UNIVERSITA' DEGLI STUDI DI PISA

Facoltà di Scienze Matematiche, Fisiche e Naturali

Corso di Laurea Specialistica in Informatica

TESI DI LAUREA

REALIZZAZIONE ED IMPLEMENTAZIONE SU UN MICROCONTROLLORE 32 BIT DI UN ALGORITMO DI NAVIGAZIONE IBRIDA INERZIALE GEOLOCALIZZATO CON SCENARI CONFIGURABILI

RELATORI:

Prof. Cecilia LASCHI

Ing. Adriano BASILE

CANDIDATO

Carmelo FORTUNATO

ANNO ACCADEMICO 2011-2012

Dedicata alla mia famiglia, agli amici veri e a Valeria, persone senza le quali questo lavoro sarebbe stato impossibile.

Indice

Pagina intenzionalmente vuota

CAPITOLO 1. INTRODUZIONE

Con l'evolversi della tecnologia e l'abbassarsi dei costi dell'hardware, concetti di uso ristretto solo in determinati ambiti di applicazione vanno via via espandendosi verso l'uso comune di massa. Non è raro ormai utilizzare sistemi per la geolocalizzazione[1] come il GPS (Global Positioning System) o sistemi inerziali all'interno degli smartphones, in applicazioni come bussole, livelle, e così via. Tali tendenze hanno inoltre portato la ricerca del settore alla progettazione di sistemi di navigazione inerziale più piccoli ed economici al fine di poter essere utilizzati in nuovi contesti quali ad esempio la robotica, la realtà virtuale e l'automotive.

In particolare il sistema GPS viene utilizzato per localizzare la posizione di veicoli o di robot. Il ricevitore GPS ricava i dati dai satelliti e calcola la nuova posizione della piattaforma mobile su cui è montato. Tuttavia si presentano casi in cui il ricevitore GPS non può svolgere la sua funzione: Esempi possono essere la presenza di tunnel stradali, in cui non vengono ricevuti dati dai satelliti, o la presenza di pochi satelliti, caso in cui i dati del ricevitore GPS risultano poco precisi[2]. Per tali motivi, soprattutto nell'ambito della robotica, vengono utilizzate tecniche per calcolare la posizione della piattaforma mobile anche in tali circostanze, calcolando la posizione corrente (posizione stimata) a partire dalla posizione precedente, utilizzando la velocità e il tempo trascorso, o altre informazioni sul movimento. Queste tecniche sono dette tecniche di **Dead Reckoning.**

In questo lavoro è stato sviluppato un algoritmo di navigazione inerziale che calcola posizione, assetto e velocità del veicolo ad ogni passo mediante l'integrazione dei dati (sensor fusion) di piattaforme inerziali e GPS. L'algoritmo implementa un **filtro di Kalman** adattato al caso della navigazione inerziale, che utilizza in ingresso le accelerazioni e gli angoli di Eulero ricavati dagli IMU XSens MTi e ST iNemo M1, dati che vengono poi processati al fine di restituire in uscita i nuovi valori di coordinate, velocità e assetto (noti insieme come **stati della navigazione**). Sono state utilizzate tali piattaforme al fine di valutare sia le prestazioni dell'algoritmo con diversi dispositivi, sia per confrontare le due

[1] Localizzazione nel globo terrestre, conoscere la posizione mediante un sistema di riferimento
[2] Esistono altri tipi di errori generati dal sistema GPS. Uno di questi è il multipath

IMU, cercando di dimostrare che i sensori inerziali ST siano arrivati a livelli di qualità tali da poter essere applicati in ambiti come la **navigazione inerziale**.

CAPITOLO 2. SISTEMI DI RIFERIMENTO E NAVIGAZIONE INERZIALE

2.1 SISTEMI DI RIFERIMENTO

Un sistema di riferimento può essere definito come l'insieme di riferimenti o coordinate utilizzate al fine di localizzare dei punti nello spazio. Se lo spazio di riferimento è una retta si avrà un sistema di riferimento monodimensionale, se lo spazio di riferimento è un piano allora si ha un sistema di riferimento bidimensionale, e così via. In qualsiasi sistema di riferimento è comunque necessario stabilire un'unità di misura. Esempi di unità di misura possono essere il metro, il centimetro, i metri al secondo e così via.

Esistono diversi tipi di sistemi di riferimento. I più comuni sono i sistemi di riferimento cartesiani (ortogonali e non, bidimensionali o tridimensionali) e il sistema di riferimento polare, nel caso bidimensionale, che poi viene esteso nei sistemi di riferimento cilindrico e sferico nel caso tridimensionale.

2.1.1 Sistema di riferimento Cartesiano

Un sistema di riferimento cartesiano nel piano è definito come una coppia di rette incidenti e orientate. Il punto dell'intersezione viene detto origine del sistema di riferimento. Se l'angolo formato dalle due rette è un angolo retto allora si parla di sistema cartesiano ortogonale. Le rette orizzontale e verticale vengono chiamate rispettivamente retta delle ascisse e retta delle ordinate, e vengono indicate dalle lettere x e y. In questo tipo di sistema di riferimento ogni punto P del piano viene indicato da un vettore $v \in \boldsymbol{R}^2$, per via della corrispondenza biunivoca tra numeri reali e punti in una retta[3].

[3] Tale tesi fu dimostrata da Richard Dedekind, che visse dal 1831 al 1916

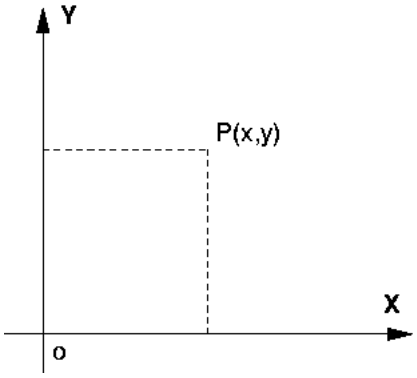

Figura 1 : Sistema Cartesiano Ortogonale

Nel caso tridimensionale si ha una terna di rette incidenti, di solito indicate con x, y e z. Ogni punto nello spazio verrà quindi individuato da un vettore $v \in \boldsymbol{R^3}$.

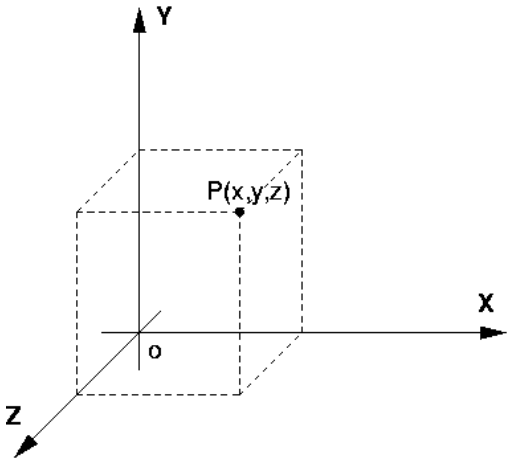

Figura 2 : Sistema Cartesiano ortogonale 3D

2.1.2 Sistema Polare

Il sistema di riferimento polare individua i punti del piano mediante due coordinate, ρ e φ. La prima rappresenta il modulo e la seconda l'angolo formati dal vettore $\boldsymbol{\rho}$ con l'asse delle ascisse.

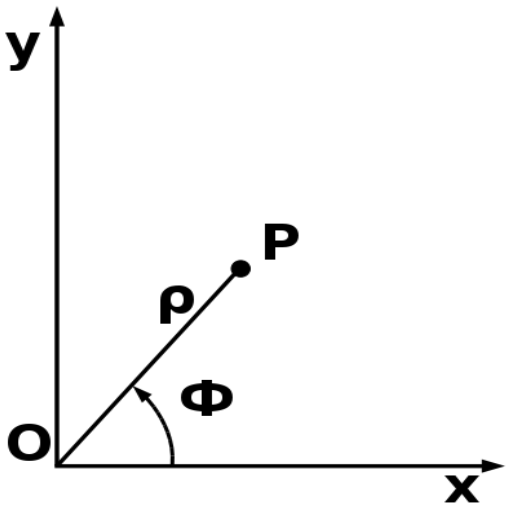

Figura 3 : Sistema Polare

2.1.3 Sistema sferico

Nel sistema di riferimento sferico un punto P è individuato da tre coordinate: ρ, θ e φ. La prima rappresenta il modulo del vettore **ρ** (ovvero la distanza di P dall'origine), la seconda è l'angolo formato dalla proiezione del vettore **ρ** nel piano xy (chiamata Q) e l'asse delle x, mentre φ è l'angolo formato da **ρ** con l'asse z.

