverso dei codici a barre bidimensionali opportunamente codificati.

In definitiva l'obiettivo del lavoro di questa tesi è lo studio di un **sistema di navigazione indoor** e lo **sviluppo di un applicativo su smartphone** in cui è presente un algoritmo in grado di stimare la nuova posizione dell'utente basandosi solo sui sensori di movimento e le altre funzionalità presenti in uno smartphone.

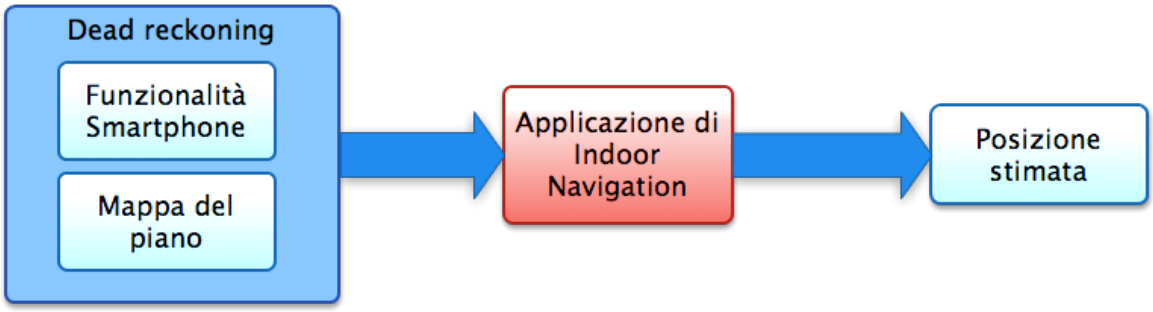

Figura 4.1 – La stima della posizione

Nelle sezioni successive vengono quindi descritti gli elementi necessari per la costruzione e il funzionamento di questo sistema. Nel paragrafo 4.1 viene descritta l'architettura, che rappresenta l'insieme degli elementi principali, e nel paragrafo 4.2 le fasi in cui questi elementi interagiscono tra di loro. La validazione del sistema attraverso una serie di test e analisi degli errori, in relazione alla correttezza del calcolo della posizione dell'utente, viene proposta nel capitolo 8 di questa tesi.

4.1 Descrizione generale dell'architettura

L'architettura del sistema descritto in questa tesi si basa sostanzialmente su quattro componenti fondamentali:

- **Uno smartphone** dotato di: accelerometro, bussola, fotocamera e connessione internet.
- **Codici a barre bidimensionali localizzati** e posti in punti strategici dell'edificio.
- **Un server web** da cui è possibile scaricare la mappa in formato digitale e i riferimenti sulla locazione dei codici a barre relativi alla mappa.
- **L'applicazione** per smartphone che realizza l'algoritmo per la stima della posizione che dialoga con i sensori dello smartphone.

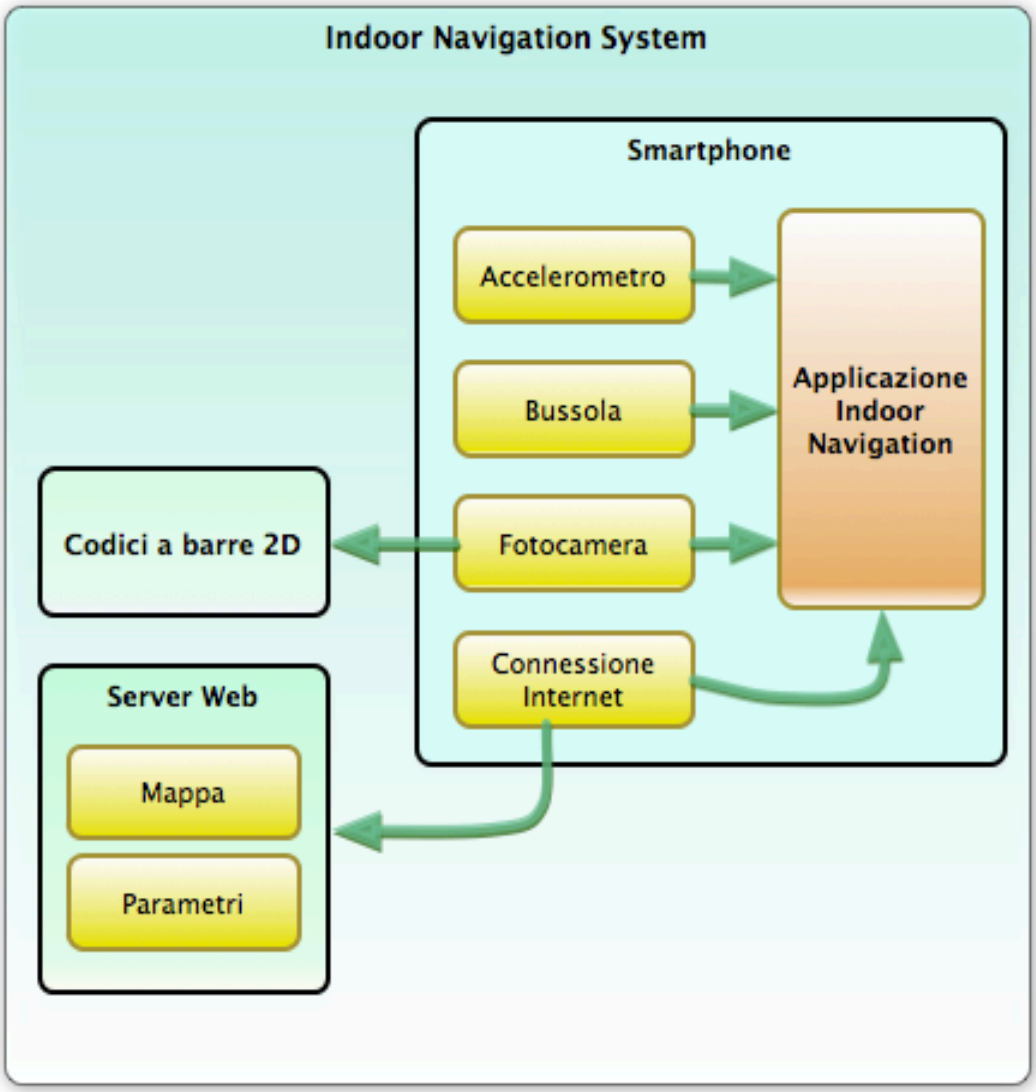

Figura 4.2 – Architettura generale del sistema

Lo smartphone è il cuore dell'intero sistema. Attraverso i suoi sensori permette di acquisire i dati sul movimento, leggere gli altri parametri necessari al funzionamento del sistema e prelevare la mappa per la navigazione.

Analizzando il movimento dell'utente, si determina la distanza percorsa che viene misurata in passi, trasformandola successivamente in metri in base alla lunghezza del singolo passo.

Questo lavoro si occupa di individuare una tecnica per determinare il posizionamento dell'utente riconoscendo la direzione e il momento in cui egli, durante la camminata,

effettua un passo. Queste operazioni, approfondite nei capitoli 5 e 6, vengono fatte utilizzando il sensore di accelerazione e la bussola elettronica.

Nei paragrafi successivi vengono descritti lo smartphone e i suoi sensori, il server web e i codici a barre bidimensionali.

4.1.1 Lo smartphone e i sensori di posizione

I sensori di posizione in grado di misurare un movimento presenti nello smartphone sono:

- **L'accelerometro**
- **Il sensore di campo magnetico**
- **Il sensore di orientamento**
- **Il giroscopio**

L'accelerometro è un sensore che misura le accelerazioni in m/s^2. Normalmente l'accelerometro presente nei dispositivi misura un'accelerazione lineare su tre dimensioni e viene chiamato accelerometro a 3 assi. Le accelerazioni misurate comprendono anche la forza gravitazionale terrestre.

I valori misurati da questo sensore vengono utilizzati per riconoscere lo spostamento dell'utente, e più precisamente, il momento in cui egli compie un passo.

Il **sensore del campo magnetico,** o bussola elettronica, misura la densità del campo magnetico intorno al dispositivo. Se non viene influenzato fortemente da fonti elettromagnetiche nelle vicinanze, il sensore è in grado di misurare il campo magnetico terrestre relativo alla posizione geografica dello smartphone. Effettua una misurazione sui tre assi relativi allo smartphone in $\boldsymbol{\mu T}$ (micro Tesla).

E' tramite questo componente che il dispositivo fornisce una misura sul suo orientamento *Nord, Sud, Est* e *Ovest* in relazione al piano tangente alla superficie terrestre.

Il **sensore di orientamento** fornisce gli angoli di inclinazione dello smartphone in relazione al piano tangente alla superficie terrestre: il **beccheggio** (o ***pitch)*** intorno

all'asse x e il **rollio** (o ***roll)*** intorno all'asse y. Tramite il sensore del campo magnetico misura invece **l'orientamento** (o ***azimuth***) in gradi da 0° a 359° (in orizzontale è l'angolo tra il nord magnetico e l'asse y).

E' in grado quindi, attraverso il sensore del campo magnetico, di fornirci un'indicazione sul polo nord magnetico ed eventualmente, tramite il calcolo della posizione GPS e della relativa declinazione magnetica, ci permette di calcolare anche la posizione precisa del nord geografico. Tuttavia il segnale GPS non viene considerato in questo lavoro in quanto è debole o del tutto assente all'interno degli edifici.

Grazie al sensore di orientamento e la misura del suo "***azimuth***", possiamo ottenere la direzione dell'utente durante la camminata.

Il **giroscopio** è un sensore costituito da un componente elettronico presente all'interno degli ultimi smartphone usciti sul mercato. E' in grado di misurare la velocità angolare in radianti/secondo intorno ai tre assi **x, y** e **z** del dispositivo. Non ha necessità di richiamare i dati misurati dall'accelerometro e dal magnetometro. Fornisce quindi una risposta più rapida sulla posizione dello smartphone nello spazio, rispetto a quella fornita dal sensore di orientamento.

Il giroscopio non è stato considerato in questo lavoro in quando non presente nei dispositivi usati per la sperimentazione descritta nel capitolo 8. Sarà oggetto di studio in uno sviluppo futuro del lavoro di tesi.

4.1.2 Il server web

Il **server web** utilizzato per il sistema di navigazione indoor è un servizio connesso alla rete internet che si occupa di fornire al dispositivo:

- **Le impostazioni iniziali del sistema**
- **Le posizioni dei codici a barre utilizzati per il posizionamento iniziale e i punti di ricalibrazione, in relazione alla mappa dell'edificio**
- **La mappa dell'edificio.**

Il file di testo contenente i parametri riguardanti la posizione iniziale e i punti di ricalibrazione (o checkpoint) viene descritto nel paragrafo 4.2. Questo file è codificato

in **XML** (***eXtensible Markup Language***), un linguaggio che permette una descrizione dettagliata dei parametri necessari.

Il **server** può essere contenuto in una risorsa hardware dedicata che risiede nell'edificio in cui viene utilizzato il sistema di navigazione, ma ciò non toglie che esso possa trovarsi anche su altri computer, ove risiedono anche altri tipi di servizi, che non stanno necessariamente nello stesso luogo (o edificio) in cui viene utilizzato il sistema.

Il collegamento delle informazioni dal server web con lo smartphone avviene, infatti, attraverso la rete internet e il protocollo **HTTP** (***Hyper Text Transfer Protocol***).

Lo spazio disco necessario per la memorizzazione dei file **XML** e delle **immagini** (in formato ***JPG, PNG o BMP)*** relativi alla mappa del piano è estremamente ridotto.

Poiché la dimensione dello spazio disco occupato da un file XML, necessario per le impostazioni, è dell'ordine di qualche centinaia di byte, in termini di spazio disco equivale alla dimensione minima del cluster (varia da 2 a 32 kB). La dimensione di un'immagine contenente la mappa è di circa 100 kB e ipotizzando che siano presenti dieci punti di ricalibrazione, e quindi dieci file XML relativi, si ha che per ogni piano sono necessari al massimo 500 kB. Ipotizzando quindi un edificio di dieci piani il server avrà bisogno di uno spazio di circa 5 MB, dimensione irrisoria se paragonata alla disponibilità di spazio disco nei sistemi odierni. Tuttavia, c'è ovviamente la possibilità di distribuire ogni immagine e ogni file XML in server web diversamente localizzati: il punto fondamentale per il funzionamento del sistema è che l'URL dei file XML sia correttamente codificato sui relativi barcode bidimensionali come descritto nel seguente paragrafo.

4.1.3 I codici a barre bidimensionali

I codici a barre bidimensionali, o barcode 2D, sono l'evoluzione dei classici codici a barre utilizzati per l'identificazione di quasi tutti i prodotti commerciali. La differenza sostanziale sta nel fatto che i codici bidimensionali oltre ad essere rappresentati in maniera differente, contengono molta più informazione dei classici barcode monodimensionali.

Figura 4.3 – Un barcode 2D (tipo QR) e barcode 1D: differenze.

Da qualche tempo ormai esistono svariati software per smartphone in grado di decodificare informazioni da questi barcode. Rispetto ai tag RFID (***Radio Frequency IDentificaton***) portano dei vantaggi in termini di costi: stampare un codice a barre su un foglio di carta è più economico e pratico dell'installazione di un chip RFID.

Esistono decine di tipi di barcode 2D [29]. All'interno di questo studio vengono utilizzati i **codici QR (*Quick Response*),** sviluppati dalla compagnia giapponese *Denso Wave* nel 1994 e rilasciati sotto licenza libera nel 1999. Sono molte al giorno d'oggi le applicazioni gratuite di lettura di tag QR distribuite sia nell'*Android Market* che nell'*Apple Store* o da altri siti web.

Per "lettura di barcode" si intende in genere una fotografia, o scansione, fatta tramite la fotocamera del dispositivo seguita da una successiva decodifica dell'informazione codificata.

La creazione di un barcode, ossia la codifica di un'informazione testuale al suo interno, può avvenire attraverso dei servizi web creati da alcune aziende, come ad esempio la ***Kaywa***. Tramite la pagina http://qrcode.kaywa.com è possibile codificare un URL, una porzione di testo semplice, un numero telefonico o un SMS scaricandone poi gratuitamente la relativa immagine del codice a barre.

Molte riviste usano i codici QR per accedere a dei contenuti multimediali. Il loro utilizzo è strettamente legato alla connessione internet: i codici, infatti, spesso vengono adoperati per contenere un URL, e tramite questo connettersi alla risorsa dedicata.

Figura 4.4 – Un codice a barre bidimensionale QR

Numeric Code	**Max. 7.089 characters**
Alphanumeric	**Max. 4.296 characters**
Binary (8 bits)	**Max. 2.953 characters**

Figura 4.5 – La quantità di informazione codificabile all'interno dei QR code.

La quantità di dati codificabile all'interno dei codici QR varia a seconda del tipo di informazione scelta. Esistono quindi delle limitazioni e si va da 2953 caratteri in formato binario a 8 bit a 7089 caratteri per i numeri come descritto nella figura 4.5 [21].

Nel nostro sistema i codici QR vengono utilizzati per memorizzare un URL che rappresenta la locazione del file XML. Si tratta in media di memorizzare circa 50 caratteri.

Una stringa di esempio può essere: http://opensource.crs4.it/mappa_settore_c.xml che rappresenta il file XML nel quale sono presenti i dati di un'ipotetica mappa dell'edificio in cui viene utilizzato il nostro sistema.

Il metodo di posizionamento che descriviamo in questa tesi sfrutta i barcode QR per la determinazione e l'identificazione iniziale della posizione dell'utente e per i successivi

checkpoint. La stima della posizione, infatti, può portare a un accumulo di errori e questo inconveniente viene risolto effettuando una ricalibrazione della posizione dell'utente attraverso dei checkpoint indicati dai barcode. Attraverso le sperimentazioni, descritte nel capitolo 8, è stato notato che è necessario correggere l'errore accumulato ogni 25 metri, di conseguenza, per avere un ottimale funzionamento del sistema i barcode andranno posizionati a circa 25 metri l'uno dall'altro o meno.

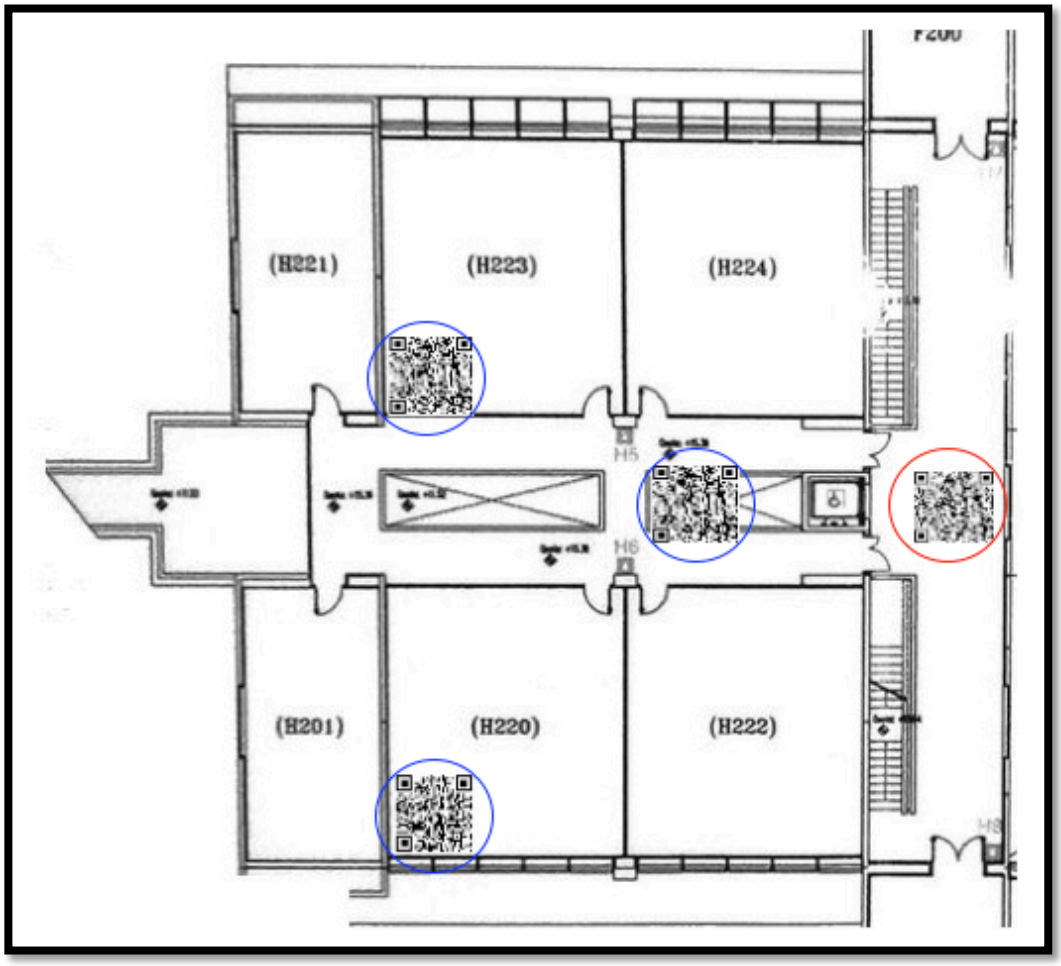

Figura 4.6 – Esempio posizionamento barcode QR

Nella figura 4.6 viene mostrato un esempio di posizionamento dei barcode nel piano dell'edificio: cerchiato in rosso il barcode di avvio del sistema che sta all'ingresso dell'edificio, in blu alcuni barcode di checkpoint distanti tra di loro circa 10 metri. La mappa nella figura 4.6 costituisce solo un'indicazione relativa a quali pareti possono essere scelte per "incollare" i barcode.

Tuttavia il primo codice a barre d'inizializzazione va preferibilmente posto all'ingresso dell'edificio o del piano che si vuole utilizzare per il sistema di navigazione, meglio se posizionato vicino alla piantina di sicurezza del piano, disponibile nella maggior parte degli edifici pubblici, come si può notare dalla figura successiva.

Figura 4.7 – Esempio di posizionamento del barcode rispetto alla mappa dell'edificio

4.2 Comportamento dinamico del sistema

Descriviamo in questo paragrafo come gli elementi dell'architettura interagiscono dinamicamente tra loro, suddividendo in fasi il comportamento del sistema.

- **Fase zero**: creazione dell'infrastruttura di barcode, la loro localizzazione, installazione dell'applicazione su smartphone e inserimento dei dati caratteristici dell'utente. Questa è l'unica fase che viene fatta una volta per tutte.
- **Fase uno**: avvio del sistema e determinazione della posizione iniziale dell'utente o ricalibrazione della posizione tramite checkpoint.
- **Fase due**: acquisizione dei dati dal server web.
- **Fase tre**: lettura dei valori forniti dai sensori di movimento.
- **Fase quattro**: elaborazione dei dati, calcolo e visualizzazione della nuova posizione.

Di seguito vediamo, rappresentati in figura 4.8, gli stati del sistema.

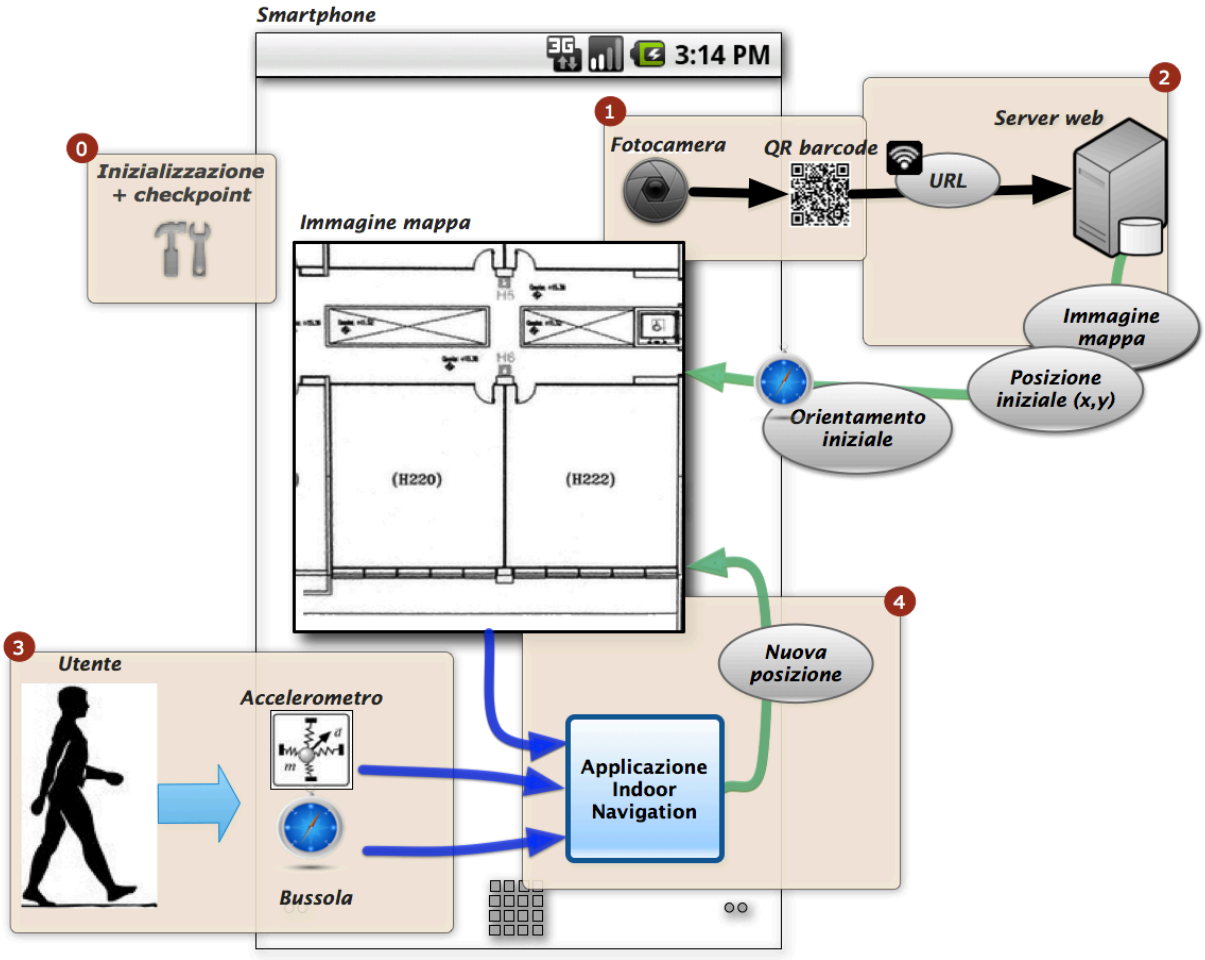

Figura 4.8 – Le fasi del sistema di navigazione indoor

Il coordinamento del sistema viene fatto dall'applicazione **Indoor Navigation** nella quale è presente un algoritmo che permette di riconoscere il movimento dell'utente, individuando il passo e la direzione in cui questo viene fatto. Lo smartphone si trasforma quindi in un contapassi direzionale, dove i passi non hanno solamente una lunghezza, ma anche un orientamento fornito dalla bussola elettronica interna allo smartphone.