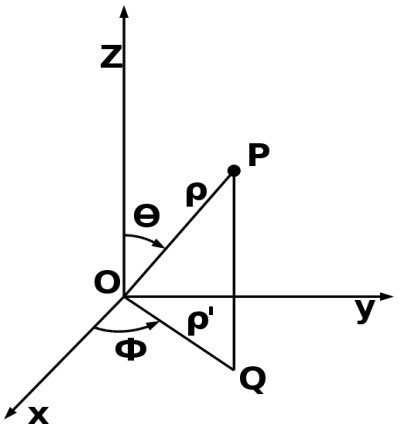

Figura 4 : Sistema sferico

Per passare da un sistema sferico ad uno rettangolare si usano le seguenti uguaglianze:

$$x = \rho \ \sin\theta \ \cos\phi \qquad y = \rho \ \sin\theta \ \sin\phi \qquad z = \rho \ \cos\theta$$

Per passare da coordinate cartesiane a sferiche:

$$\rho = \sqrt{x^2 + y^2 + z^2}$$

$$\phi \ = \arctan\left(\frac{y}{x}\right) = \arcsin\left(\frac{y}{\sqrt{x^2 + y^2}}\right)$$

$$\theta \ = \arccos\left(\frac{z}{\sqrt{x^2 + y^2 + z^2}}\right) = \operatorname{arccot}\left(\frac{z}{\sqrt{x^2 + y^2}}\right)$$

2.2 SISTEMI DI RIFERIMENTO TERRESTRI E FRAMES

I sistemi di riferimento terrestri partono certamente dai concetti matematici su cui si basano i sistemi di riferimento suddetti. Tuttavia, subentrano altri fattori in gioco che è opportuno specificare. Per tale motivo descriviamo innanzitutto i concetti di sistema spazio-temporale e sistema di riferimento inerziale.

- Un **sistema di riferimento spazio-temporale** è una legge che ad ogni istante t assegna un sistema di riferimento spaziale.
- Un **sistema di riferimento inerziale** è un sistema di riferimento in cui ogni punto materiale, sottratto a tutte le forze esterne, resta in quiete o si muove di moto rettilineo uniforme, ovvero rispetta le *prima legge fondamentale della dinamica*[4].

Le stelle fisse, che hanno un'interazione gravitazionale trascurabile rispetto alla terra, possono essere considerate un buon punto di riferimento per la definizione di un sistema inerziale. Tuttavia per individuare un'auto nella superficie terrestre, un sistema di riferimento di questo tipo sarebbe piuttosto scomodo da utilizzare, per cui è necessario avere a che fare con sistemi di riferimento più comodi, magari con origine al centro della terra, ma che per ovvie ragioni[5] non possono essere considerati inerziali. Un altro concetto su cui soffermarsi è la differenza tra sistema di riferimento e *frame*: il primo è il sistema di riferimento teorico mentre il secondo è l'istanziazione del primo nello spazio reale. I frame non sono sempre ottenibili dai sistemi di riferimento in maniera diretta, soprattutto quando si tratta di frame inerziale. Ad esempio, i punti della superficie terrestre non si muovono tutti con la stessa velocità, quindi un sistema di riferimento terrestre subirà una rotazione non uniforme rispetto al sistema precedente.

[4] Il primo principio della dinamica: "Un corpo in quiete o in moto rettilineo uniforme rimane tale finchè altre forze non agiscono su di esso". Newton formulò tale principio nella sua opera *Philosophiae Naturalis Principia Mathematica,* del 1687, ma già Galileo Galilei nel periodo tra il 1632 e il 1638 descrisse tale fenomeno, tanto che il primo principio della dinamica viene a volte detto principio di Galileo.

[5] La terra non può essere considerata un buon punto di riferimento per un sistema inerziale, in quanto in essa agiscono le forze derivanti dalla rotazione terrestre ed altre forze come quella di Coriolis, o quelle derivanti dai movimenti della terra di precessione e nutazione.

2.2.1 Earth-Centered Inertial Frame

L'ECI-frame o i-frame[6] ha origine al centro di massa della terra, il piano xy coincidente col piano equatoriale terrestre e l'asse z passante dal polo nord, ortogonale al piano equatoriale. Gli assi x e y sono fissi e non ruotano insieme alla terra. Tale sistema di riferimento è per lo più utilizzato per individuare corpi nello spazio rispetto alla terra.

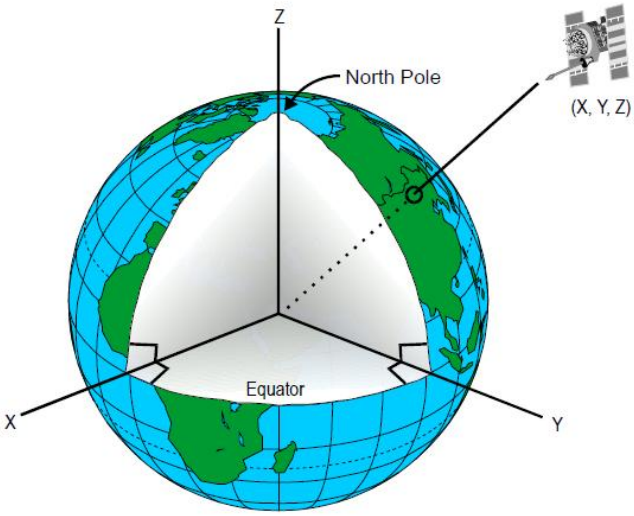

Figura 5 : ECI Frame

2.2.2 ECEF Frame

Per localizzare oggetti in movimento nel globo terrestre utilizzare l'ECI frame è scomodo, in quanto col movimento della terra i punti della superficie terrestre cambiano continuamente coordinate. Per tale motivo è conveniente utilizzare un diverso sistema di coordinate chiamato Earth-Centered Earth-Fixed frame (e-frame), detto anche sistema terrestre convenzionale. Tale sistema, come l'ECI frame, ha il piano xy coincidente col piano equatoriale e l'asse z ortogonale ad esso, ma gli assi x e y sono fissati: in particolare si è deciso che l'asse x passi dal meridiano di Greenwich, detto meridiano internazionale di riferimento (IRM), mentre l'asse y sia disposto ortogonalmente all'asse x sempre sul piano equatoriale, secondo la regola della mano destra. Il frame ECEF quindi ruota insieme alla terra, per cui un qualsiasi punto del globo terrestre mantiene le sue coordinate. Di fatto

[6] Viene anche detto sistema di riferimento pseudo-inerziale o quasi-inerziale.

queste caratteristiche lo rendono un sistema di riferimento abbastanza comodo per la geolocalizzazione, ragion per cui il sistema di posizionamento globale (GPS) utilizza le coordinate ECEF.

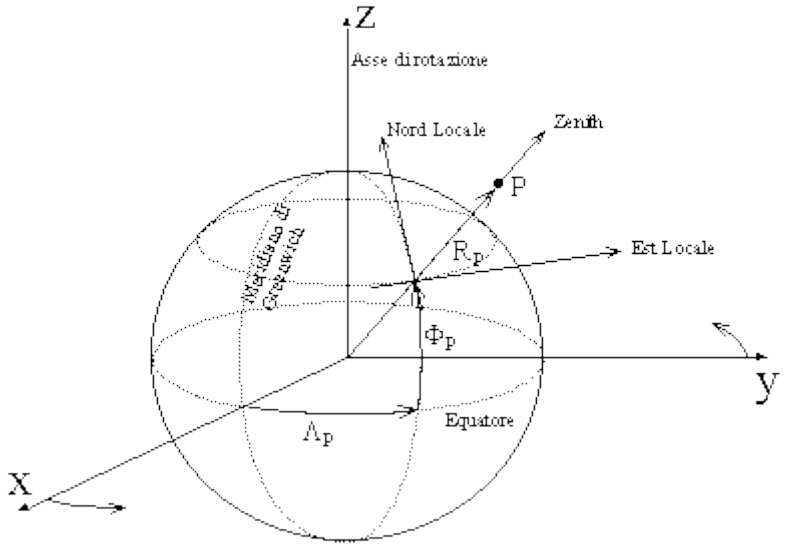

Figura 6 : ECEF frame

Un punto *P* viene individuato da 3 componenti : *Λ*, *Φ* e *d*. La prima componente, come nel caso del sistema di coordinate sferico, è l'angolo tra l'asse *x* e la proiezione del vettore ***d*** (formato da *P* con l'origine) col piano *xy*. La seconda componente è l'angolo tra il vettore ed il piano *xy*, mentre *d* è la distanza del punto *P* dall'origine, ovvero il modulo del vettore ***d***. Quest'ultima è composta dalla somma di due componenti: il raggio della terra *R* e la parte rimanente, ovvero la distanza *h* tra il punto *P* e la sua proiezione ortogonale[7] nella superficie terrestre.

$$d = R + h$$

Φ, Λ e *h* sono dette **latitudine, longitudine** e **altitudine (o altezza ellissoidrica).**

[7] In realtà il prolungamento del vettore ortogonale alla superficie terrestre del punto P verso l'interno del globo non coincide esattamente con l'origine del sistema di riferimento a meno che il punto P non si trovi sul piano equatoriale o nell'asse di rotazione terrestre, per via della forma ellissoidale della terra. Durante il calcolo di *d* stiamo quindi supponendo che la terra sia sferica, un'approssimazione della distanza reale.