L'algoritmo di calcolo della posizione successiva è un processo che rimane interno allo smartphone e non richiede il supporto di un server per il calcolo dei dati. La comunicazione con il "mondo esterno" riguarda solo la lettura del codice a barre e la connessione al server web per l'acquisizione dei dati iniziali e dei checkpoint.

Nei paragrafi successivi sono descritte nel maggior dettaglio le singole fasi.

4.2.1 Fase zero: creazione dell'infrastruttura di barcode e inserimento dei dati utente

Prima dell'avvio del sistema, sarà necessaria una fase di inizializzazione, che chiameremo fase zero, la quale comprenderà:

1. Installazione e configurazione di un server web per ospitare i file XML e le immagini.
2. Creazione dei file XML con i dati relativi alle mappe e alla localizzazione e caricamento nel server.
3. Generazione, stampa e posizionamento dei codici QR (associati ai file XML) su determinate pareti all'interno dell'edificio secondo le modalità descritte nel paragrafo 4.1.
4. Installazione dell'applicativo nello smartphone e inserimento dei dati utente.

I punti 1, 2, 3 costituiscono l'infrastruttura che va impostata una sola volta ed è valida per tutti gli utilizzi futuri. A differenza dei complessi sistemi radio, descritti nel secondo capitolo di questa tesi, nel nostro caso abbiamo semplicemente bisogno di un server web, un pc collegato in rete in grado di accedere ai servizi di generazione dei barcode e una stampante in grado di stampare su della carta comune questi codici a barre.

La posizione iniziale e quelle dei checkpoint, vengono rappresentate in coordinate relative all'immagine della mappa.

Il punto 4 rappresenta il punto di partenza per la personalizzazione dell'applicazione. Nell'applicazione sviluppata, descritta nel capitolo 7, si è scelto di utilizzare come dato utente solamente la lunghezza del passo, misurato attraverso un semplice metro. Vedremo nei capitoli 5 e 6 come viene riconosciuto il posizionamento e nel capitolo 8 i risultati ottenuti dalla sperimentazione.

4.2.2 Prima fase: avvio del sistema

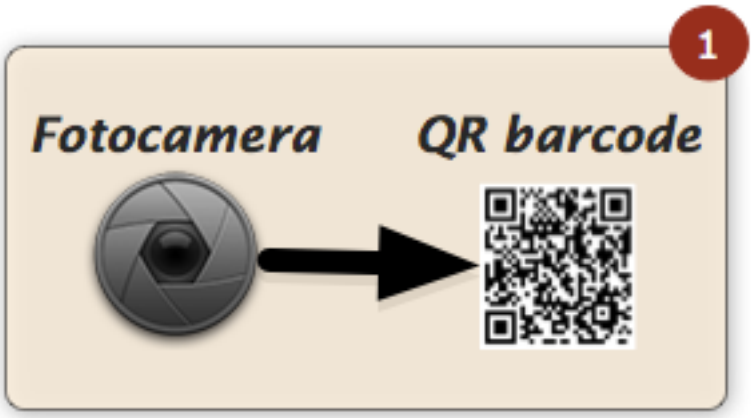

Figura 4.9 – Fase 1

La prima fase consiste nell'avvio dell'applicazione dallo smartphone e la decodifica del barcode iniziale o di quelli successivi. Questa operazione verrà ripetuta per ogni checkpoint e ci permetterà di ricalibrare la posizione.

Nella prima fase viene utilizzata la fotocamera dello smartphone, richiamata all'interno dell'applicativo, la quale effettua la lettura di un barcode posizionato, ad esempio, sulla mappa del piano disponibile all'ingresso dell'edificio (calibrazione iniziale) come descritto nel paragrafo 4.1.3 e in figura 4.7.

La scansione del barcode dovrà essere fatta tenendo il dispositivo in posizione verticale, parallelo alla parete e perpendicolare al pavimento. I dettagli vengono descritti nel capitolo 6.

Il software di decodifica del barcode permette all'utente di evitare di scrivere manualmente i dati. All'interno del codice a barre infatti, è presente una stringa di caratteri, più precisamente un URL, che permette di acquisire il relativo XML nella seconda fase.

4.2.3 Seconda fase: acquisizione dei dati dal server

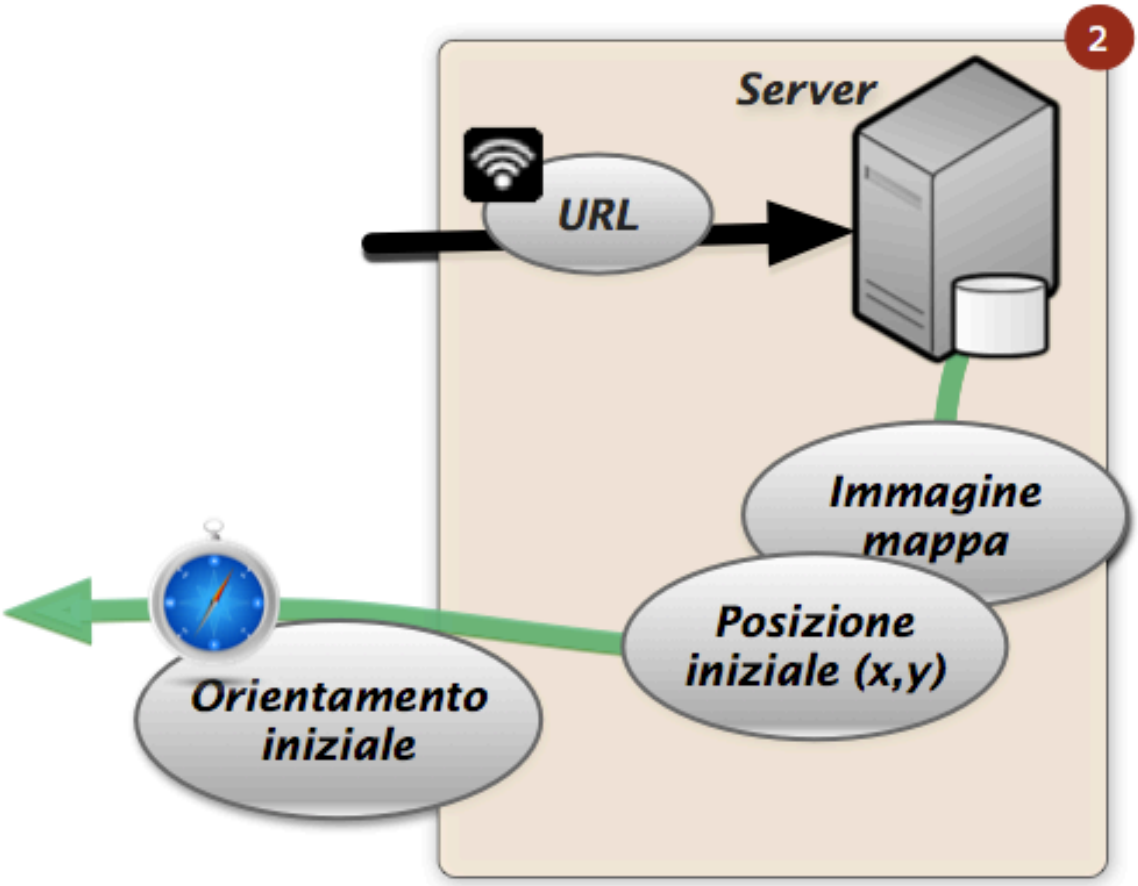

Figura 4.10 – Fase 2

La seconda fase è interna allo smartphone e non richiede nessun tipo di interazione con l'utente. L'URL acquisito, tramite il codice a barre, rappresenta la risorsa equivalente ad un documento XML di questo tipo:

```
<?xml version="1.0" encoding="UTF-8"?>
<map>
   <name>Mappa piano terra</name>
   <fileName>http://opensource.crs4.it/mappa0.jpg</fileName>
   <pixel_mt>10</pixel_mt>
   <startX>100</startX>
   <startY>256</startY>
</map>
```

Validato rispetto alla grammatica definita dal seguente XML Schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.crs4.it/schema/indoor"
    xmlns:tns="http://www.crs4.it/schema/indoor"
    elementFormDefault="qualified">
     <xs:element name="map" type="tns:mapType"/>
     <xs:complexType name="mapType" >
          <xs:sequence>
             <xs:element name="name"  type="xs:string"/>
             <xs:element name="fileName"  type="xs:anyURI"/>
             <xs:element name="pixel_mt" type="xs:integer"/>
             <xs:element name="startX"  type="xs:integer"/>
             <xs:element name="startY"  type="xs:integer"/>
          </xs:sequence>
     </xs:complexType>

</xs:schema>
```

I dati principali per la taratura iniziale del sistema e dei checkpoint successivi, contenuti nel documento XML, sono ***fileName***, ***pixel_mt*** e i due valori che indicano la posizione dei barcode sulla mappa ovvero ***startX*** *e* ***startY****.* Nel dettaglio:

- ***fileName:*** è l'URI *(Uniform Resource Identifier)* completo del file che rappresenta l'immagine della mappa che viene visualizzata nei display dello smartphone.
- ***pixel_mt:*** rappresenta il rapporto in pixel/metri nella mappa. Ossia a quanti pixel della mappa, equivale un metro nel mondo reale.
- ***startX:*** il valore della posizione sull'asse delle ascisse nella mappa espresso in pixel.
- ***startY:*** il valore della posizione sull'asse delle ordinate nella mappa espresso in pixel.

La posizione relativa ad ogni checkpoint viene fornita dai valori **startX** e **startY**: sono questi due parametri che variano in ogni barcode presente nel piano.

I valori ricavati dal file XML vengono affiancati all'orientamento reale del dispositivo, espresso in gradi calcolato al momento di acquisizione del barcode, dalla bussola.

Al termine della seconda fase è possibile quindi visualizzare sul display dello smartphone la mappa, l'orientamento e la posizione dell'utente che corrisponde all'esatta locazione in cui egli effettua la scansione del barcode.

4.2.4 Terza fase: lettura dei valori forniti dai sensori di movimento

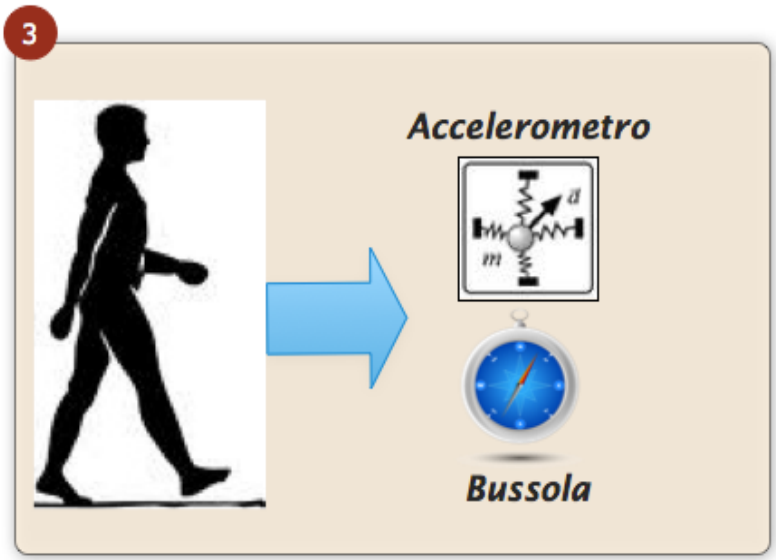

Figura 4.11 – Fase 3

Nella terza fase l'utente inizia a camminare e i sensori, ossia l'accelerometro e il magnetometro, rilevano i dati relativi rispettivamente al movimento e la direzione. L'utente ha l'unico vincolo di tenere lo smartphone fermo in mano in posizione verticale, o leggermente inclinata, di fronte a lui in modo da fornire un orientamento corretto. Sarà poi compito dell'applicativo, nella fase 4, visualizzare la nuova posizione dell'utente sulla mappa.

Il contapassi, il calcolo della direzione e le soluzioni proposte per il corretto posizionamento vengono descritti nel dettaglio nei capitoli 5 e 6.

4.2.5 Quarta fase: elaborazione dati e calcolo della nuova posizione

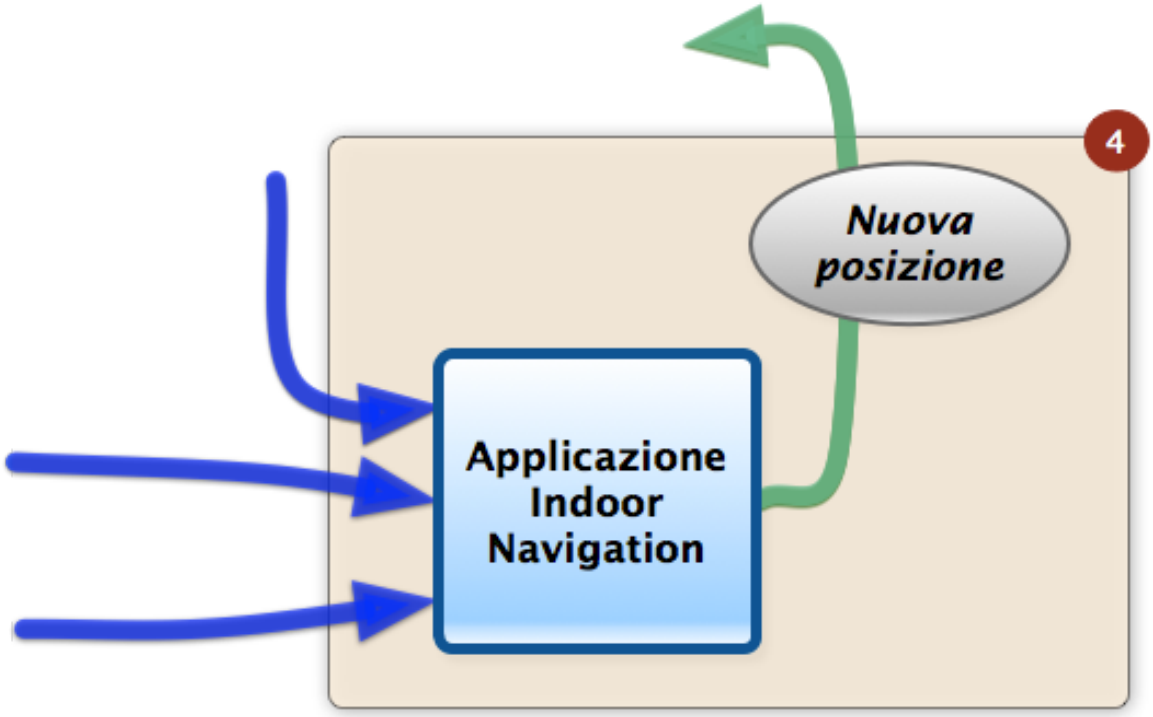

Figura 4.12 – Fase 4

Nella quarta fase avviene il calcolo della nuova posizione grazie all'applicativo di navigazione indoor installato nello smartphone. L'applicativo utilizza un algoritmo che prende in input i dati dei sensori di movimento, la posizione corrente e la mappa e produce come risultato una nuova posizione, in termini di coordinate *xy* su mappa, e la rappresenta nel display.

Al suo interno è presente del codice in grado di riconoscere se l'utente ha compiuto un passo e verso quale direzione l'ha fatto. I dettagli sono descritti nei prossimi capitoli.

5 Il contapassi

Per poter costruire un navigatore indoor, che utilizza la tecnica del dead reckoning, è necessario stabilire in che modo misurare la distanza percorsa dall'utente. In questo lavoro l'accelerometro rappresenta la risorsa utilizzata per la determinazione di questa misura. Non avendo a disposizione un posizionamento costante e continuo tramite trilaterazione, come avviene nel GPS, la distanza percorsa viene misurata in passi utente e, conoscendo la lunghezza di un singolo passo, rapportata in metri. Il dispositivo in grado di contare i passi di una persona viene chiamato **contapassi**, o in inglese, *step counter*.

I contapassi disponibili sul mercato vengono utilizzati soprattutto in ambito sportivo. Da alcuni anni questa tecnologia è stata integrata anche negli smartphone tramite delle applicazioni in grado di contare i passi di una persona attraverso la misura dei suoi movimenti corporei. Dato che le misure variano da persona a persona, solitamente viene richiesta dal sistema una calibrazione iniziale, ad esempio l'inserimento della lunghezza stessa del passo, in modo da fornire durante l'utilizzo una grandezza standardizzata della distanza percorsa in metri.

Tipicamente il contapassi è accurato se i passi vengono fatti su una superficie piana e se il dispositivo viene legato ai pantaloni della tuta sportiva o alla cinta in posizione verticale. E' stato calcolato che l'errore medio nella misurazione di un tipico contapassi è di circa il 5% [22].

I contapassi sportivi di prima generazione utilizzavano un interruttore meccanico che scattava al passaggio di una piccola sfera di piombo in un cilindro verticale nel momento in cui si effettuava un passo. L'interruttore incrementava il valore di un contatore tenendo così in memoria il numero di passi fatti.

Negli ultimi anni la sfera di piombo e l'interruttore sono stati sostituiti da sensori inerziali e da programmi in grado di rilevare i passi a seconda delle misurazioni del sensore. I sensori di accelerazione presenti sugli smartphone individuano il passo indipendentemente da com'è posizionato il dispositivo nello spazio: in verticale, in orizzontale, inclinato o capovolto.

5.1 L'individuazione del passo

Lo studio dei metodi per l'individuazione del passo è una materia importante di questo lavoro di tesi. Quando l'utente inizia a camminare, l'applicazione di navigazione deve mostrare in tempo reale passo dopo passo la sua posizione sulla mappa nel display dello smartphone. I passi vengono individuati tramite i valori di accelerazione registrati dallo smartphone.

L'accelerometro viene quindi interrogato per conoscere i valori di accelerazione nei tre assi **x, y** e **z** solidali al dispositivo. Utilizziamo le componenti dell'accelerazione, misurate in m/s^2, per calcolare il corrispondente valore assoluto o modulo:

$$|m| = \sqrt{{a_x}^2 + {a_y}^2 + {a_z}^2}$$

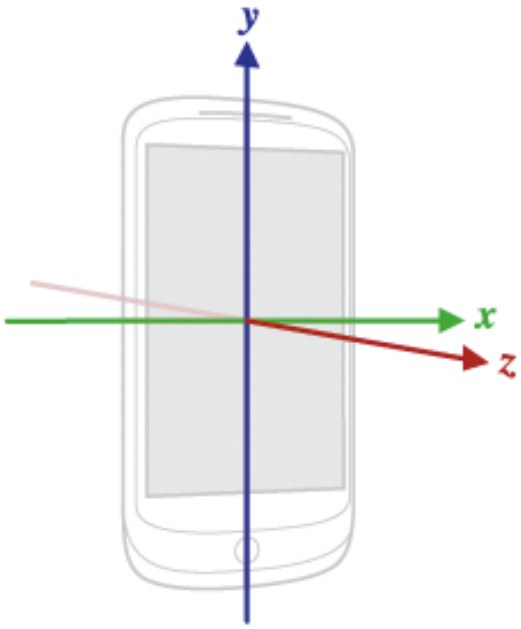

Figura 5.1 – Il sistema di coordinate nello smartphone

Nell'algoritmo successivo viene descritto come si può individuare un passo. Se il modulo dell'accelerazione (***Acc_module*** nella figura 5.2) è minore della soglia minima impostata (***th_min***) l'algoritmo si trova in una cosiddetta "*fase di discesa*" nella quale ad un certo punto si presenta un picco inferiore di accelerazione seguito subito dopo da una "*fase di salita*". Se nella fase di salita il picco superiore di accelerazione è maggiore della soglia massima impostata (***th_max***) allora siamo in presenza di un

passo. I valori di soglia vengono impostati a priori ma possono essere modificati nell'applicazione durante il suo utilizzo.

Algoritmo per l'individuazione del passo:

```
stepFlag = true;
if (!stepFlag && (acc_module > th_max) ){
    stepFlag = true;
    stepDetected();
}
else if (stepFlag && (acc_module < th_min) ){
     stepFlag = false;
}
```

E' stato scelto di non rimuovere la componente gravitazionale dalla misura dell'accelerometro in quando trascurabile, una volta individuato l'andamento dei picchi di accelerazione, per la rilevazione del passo. Di conseguenza, tutte le successive misurazioni che saranno descritte come "accelerazione" saranno intese includendo anche la componente di accelerazione di gravità, che equivale a circa 9.81 m/s^2.

La seguente figura 5.2 mostra un esempio di misurazione dei valori di accelerazione per un utente con la lunghezza media del passo di circa 65 cm, con un campionamento costante effettuato ogni 100ms. Le soglie scelte per una sensibilità normale sull'individuazione del passo sono *Th_high* = 11.0 e *Th_low* = 9.5.

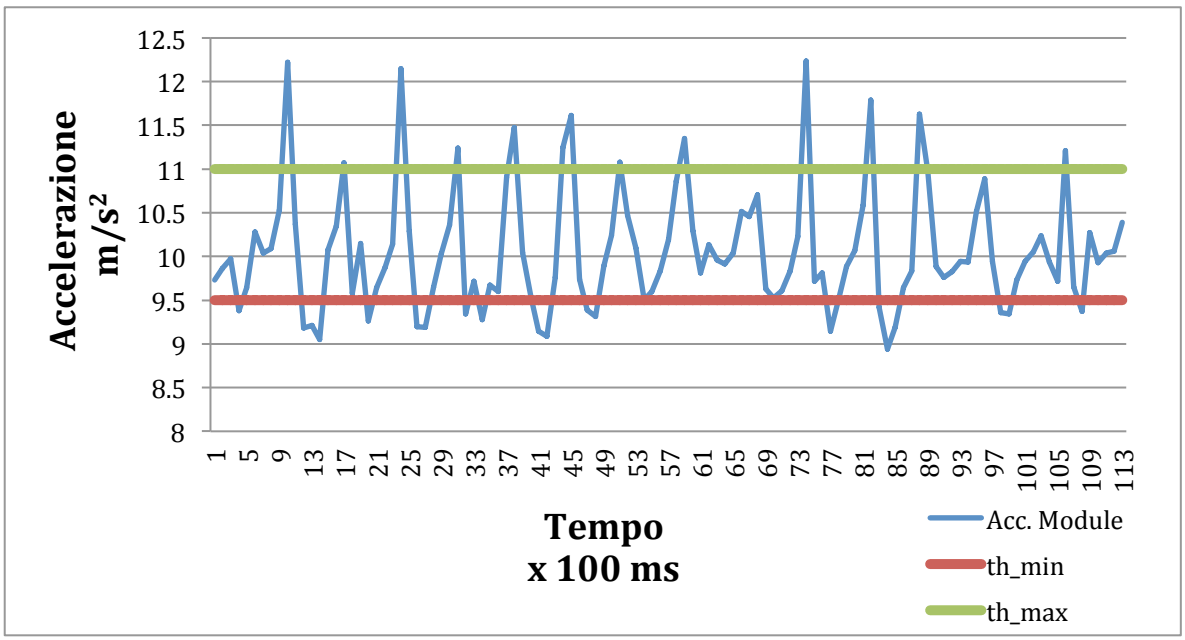

Figura 5.2 – Grafico tipico dei valori in modulo forniti dall'accelerometro

Nella figura 5.2 vengono rappresentati l'andamento temporale del **modulo dell'accelerazione**, la **soglia massima** (in verde) e la **soglia minima** (in rosso). L'andamento dell'accelerazione e i suoi picchi determinano il riconoscimento del passo.

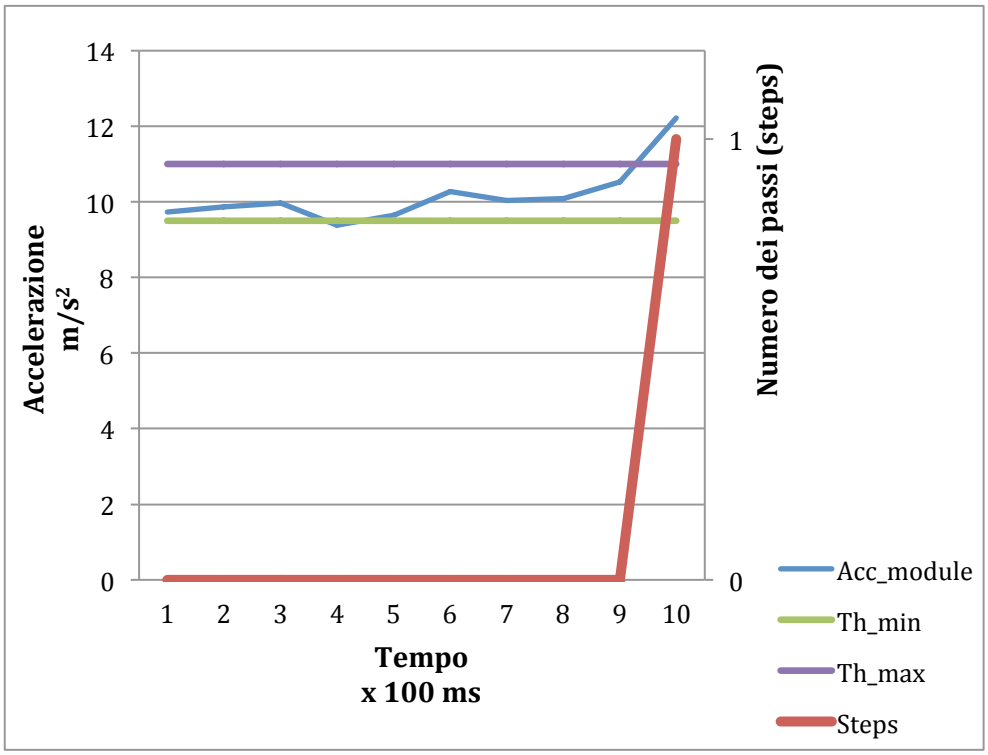

Figura 5.3 – Dettaglio funzionamento algoritmo

Nella figura 5.3 vediamo un esempio di campionamento di dieci dati di accelerazione equivalenti ad una finestra temporale di un secondo. Nel dettaglio viene rappresentato come il passo viene individuato nel momento in cui l'accelerazione, dopo aver compiuto la fase di discesa superando la soglia minima (a circa *400 ms*), supera nella fase di risalita la soglia massima preimpostata (a circa *900 ms*).

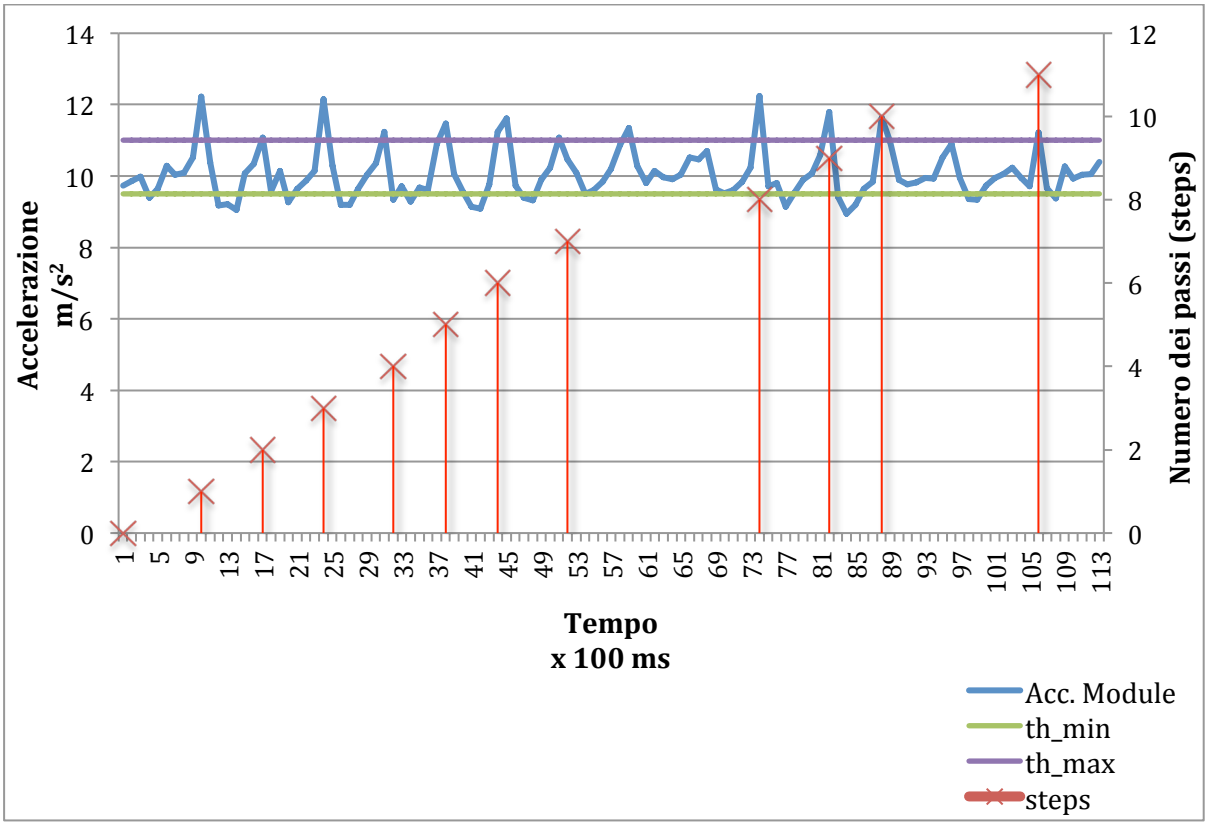

Figura 5.4 – La misurazione di 11 passi

Nella figura 5.4 viene rappresentato uno specchio temporale più ampio di campionamento dei dati: le linee verticali rosse rappresentano il valore del contapassi che cresce man mano che l'algoritmo rileva un nuovo passo attraverso l'esame dei picchi di accelerazione.

Come già descritto all'inizio del paragrafo la scelta di utilizzare il valore assoluto dell'accelerazione permette di rilevare i passi dell'utente indipendentemente dalla posizione dello smartphone nello spazio. In altre parole, così facendo, il passo dell'utente viene individuato sia se egli tiene il cellulare con una mano in posizione orizzontale, sia in posizione verticale, sia in posizione obliqua.

Nelle figure successive vengono mostrate le tre componenti di accelerazione originarie che compongono il modulo ("***Acc_module***" in figura 5.4).

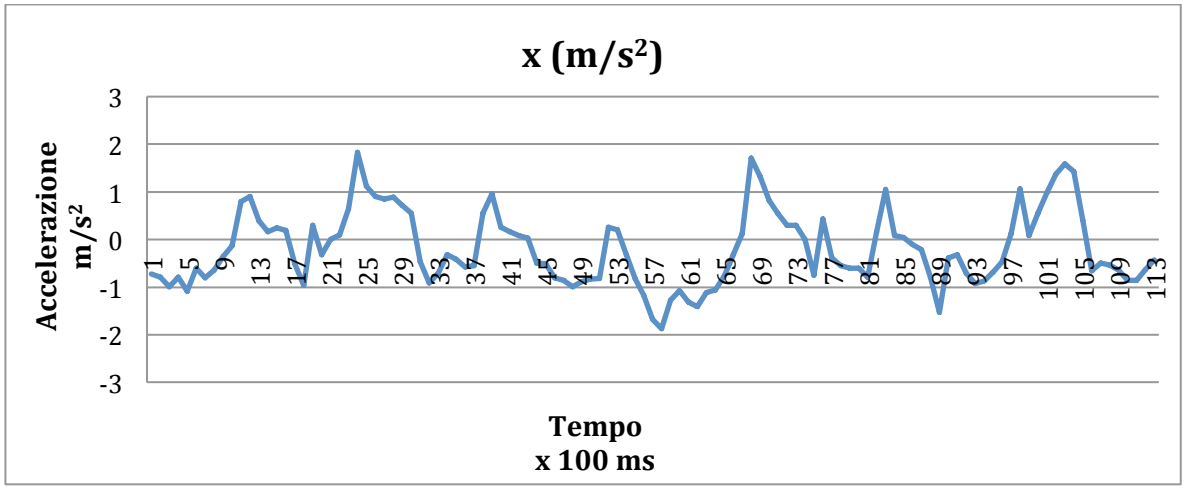

Figura 5.5 – Grafico accelerazione rispetto all'asse x

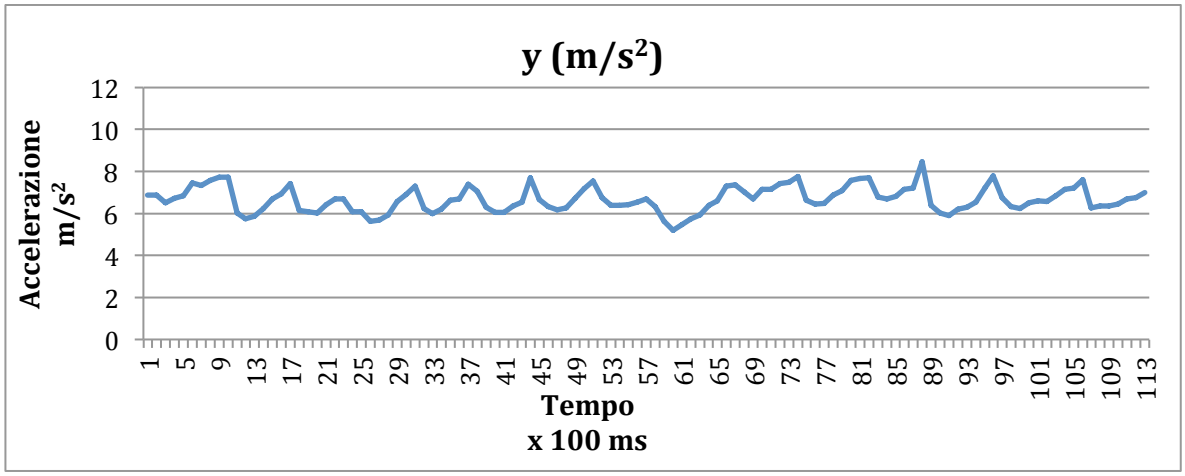

Figura 5.6 – Grafico accelerazione rispetto all'asse y

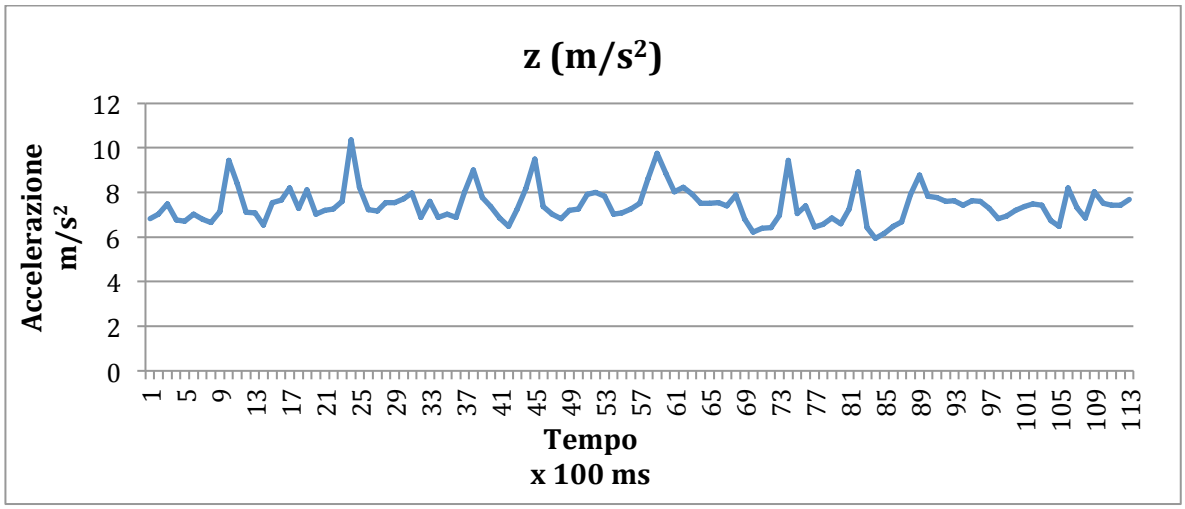

Figura 5.7 – Grafico accelerazione rispetto all'asse z

Le figure 5.5, 5.6 e 5.7 rappresentano le accelerazioni rispettivamente sugli assi **x, y** e **z** (dettaglio degli assi del dispositivo nella figura 5.1 all'inizio del paragrafo). Durante la registrazione di questi valori, il cellulare è stato tenuto con una mano in posizione verticale leggermente inclinata: si nota, infatti, che la componente gravitazionale influisce maggiormente sugli assi y e z del dispositivo (figura 5.6 e figura 5.7) e risulta "neutra" sull'asse x. La scelta del modulo delle accelerazioni ha facilitato quindi il lavoro dell'algoritmo. In caso contrario, infatti, ci sarebbe stato bisogno di conoscere in ogni momento la precisa disposizione dello smartphone nello spazio, individuando su quale componente ha influito maggiormente il passo attraverso l'analisi degli angoli di imbardata, rollio e beccheggio dello smartphone.

5.2 La lunghezza del passo

Esistono fondamentalmente due approcci per determinare la distanza percorsa dall'utente. Questa distanza è data dalla lunghezza del passo che può essere:

- **Costante nel tempo** e basata sui dati fisici dell'utente.
- **Variabile** e basata sui dati dei sensori inerziali.

Nei nostri primi esperimenti, e nell'applicativo sviluppato per questa tesi, è stato utilizzato il primo metodo ottenuto attraverso l'inserimento dei dati dell'utente (la lunghezza del passo). La tecnica di lunghezza di passo adattiva verrà implementata negli sviluppi futuri.

5.2.1 Lunghezza costante basata sui dati fisici dell'utente

Con questa tecnica si afferma che data l'età, il sesso, l'altezza, il peso e se disponibile anche la misura della calzatura, è possibile stimare la lunghezza del passo che, con una probabilità abbastanza alta, si manterrà costante per una determinata lunghezza di percorso.

Questa è principalmente la tecnica che usano i contapassi sportivi disponibili in commercio. Altri contapassi invece, più semplici ma non meno precisi, richiedono di inserire la lunghezza del proprio passo, suggerendo prima in che modo misurarlo.

Solitamente si consiglia di misurare la distanza che intercorre dall'impronta lasciata dal tallone in un piede all'impronta del tallone dell'altro piede. Questa misura equivale a un singolo passo in avanti.

Nel manuale di istruzioni di alcuni contapassi viene indicato che la lunghezza media del passo, durante una corsa, è circa 65 cm per una donna adulta e circa 75 cm per un uomo adulto. Tuttavia la precisione di questi valori dipende molto dall'altezza della persona [23]. Attraverso le sperimentazioni effettuate in questo lavoro, descritte nel capitolo 8, è stato rilevato che la lunghezza media, su uomo e donna durante una camminata, è circa 68 cm.

Di seguito viene elencata una lista di alcuni contapassi commerciali che richiedono l'inserimento della lunghezza del passo:

- *Yamax Digi-Walker SW700*
- *Yamax Digi-Walker Powerwalker Accelerometer*
- *Omron HJ113 and HJ109*
- *New Lifestyle 1000*

In alcuni di questi dispositivi sono presenti all'interno del manuale di instruzioni alcune tecniche per la misurazione della lunghezza del passo. Un esempio viene fornito dall'NL-1000[24]. Questo contapassi suggerisce tre diverse tecniche:

- Il metodo Lap-Around-Track
- Il metodo dei dieci passi
- Approssimazione basata sull'altezza

Il metodo *Lap-Around-Track* risulta essere il più accurato e consiste nel camminare lungo percorso di una lunghezza nota, ad esempio i cento metri in una pista di atletica, contando i passi fino alla fine. La lunghezza finale del passo si ottiene dividendo la lunghezza del tracciato per il numero di passi.

Il *metodo dei dieci passi* è una semplificazione del metodo precedente. Consiste nel tenere traccia del punto di partenza, fare dieci passi e misurare la distanza percorsa. La misura del passo equivale a un decimo della distanza percorsa.

Nella terza scelta "*Approssimazione basata sull'altezza*" si consiglia di moltiplicare l'altezza dell'utente per un preciso fattore, diverso per uomo e donna. Per quanto riguarda la donna, il fattore è 0.413, per l'uomo invece è 0.415.

Figura 5.8 – Contapassi New Lifestyle 1000
(Per gentile concessione della NEW-LIFESTYLES, INC.)

5.2.2 Lunghezza variabile basata sui dati dei sensori inerziali.

Con la seconda tecnica alcuni studi [25] dimostrano che è possibile avere una misurazione continua della lunghezza del passo di una persona mentre cammina. Il passo, infatti, può variare secondo la velocità della camminata e tali studi calcolano la sua lunghezza a seconda dei dati di accelerazione forniti dai sensori inerziali.

Da un'accurata analisi dei valori di accelerazione, ci si rende conto che la lunghezza del passo viene influenzata in modo lineare a partire dalla frequenza della camminata e dalla varianza del segnale dell'accelerometro. Pertanto il passo può essere calcolato come combinazione lineare di questi valori.

Il calcolo adattivo della lunghezza del passo permette migliori performance del sistema e sarà oggetto di studio e d'implementazione negli sviluppi futuri.

6 Il calcolo della direzione

La direzione è un parametro fondamentale per un posizionamento indoor corretto basato sulla navigazione stimata. Essa viene misurata tramite il **sensore di orientamento** presente all'interno dello smartphone.

L'orientamento, fornito dalla bussola elettronica dello smartphone, viene misurato in gradi da 0° a 359° e ci indica la posizione del polo nord magnetico. Il polo nord magnetico non corrisponde esattamente al polo nord geografico: l'angolo compreso tra il nord magnetico e il nord geografico, detto declinazione magnetica, cambia a seconda del luogo in cui ci troviamo e nel corso del tempo.

Anche l'eventuale presenza di forti campi magnetici esterni compromette la corretta rilevazione del nord magnetico mediante una bussola. Lo stesso accade se la bussola è posizionata vicino a masse metalliche.

6.1 Calibrazione della bussola

Onde evitare errori sul corretto orientamento dei passi dell'utente è opportuno ricordare che la bussola ha bisogno di una calibrazione "manuale" iniziale.

Figura 6.1 – Messaggio di interferenza sulla bussola dell'iPhone

Alcuni software di orientamento presenti negli smartphone visualizzano un messaggio di avviso, relativo alla calibrazione, quando rilevano la presenza di forti campi elettromagnetici (figura 6.1). Questa calibrazione consiste nello spostare il dispositivo lontano da fonti elettromagnetiche e muoverlo lungo tutti i suoi assi formando una figura a otto rovesciato, come indicato in figura 6.2, fino a quando la schermata di calibrazione scompare. Solitamente bastano due o tre giri completi.

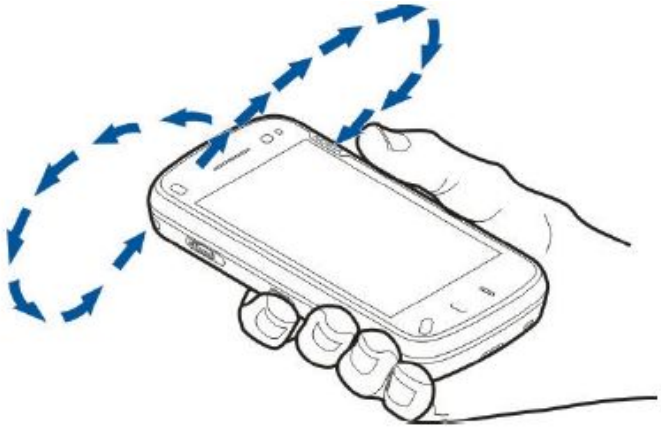

Figura 6.2 – Il movimento da eseguire per una corretta calibrazione della bussola

6.2 La rappresentazione del passo sulla mappa

Una volta calibrata la bussola elettronica del dispositivo, l'utente scatta la foto del codice a barre disponendo il dispositivo parallelamente alla parete e perpendicolarmente al pavimento, come descritto nel capitolo 4. L'angolo di orientamento misurato dalla bussola dello smartphone rappresenta la calibrazione iniziale del nostro sistema rapportata alla mappa nel dispositivo.

Questo viene fatto in modo da avere una corrispondenza corretta fra la direzione dei passi dell'utente e la sua raffigurazione sulla mappa nel display dello smartphone.

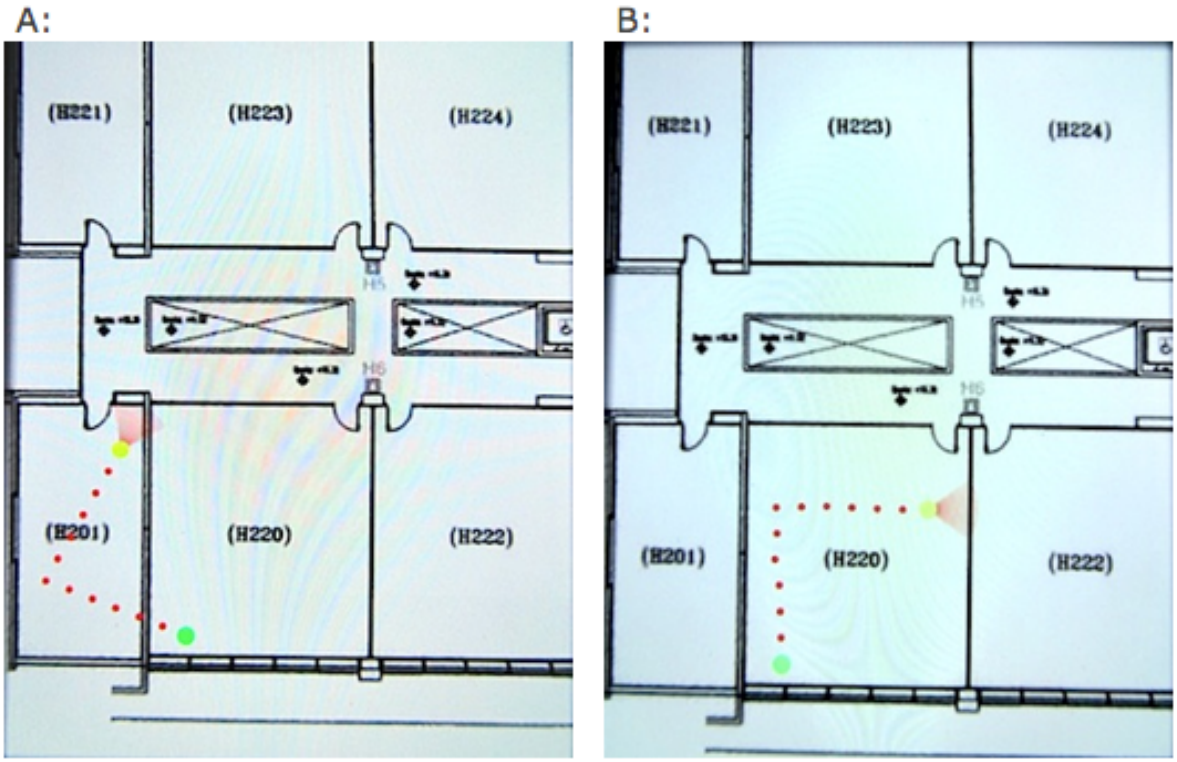

Figura 6.3 – La calibrazione del sistema

La figura 6.3 mostra le due situazioni a confronto. La figura 6.3A rappresenta un tracciamento dei passi fatti a seguito di un'errata calibrazione dell'orientamento iniziale (falsata di circa 60°) indicando addirittura l'attraversamento di un muro dopo il primo passo. A differenza del caso precedente, nella figura 6.3B, con lo stesso movimento e lo stesso percorso reale, il dispositivo risulta calibrato e coordinato alla mappa. Questo perché nel caso della figura 6.3B c'è stata un'impostazione iniziale corretta tra la mappa nello smartphone, l'orientamento iniziale del dispositivo e le direzioni successive dell'utente durante la camminata.

La mappa presente nel display del dispositivo, dovrà fornire all'utente una esatta corrispondenza in tempo reale. Ci sarà quindi un aggiornamento costante relativo al posizionamento e all'orientamento indicato dalla bussola elettronica.

Il disegno della nuova posizione verrà fatto calcolando il *seno* e il *coseno* dell'angolo di orientamento. Il risultato fornirà la nuova posizione misurata in pixel su mappa, rapportandola in seguito ad una misura reale (centimetri o metri) attraverso il

rapporto dato dal valore pixel/metro (presente nel file XML secondo le modalità descritte nel capitolo 4).

Nella figura 6.4 viene rappresentato il calcolo della nuova posizione ***x'y'*** ottenuto attraverso una vecchia posizione nota ***xy***, la lunghezza del passo ***l*** e l'angolo di orientamento **α** fornito dalla bussola durante il passo.

xy = vecchia posizione;

α = orientamento espresso in gradi. Misura fornita dalla bussola e calibrata secondo l'orientamento iniziale;

l = lunghezza del passo;

x'y' = nuova posizione calcolata;

$$x' = \cos(\alpha) * l$$

$$y' = \sin(\alpha) * l$$

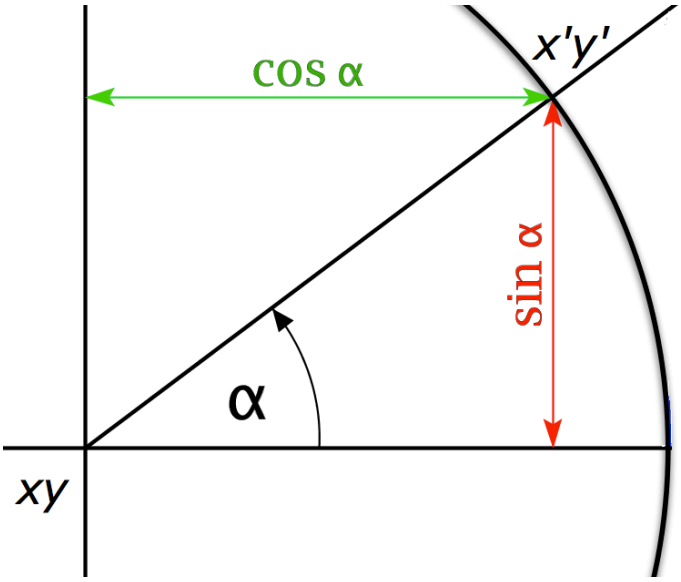

Figura 6.4 – Calcolo della nuova posizione.

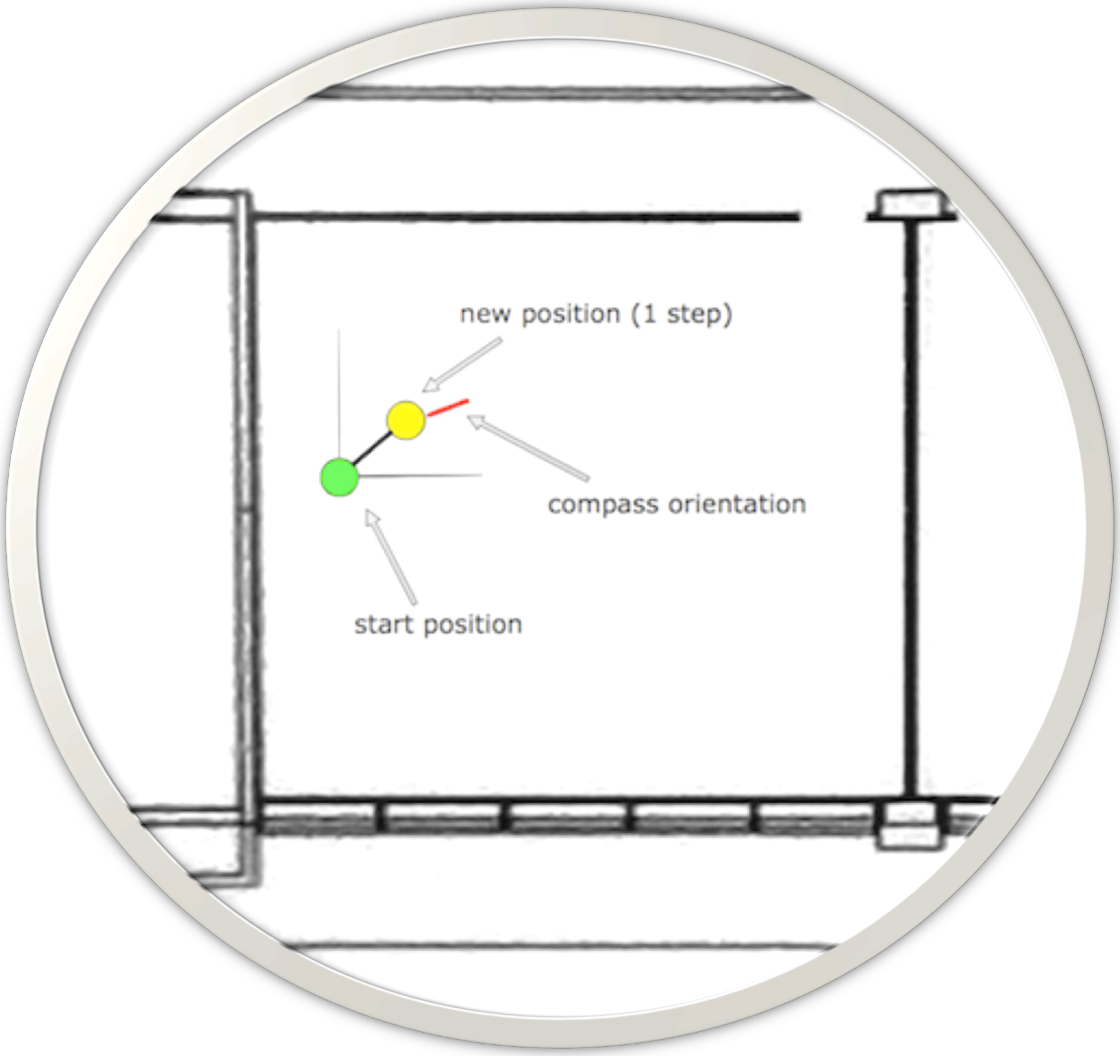

Figura 6.5 – Dettaglio sul disegno della nuova posizione sulla mappa.

Nella figura 6.5 vediamo il dettaglio della rappresentazione del passo sulla mappa. In verde la posizione iniziale ***xy***, in giallo la nuova posizione ***x'y'*** e l'orientamento dello smartphone indicato con la linea di colore rosso.

La calibrazione iniziale può essere soggetta a un errore di misurazione dovuto all'errato posizionamento dello smartphone durante la lettura del codice a barre. L'utente, infatti, potrebbe non tenere correttamente il dispositivo, ossia in modo non perfettamente perpendicolare alla superficie del pavimento e non parallelo alla parete, e quindi "falsare" il suo reale orientamento. Il sistema progettato quindi prevede sostanzialmente due casi:

- Il caso in cui l'utente scatta correttamente una fotografia al codice a barre in posizione parallela alla parete e perpendicolare al pavimento;
- Il caso in cui l'utente non tiene correttamente allineato il dispositivo;

Il primo caso è il caso più semplice, e non richiede un calcolo sulla effettiva posizione nello spazio del dispositivo: l'importante è che la sua bussola indichi lo stesso orientamento dell'utente di fronte alla parete.

Il secondo caso è più complicato, in quanto non si ha una corrispondenza diretta tra l'orientamento dello smartphone durante la lettura del barcode, e l'effettiva posizione dell'utente. Egli potrebbe decodificare il codice a barre posizionandosi non parallelamente alla parete: in questo caso il sistema ha bisogno di un calcolo che permetta di conoscere l'effettiva distanza e angolo di orientamento dello smartphone rispetto al codice a barre e quindi alla parete.

6.3 Calibrazione del sistema attraverso i barcode

Per risolvere il problema del corretto posizionamento del cellulare, rispetto ai barcode, è stata utilizzata in questa tesi una tecnica che permette di determinare la posizione dello smartphone rispetto al codice a barre, in termini di distanza e angolo di orientamento.

Questa parte del lavoro di tesi è stata sviluppata in collaborazione con il *"Multimedia Communications laboratory"* (**MCLab**) del ***Dipartimento di Ingegneria Elettrica ed Elettronica (DIEE)*** di Cagliari.

Analizziamo quindi per prima cosa le componenti di misurazione della posizione dello smartphone nello spazio.

Come già accennato nel capitolo 4, il sensore di orientamento fornisce tre misurazioni rispetto ai tre assi ***x, y, z*** solidali con lo smartphone e indicati in figura 6.6 [27].

- **Azimuth o Yaw**: è la rotazione intorno all'asse **z** del dispositivo. Fornisce un valore in gradi che va da 0° a 359°. 0 = Nord, 90 = Est, 180 = Sud, 270 = Ovest.

- **Pitch**: è la rotazione intorno all'asse **x** del dispositivo in relazione alla sua posizione orizzontale. Il suo range misurato in gradi, va da -180°, a 180°. Il valore è positivo quando il movimento e è in senso orario.
- **Roll**: è la rotazione intorno all'asse **y** relativamente alla sua posizione orizzontale. Il suo range è -90°, +90° (gradi). Il valore è positivo quando il movimento va dall'asse z all'asse x.

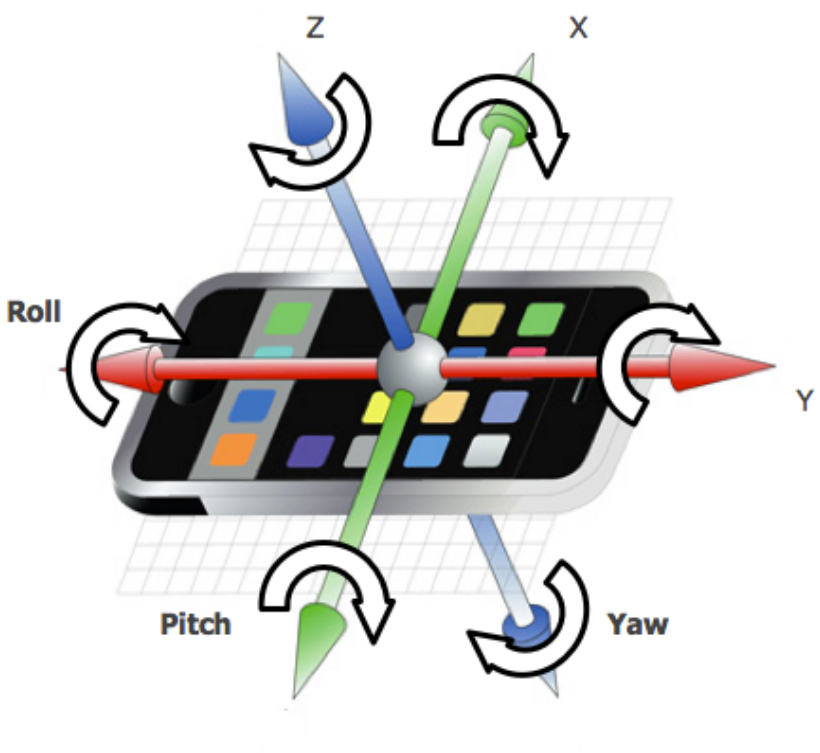

Figura 6.6 – Yaw, pitch e roll su un dispositivo mobile

Le definizioni di yaw, pitch e roll differiscono dalle tradizionali definizioni usate in aviazione dove l'asse x è posto lungo il corpo dell'aereo (dalla testa alla coda).

La direzione misurata per il sistema di posizionamento descritto in questa tesi non risulta soggetta al pitch del dispositivo: questo significa che l'azimuth, o l'orientamento rispetto al campo magnetico, è misurabile sia con il dispositivo posizionato in orizzontale, sia in verticale.

Per determinare la distanza e l'orientamento dello smartphone, durante la scansione del codice a barre, si fa uso anche dell'omografia planare.

In un modello di fotocamera a proiezione centrale, un punto nello spazio 3D viene proiettato in una immagine planare per mezzo di rette visuali dal punto nello spazio al centro dell'ottica. Matematicamente questo può essere descritto con l'utilizzo di una matrice *P* 3x4 che prende un punto in uno spazio 3D in coordinate omogenee $(X, Y, Z, 1)^T$ e lo trasforma in n punto nel piano di immagine 2D $(x, y, 1)^T$.

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = [P] \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

La matrice di proiezione P può essere calcolata secondo i parametri della fotocamera:

$$P = K\,[R|T]$$

dove K è una matrice (triangolare superiore) 3 x 3 chiamata matrice di calibrazione, e include i parametri della fotocamera (lunghezza focale e inclinazione). La [R|T] definisce le rotazioni e le traslazioni del dispositivo.

Nel caso di superfici planari (Z = 0), la trasformazione viene chiamata *plane-to-plane homography*:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = [H] \begin{pmatrix} X \\ Y \\ Z = 0 \\ 1 \end{pmatrix}$$

La matrice H 3x3 è chiamata *Homography Matrix* e risulta più semplice della matrice P. Può essere ridotta a:

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} = [K] \; [\mathbf{r_1} \quad \mathbf{r_2} \quad \mathbf{t}]$$

con K matrice che modella le caratteristiche della fotocamera e r_1 e r_2 sono le colonne corrispondenti della matrice di rotazione R e t = - RC con C il centro della fotocamera.

La figura seguente mostra il mapping tra un punto 2D *x'* nel piano *π'*, dove giace il codice a barre, in un punto 2D *x* nel piano *π* che rappresenta l'immagine acquisita dalla fotocamera.

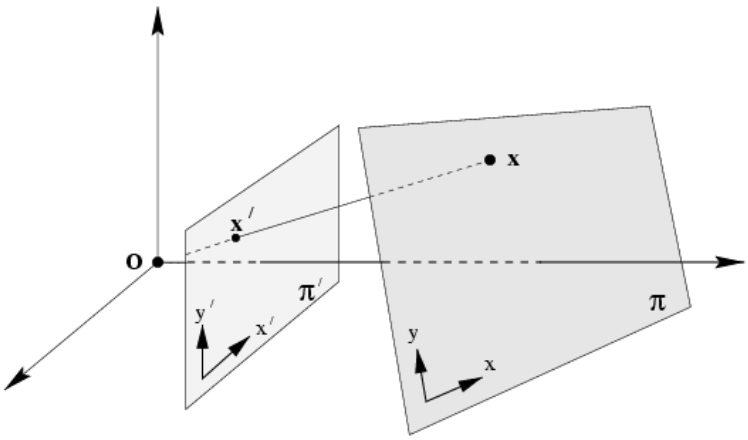

Figura 6.7 – Mapping tra piani

Dati quattro punti nel piano della scena, con al massimo due punti collineari, e la loro corrispondente posizione nell'immagine la matrice H viene determinata univocamente.

Consideriamo $P_1'(x_1', y_1')$, $P_2'(x_2', y_2')$, $P_3'(x_3', y_3')$ *e* P_4' (x_4', y_4') i quattro punti dell'oggetto rettangolare (barcode) e $P_1(x_1, y_1)$, $P_2(x_2, y_2)$, $P_3(x_3, y_3)$ *e* P_4 (x_4, y_4) le loro proiezioni ottenute usando una trasformazione di omografia planare.

I corrispondenti punti nelle due immagini in relazione all'omografia sono:

$$x_i' = \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}}$$

$$y_i' = \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}}$$

L'obiettivo di questo calcolo è rimuovere le componenti proiettate in modo da ottenere una versione scalata e ruotata dell'immagine originale.

Per fare questo identifichiamo i quattro angoli nella foto scattata, e mappiamo ogni vertice del quadrilatero nel corrispondente vertice del rettangolo conosciuto. Usando l'equazione precedente troviamo i coefficienti della matrice di omografia H e infine correggiamo l'immagine nella vista frontale.

Una volta che otteniamo la vista frontale, possiamo decomporre la matrice H utilizzando la *QR Factorization* nella sua matrice **([r1 r2 t])** ortogonale e la matrice K.

Dalla matrice ortogonale possiamo ottenere le varie rotazioni e le traslazioni e determinare l'orientamento della fotocamera nella scena rispetto al barcode.

Tramite i valori conosciuti a priori di lunghezza focale *f*, l'apertura della fotocamera, le dimensioni reali del barcode 2D (altezza *H* nella figura 6.8) e le corrispondenti dimensioni in pixel (altezza H' nella figura 6.8), possiamo calcolare la distanza ***l*** dell'utente dal barcode. Una semplificazione del problema viene presentata nella seguente figura:

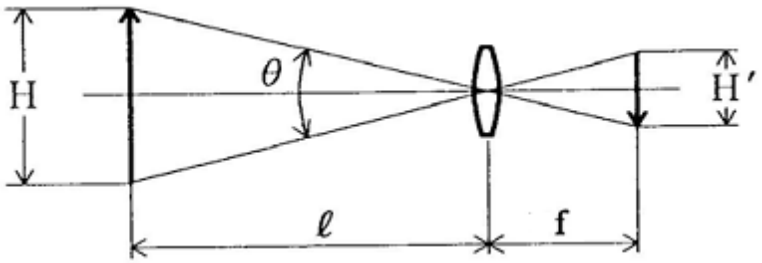

Figura 6.8 – Distanza e focale

Utilizzando i risultati del calcolo (angolo e distanza) la posizione iniziale dell'utente e il suo orientamento rispetto alla scansione del barcode 2D vengono calcolati in maniera più precisa. Ciò risulta di importanza cruciale per la corretta valutazione dei dati generati successivamente dallo smartphone, soprattutto in termini di orientamento.

6.4 Correzione della traiettoria

La locazione e la direzione del dispositivo non sempre corrispondono con l'effettiva locazione e direzione dell'utente. Infatti l'utente può tenere il dispositivo in diversi modi. Più precisamente si considerano le seguenti posizioni:

- Dispositivo di fronte all'utente in posizione verticale tenuto con una mano
- Dispositivo in una tasca dei pantaloni
- Dispositivo nel taschino frontale della camicia

Se il dispositivo è posizionato di fronte all'utente, ciò non pone nessun problema di riferimento sulla mappa: la posizione e la direzione del dispositivo sono uguali a quella dell'utente per tutto il percorso.

Diverso è il caso in cui lo smartphone viene tenuto in una tasca dei pantaloni. In questo caso l'orientamento indicato dall'applicazione dovrà essere ricalibrato rispetto alla reale direzione dell'utente.

Il metodo proposto permette di tracciare una serie di traiettorie possibili a partire da un punto di partenza, eliminando successivamente le traiettorie scorrette, ossia quelle in cui il tracking dell'utente sulla mappa va a scontrarsi con la rappresentazione di ostacolo fisico (ad esempio un muro).

Questo permette di avere una calibrazione della direzione assoluta del dispositivo, rapportata alla direzione del passo dell'utente.

Nella figura seguente: a partire da un punto iniziale vengono tracciate su 360° una serie di traiettorie possibili eliminando quelle che si scontrano con un ostacolo (il muro indicato sulla mappa). Le traiettorie eliminate sono indicate in blu nella figura 6.9, mentre in verde viene indicata la traiettoria corretta dell'utente riconosciuta dal dispositivo.

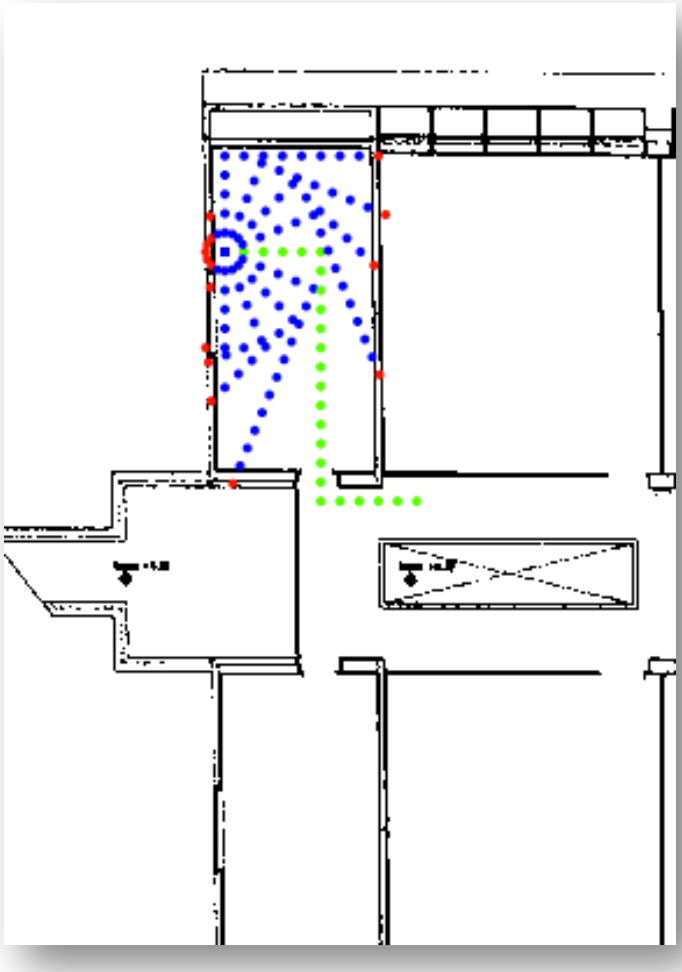

Figura 6.9 – La correzione della traiettoria con la mappa. In verde la traiettoria corretta.

Il procedimento descritto è in qualche modo simile a quello che avviene in molti navigatori commerciali: l'icona del veicolo viene sempre rappresentata in una strada sulla mappa, individuata come la più vicina o la più probabile nel caso in cui il segnale GPS non corrisponda direttamente alla strada stessa.

Il navigatore che utilizza questa tecnica ha solitamente al suo interno una mappa digitalizzata di tipo vettoriale che permette la rappresentazione della posizione vincolandola a una serie di posizioni possibili.

7 Lo sviluppo dell'applicazione su Android

In questo capitolo descriviamo l'applicazione sviluppata e daremo alcuni cenni sulla struttura del sistema operativo Android.

7.1 Il sistema operativo Android

Android è un sistema operativo opensource per dispositivi mobile (smartphone e tablet) basato sul Kernel Linux. E' stato sviluppato da "*Android Inc*." in seguito acquisita da Google. La prima piattaforma venne presentata nel 2007, ed era basata sulla versione 2.6 del Kernel Linux. Android supporta lo standard OpenGL ES 2.0 per la grafica tridimensionale, utilizza il database SQLite e ha una libreria SGL dedicata alla grafica bidimensionale [26].

La *Software Development Kit* (SDK) di Android include gli strumenti di sviluppo, le librerie, un emulatore del dispositivo, la documentazione, alcuni progetti di esempio e dei tutorial. A differenza dell'SDK iOS per Apple iPhone, con Android è possibile programmare su qualsiasi computer x86 compatibile che usi come sistema operativo Windows, Mac OSX o Linux. L'IDE ufficialmente supportato per lo sviluppo di applicazioni è Eclipse e il linguaggio utilizzato è JAVA. Il primo dispositivo mobile dotato di piattaforma Android è stato l'HTC T-Mobile G1, che abbiamo utilizzato in questa tesi durante la sperimentazione (descritta nel capitolo 8).

Android utilizza una *virtual machine* ottimizzata per il funzionamento nei dispositivi mobile, chiamata Dalvik.

Nella seguente figura 7.1 vediamo rappresentata l'architettura del sistema operativo Android composta da:

- **Kernel Linux**
- **Librerie**
- **Android Runtime (*Dalvik Virtual Machine*)**
- **Framework di applicazione**