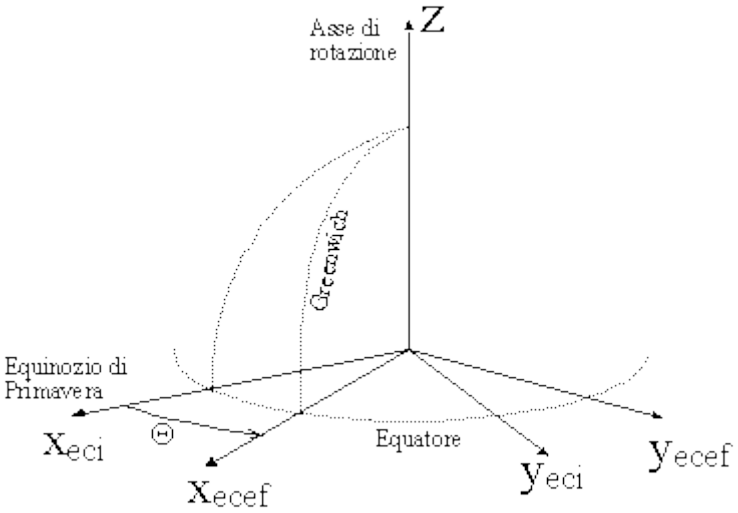

Figura 7 : Differenze tra e-frame e i-frame

2.2.3 Frame Navigazionale

Il frame navigazionale o n-frame non è un sistema di riferimento terrestre, bensì è legato ad un corpo o ad un punto di cui si vogliono conoscere gli stati della navigazione. Esso ha origine solitamente al centro di massa del corpo in questione, ed assi ortogonali tra loro a formare una terna destrosa. In base all'orientazione degli assi dell'n-frame possiamo avere diversi sistemi di riferimento. I frame utilizzati sono:

- NED o North-East-Down
- NWU o North-West-Up
- ENU o East-North-Up

Il primo viene in genere utilizzato (ma non solo) per applicazioni aeronautiche o spaziali, mentre gli altri due per applicazioni terrestri

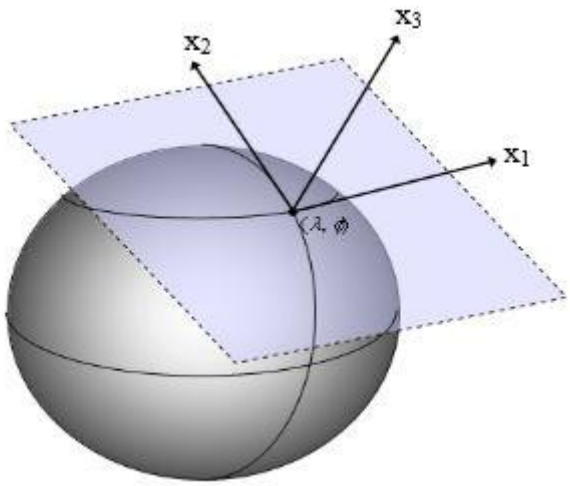

Figura 8 : Frame navigazionale ENU

2.2.4 Body frame

Il frame navigazionale non tiene conto dell'orientazione del corpo (sia esso un veicolo, un veivolo o un robot), informazione necessaria quando dei sensori sono montati rigidamente sul corpo (configurazione strapdown[8]) e che quindi rilevano accelerazioni e velocità secondo un loro sistema di riferimento. Per tale motivo viene utilizzato un altro sistema di riferimento detto body frame o b-frame, i cui assi fanno riferimento all'orientazione del corpo. Le coordinate in n-frame sono senz'altro più facili da trasformare in un sistema di riferimento terrestre, ragion per cui in genere le grandezze misurate da sensori (ottenute rispetto al b-frame) vengono prima trasformate in valori rispetto l'n-frame, per essere poi trasformate in riferimento al frame terrestre. I due frame in genere hanno l'origine degli assi in comune, quindi ciò che li distingue sono delle rotazioni degli assi stessi, ovvero degli angoli con cui il b-frame dev'essere ruotato per farlo coincidere con l'n-frame. Tali angoli sono detti **roll**, **pitch**, e **yaw,** detti anche Angoli di Eulero. Il roll o rollìo è la rotazione rispetto all'asse *x* (che nel b-frame genericamente identifica la verticale del corpo con verso in avanti), il pitch o beccheggio è la rotazione rispetto all'asse *y* mentre lo yaw o imbardata è la rotazione rispetto all'asse *z*. La combinazione di rollio e beccheggio viene spesso

[8] Verrà spiegato con maggiore dettaglio nei paragrafi successivi

chiamata attitudine (*attitude*) del corpo, mentre l'imbardata è l'orientazione o *heading* del corpo.

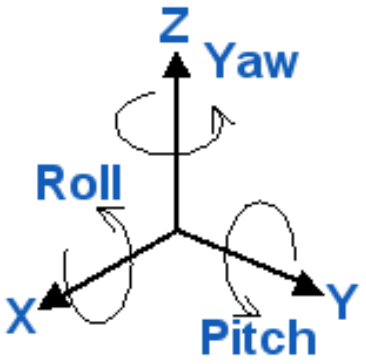

Figura 9 : Angoli di Eulero

2.3 PASSAGGI TRA SISTEMI DI RIFERIMENTO

Descriviamo adesso l'insieme di trasformazioni necessarie per portare le coordinate di un punto (o le misure di una grandezza fisica) da un sistema di riferimento all'altro, soffermandoci in particolare sulle trasformazioni utilizzate nel presente lavoro.

2.3.1 Da body frame a frame navigazionale

In piattaforme montate su sospensioni cardaniche (dette piattaforme *Gimbaled*) gli assi del b-frame vengono riallineati automaticamente con gli assi dell'n-frame. Tuttavia tali piattaforme sono più costose ed ingombranti, per cui la tendenza è quella di utilizzare piattaforme *strapdown*, con le quali però non è presente un riallineamento automatico. Queste ultime infatti sono fissate al corpo allineando gli assi dei sensori agli assi del b-frame. Per tale motivo è necessario effettuare il riallineamento computazionalmente. Uno dei metodi più utilizzati in tale ambito è il **metodo degli angoli di eulero[9].**

[9] Esistono altri metodi per svolgere tale compito. Un altro esempio è il "Metodo dei Parametri di Eulero". Altri esempi di soluzione possono essere visti in [15] e [16]

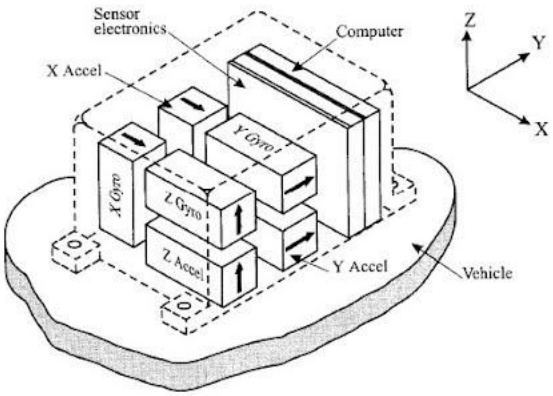

Figura 10 : Piattaforma strapdown

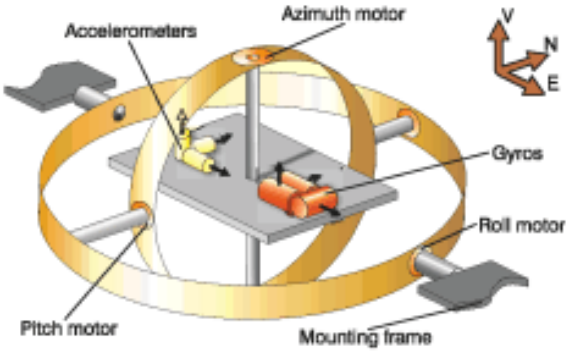

Figura 11 : Piattaforma Gimbaled

Siano Φ, θ e Ψ rispettivamente gli angoli di roll, pitch e yaw. Per passare dal body frame al frame navigazionale è sufficiente applicare le tre matrici di rotazione corrispondenti ai tre angoli [1]:

$$R_x(\Phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\Phi & -\sin\Phi \\ 0 & \sin\Phi & \cos\Phi \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_z(\Psi) = \begin{bmatrix} \cos\Psi & -\sin\Psi & 0 \\ \sin\Psi & \cos\Psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Il prodotto tra queste matrici porta alla definizione della matrice di rotazione totale detta anche *matrice dei coseni direttori (DCM)* [2].