- **Le applicazioni**

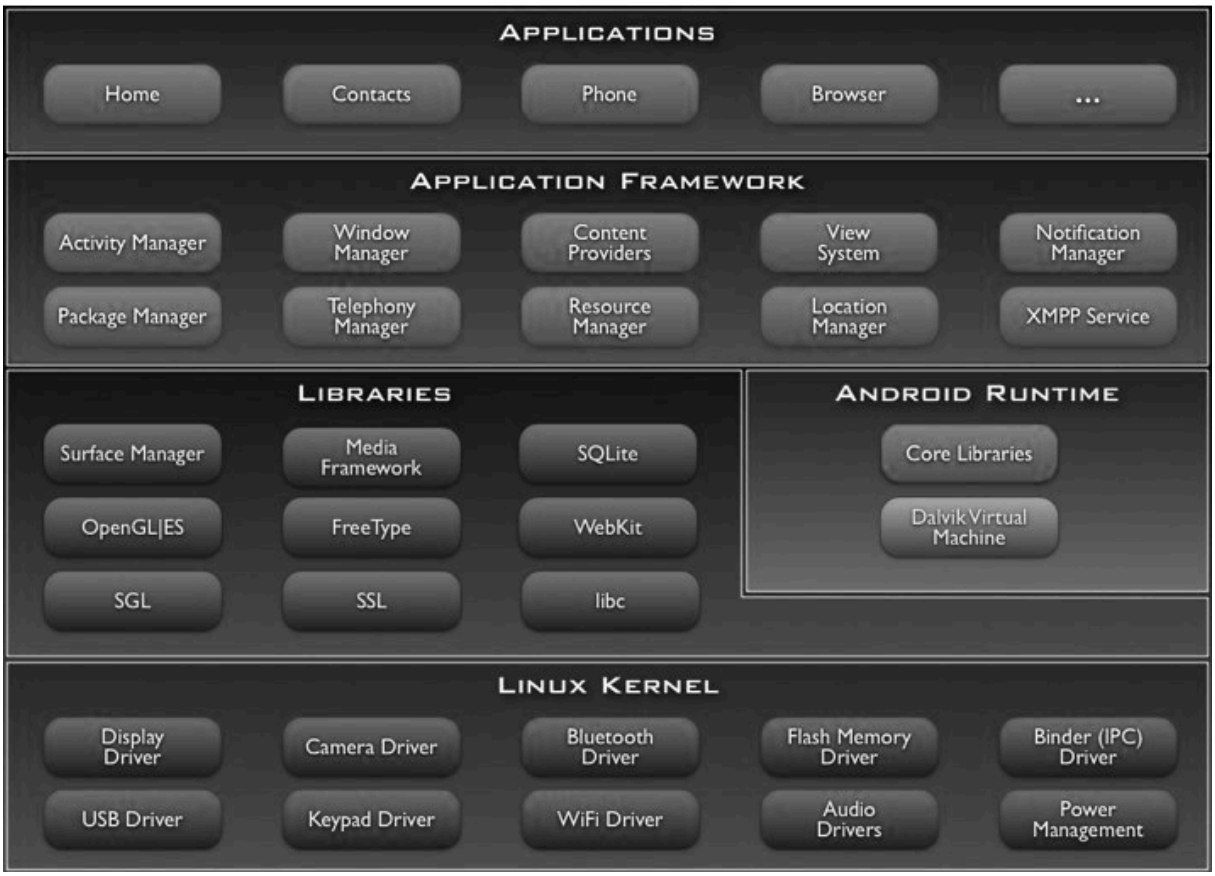

Figura 7.1 – L'architettura del sistema operativo Android.

Un'applicazione per Android viene scritta attraverso la combinazione di **Activity, Intent, Service e Content Provider**. Dopo aver determinato quali di questi blocchi devono essere utilizzati, essi vengono indicati e descritti nel file *AndroidManifest.xml,* che rappresenta il file di impostazioni dell'applicazione.

7.1.1 Le Activity

Le activity sono gli elementi base che vanno a costruire un'applicazione Android e rappresentano una schermata d'interazione con l'utente. La piattaforma Android mantiene in primo piano una sola activity alla volta, portando in secondo piano tutte le altre, memorizzandole in un *"activity stack"* gestito dall'Activity Manager che sta nel terzo strato dell'architettura di Android (figura 7.1).

Una activity può trovarsi nei seguenti stati:

- **Running**: l'activity è in primo piano e pronta per l'interazione con l'utente
- **Paused**: l'utente non può interagire con l'activity ma il layout rimane visibile.
- **Stopped**: quando l'utente passa ad un'altra activity, la precedente viene portata in secondo piano e messa in stato di stop. Le activity in "stopped" sono le prime ad essere terminate in caso di scarsità di risorse.

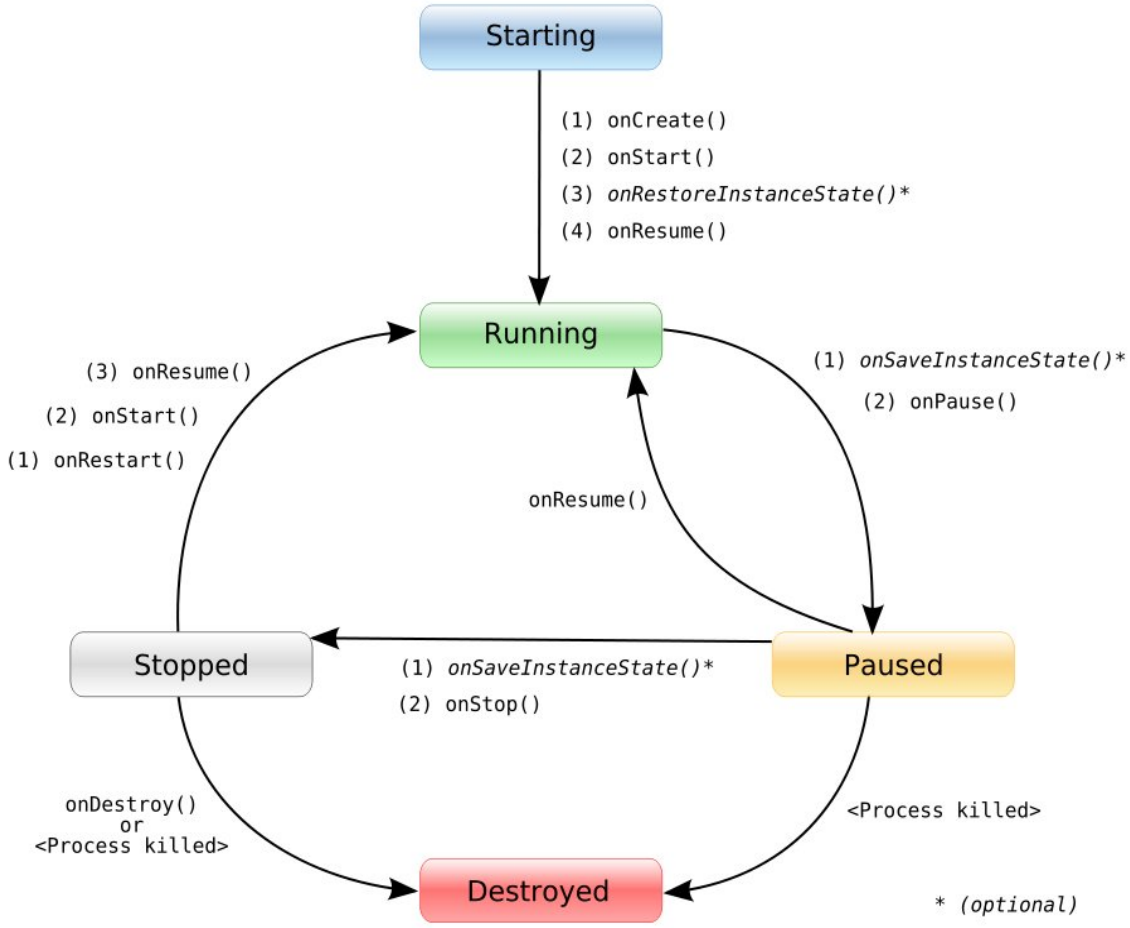

Figura 7.2 – Il ciclo di vita delle activity su Android

Nella figura 7.2 viene descritto il ciclo di vita delle activity di Android. I metodi elencati nella figura sono gestiti da Android stesso, e possono subire un *override* per permettere allo sviluppatore di decidere quali azioni intraprendere a seguito di un particolare stato. Descriviamo ora i metodi e i cambiamenti di stato del ciclo di vita delle activity.

- **onCreate():** è richiamato quando l'activity viene avviata per la prima volta ed è il metodo necessario alla sua inizializzazione.
- **onStart():** indica che l'activity viene visualizzata sullo schermo
- **onResume():** richiamato quando l'activity inizia l'interazione con l'utente
- **onPause():** richiamato quando l'activity sta per andare in background, solitamente perché è stata avviata un'altra activity pronta ad andare in primo piano. E' qui che il programmatore specifica le operazioni riguardanti il salvataggio dello stato raggiunto dall'applicazione.
- **onStop():** richiamato quando l'activity non è più visibile all'utente.
- **onRestart():** l'activity diventa visibile dopo lo stato ***PAUSE***.
- **onDestroy():** richiamato un attimo prima che l'activity viene distrutta
- **onSaveInstanceState():** Android invoca questo metodo per salvare alcune informazioni di stato dell'activity.
- **onRestoreInstanceState():** richiamato solo se alcuni stati dell'activity sono stati precedentemente salvati con il metodo **onSaveInstanceState**().

La visualizzazione sul display viene fornita dalle activity attraverso le viste o View. Una View è la classe di base per la definizione dell'interfaccia grafica e la gestione degli eventi nati dall'interazione con l'utente. L'interfaccia grafica di un'Activity viene in genere definita da una o più **View**, organizzate in una struttura ad albero e rappresentate sullo schermo tramite un gruppo di viste ,**ViewGroup**, chiamato anche **Layout** (figura 7.3).

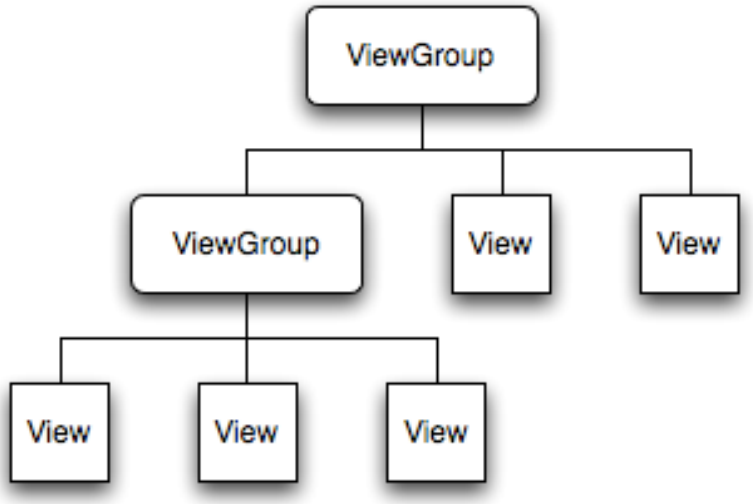

Figura 7.3 – Il ViewGroup su Android

Sia la View che il Layout possono essere definite completamente attraverso codice JAVA oppure tramite un file XML. L'utilizzo del file XML è consigliato dalla documentazione Android in quanto meno complicato e facilmente modificabile. Tuttavia Android fornisce il pieno supporto per entrambe le tipologie di programmazione.

7.1.2 Gli Intent

Per il passaggio da un'activity all'altra e per l'avvio di qualsiasi activity, Android utilizza gli Intent. Gli Intent rappresentano un meccanismo di gestione delle activity attraverso la descrizione astratta di un'operazione che dovrà essere compiuta. Un intent può richiamare un activity richiedendone, se necessario, la restituzione di un risultato.

Le informazioni più importanti contenute all'interno della chiamata con l'intent sono:

- **Action**: ovvero l'azione che deve essere compiuta, come ad esempio: ACTION_VIEW, ACTION_EDIT, ACTION_MAIN.
- **Data**: Il dato in cui deve operare, come ad esempio un record relativo a una persona nel database dei contatti, espresso attraverso un URI.

Esistono inoltre una serie di attributi secondari che possono essere inclusi, come ad esempio:

- **Category**: la categoria dell'action (ad esempio: LAUNCHER, ALTERNATIVE, ecc..).
- **Type**: il tipo dei dati.
- **Component**: specifica un nome esplicito di componente di classe da utilizzare per l'intent.
- **Extras**: un "pacchetto" in grado di contenere ulteriori informazioni per l'intent.

Il meccanismo degli Intent è molto importante nella programmazione Android e offre due notevoli vantaggi:

- Le activity possono riutilizzare funzionalità offerte da altri componenti, semplicemente costruendo un corretto Intent e inserendo in quest'ultimo le informazioni necessarie a soddisfare la richiesta.
- Un'activity può essere completamente sostituita da un'altra.

7.1.3 I Service

Il Service è un componente che viene eseguito in background per un periodo di tempo indefinito, e non presenta interazione con l'utente. Esso viene incluso nello stesso processo dell'applicazione della quale fa parte.

La piattaforma Android fornisce una serie di service predefiniti, richiamabili nell'activity attraverso il metodo getSystemService().

Un service può essere utilizzato per le gestione di transazioni di rete, avvio di musica, operazioni di I/O o l'interazione in background con un content provider e può essenzialmente assumere due forme:

- **Started**: quando un componente di una applicazione, come ad esempio un'activity, avvia un servizio tramite la chiamata startService(). Una volta avviato il servizio rimane in background per un tempo indefinito anche se il componente che lo ha avviato viene distrutto. Solitamente un servizio avviato esegue una singola operazione e non restituisce nessun risultato al chiamante. Ad esempio esso può scaricare o caricare un file attraverso la rete: quando l'operazione si conclude, il servizio si interrompe automaticamente.
- **Bound**: viene effettuato un "bound" (legame) del servizio quando un componente dell'applicazione si lega ad esso richiamando il metodo bindService(). Un servizio bound offre un'interfaccia client-server che permette ai componenti dell'applicazione di interagire con il servizio, ossia di inviare richieste e ottenere risultati. Un servizio bound rimane in esecuzione per lo stesso tempo in cui esso rimane legato al componente chiamante. Componenti multipli possono legarsi al servizio uno alla volta, ma quando questi vengono tutti slegati, il servizio viene distrutto.

Il componente che avvia o fa il bound del Service, può utilizzare il servizio nella stessa maniera in cui esso può usare un'activity avviata tramite intent.

7.1.4 I Content provider

Hanno la funzione di garantire l'accesso a dati condivisi da più applicazioni. Costituiscono l'unica maniera per condividere dati attraverso le applicazioni: non esiste, infatti, un'area comune di memoria alla quale tutti i package di Android possono accedere. Android permette di condividere una serie di tipi di dati anche multimediali come ad esempio file audio, file video, immagini e contatti personali.

E' compito del programmatore gestire l'implementazione del salvataggio dei dati in memoria di massa e devono quindi essere implementate delle interfacce che specificano il modo con il quale le interrogazioni dovranno essere effettuate e la modalità di presentazione del risultato. Ogni Content Provider fornisce un *URI* (***Uniform Resource Identifier***) ai client che vogliono utilizzare i suoi dati: è tramite questa stringa che è possibile effettuare richieste o inserire dati.

7.2 L'applicazione

L'applicazione di questa tesi, sviluppata in JAVA su Android SDK 1.6, e successivamente portata su Android SDK 2.1, è stata strutturata in modo da permettere il funzionamento del sistema descritto nel capitolo 4.

L'ambiente di sviluppo utilizzato per la programmazione è Eclipse unito alla plug-in per la programmazione su Android. Il programma contiene dei moduli in grado di:

- Permettere l'inserimento dei dati utente;
- Leggere e decodificare i codici a barre;
- Garantire la connessione al server per il download dei dati;
- Leggere i dati provenienti dai sensori di movimento;
- Calcolare la nuova posizione attraverso i dati letti;
- Mostrare la posizione all'utente, in tempo reale, su una mappa visualizzata nel display del dispositivo;

Per l'implementazione di queste funzionalità sono state create diverse activity, ed è stato utilizzato il meccanismo degli intent per l'utilizzo di moduli esterni all'applicazione, come ad esempio il lettore di codice a barre.

7.2.1 La struttura dell'applicazione

Nella figura 7.4 vediamo il dettaglio della struttura dell'applicazione chiamata **IndoorNav**.

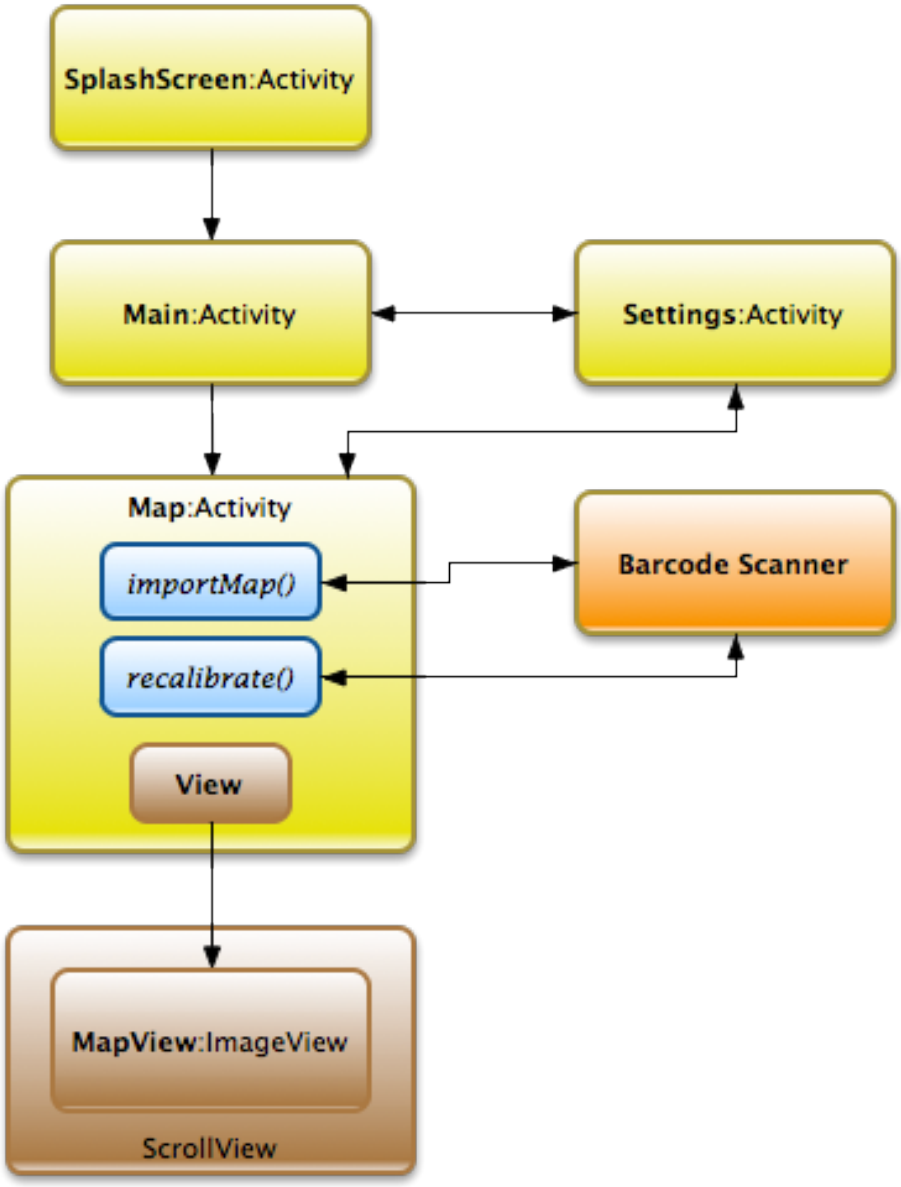

Figura 7.4 – Outline dell'applicazione

L'applicazione è composta dalle seguenti parti:

1. Un'activity di presentazione (***Splashscreen***).
2. Un'activity di calibrazione del sistema e inserimento dei dati iniziali (***Main***).
3. Un'activity per le impostazioni (***Settings***).
4. Un'activity per la mappa (***MapActivity***).
5. Una view che rappresenta la mappa stessa, navigabile tramite lo scrolling (***MapView***).
6. Due metodi per la comunicazione con il **barcode scanner** e l'acquisizione dei dati (***importMap*** e ***recalibrate***).

I dati letti dai sensori, combinati con la mappa, forniscono un'informazione sulla posizione attuale dell'utente. Riassumendo:

- L'utente fa partire l'applicazione
- Inserisce i parametri iniziali.
- Scatta una foto del barcode di partenza.
- L'applicazione scarica la mappa del piano, imposta la posizione iniziale sulla mappa.
- L'applicazione traccia la posizione dell'utente sulla mappa del dispositivo mentre cammina.
- L'utente scatta nuovamente le foto ai barcode che incontra nei checkpoint per la ricalibrazione.

7.2.2 Activity di presentazione

Una semplice activity utilizzata per l'introduzione all'applicazione, contiene una schermata di avvio e un breve tutorial sull'utilizzo dell'applicativo. E' Il modo in cui si presenta il programma all'utilizzatore.

A partire da questa activity è possibile passare solo alla fase di inserimento dei dati iniziali.

7.2.3 Activity "Main" di inserimento dei dati iniziali

Da questa activity si possono inserire le soglie di misurazione e gli altri parametri che vengono memorizzati in un file di preferenze gestito dall'activity per le impostazioni (***SettingsActivity***). In questa versione viene inserita la lunghezza del passo e le soglie per il calcolo del contapassi. Le versioni future dell'applicazione permetteranno all'utente di inserire i suoi dati personali, come descritto nel capitolo 5, e in seguito effettuare una stima dei parametri relativi al passo.

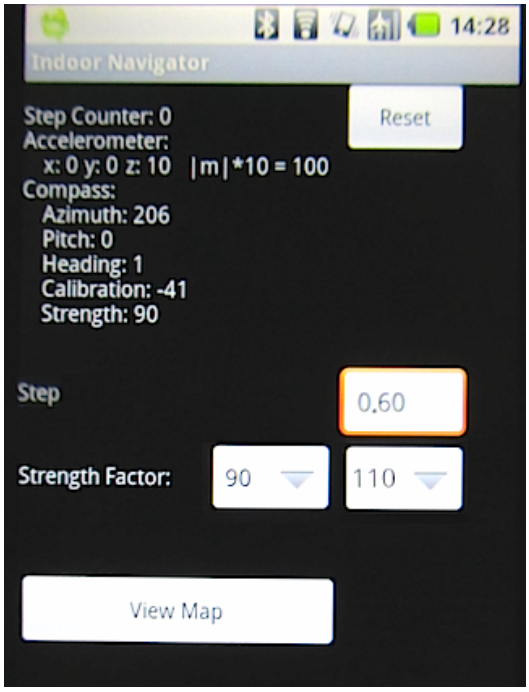

Figura 7.5 – Screenshot dell'activity Main. I valori sono calcolati in base a "modulo*10"

Quest'activity è in grado da sola di funzionare da contapassi: infatti, oltre a permettere l'impostazione dei dati iniziali, esegue la lettura dei dati dei sensori e riconosce il momento in cui è stato compiuto il passo, fornendo un'indicazione sui dati letti da accelerometro e bussola.

I moduli utilizzati per la lettura dei dati dai sensori sfruttano la classe *android.hardware.Sensor*, tramite la quale è possibile implementare dei "*manager*", ossia dei gestori di eventi relativi ai sensori. La classe *android.hardwar.SensorManager* si occupa di gestire questi eventi e la classe *android.hardware.SensorEventListener* ha il

compito di “ascoltarli”. Una parte importante della classe dell’activity dell’activity riguarda la creazione dei sensori e di un SensorManager.

```
SensorManager myManager;
Sensor accSensor;
Sensor orSensor;
myManager =
(SensorManager)getSystemService(Context.SENSOR_SERVICE); accSensor
= myManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER); orSensor
= myManager.getDefaultSensor(Sensor.TYPE_ORIENTATION);
```

A questo punto il manager dei sensori (“*myManager*)” registra un Listener in grado di recuperare i valori dai sensori.

```
myManager.registerListener(mySensorListener, accSensor,
SensorManager. SENSOR_DELAY_FASTEST);
myManager.registerListener(mySensorListener, orSensor,
SensorManager.SENSOR_DELAY_FASTEST);
```

Questa porzione di codice andrà inserita nel metodo *onResume()* dell’activity. La registrazione del *listener* viene quindi annullata nel metodo *onStop()* attraverso la seguente chiamata di metodo:

```
myManager.unregisterListener(mySensorListener);
```

Prima del suo utilizzo il listener deve essere definito indicando quali sensori ha il compito di “ascoltare”. Di seguito vediamo una sua possibile implementazione:

```
mySensorListener = new SensorEventListener() {
          public void onSensorChanged(SensorEvent event) {
               int type = event.sensor.getType();

               switch (type) {
                    case Sensor.TYPE_ACCELEROMETER:
                         accelValues = event.values.clone();
                         break;
                    case Sensor.TYPE_ORIENTATION:
                         orientValues = event.values.clone();
                         break;
               }
               updateValues(accelValues, orientValues);
          }
     }
};
```

Il metodo ***updateValues***, richiamato al variare dei dati dei sensori, riceve le informazioni dall'accelerometro e dal sensore di orientamento, sotto forma di array di dimensione pari a tre. Sono tre infatti i valori registrati dai sensori, relativi ai tre assi (come descritto nel capitolo 3).

```
float accelerationX = accelValues[0];
float accelerationY = accelValues[1];
float accelerationZ = accelValues[2];
float azimuth = orientValues[0];
float pitch = orientValues[1];
float roll = orientValues[2];
```

Attraverso queste variabili relative all'accelerazione e all'orientamento, è possibile gestire l'implementazione dell'algoritmo per il riconoscimento del passo secondo le modalità descritte nel capitolo 5. Con il pulsante *"View Map"* dalla schermata Main è possibile accedere all'activity contenente la mappa.

7.2.4 Activity e View per la gestione della mappa

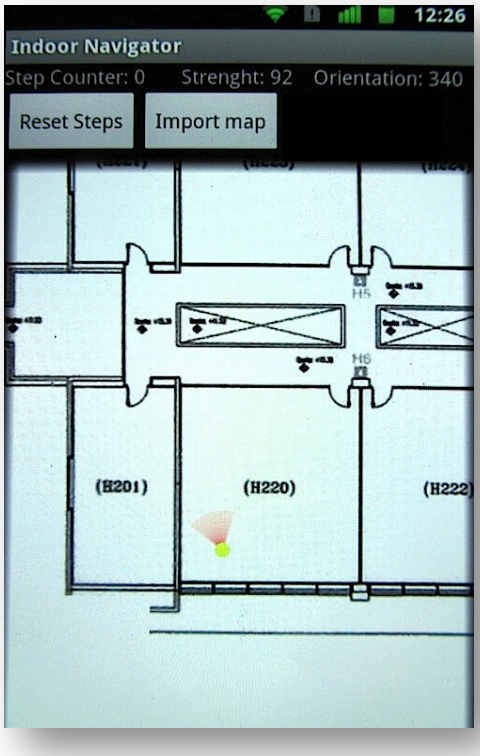

Figura 7.6 – Screenshot della schermata relativa alla mappa

Nella figura 7.6 viene mostrato lo screenshot con la view gestita dall'activity della mappa, registrato subito dopo l'acquisizione dei dati dal codice a barre.

L'activity relativa alla mappa permette all'utente di tenere traccia del suo spostamento, attraverso la visualizzazione grafica in una mappa del piano.

E' strutturata in modo da avere una view con un contatore di passi, un tasto di reset (del numero di passi) e un pulsante per l'import della mappa e dei dati iniziali in grado di richiamare l'activity esterna di lettura dei codice a barre attraverso i metodi ***importMap*** e ***recalibrate***.

Anche con quest'activity è possibile impostare le preferenze di **IndoorNav**, ossia la lunghezza del passo e in uno sviluppo futuro i dati completi dell'utente. Questo permette l'utilizzo dell'applicativo da parte di più utenti, con una lunghezza di passo diversa, senza passare attraverso l'activity principale.

I moduli per l'interpretazione del movimento dell'utente sono uguali a quelli descritti nel paragrafo precedente (activity ***Main***), con la differenza che in questo caso i dati sono elaborati in modo da avere un riferimento visivo diretto sulla mappa caricata nel display. Il passo viene determinato secondo l'algoritmo e le modalità descritte nel capitolo 5. Il calcolo della direzione utente e la rappresentazione della posizione viene fatto secondo la descrizione presente nel capitolo 6.

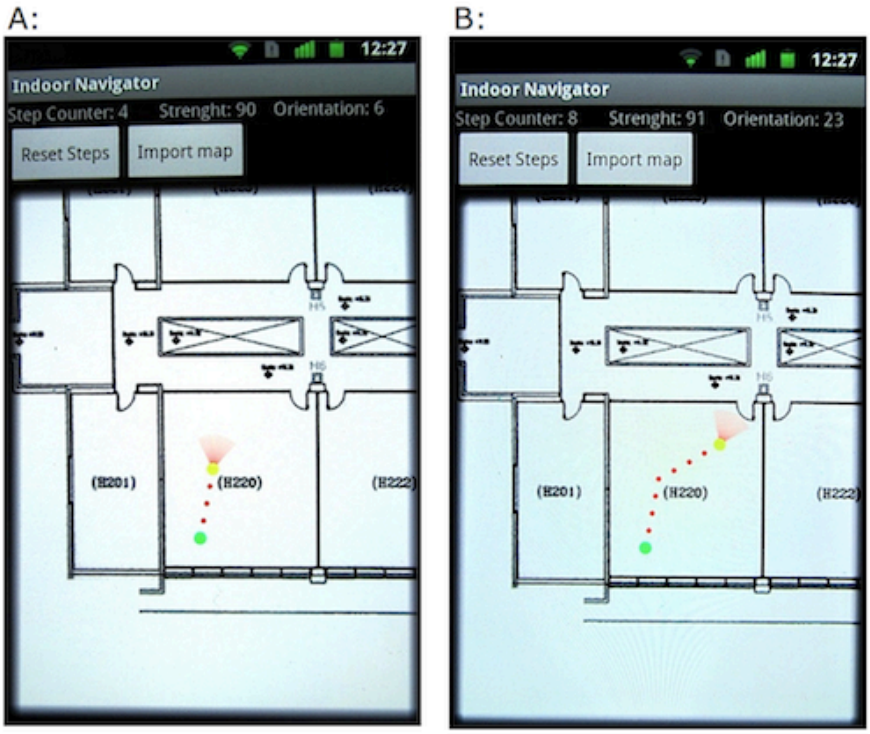

Figura 7.7 – I passi dell'utente sulla mappa.

Nella figura 7.7 vediamo un esempio di posizionamento e tracciamento dell'utente sulla mappa, durante l'utilizzo di *IndoorNav*. Ogni passo viene disegnato e rappresentato da un punto rosso, a partire da una posizione di partenza indicata con il punto verde. La figura 7.7A mostra la posizione dell'utente dopo quattro passi (step counter = 4) e la figura 7.7B mostra la posizione dopo otto passi. Si nota dalla figura il cambio di direzione dopo quattro passi.

7.2.5 Modulo esterno per la scansione del barcode

Questa parte della tesi è stata sviluppata in collaborazione con il ***DIEE*** di Cagliari.

Il lettore di codice a barre utilizzato consiste in un modulo esterno all'applicativo vero e proprio richiamato grazie al meccanismo degli Intent.

Si tratta di una versione modificata dell'applicazione di lettura del codice a barre, *Barcode Scanner*, che sfrutta la libreria ***ZXing*** (si pronuncia *Zebra Crossing*), in modo

da funzionare anche da misuratore di distanza e orientamento rispetto al barcode come descritto nel capitolo 6.

La libreria ***ZXing*** (http://code.google.com/p/zxing/) è una libreria open-source di decodifica di barcode monodimensionali e bidimensionali sviluppata in linguaggio Java da Google. Si tratta di una libreria che permette l'utilizzo della fotocamera dello smartphone per la decodifica di barcode, senza nessun tipo di interazione con un server di supporto.

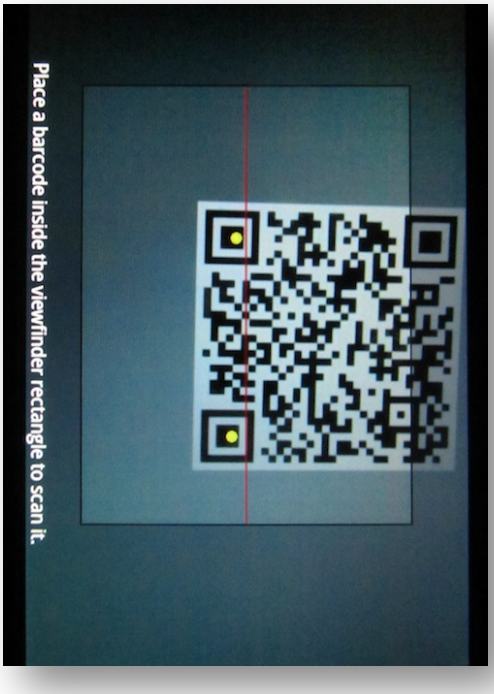

Figura 7.8 – Screenshot della schermata relativa all'acquisizione barcode

Il modulo per la scansione del barcode permette due modalità di utilizzo:

- Una "semplice" modalità in cui è chiesto all'utente di disporre il dispositivo ad una distanza di circa 20-30 cm dal barcode in posizione parallela alla parete
- Una "complessa" in cui l'utente può fare la scansione del barcode anche in posizione non parallela alla parete. E' in questo caso che vengono calcolate la distanza corretta e l'orientamento dello smartphone rispetto al codice a barre.

E' compito dell'activity della mappa richiamare la scansione del barcode.

La porzione di codice per la comunicazione con il "barcode scanner" consiste nel creare un nuovo intent descrivendo la libreria desiderata attraverso il tag *"com.google.zxing.client.android.SCAN"*. L'Intent relativo viene quindi richiamato in questo modo:

```
Intent intent = new
Intent("com.google.zxing.client.android.SCAN");
intent.putExtra("SCAN_MODE", "QR_CODE_MODE");
startActivityForResult(intent, 0);
```

Attraverso questa chiamata, l'activity di scansione del barcode riceve due indicazioni: una indica la modalità di scansione ("SCAN_MODE"), l'altra indica il tipo di codice a barre che desideriamo decodificare ("QR_CODE_MODE").

La stringa decodificata verrà letta dal metodo ***onActivityResult***, richiamato al termine della scansione all'interno dell'activity chiamante, attraverso questa riga di codice:

```
String contents = intent.getStringExtra("SCAN_RESULT");
```

Con il tag "SCAN_RESULT" si richiede il risultato della scansione e la stringa "*contents*" conterrà la decodifica del codice a barre che nel nostro caso equivale all'URL del file **XML** associato al **codice a barre QR**.

8 La sperimentazione

La sperimentazione descritta in questo capitolo ha permesso di validare il corretto funzionamento dell'applicativo ***IndoorNav***, e analizzare attraverso i test, in che modo il sistema dovrà essere perfezionato negli sviluppi futuri. Lo scenario scelto è quello dell'edificio 1 del *Parco Scientifico Tecnologico* di Pula, localizzato più precisamente in un settore della sede del ***CRS4***.

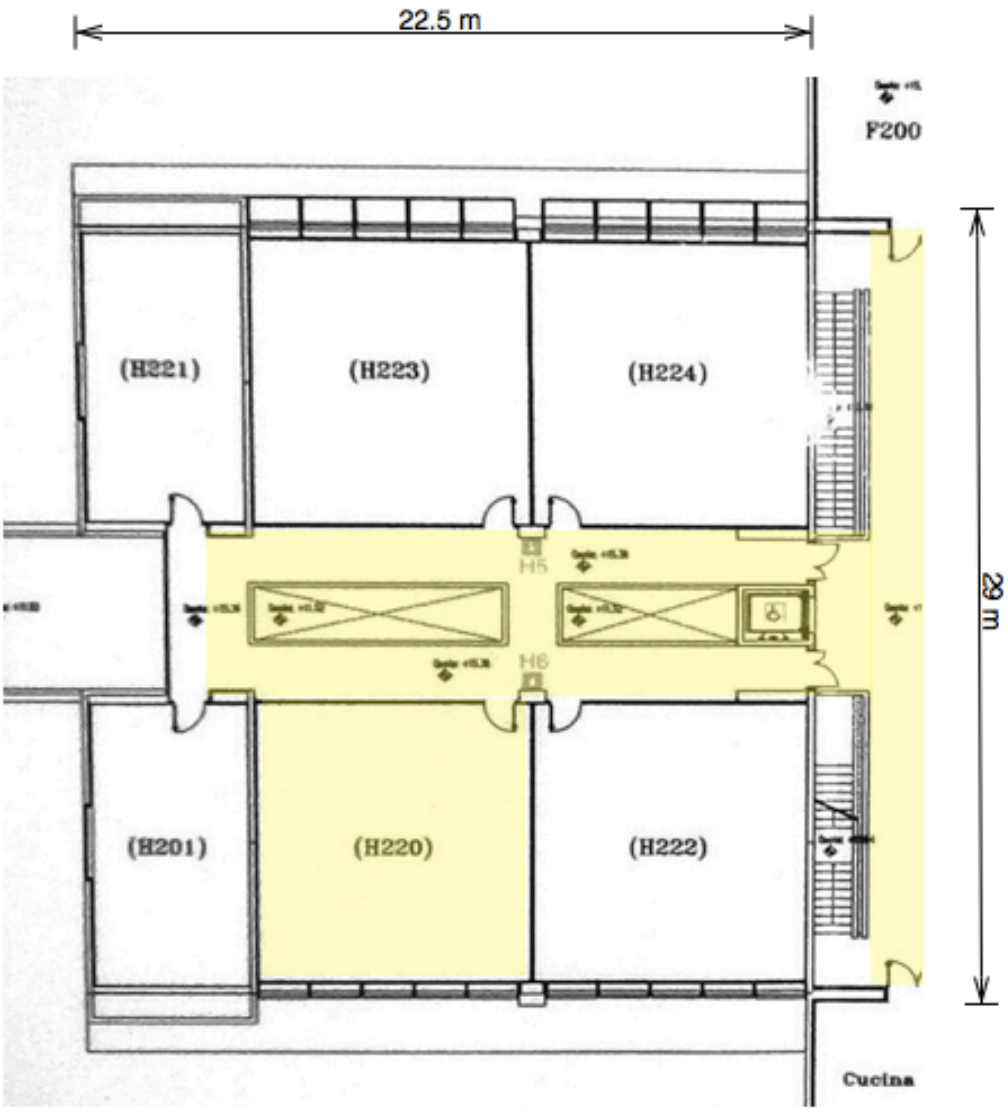

Figura 8.1 – In evidenza le aree in cui è stato sperimentata l'applicazione.

Nello scenario sono stati definiti:

- I percorsi per i test
- Il server web
- I file XML, i barcode e la localizzazione dei checkpoint
- La stampa e la disposizione dei barcode
- L'installazione dell'applicazione nello smartphone

Gli smartphone utilizzati per le prove sono l'**HTC T-Mobile G1** e il **Motorola Milestone** con sistema operativo Android 1.6, e Android 2.1 rispettivamente. Nei paragrafi successivi sono descritti nei dettagli i dispositivi e i test effettuati.

8.1 Caratteristiche dei dispositivi usati

I dispositivi devono possedere determinate caratteristiche, ovvero devono essere dotati di sensori di movimento, display touch medio-grande per la visualizzazione della mappa, fotocamera e connessione a internet.

Nelle tabelle seguenti è presente una descrizione dettagliata dell'HTC G1 e del Motorola Milestone.

HTC T-Mobile G1:

GENERAL	**2G Network**	GSM 850 / 900 / 1800 / 1900
	3G Network	HSDPA 2100
SIZE	**Dimensions**	117 x 55.7 x 17.1 mm
	Weight	158 g
DISPLAY	**Type**	TFT capacitive touchscreen, 65K colors
	Size	320 x 480 pixels, 3.2 inches
SOUND	**Alert types**	Vibration; MP3, WAV ringtones
	Speakerphone	Yes, with stereo speakers

MEMORY	**Internal**	192 MB RAM, 256 MB ROM
	Card slot	microSD
DATA	**GPRS**	Class 10 (4+1/3+2 slots), 32 - 48 kbps
	EDGE	Class 10, 236.8 kbps
	3G	HSDPA, 7.2 Mbps; HSUPA, 2 Mbps
	WLAN	Wi-Fi 802.11 b/g
	Bluetooth	Yes, v2.0 with A2DP , headset support only
	USB	Yes, miniUSB
CAMERA	**Primary**	3.15 MP, 2048 x 1536 pixels, autofocus
	Video	Yes
FEATURES	**OS**	Android OS 1.6
	CPU	528 MHz ARM 11 processor, Adreno 130 GPU, Qualcomm MSM7201A chipset
	GPS	Yes
	Digital Compass	Yes
	Accelerometer	Yes

Figura 8.3 – HTC T-Mobile G1

Motorola Milestone:

GENERAL	**2G Network**	GSM 850 / 900 / 1800 / 1900
	3G Network	HSDPA 900 / 2100
		UMTS 850 / 1900 - American version
SIZE	**Dimensions**	115.8 x 60 x 13.7 mm
	Weight	165 g
DISPLAY	**Type**	TFT capacitive touchscreen, 16M colors
	Size	480 x 854 pixels, 3.7 inches
		- Multi-touch input method
SOUND	**Alert types**	Vibration; MP3, WAV ringtones
	Speakerphone	Yes, with stereo speakers
MEMORY	**Internal**	133 MB storage, 256 MB RAM
	Card slot	microSD, up to 32GB, 8GB included
DATA	**GPRS**	Class 12 (4+1/3+2/2+3/1+4 slots), 32 - 48 kbps
	EDGE	Class 12
	3G	HSDPA, 10.2 Mbps; HSUPA, 5.76 Mbps
	WLAN	Wi-Fi 802.11 b/g
	Bluetooth	Yes, v2.1 with A2DP
	USB	Yes, microUSB v2.0
CAMERA	**Primary**	5 MP, 2592 x 1944 pixels, autofocus, dual-LED flash
	Features	Geo-tagging
FEATURES	**OS**	Android OS, v2.1
	CPU	ARM Cortex A8 600 MHz, PowerVR SGX530 graphics
	GPS	Yes, with A-GPS support
	Digital Compass	Yes
	Accelerometer	Yes

Figura 8.2 – Motorola Milestone

La differenza sostanziale nelle specifiche dei due smartphone sta nel fatto che l'HTC G1 ha una capacità di calcolo e un quantitativo di memoria RAM minore rispetto al Milestone, e sommando a ciò la presenza di una versione dell'Android SDK più vecchia (1.6 contro 2.1), probabilmente tutto questo influirà sui vari errori di calcolo da parte dei sensori durante l'utilizzo dell'applicazione.

Andando ancora più nel dettaglio per quanto riguarda la sensoristica nell'HTC G1 è presente il chip AK8976A della AKM. E' un sensore a 6 assi composto da un magnetometro a 3 assi e un accelerometro (sempre a 3 assi). Questo ha il vantaggio che se si interroga lo stato dell'accelerometro, del magnetometro o di entrambi, il sensore consuma lo stesso quantitativo di energia. [27]

Sul Motorola Milestone invece la misura dell'accelerazione viene fornita dal chip LIS331DLH della ST [28]. Per quanto riguarda invece la misurazione del campo magnetico è stato installato il chip AK8973 della AKM [27]. In questo caso, a differenza dell'HTC G1, le misurazioni vengono fatte da due sensori diversi.

8.2 Risultati ottenuti

In questo paragrafo sono descritti i test effettuati con l'applicazione, e precisamente sull'algoritmo del conteggio dei passi e del calcolo della nuova posizione. Sono stati fatti tre tipi di test:

- **Test sull'affidabilità del contapassi.**
- **Test relativo al posizionamento su mappa.**
- **Test sulla calibrazione della distanza e dell'orientamento rispetto ai barcode.**

8.2.1 Test sull'affidabilità del contapassi

Questo test mira a verificare l'affidabilità del conta passi nel riconoscere un passo. Durante la fase di camminata, infatti, possono verificarsi dei *ghost steps (*o passi fantasma), ossia dei passi misurati senza che l'utente abbia compiuto nessuno spostamento. Solitamente i passi fantasma si verificano a seguito di un errato utilizzo del dispositivo, ad esempio un brusco movimento della mano o del braccio, oppure a seguito di una impostazione di soglie di accelerazione non adatte all'utente.

Hanno partecipato a questo test sei persone adulte, tre di sesso maschile, tre di sesso femminile. E' stato chiesto loro di seguire una traiettoria in linea retta tracciata sul pavimento lunga circa 27.2 metri (indicata nella figura 8.4), eseguendo esclusivamente 40 passi.

Volutamente durante il test, si è scelto di mantenere una soglia impostata ai valori prossimi a quelli misurati sul primo tester (l'utente M1) che equivale a ***Th_max: 11*** (soglia massima)*,* ***Th_min: 9.5*** (soglia minima) espressi in m/s^2, relativi al modulo dell'accelerazione*.* La lunghezza del passo è stata impostata a 0.68 metri, ma in questo caso costituisce un dato superfluo poiché il test prevede solo la misurazione della bontà del contapassi in base ad una soglia fissata all'inizio del test e su un campione di individui scelti a caso.

I dispositivi utilizzati durante la prova sono l'**HTC G1**, e il **Motorola Milestone** descritti nel paragrafo 8.1. E' stato chiesto ai tester di tenere con una mano lo smartphone in posizione verticale, a circa 50° d'inclinazione rispetto al pavimento, durante la camminata, senza effettuare bruschi movimenti.

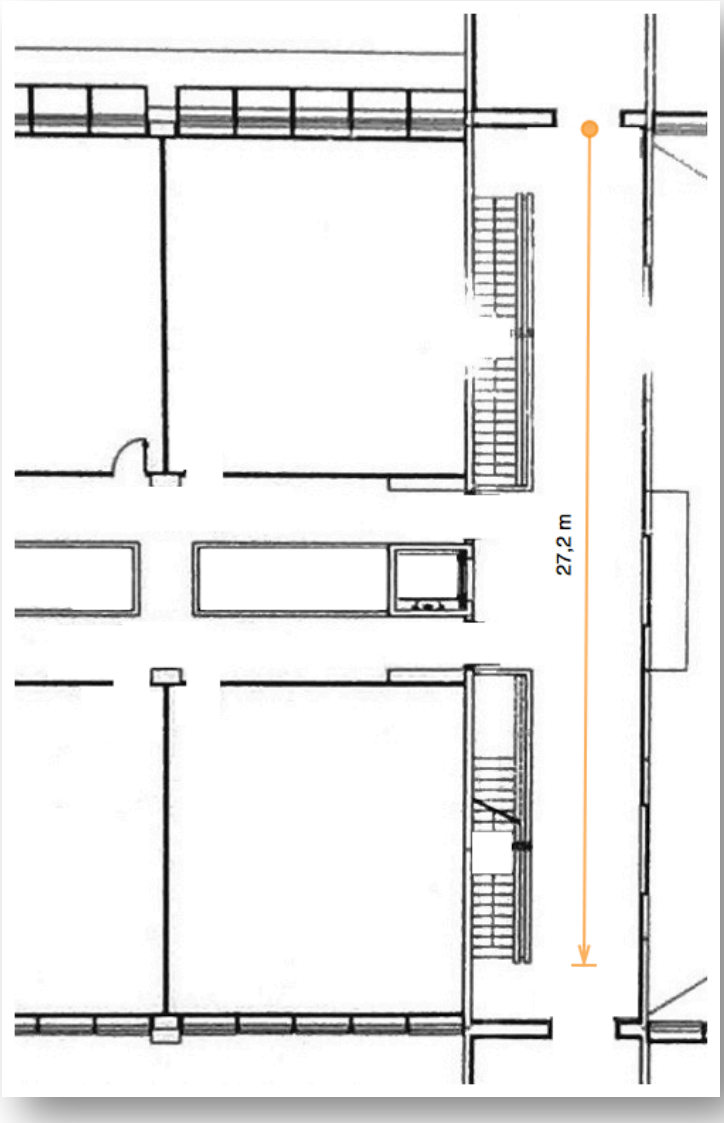

Figura 8.4 – In arancione il percorso rettilineo usato per i test.

8.2.1.1 Test HTC G1

HTC G1

Totale step: 40 **Lunghezza: 27,2 m** **Lunghezza step media: 0.68 m** **th_min: 9.5** **th_max:11.0**

Testers	Run 0	Run 1	Run 2	Run 3	Run 4	Run 5	Run 6	Run 7	Media passi	Errore medio %
M1	41	38	39	40	40	41	40	40	39.875	0.3125
M2	38	40	43	42	40	42	42	42	41.125	2.8125
M3	38	39	38	39	39	39	40	40	39	2.5
F1	37	38	39	38	37	36	40	38	37.875	5.3125
F2	39	39	37	38	38	38	37	35	37.625	5.9375
F3	35	36	38	38	38	39	37	40	37.625	5.9375

	Errore Medio Totale %
M	1.875
F	5.729166667
TOT M+F	3.802083333

L'errore sui dati riguardanti gli utenti di sesso femminile, F1, F2, F3 risulta maggiore degli utenti di sesso maschile M1, M2, M3. Questo perché le soglie impostate, come già accennato all'inizio del paragrafo, erano relative all'utente M1 (40 passi per 27,2 m). La percentuale di errore su M1, infatti, lo conferma: 0.31 %, ossia meno di un passo di errore su 40. La bassa percentuale totale di errore degli utenti maschi (Errore Medio Totale M %) dimostra anche che i tester M2 e M3 molto probabilmente presentano caratteristiche fisiche simili, e quindi stesso tipo di camminata, al tester M1. Anche in questo caso il contapassi si è dimostrato abbastanza affidabile, poco più di un *ghost step* su 40.

In definitiva questi risultati dimostrano che con un'impostazione delle soglie accurata si può avere un errore molto piccolo di misurazione, con una bassa probabilità di *passi fantasma*. Viceversa, la stessa soglia su una persona di diverso sesso e diversa corporatura può portare a un errore del 5-6%, su una distanza di 27,2 metri, ossia circa 2-3 passi erroneamente misurati su 40.

Questo test porta ad affermare che la calibrazione iniziale è una fase molto importante per il successivo processo di elaborazione dati, e suggerisce allo stesso tempo, una distanza ottimale nella quale effettuare una ricalibratura del sistema. In questo caso un checkpoint ogni 25 metri o meno.

8.2.1.2 Test Motorola Milestone

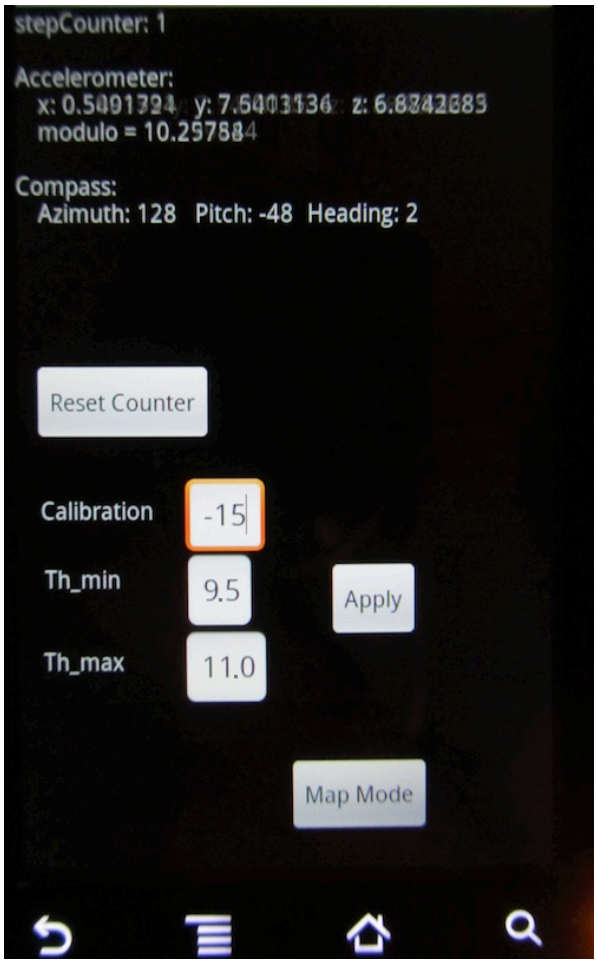

Figura 8.5 – Schermata dell'IndoorNav durante il test sul contapassi.

Dalla schermata rappresentata nella figura 8.5 si nota il valore di pitch a 48° che indica l'inclinazione verticale dello smartphone tenuto in mano dal tester, e i valori di soglia massima (***th_max***) e minima (***th_min***).

Motorola Milestone

Totale step: 40 **Lunghezza: 27,2 m** **Lunghezza step media: 0.68 m** **th_min: 9.5** **th_max:11.0**

Testers	Run 0	Run 1	Run 2	Run 3	Run 4	Run 5	Run 6	Run 7	Media passi	Errore medio %
M1	40	40	40	40	40	40	41	40	40.125	0.3125
M2	39	39	41	40	39	45	42	42	40.875	2.1875
M3	40	39	43	39	39	45	40	40	40.625	1.5625
F1	37	38	39	37	38	39	40	38	38.25	4.375
F2	39	39	40	38	38	38	39	39	38.75	3.125
F3	40	39	37	38	37	39	36	40	38.25	4.375

	Errore medio Totale %
M	1.354166667
F	3.958333333
TOT M+F	2.65625

Il Motorola Milestone, con il suo sensore inerziale LIS331DLH dedicato alla misura delle accelerazioni, è meno soggetto a errori rispetto all'HTC.

La prova dimostra che il contapassi è molto accurato e ha fornito un risultato accettabile su una lunghezza di 27.2 metri, con un campione scelto a caso, e con una soglia scelta in partenza e non modificata durante lo svolgimento dell'esperimento.

8.2.2 Test relativo al posizionamento su mappa

E' un tipo di test differente da quello descritto nel paragrafo precedente. Nel caso descritto in questo paragrafo non si misura la precisione del contapassi, il quale si è dimostrato molto affidabile impostando una soglia adatta all'utente, ma il corretto posizionamento dell'utente sulla mappa di riferimento nel dispositivo.

Si misura quindi la performance del *dead reckoning* e le stime dei posizionamenti successivi, analizzando un dato che contiene al suo interno una rappresentazione degli errori relativi sia all'orientamento della bussola sia al contapassi.

Questo esperimento è stato condotto da un singolo utente con soglia di misurazione del passo correttamente preimpostata, e sono stati eseguiti sia sull'HTC G1 sia sul Motorola Milestone.

E' stato scelto un percorso rettangolare, in modo da avere una misurazione accurata su quattro svolte successive da parte della bussola, lungo 9,07 metri e largo 3,02 metri.

Lunghezza totale percorso: 24,18 m.

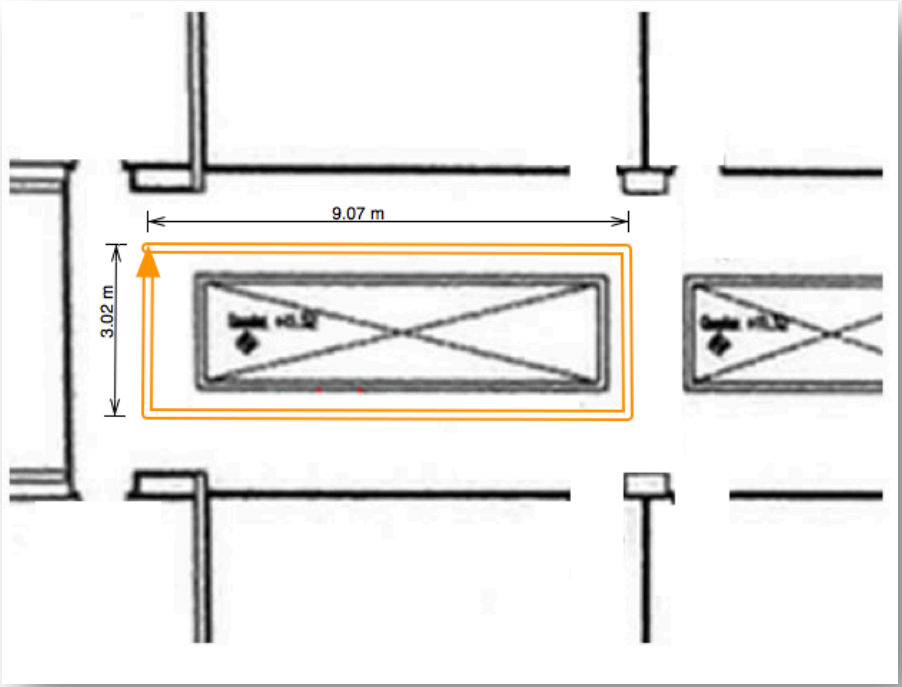

Figura 8.6 – Il dettaglio del percorso scelto per il test. Lunghezza totale 24.18 metri.

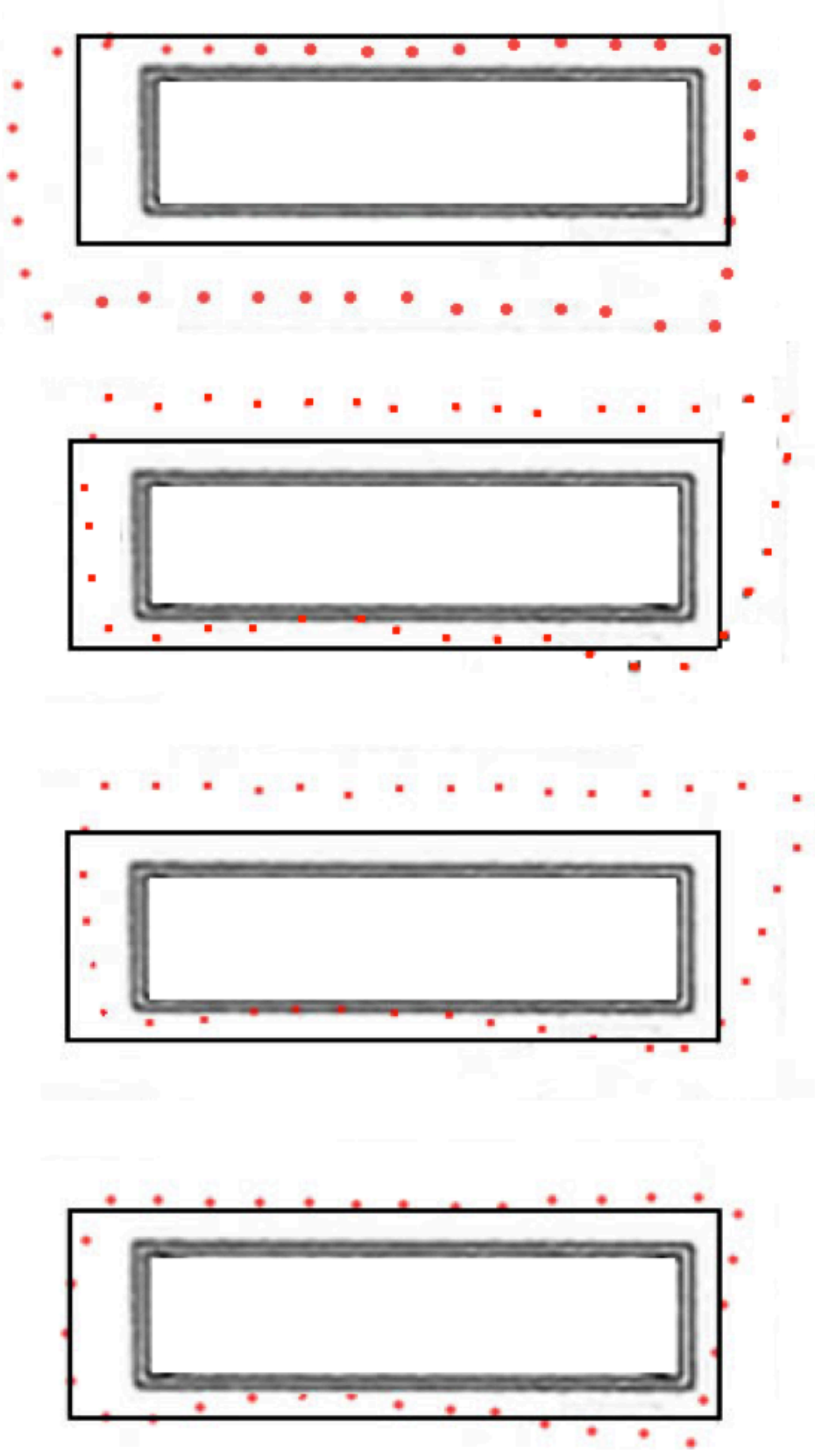

Figura 8.7 – I quattro test relativi al posizionamento

Nella figura 8.7 sono rappresentati i test sul posizionamento. Le prime due immagini a partire dall'alto, sono relative all'HTC, le altre due al Motorola.

I test sono stati eseguiti facendo un confronto grafico dell'immagine di punti registrata durante i passi rispetto al percorso originario tracciato sul pavimento e indicato sulla mappa visualizzata dal dispositivo. L'errore è stato prima misurato in pixel e poi rapportato in metri. In relazione ad ogni passo, l'errore equivale alla distanza del punto del passo registrato rispetto al perimetro del percorso originario.

In questi test si tiene conto anche di un eventuale errore di 10 cm, dovuto al fatto che l'utente che si muove con il telefono in mano, non esegue perfettamente il passo lungo la linea.

Di seguito sono indicati i grafici relativi ai dati registrati e gli errori.

8.2.2.1 Test HTC G1

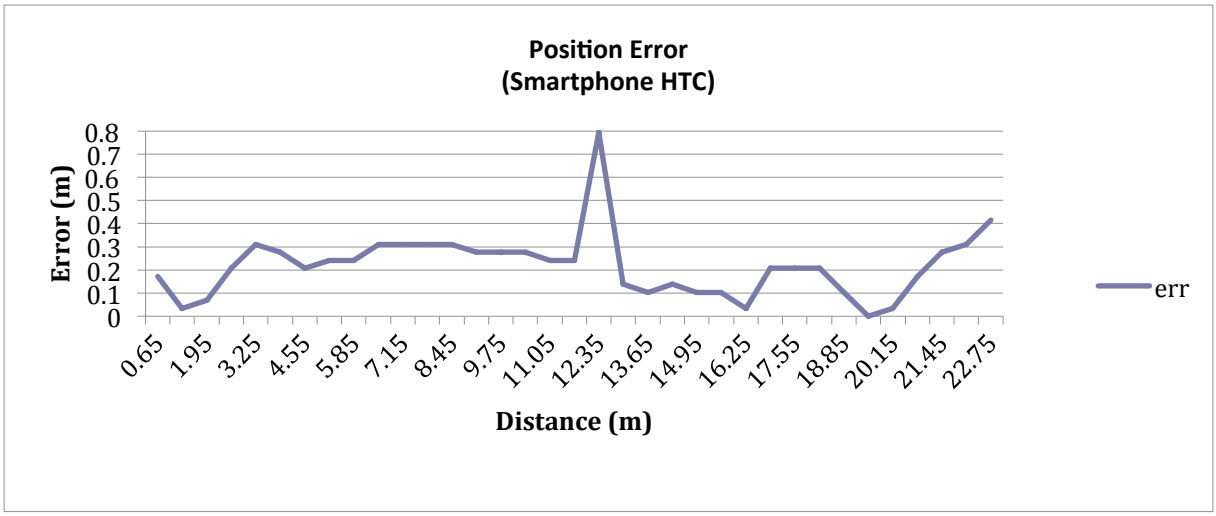

Figura 8.8 – Grafico errori posizionamento relativi al primo test con l'HTC G1

Errore medio = 0.22 m

Scarto quadratico medio errore = 0.14 m

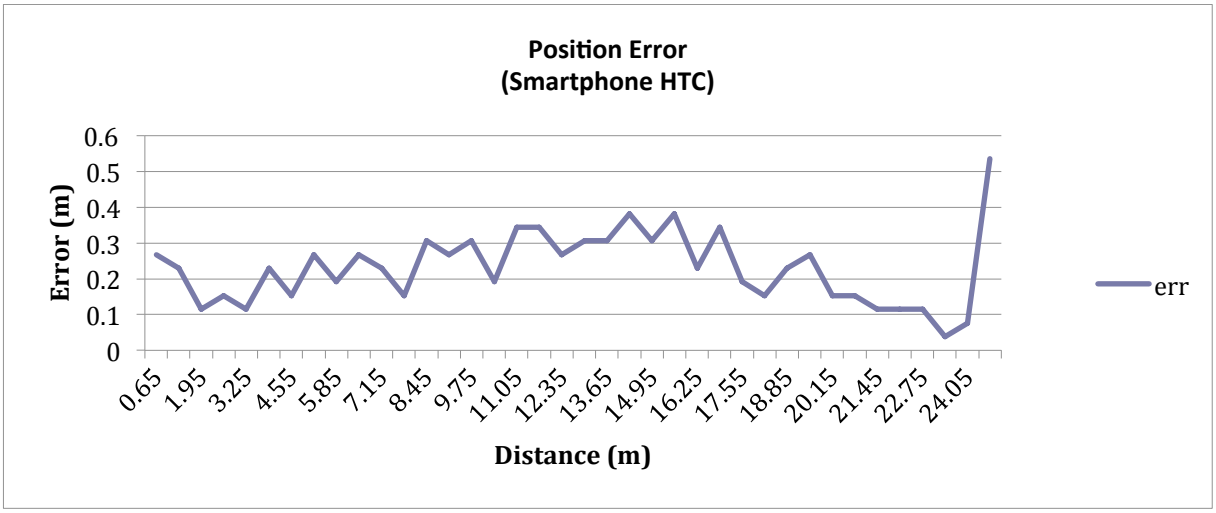

Figura 8.9 – Grafico errori posizionamento relativi al secondo test con l'HTC G1

Errore medio = 0.23 m

Scarto quadratico medio errore = 0.10 m

8.2.2.2 Test Motorola Milestone

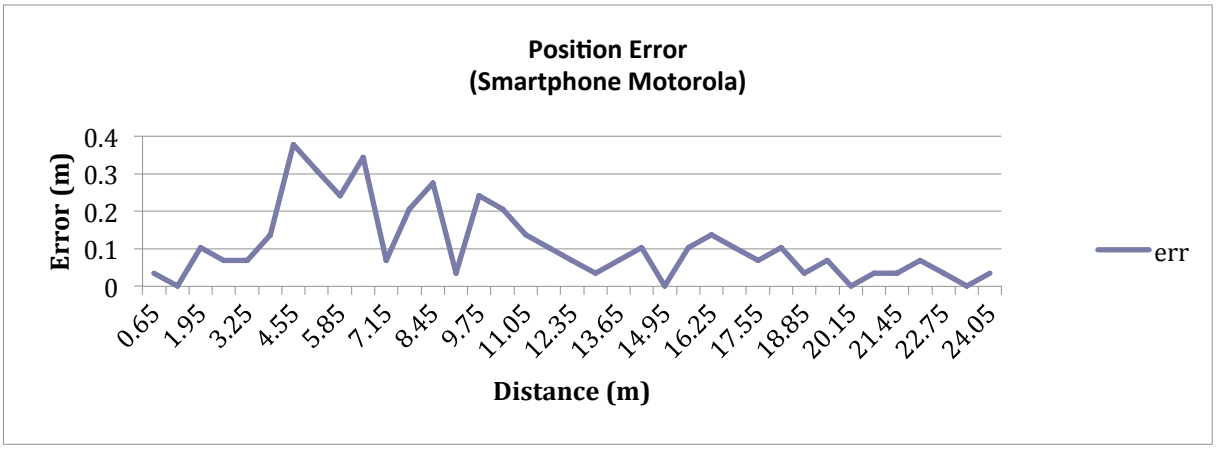

Figura 8.10 – Grafico errori posizionamento relativi al primo test con il Motorola Milestone

Errore medio = 0.11 m

Scarto quadratico medio errore = 0.10 m

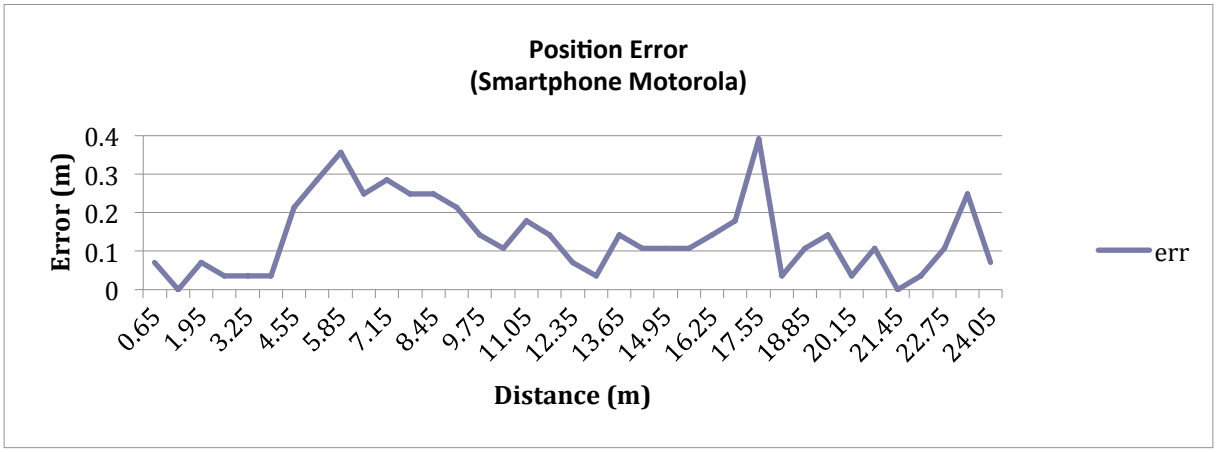

Figura 8.11 – Grafico errori posizionamento relativi al secondo test con il Motorola Milestone

Errore medio = 0.13 m

Scarto quadratico medio errore = 0.10 m