$$R_b^n = R_z(\Psi)R_y(\theta)R_x(\Phi) =$$

$$\begin{bmatrix} \cos\theta\cos\Psi & \cos\Phi\sin\Psi + \sin\Phi\sin\theta\cos\Psi & \sin\Phi\sin\Psi - \cos\Phi\sin\theta\cos\Psi \\ -\cos\theta\sin\Psi & \cos\Phi\cos\Psi - \sin\Phi\sin\theta\sin\Psi & \sin\Phi\cos\Psi + \cos\Phi\sin\theta\sin\Psi \\ \sin\theta & -\sin\Phi\cos\theta & \cos\Phi\cos\theta \end{bmatrix}$$

Il passaggio da frame navigazionale a body frame si ottiene con la matrice inversa di R_b^n che, sfruttando l'ortogonalità delle matrici, equivale alla trasposta

$$R_n^b = R_x(-\Phi)R_y(-\theta)R_z(-\Psi) =$$

$$\begin{bmatrix} \cos\theta\cos\Psi & -\cos\theta\sin\Psi & \sin\theta \\ \cos\Phi\sin\Psi + \sin\Phi\sin\theta\cos\Psi & \cos\Phi\cos\Psi - \sin\Phi\sin\theta\sin\Psi & -\sin\Phi\cos\theta \\ \sin\Phi\sin\Psi - \cos\Phi\sin\theta\cos\Psi & \sin\Phi\cos\Psi + \cos\Phi\sin\theta\sin\Psi & \cos\Phi\cos\theta \end{bmatrix}$$

2.3.2 Dal frame navigazionale al frame terrestre

Il frame navigazionale è un frame cartesiano. Il frame terrestre invece può rappresentare i punti nel globo terrestre mediante due sistemi: il sistema di coordinate cartesiane ed il sistema di coordinate geodetiche. Quest'ultimo è il più usato, e riferisce i punti mediante le tre coordinate Φ, Λ ed h, ovvero latitudine,longitudine e altitudine.

Quindi, per rappresentare punti riferiti nel sistema navigazionale nel sistema terrestre è innanzitutto necessario trasformare le coordinate geodetiche in coordinate cartesiane terrestri. Per fare ciò è sufficiente applicare le formule di conversione presentate nel paragrafo 1.2.3 alle coordinate geodetiche (ricordando però che la terra non è esattamente una sfera, ma un ellissoide). Fatto ciò si deve applicare la trasformazione per portare le coordinate cartesiane locali in coordinate e-frame. Tale passo è rappresentato dalla matrice R_n^e.

Si ha quindi[10] :

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} (N+h)\cos\Phi\cos\Lambda \\ (N+h)\cos\Phi\sin\Lambda \\ \left((1-e^2)N+h\right)\sin\Phi \end{bmatrix} + R_n^e \begin{bmatrix} x^n \\ y^n \\ z^n \end{bmatrix}$$

Dove *e* è l'eccentricità del globo terrestre mentre N è il raggio di curvatura terrestre (Normal Radius) calcolato mediante la seguente formula:

$$N = \frac{a}{\sqrt{1-e^2\sin\Phi}}$$

In cui *a* è il semiasse maggiore del globo terrestre, a volte indicato con R_{ew} (il semiasse maggiore è corrispondente al raggio equatoriale della terra, a differenza del semiasse minore che è il raggio polare, a volte chiamato R_{ns}. I valori dei semiasse e l'eccentricità terrestre vengono calcolati dall'IERS, o *International Earth Rotation and Reference System Service*[11]).

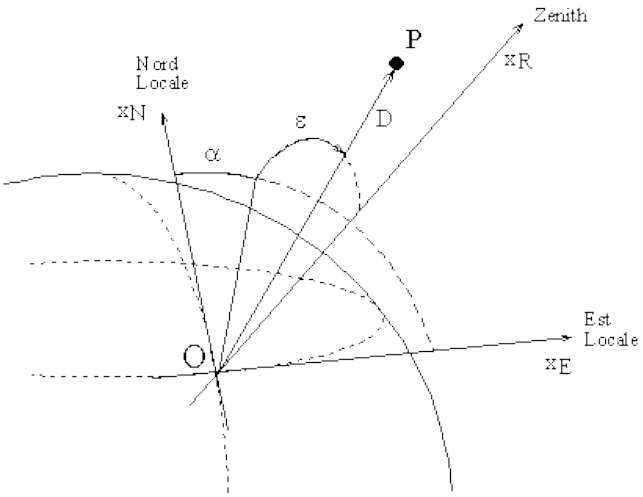

Figura 12 : Sistema di riferimento cartesiano locale

[10] Dove x_1 , x_2 e x_3 sono le coordinate nel sistema cartesiano terrestre
[11] Si veda [3] per le procedure di determinazione

La matrice di rotazione R_n^e dipende soprattutto dal sistema di riferimento adottato nel frame navigazionale. Ad esempio, per il frame di tipo NED la matrice è la seguente, ottenuta dalle matrici di rotazione intorno gli assi y e z [4][5] :

$$R_n^e = \begin{bmatrix} -\sin\Phi\cos\Lambda & -\sin\Lambda & -\cos\Phi\cos\Lambda \\ -\sin\Phi\sin\Lambda & \cos\Lambda & -\cos\Phi\sin\Lambda \\ \cos\Phi & 0 & -\sin\Phi \end{bmatrix}$$

La matrice per la trasformazione inversa, R_e^n, si ottiene sfruttando ancora l'ortogonalità dei frames, quindi si avrà:

$$R_e^n = \begin{bmatrix} -\sin\Phi\cos\Lambda & -\sin\Phi\sin\Lambda & \cos\Phi \\ -\sin\Lambda & \cos\Lambda & 0 \\ -\cos\Phi\cos\Lambda & -\cos\Phi\sin\Lambda & -\sin\Phi \end{bmatrix}$$

2.4 NAVIGAZIONE INERZIALE

Storicamente per **navigazione** si intende l'insieme di tecniche e strumenti usati per determinare la posizione e la rotta di una nave in mare. L'uso del termine si è via via esteso ad una serie di altri ambiti applicativi al di là di quello marittimo, come per esempio quelli terrestre, aeronautico e spaziale.

Per svolgere tale funzione quindi è necessario disporre di tecniche e strumenti, che permettano di calcolare gli *stati della navigazione*, ovvero, posizione, velocità e assetto.

Nel caso della navigazione terrestre gli strumenti utilizzati sono in genere sensori di vario tipo[12]:

- Dispositivo GPS
- Odometri (sensori che vengono utilizzati nella tecnica di odometria)
- Sensori inerziali (accelerometri e giroscopi)
- …

Esistono diverse tecniche per ricavare gli stati della navigazione. Una tra le più utilizzate è la cosiddetta tecnica di Dead Reckoning. Tale tecnica è il processo di calcolo della posizione attuale tramite l'utilizzo di una posizione precedentemente determinata, sulla base di velocità conosciute o stimate all'istante precedente. L'implementazione di tale tecnica può essere realizzata essenzialmente con tre metodi, che possiamo distinguere in metodo cinematico, metodi analitico e metodo dinamico. I metodo cinematico è il più semplice e basa la risoluzione delle equazioni di navigazione mediante integrazione numerica [6]. Il metodo analitico risolve le equazioni mediante un modello analitico di derivazione complessa, ma che fornisce una soluzione molto precisa. Un esempio di tale metodo è presente in [7]. Infine, il metodo dinamico determina gli stati della navigazione calcolando una matrice di stato che contiene la dinamica del sistema, ad esempio mediante l'utilizzo del filtro di Kalman. Tale soluzione è applicabile persino in condizioni in cui si utilizzano sensori economici (come ad esempio sensori MEMS[13]) caratterizzati da sensibilità a rumore e deriva di misurazione (drift) a lungo termine, a differenza dei metodi cinematico e analitico.