I quattro test, rappresentati nei grafici precedenti (Figure 8.8, 8.9, 8.10 e 8.11), hanno mostrato che il Motorola Milestone riesce a fornire un posizionamento più accurato su mappa rispetto all'HTC G1. Questo è molto probabilmente dovuto al fatto che il dispositivo ha una capacità di calcolo superiore all'HTC, un firmware più recente, e utilizza due sensori separati per l'accelerazione e il calcolo del campo magnetico terrestre.

Tuttavia è da notare che l'errore medio non è dovuto solo da un errato posizionamento dell'utente sulla mappa da parte dello smartphone, ma anche da una componente di errore relativa all'utente stesso mentre tiene il cellulare. E' stato deciso quindi di aggiungere un'indeterminazione di circa 5 cm nel compiere il passo sulla linea tracciata e un altro errore di circa 5 cm riguardante gli altri movimenti errati.

8.2.3 Test sulla calibrazione della distanza e dell'orientamento rispetto ai barcode

Questo test effettua una stima degli errori sul posizionamento dello smartphone rispetto a dei barcode posizionati su una parete. I barcode utilizzati sono:

- Un barcode QR contenente un'informazione di **32 caratteri** stampato con un lato di **18 cm** (figura 8.12)
- Un barcode QR con una informazione di **64 caratteri** stampato con un lato **di 19 cm** (figura 8.13)

Figura 8.12 – Il barcode QR da 18 cm con 32 caratteri di informazione codificati

Figura 8.13 – Il barcode QR da 19 cm con 64 caratteri di informazione codificati

I codici a barre QR crescono in pixel al crescere dell'informazione, assumendo una struttura "più complessa". E' per questo motivo che per 32 caratteri si è scelta una stampa di lato 18 cm e per 64 caratteri una stampa di lato 19 cm.

I test fatti consistono nel misurare la distanza e l'angolo di orientamento dello smartphone, perpendicolare al pavimento, rispetto al barcode sulla parete.

Il grafico seguente mostra le quattro misurazioni effettuate a distanze diverse (da 40 a 120 cm) e angolo diverso (da -40° a 40°) sui barcode descritti in precedenza.

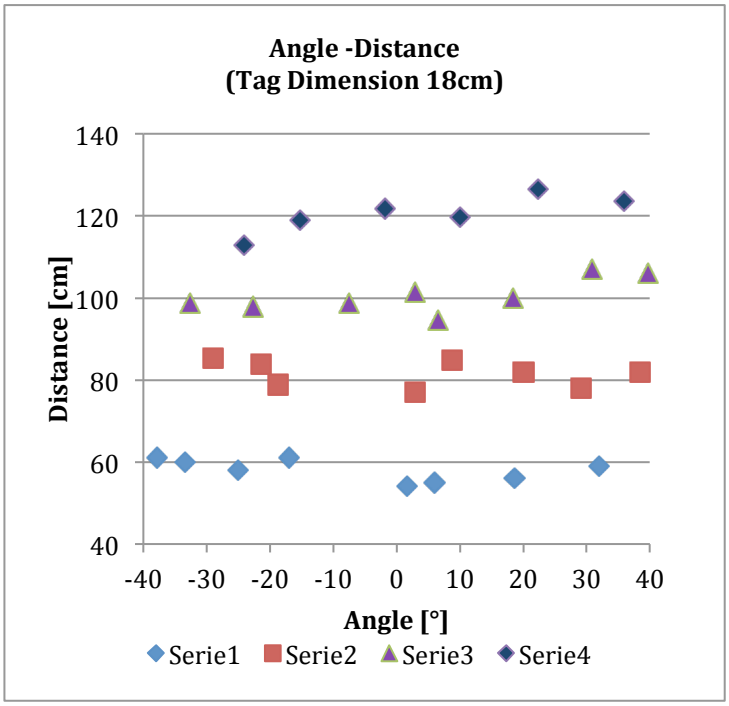

Figura 8.14 – Gli errori di misurazione rispetto al barcode con lunghezza del lato 18 cm

Errore medio:

Distanza = 1.67% Angolo = 8.12%

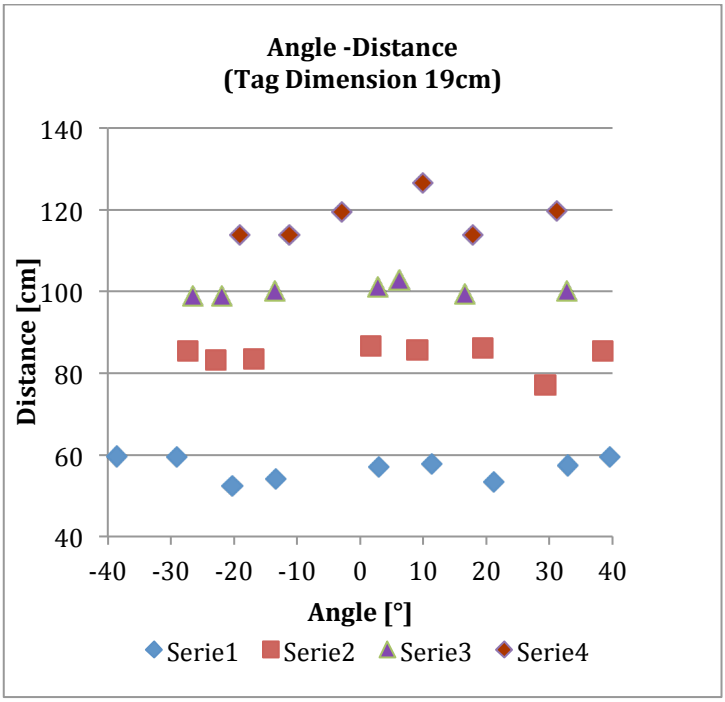

Figura 8.15 – Gli errori di misurazione rispetto al barcode con lunghezza del lato 19 cm

Errore medio:

Distanza = 2.03% Angolo = 6.1%

La figura 8.14 rappresenta le misurazioni sul codice a barre QR grande 18 cm e la figura 8.15 presenta invece le misure che si riferiscono a un codice a barre QR di 19 cm con una stringa di 64 caratteri alfanumerici codificata al suo interno.

Dai vari test effettuati con i barcode si afferma, in conclusione, che per avere un esito positivo nella misurazione della distanza, è necessario avere dei codici a barre le cui dimensioni risultino essere sempre maggiori all'aumentare della quantità di dati che questi devono codificare. Se si vuole aumentare la distanza di decodifica inoltre, devono aumentare anche le dimensioni del codice. Nel calcolo dell'angolo d'inclinazione non sono stati riscontrate grosse differenze nella misurazione.

9 Conclusioni

La presente tesi ha dimostrato che è possibile sviluppare un sistema di posizionamento inerziale per ambienti indoor basato sull'utilizzo di un comune smartphone moderno. Il tracking in tempo reale, effettuato con la tecnica del dead reckoning, unito a un'infrastruttura di barcode ed una mappa, si è dimostrato abbastanza affidabile. Tuttavia si incorre in errori cumulativi trascurabili su brevi distanze (relativamente ad un ambiente indoor) come descritto nel capitolo 8 riguardante le fase di sperimentazione e test del sistema.

La sperimentazione ha inoltre dimostrato che l'affidabilità del sistema dipende dalla potenza di calcolo del dispositivo e dalla corretta impostazione iniziale dei parametri relativi all'utente. Per il funzionamento del sistema è importante la corretta installazione dell'infrastruttura di posizionamento, rappresentata dai barcode: gli esperimenti effettuati hanno dimostrato che è possibile avere una buona stima del posizionamento in tempo reale attraverso una disposizione dei barcode entro circa 25 metri.

Il lavoro svolto dimostra inoltre che non è necessaria un'infrastruttura importante di posizionamento (nessun sistema Wi-Fi, Bluetooth, o apparato radio di localizzazione) e soprattutto si possono evitare ingombranti attrezzature di misurazione montate sull'utente: un posizionamento indoor è possibile anche solo tramite uno dei molti modelli di smartphone disponibili in commercio dotato di una buona sensoristica di posizionamento.

Sviluppi Futuri

Il lavoro futuro includerà sicuramente il miglioramento dei metodi di misurazione dei passi in modo da superare la serie di errori cumulativi dovuti, per ora, all'impostazione fissata della lunghezza del passo. Come dimostrato in alcuni studi [25], la stima istantanea della lunghezza del passo può essere ottenuta misurando la forza di accelerazione (attraverso un algoritmo probabilistico) e i dati relativi alle informazioni personali dell'utente (sesso, età, altezza, peso, ecc.…).

Quello che è davvero mancato nei dispositivi utilizzati durante i vari test, è stato il giroscopio. Il **giroscopio** sarebbe stato di grande aiuto nel correggere gli errori riguardanti la misura dell'orientamento, sia quelli standard sia quelli provocati da campi magnetici nelle vicinanze. Negli sviluppi futuri quindi, sarà preferibile la scelta di smartphone che contengono al loro interno un sensore giroscopico.

In seguito ai risultati ottenuti in altri esperimenti, documentati nella letteratura scientifica relativa al dead reckoning, il miglioramento dell'interpretazione dei dati può avvenire tramite l'utilizzo dei filtri di stima del posizionamento, come il filtro di Kalman e il Particle filter [19] [20] [30].

L'applicazione verrà migliorata anche dal punto di vista della facilità di utilizzo da parte dell'utente: potrà ad esempio guidarlo passo passo verso la locazione di una stanza o un oggetto nell'edificio, indicare una lista di persone nelle vicinanze e fornire una guida vocale a un utente che presenta difficoltà nell'ambientarsi a causa di deficit della vista.

Ringraziamenti

Dedico questo mio lavoro a mia madre Serena, mio padre Luciano e mio fratello.

Ringrazio i miei genitori per essere come sono, perché mi sono sempre stati vicini e mi hanno sempre incoraggiato giorno dopo giorno. Ma soprattutto perché mi hanno permesso di raggiungere questo importante traguardo. Grazie Infinite.

Un caloroso ringraziamento a nonno Antonio e nonna Grazietta, a Monica, a Milena e Mario. Mi avete sempre incoraggiato e siete sempre stati disponibili e comprensivi in ogni momento. Grazie.

Grazie a prof. Andrea Bosin per la disponibilità, la cortesia, la competenza e l'aiuto fornito durante la stesura di questa tesi e di quella triennale, che a suo tempo mi fece da trampolino di lancio nel mondo del lavoro. Grazie.

Colgo l'occasione per ringraziare tutti i docenti del Corso di Laurea in Tecnologie Informatiche per la loro professionalità, la competenza e tutti gli insegnamenti.

Ringrazio Antonio per la disponibilità, per tutti i preziosi suggerimenti, il supporto e l'incoraggiamento che non sono mai mancati in questi ultimi mesi.

Ringrazio Davide che ha reso possibile questo progetto e il CRS4 che mi ha fornito il materiale per la ricerca e lo sviluppo per il progetto di questa tesi.

Grazie ai colleghi del DIEE di Cagliari e del CRS4 che hanno collaborato con me a questo progetto: Vlad, Tiziana, Luigi Atzori e Valentina. Grazie.

Inoltre ringrazio i compagni di studio Cino, Claudio, Cozio, DDany, Fabrizio, Kate, Maurizio e Rosso per la loro disponibilità e tutti i bei momenti passati assieme durante questi anni di studio. Sono stati per me più veri amici che semplici colleghi.

Un pensiero speciale va a nonno Angelo e zio Lello. Durante gli ultimi anni sempre pronti nel tenersi informati sullo stato dei miei studi.

Ringrazio Fabiana. Per tanti motivi e soprattutto perché mi è sempre stata vicina, particolarmente nell'ultimo periodo, supportandomi e incoraggiandomi in ogni momento.

Grazie a tutti!

Bibliografia

[1] Alberto Serra, Davide Carboni, Valentina Marotto “Indoor pedestrian navigation system using a modern smartphone” In Proceedings of MobileHCI2010. ACM Press. Lisbon, Portugal. 2010

[2] Alberto Serra, Tiziana Dessi, Davide Carboni, Vlad Popescu, Luigi Atzori “Inertial Navigation Systems for User-Centric Indoor Applications ” In Proceedings of NEM Summit - Towards Future Media Internet Barcelona, Spain.2010

[3] U.S. Government, “Official U.S. Government information about the Global Positioning System ”, http://www.gps.gov

[4] P. F. Lammertsma, “Satellite Navigation”, Institute of Information and Computing Sciences. Utrecht University, 2005

[5] Andrew Howard, Sajid Siddiqi, Gaurav S Sukhatme, "An Experimental Study of Localization Using Wireless Ethernet", Proceedings of the 4th International Conference on Field and Service Robotics (FSR'03), Lake Yamanaka, Japan, July 2003

[6] Ahmed Ali Sabbou, “Indoor Localization using Wireless LAN/WiFi Infrastructure”, http://sabbour.wordpress.com/2008/10/20/indoor-localization-using-wireless-lan-infrastructure/, 2007

[7] Vlad Badea, Rikard Eriksson. “Indoor navigation with pseudolites (fake GPS sat.)”, 2005

[8] Ubisense, “Real-time Location” http://www.ubisense.net/

[9] Kiran Thapa and Steven Case. “An indoor positioning service for Bluetooth ad hoc networks”. Midwest Instruction and Computing Symposium, MICS, 2003.

[10] M. Sugano, T. Kawazoe, Y. Ohta, and M. Murata (2006). “Indoor Localization System using RSSI Measurement of Wireless Sensor Network based on ZigBee Standard”. From Proceeding (538) Wireless Sensor Networks

[11] Nokia Research Center, “Nokia Indoor Navigator” http://research.nokia.com/news/9505, Nokia World 2010

[12] Kougori, M. and Kurata, T. 2003. “Personal Positioning based on walking locomotion Analysis with self-contained sensor and a wearable camera.” In Proceedings of ISMAR2003, pp. 103-112

[13] Krach, B. and Robertson, P. 2008. “Integration of Foot-Mounted Inertial Sensors into a Bayesian Location Estimation Framework”. In Proceedings of 5th Workshop on Positioning, Navigation and Communication 2008 (WPNC 2008), Hannover, Germany.

[14] Gehring S., Löchtefeld M., Schöning J. and KrügerA. “Exploring the Usage of an Electronic Compass for Human Navigation in Extreme Environments”. Haptimap 2010: Multimodal Location Based Techniques for Extreme Navigation, In conjunction with Pervasive 2010, Helsinki, Finland, 2010.

[15] Wang, X., Klette, R. and Rosenhahn, B. (2005) “Geometric and photometric correction of projected rectangular picture”, Image and Vision Computing New Zealand, November 2005.

[16] A.J. Bernheim Brush, Amy K. Karlson, James Scott, Raman Sarin, Andy Jacobs, Barry Bond, Oscar Murillo, Galen Hunt, Mike Sinclair, Kerry Hammil, Steven Levi "User experiences with activity-based navigation on mobile devices". In Proceedings of MobileHCI 2010, Lisbon 2010

[17] Mulloni A., Wagner D., Barakonyi I., Schmalstieg D., "Indoor Positioning and Navigation with Camera Phones", IEEE Pervasive Computing, vol. 8, no. 2, pp. 22-31, Apr.-June 2009.

[18] C. Randell, C. Djiallis, H. Muller, "Personal position measurement using dead reckoning," in Proc. IEEE ISWC, pp. 166-173, Oct. 2000.

[19] R.E. Kalman. A new approach to linear filtering and prediction problems. Journal of basic Engineering, 82(1):35–45, 1960.

[20] M.S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, D. Sci, T. Organ, and S.A. Adelaide. "A tutorial on particle filters for online nonlinear/non-GaussianBayesian tracking". IEEE Transactions on signal processing, 50(2):174–188, 2002.

[21] Denso-Wave INC, About QR Code. http://www.denso-wave.com/qrcode/aboutqr-e.html

[22] C G Ryan, P M Grant, W W Tigbe, M H Granat (2006). "The validity and reliability of a novel activity monitor as a measure of walking". *British Journal of Sports Medicine*.

[23] The New York Times COMPANY, About.com. http://walking.about.com/cs/pedometers/a/pedometerset.htm

[24] NEW–LIFESTYLE INCORPORATED. NL-1000 pedometer. http://www.new-lifestyles.com/NL-1000_Users_Guide.pdf

[25] S.H. Shin, “Adaptive Step Length Estimation Algorithm Using Low-Cost MEMS Inertial Sensors” IEEE Sensors Appl. Symp., San Diego, USA, Feb 6-8, 2007.

[26] Google INC. Android.com, Android Reference Library. http://developer.android.com/index.html

[27] Asahi Kasei Microdevices Corporation (AKM), http://www.asahi-kasei.co.jp/akm/en/index.html

[28] ST Microelectronics, LIS331DLH datasheet, http://www.st.com/stonline/products/literature/ds/15094/lis331dlh.pdf

[29] 2D Barcode Scanner, “List of Online Barcode Generators” http://www.2dbarcodescanner.info/2d-barcode-generator-online.html

[30] Jonas Callmer, David Törnqvist, Fredrik Gustafsson, “Indoor Navigation using an iPhone”, Institutionen för systemteknik - Department of Electrical Engineering, Linköping 2010