[12] Si veda il capitolo seguente per una panoramica sui sensori

[13] Micro Electro Mechanical Sensors, sensori integrati in chip dell'ordine del micron (un millesimo di millimetro). In realtà si può già parlare di NEMS, ovvero Nano Electro Mechanical Sensors, sensori ancora più piccoli dei precedenti.

2.4.1 Filtro di Kalman

Il filtro di Kalman è un algoritmo ricorsivo che risolve il problema della stima ottima dello stato per sistemi lineari a tempo discreto con rumore bianco gaussiano additivo[14] che agisce sullo stato e sulle osservazioni dell'uscita. Per stima ottima si intende *stima a minima varianza dell'errore*.

La teoria del suddetto filtro è ancora oggi oggetto di ricerca, al fine di estenderne i risultati e l'applicabilità.

Un primo risultato, in tal senso, è costituito dal *filtro di Kalman esteso (EKF)*, utilizzato per la stima dello stato nei sistemi non lineari. Esso consiste nell'applicare alla linearizzazione del sistema intorno alla stima corrente il filtro di Kalman nella sua formulazione classica. Tecniche di discretizzazione consentono inoltre di applicare il filtro a sistemi "continui" mediante osservazioni discrete. Per un quadro più completo sull'uso del filtro di Kalman si veda [8].

Essenzialmente il filtro di Kalman è composto da due fasi: Fase di predizione e fase di correzione (o update). Nella prima fase si utilizza la stima dello stato ottenuta al passo precedente per effettuare la stima (a priori) dello stato corrente. Nella fase di correzione la stima ottenuta dalla fase di predizione viene corretta con delle informazioni di misura corrente. La stima ottenuta dalla correzione viene detta stima a posteriori.

Fase di Predizione

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{F}_k \hat{\boldsymbol{x}}_{k-1|k-1} + \boldsymbol{G}_k \boldsymbol{u}_k$$

Predizione dello stato: $\hat{\boldsymbol{x}}_{k|k-1}$ è la stima dello stato al passo k data la stima dello al passo k-1

$\boldsymbol{F}_k$ rappresenta la matrice di transizione di stato da applicare allo stato precedente $\hat{\boldsymbol{x}}_{k-1|k-1}$, detta anche matrice di update.

[14] Errore Gaussiano a media nulla

	$\boldsymbol{G_k}$ è la matrice di controllo dell'input, che viene applicata al vettore di input
$\boldsymbol{P_{k\|k-1} = F_k\, P_{k-1\|k-1}\, F_k^T + Q_k}$	**Predizione della covarianza:** $\boldsymbol{P_{k\|k-1}}$ è la matrice di covarianza allo stato corrente dato lo stato k-1
	$\boldsymbol{Q_k}$ è la matrice di covarianza del rumore di processo

Fase di Correzione

$\boldsymbol{\tilde{y}_k = z_k - H_k \hat{x}_{k\|k-1}}$	$\boldsymbol{\tilde{y}_k}$ è detto innovazione o residuo di misura.
	$\boldsymbol{H_k}$ è la matrice di estrazione o modello di osservazione
	$\boldsymbol{z_k}$ è l'osservazione al passo k
$\boldsymbol{S_k = H_k P_{k\|k-1} H_k^T + R_k}$	$\boldsymbol{S_k}$ è l'innovazione o residuo della covarianza
	$\boldsymbol{R_k}$ è la matrice di covarianza del rumore di misura
$\boldsymbol{K_k = P_{k\|k-1} H_k^T\, S_k^{-1}}$	$\boldsymbol{K_k}$ è il guadagno ottimo di Kalman
$\boldsymbol{\hat{x}_{k\|k} = \hat{x}_{k\|k-1} + K_k\, \tilde{y}_k}$	**Correzione dello stato**
$\boldsymbol{P_{k\|k} = (I - K_k\, H_k)\, P_{k\|k-1}}$	**Correzione della covarianza**

CAPITOLO 3. STATO DELL'ARTE DEI SENSORI

Oggi giorno i sensori sono di così largo uso da poter essere inseriti in ogni tipo di contesto o applicazione: Elettrodomestici, abbigliamento, smartphones, automobili, veivoli, console, accessori, robotica e realtà virtuale sono solo alcuni degli ambiti di utilizzo. In diversi tipi di applicazione stanno assumendo sempre maggiore importanza i MEMS, ovvero Micro Electro-Mechanical Sensors, sensori miniaturizzati ed integrati in circuiti dell'ordine di grandezza del micron. Mediante tale tecnologia è stato possibile costruire delle Inertial Measurement Unit (IMU)[15] piccole e relativamente economiche.

Un sensore è un trasduttore (ovvero un dispositivo che trasforma grandezze di un dato sistema energetico in ingresso in grandezze proporzionali di un altro sistema energetico) che ha come grandezza in uscita un segnale elettrico.

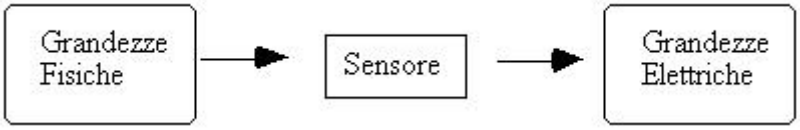

Figura 13 : Schema di un Sensore

3.1 CARATTERISTICHE DEI SENSORI E CALIBRAZIONE

Un sensore associa la grandezza misurata (ovvero la grandezza fisica in ingresso) al corrispondente segnale elettrico mediante una legge matematica, detta FUNZIONE DI TRASFERIMENTO. Tale funzione può essere più o meno complessa, lineare rispetto alla dimensione dell'input (caso in cui si parlerà di sensore lineare) oppure no.

[15] Si veda il paragrafo 3.4 per una maggiore descrizione

Un sensore, così come un qualsiasi trasduttore elettronico, ha delle proprietà che lo caratterizzano, dette appunto caratteristiche del sensore.

Al fine di conoscere tali caratteristiche sul sensore viene applicato una procedura nota come processo di calibrazione o **ciclo di calibrazione**: vengono scelte delle grandezze fisiche con valore noto che vanno dalla minima alla massima grandezza misurabile dal sensore e vengono misurate una dopo l'altra.

Tale processo si divide essenzialmente in due fasi:

- **Cammino crescente** : le grandezze note vengono misurate dalla più piccola alla più grande
- **Cammino decrescente**: le grandezze note vengono misurate dalla più grande alla più piccola

L'esempio tipico di questo processo è quello di una bilancia. Vengono scelti dei pesi che vengono ordinati e misurati, in una prima fase dal più leggero al più pesante, secondariamente alla rovescia.
Un'altra procedura che solitamente si fa nella fase di calibrazione è quella di pesare ripetutamente la stessa grandezza al fine di poter stimare gli errori casuali di misura.
Mediante tale procedura è inoltre possibile conoscere la curva di calibrazione del sensore, che corrisponde graficamente alla funzione di trasferimento. Tanto più simile è la curva ad una retta, tanto più si dice che il sensore ha una buona linearità. Questo tipo di calibrazione purtroppo non può essere applicato ad ogni tipo di sensore.

Per i sensori inerziali in particolare non è possibile conoscere a priori le grandezze misurate se non in particolari casi. Per tali motivi si usano altre tecniche, come ad esempio il test statico delle sei facce, di cui una descrizione si trova in [5] e [10].

3.1.1 Descrizione delle caratteristiche

Tra le caratteristiche più importanti abbiamo:

- Risoluzione
- Sensibilità
- Accuratezza
- Isteresi
- Ripetibilità
- Tempo di vita utile
- Stabilità

- La risoluzione definisce e misura il più piccolo scarto della grandezza misurata che un sensore è in grado di rilevare. Ad esempio, un encoder dotato di 3600 tacche sulla sua periferia potrà risolvere un decimo di grado e non meno. La risoluzione è un parametro importante perché condiziona sia la ripetibilità sia l'accuratezza.

  La risoluzione viene di solito indicata dal costruttore nelle specifiche tecniche del sensore.

- La sensibilità è il rapporto tra variazione della grandezza reale e variazione della grandezza misurata e può essere indicata mediante la formula:

$$S = \frac{\Delta \mathrm{r}}{\Delta \mathrm{m}}$$

  dove r è la grandezza reale mentre m è la grandezza misurata (in uscita dal sensore).

- L'accuratezza rappresenta l'errore massimo tra la grandezza reale e la grandezza misurata. Tale caratteristica cerca di esprimere l'assenza di errori sistematici nelle misure.
- L'isteresi indica l'errore tra grandezza misurata nel cammino decrescente e la stessa grandezza misurata nel cammino crescente durante il processo di calibrazione del sensore.

- La ripetibilità rappresenta in qualche modo la misura degli errori casuali presenti nelle misure del sensore. Un'alta ripetibilità indica che le misure date dal sensore sono piuttosto "stabili", ovvero una grandezza misurata più volte da in output lo stesso valore misurato, o comunque valori con deviazione standard inferiore ad una certa soglia stabilita.
- Il tempo di vita utile è il periodo in cui il sensore opera senza variare le sue prestazioni, ovvero senza subire deterioramento.
- Infine la stabilità rappresenta l'intervallo di tempo (lungo, medio o breve) in cui il sensore può garantire inalterate le sue caratteristiche.

Caratteristiche spesso citate dei sensori sono il bias ed il fattore di scala.

Il **bias** è l'errore sistematico totale o errore assoluto:

$$\textit{BIAS} = \textit{valore ottenuto} - \textit{valore vero}^{*}$$

Tale relazione ci dice che il bias è strettamente correlato all'accuratezza del sensore ed in effetti potrebbe essere scelto come misura dell'accuratezza (o meglio dire dell'inaccuratezza).

Il **fattore di scala** invece è il rapporto tra il valore vero ed il valore ottenuto. Esso quindi è strettamente correlato alla sensibilità.

In molti casi non è possibile conoscere il "valore vero",ovvero quello reale di una grandezza da parte di un sensore. In queste circostanze, per poter approssimare al meglio tale valore si effettuano varie misure della stessa grandezza al fine di scegliere il valore più indicativo (valore mediano) oppure il valore medio per tale grandezza.

3.2 TIPI DI SENSORI

Adesso vedremo quali sensori vengono spesso usati in ambiti come la robotica, suddividendoli in categorie dipendenti dalle loro peculiarità e fornendo quando possibile una visione qualitativa del loro funzionamento. Per maggiori dettagli si veda [11].

I sensori possono essere distinti in due categorie: Sensori attivi e sensori passivi.

I **sensori attivi** sono quei sensori che, per poter espletare il loro funzionamento, hanno bisogno di energia ausiliaria, di un qualsiasi sistema energetico fissato. Esempi di sensori di questo tipo sono gli encoder ottici e gli encoder magnetici, che utilizzano rispettivamente laser e campo magnetico per misurare rotazioni o movimenti.

I **sensori passivi** invece convertono direttamente la grandezza in ingresso nella grandezza in uscita, senza cioè l'utilizzo di energia ausiliaria. Esempi di questo tipo di sensori sono fotodiodi e potenziometri.

In base alla grandezza misurata possiamo classificare i sensori in:

Sensori di Forza	Estensimetri (Strain Gauge), sensori di Forza-Coppia,…
Sensori Magnetici	Sensori a Effetto Hall, Magnetometri, …
Sensori Ottici	CMOS, CCD, fotodiodi, fotocellule,…
Sensori di Calore	Bolometri, Calorimetri, …
Sensori Inerziali	Accelerometri, Giroscopi, …
Sensori di Prossimità	Infrarossi, laser, ultrasuoni, …
Sensori di Posizione	Switch, Encoders, Potenziometri,…

...

I **Sensori di forza** vengono utilizzati per conoscere le forze agenti in genere su una certa superficie. Esempi d'uso sono bilance digitali, guanti sensoriali, mani robotiche e ascensori .

I **sensori magnetici** vengono utilizzati per diversi scopi, sia per l'orientamento che per conoscere posizione e prossimità. Per quest'ultimo compito tali sensori si utilizzano nel caso si debba conoscere in un certo intorno la presenza di corpi ferromagnetici. I magnetometri invece vengono spesso utilizzati per conoscere la direzione del nord magnetico e quindi l'orientazione.

I **sensori ottici** trasformano le immagini (fotogrammi) in ingresso in segnale elettrico. Sono utilizzati nelle fotocamere digitali, nelle webcam e nelle videocamere, per i compiti comuni di tali dispositivi, ma non solo. Un esempio di applicazione alternativo potrebbe essere la conoscenza della posizione di un oggetto mediante triangolazione passiva (per il task è necessario avere due camere disposte sullo stesso piano a distanza nota tra loro, in modo da poter utilizzare le conoscenze trigonometriche e algebriche per il calcolo della distanza). Per triangolazione passiva si intende una tecnica di calcolo della distanza (e quindi posizione rispetto ad un punto noto) che fa soltanto uso di sensori, a differenza della triangolazione attiva che fa uso di almeno una coppia sensore – emettitore (tecnica utilizzabile ad esempio con un emettitore laser ed un fotoricettore).

I **sensori di posizione** trasformano movimenti o rotazioni in segnali elettrici. Gli Switch sono tra i più semplici di questo tipo e forniscono soltanto l'informazione "contatto/non contatto". Per tali motivi possono essere utilizzati per rilevare posizioni specifiche, come ad esempio la posizione di fine corsa del giunto di un robot. Gli encoder sono invece molto più complessi. A seconda dalla struttura, dal loro uso e dall'energia ausiliaria che utilizzano gli encoder possono essere una combinazione tra le seguenti:

LINEARE / ROTATIVO

OTTICO / MAGNETICO

INCREMENTALE / ASSOLUTO

La prima voce fa riferimento alla struttura del sensore, che usa una superficie rettilinea nel caso dell'encoder lineare, o un disco nel caso di encoder rotativo. La seconda voce fa riferimento al principio di funzionamento e quindi al tipo di energia ausiliaria utilizzata. Nel caso di encoder ottico viene usata almeno una coppia sensore – emettitore ottici e nella superficie movente o rotante sono presenti dei fori per permettere il passaggio del raggio. Al muoversi della superficie il raggio viene ripetutamente occluso e, conoscendo la distanza tra i fori è possibile ricavare lo spostamento rispetto al punto di partenza. I sensori magnetici invece utilizzano lo stesso principio ma utilizzano un sensore magnetico e delle piccole superfici ferromagnetiche poste sulla superficie movente in modo che, al muoversi della stessa, venga ripetutamente variato il campo magnetico percepito dal sensore. Gli encoder incrementali, dotati di un solo anello di fori, forniscono soltanto l'indicazione dello spostamento rispetto al punto di partenza, che quindi deve essere noto. Per contro, gli encoder assoluti, dotati di diversi anelli concentrici di fori e una coppia sensore – emettitore per ogni anello, forniscono una stringa binaria (1/0 come acceso/spento, luce/non luce, campo magnetico variato/non variato) che, secondo un'opportuna codifica[16], fornisce la posizione assoluta del sistema. Gli encoder rotativi ottici sono tra quelli di uso più comune, e possono essere utilizzati per conoscere la posizione dei giunti in esoscheletri o in robots oppure per calcolare la distanza approssimativamente percorsa da un veicolo o un robot se montati negli assi delle ruote, tecnica che viene chiamata Odometria.

[16] La codifica della sequenza di bit restituiti da un encoder assoluto segue il cosiddetto "Codice Gray" [11].

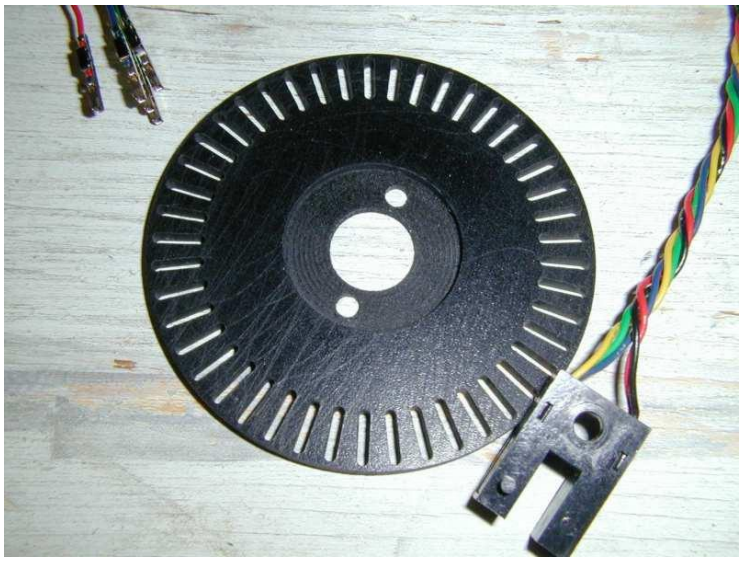

Figura 14 : Disco di un encoder ottico/rotativo/incrementale

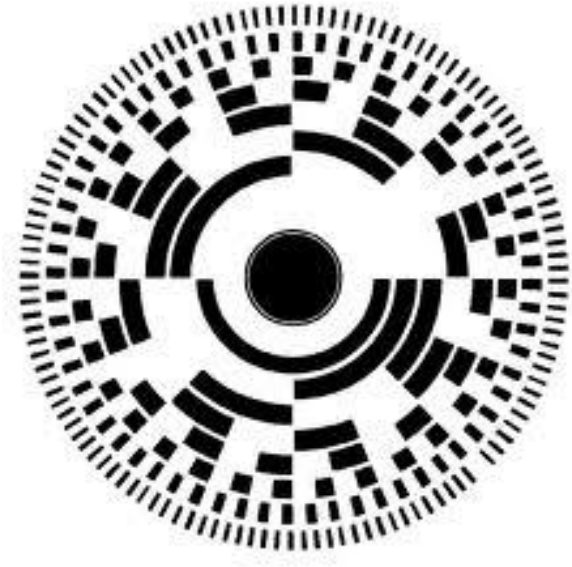

Figura 15 : Schema di foratura di un encoder rotativo/assoluto

3.3 SENSORI INERZIALI

Ai fini del lavoro qui proposto è interessante considerare i principi di funzionamento dei **sensori inerziali** (ovvero accelerometri e giroscopi) e dei magnetometri.

3.3.1 ACCELEROMETRO

Un accelerometro è un sensore in grado di misurare le accelerazioni (forze) percepite in genere lungo una direzione, detta direzione sensibile. Esistono diversi tipi di accelerometri, ognuno dei quali utilizza un differente principio di funzionamento per misurare l'accelerazione: Piezoresistivi, capacitivi, piezoelettrici, estensimetrici, laser e così via. In figura viene riportato lo schema di funzionamento tipico di un accelerometro.

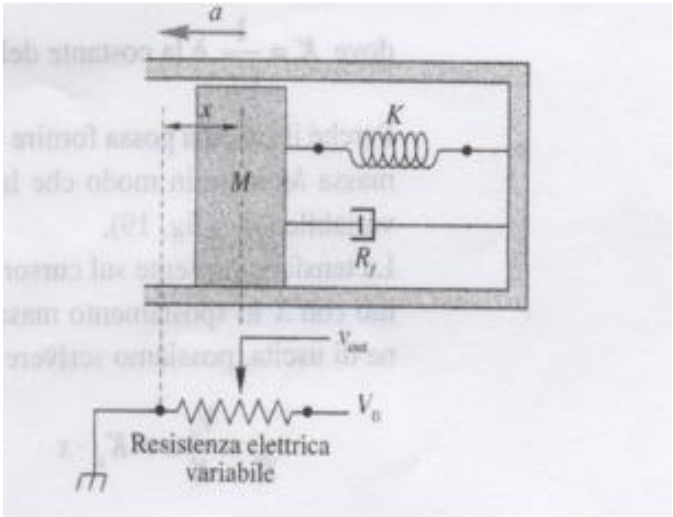

Figura 16: Schema di funzionamento di un Accelerometro

Nella maggior parte degli accelerometri, il principio di funzionamento è il medesimo: si basa sulla rilevazione dell'inerzia di una massa quando viene sottoposta ad un'accelerazione. La massa viene sospesa ad un elemento elastico, mentre un qualche tipo di sensore ne rileva lo spostamento rispetto alla struttura fissa del dispositivo. In presenza di un'accelerazione, la massa (che è dotata di una propria inerzia) si sposta dalla propria posizione di riposo in

modo proporzionale all'accelerazione rilevata. Il sensore trasforma questo spostamento in un segnale elettrico acquisibile dai moderni sistemi di misura. Utilizzando tre accelerometri disposti ortogonalmente l'uno rispetto al piano dell'altro è possibile conoscere le componenti dell'accelerazione totale agente sul sensore nei tre assi del sistema di riferimento

Se fino a pochi anni fa gli accelerometri erano destinati ad usi scientifici, militari o civili "speciali", oggi con l'evoluzione dell'elettronica, la riduzione dei costi e lo sviluppo delle applicazioni, gli accelerometri vengono utilizzati sempre più su oggetti d'uso comune.

Alcuni accelerometri miniaturizzati si ritrovano in apparecchi portatili allo scopo di ruotare automaticamente l'orientamento della visualizzazione sullo schermo (da verticale a orizzontale e viceversa), a seconda se il dispositivo sia posto in orizzontale o verticale. La medesima tecnologia è a bordo dei gamepad di alcune console giochi, permettendo, con la sola inclinazione dei medesimi, di comandare lo svolgimento dei giochi. Esempio, nella piattaforma Wii della Nintendo, l'uso di accelerometri nei telecomandi, permette un'interattività molto superiore alla concorrenza. Un'altra applicazione sempre più comune, è quella utilizzata per la rilevazione dell'accelerazione laterale nei veicoli, allo scopo di controllare le sbandate azionando opportunamente il sistema di frenatura.

3.3.2 GIROSCOPIO

Un giroscopio è un dispositivo per la misurazione o il mantenimento dell'orientazione basato sui principi di conservazione del momento angolare. Meccanicamente, un giroscopio è un disco nel quale l'asse è capace di assumere qualsiasi orientazione.

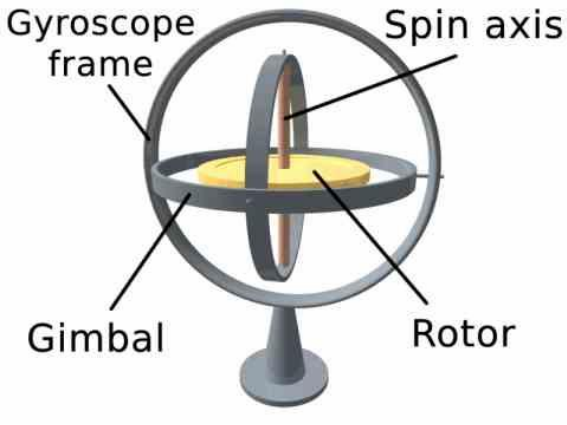

Figura 17 : Giroscopio meccanico

Il primo giroscopio era formato da un rotore montato su una staffa che lascia l'asse di rotazione libero di muoversi soltanto su un piano. In fase di movimento viene generato un momento dell'asse stesso in direzione ortogonale alla forza coppia impressa. La misura di tale moto fornisce una misura della velocità angolare nella rotazione.

Con l'evolvere del tempo sono stati sviluppati diversi tipi di giroscopi, basati su differenti principi operativi, Alcuni esempi sono i giroscopi MEMS, presenti in molti dispositivi di elettronica di consumo, i giroscopi a fibra ottica e i giroscopi quantici, estremamente sensibili.

In particolare, i giroscopi MEMS a sistema vibrante sono costituiti da una massa sospesa attraverso dei sistemi elastici che le permettono di muoversi in entrambe le direzioni x e y. Un'elaborazione del segnale permette di ricavare la velocità angolare. I giroscopi MEMS di ultima generazione, come ad esempio quelli sviluppati in ST Microelectronics, sono principalmente a un asse (giroscopi yaw) e a due assi (giroscopi roll-pitch e pitch-yaw). I nuovi sistemi MEMS vengono utilizzati con alte prestazioni nel rilevamento dello spostamento angolare nelle applicazioni di interazione uomo – macchina, nei sistemi di navigazione portatili (integrati in smartphone e navigatori per auto) e nella stabilizzazione delle immagini di fotocamere e videocamere digitali.

3.3.3 MAGNETOMETRO

Strumento che serve a misurare la direzione e l'intensità di un campo magnetico e in particolare del campo magnetico terrestre. Si hanno vari tipi di magnetometri per la misurazione degli elementi del campo magnetico terrestre (componente orizzontale H, componente verticale Z, inclinazione I, declinazione D, intensità totale F). Storicamente, sono stati realizzati per primi i magnetometri ad ago per la misurazione della componente orizzontale. Anche i magnetometri hanno subito il processo di miniaturizzazione in atto sui sensori sopra descritti.

Figura 18 : Magnetometro in un circuito integrato

3.4 INERTIAL MEASUREMENT UNIT

Combinando Accelerometri e giroscopi si ottengono le IMU. Le IMU sono sistemi elettronici basati su sensori come accelerometri e giroscopi. In passato tali dispositivi avevano dimensioni macroscopiche e costi non trascurabili, per cui il loro utilizzo era ristretto a particolari campi di applicazione. Con l'avvento dei sensori MEMS si è riusciti a integrare tutto in board di pochi centimetri, aprendo nuove prospettive di utilizzo.

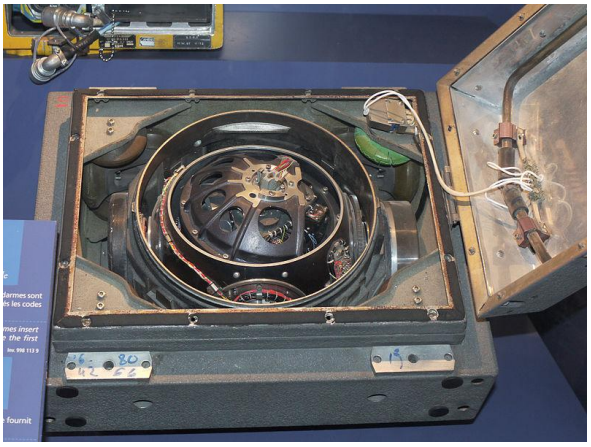

Figura 19 : IMU utilizzata nel missile S3

Il principale utilizzo delle IMU è dovuto ad applicazioni aerospaziali, navali e aeronautiche, essendo la componente fondamentale dei sistemi di guida inerziale. Esse da un po' di tempo vengono utilizzate anche per il motion tracking, ovvero per conoscere posizione e movimenti di un utente (posizioni nello spazio di braccia, gambe e busto), cercando di sostituire gli esoscheletri sensoriali (i cui sensori principali sono ad esempio gli encoder) che oltre ad essere abbastanza costosi sono piuttosto ingombranti. All'aumentare della precisione e della stabilità dei sensori integrati in una IMU il costo aumenta sensibilmente, per cui, per applicazioni come quella del motion tracking o di Dead Reckoning, è necessario trovare un compromesso tra economicità e qualità. Nelle board comprendenti tali dispositivi vengono in genere integrati anche dei magnetometri.

3.4.1 Principi di funzionamento

I sensori inerziali integrati in una IMU sono in genere una terna di giroscopi, una terna di accelerometri ed una terna di magnetometri. Tale configurazione può variare nel numero di sensori, ad esempio montando giroscopi a due assi per roll e pitch e un giroscopio ad un asse per lo yaw [17].

Oltre ai sensori inerziali appena elencati, altri sensori possono essere integrati per fornire informazioni addizionali o correggere i dati dei sensori. Ad esempio vengono spesso usati sensori di temperatura e di pressione. All'interno della board è sempre presente un microcontrollore che raccoglie i dati dei sensori, esegue dei calcoli per ricavare dati aggiuntivi (come ad esempio gli angoli di Eulero, ricavati da accelerazioni, velocità angolari e dati del magnetometro), per correggere eventuali derive dovuti a temperatura e pressione o errori del magnetometro dovuti a campi magnetici nell'ambiente circostante ed infine li trasforma al fine di poter essere restituiti in output (ad esempio ad un personal computer). L'algoritmo che restituisce gli angoli di Eulero integrato nel firmware della piattaforma è un algoritmo di fusione sensoriale che viene detto algoritmo AHRS che è l'acronimo di Attitude and Heading Reference System, basato su un filtro di Kalman. Tale algoritmo ricava i valori degli angoli di Eulero, necessari per riportare i valori di accelerazioni e velocità angolari dal body frame al frame navigazionale. Inoltre effettua delle correzioni ai valori uscenti dai sensori in base a delle matrici di covarianza interne. Le IMU microelettroniche possono girare a diverse frequenze. Ad ogni campionamento in genere viene creato un pacchetto dati (o bundle) comprendente tutte le informazioni ricavate dai sensori (accelerazioni in x, y e z, velocità angolari in intorno a x, y e z, intensità del campo magnetico nei tre assi, angoli di eulero) ed inserito in un buffer interno. Con delle funzioni di lettura dati, fornite di solito coi wrapper software (in dll o altri formati di libreria) è possibile estrarre tali pacchetti ed utilizzarli nella propria applicazione. Se il buffer interno viene riempito di default i nuovi pacchetti andranno a sovrascrivere i pacchetti più vecchi, ed alla richiesta di lettura dati verranno forniti i pacchetti più vecchi. Per evitare questo comportamento, nelle API fornite viene di solito fornita una funzione per settare la configurazione della piattaforma, con la quale è possibile specificare ad esempio che debba essere restituito il pacchetto più recente. In alternativa in certe piattaforme è possibile

impostare la lunghezza della coda a 1. In tal modo ad ogni campionamento si andrà a sovrascrivere il campione precedente e quindi alla richiesta di lettura verrà restituito sempre l'ultimo pacchetto.

3.5 GLOBAL POSITIONING SYSTEM

Il sistema di posizionamento globale è un sistema di localizzazione e navigazione satellitare civile. Ebbe la nascita nel 1991 negli Stati Uniti col nome di SPS[17], acronimo di Standard Positioning System, andando a sostituire il sistema precedente, il *Transit.* Le componenti di base del GPS sono [18]:

- **Un complesso di 32 satelliti**, disposti in orbita attorno alla terra, che trasmettono i messaggi di effemeride, che contengono la data e l'ora del messaggio ed i dati orbitali del satellite.
- **Una rete di tracking station,** il cui compito è ricavare i messaggi dei satelliti al fine di determinare i parametri dell'orbita
- **Un centro di calcolo,** che raccoglie i dati dalle tracking station da cui calcola i parametri orbitali
- **Delle injection station,** che raccolgono i parametri orbitali ricavati dal centro di calcolo e li inviano al satellite interessato. Il satellite registrara tali dati e li invia agli utenti
- **Un ricevitore GPS,** il cui compito è appunto quello di ricevere i messaggi del sistema per calcolare le proprie coordinate.

Un messaggio inviato dal satellite ha tre componenti: un codice pseudo casuale, gli effemeridi e i dati dell'almanacco. Il primo è un codice identificativo del satellite. I secondi invece sono i parametri del satellite, ovvero stato del satellite, data e ora correnti. I dati dell'almanacco raccolgono invece le informazioni orbitali del satellite. Il ricevitore GPS, i dati da più satelliti e ricavando le distanze in base al tempo di volo (confrontando l'ora

[17] In america l'SPS fu differenziato dal PPS, il Precision Positioning System, che veniva utilizzato per scopi militari. L'SPS era di fatto volutamente meno preciso del PPS per questioni di sicurezza.

interna del ricevitore con l'ora del pacchetto) riesce a calcolare approssimativamente le coordinate di latitudine, longitudine ed altitudine in cui si trova. Per fare ciò sono comunque necessari almeno 3 satelliti, e almeno 4 per avere una stima corretta dell'altitudine. Maggiore è il numero di satelliti, più precise saranno le coordinate ricavate. La procedura per il calcolo delle coordinate viene detta *trilaterazione* che è simile alla triangolazione meno il fatto che non vengono usate informazioni sugli angoli.

I dati vengono ricevuti secondo il protocollo NMEA. Tale protocollo è stato creato e viene tutt'ora gestito dalla National Marine Electronics Association. NMEA definisce una serie di standard: tra questi vi è l'NMEA 0183 che è lo standard utilizzato dal GPS.

3.5.1 Standard NMEA

Tale protocollo è uno standard di comunicazione tra dispositivi digitali. In particolare, NMEA 0183 è lo standard di comunicazione in un bus con baud rate di 4800bps. Nato principalmente per scopi nautici, viene adesso utilizzato in molti ambiti, tra cui la comunicazione dei dati di navigazione da un ricevitore GPS ad un computer. Il modello dello standard si basa su un dispositivo capace solo di inviare dati, detto **talker,** ed uno o più dispositivi in grado di riceverli, detti **listener.** Essendo uno standard per molti dispositivi, i formati di dati del sistema sono numerosi.

La comunicazione viene organizzata in frasi NMEA o sentences. Tali frasi hanno tutte la seguente struttura:

$PREFISSO,dato1,dato2 ... datoN-1,datoN*CHECKSUMCRLF

Esse possono contenere un massimo di 82 caratteri, inclusi i caratteri speciali $ e CRLF. Tali caratteri segnano l'inizio e la fine di una frase NMEA. Il prefisso indica, nei primi due

caratteri, il tipo di talker che sta trasmettendo. Il GPS viene identificato dal prefisso GP. I rimanenti tre caratteri danno informazioni sul contenuto e sul tipo della frase.

Le frasi più utilizzate in ambito GPS sono le seguenti:

- $GPRMA Recommended Minimum specific loran-C data
- $GPRMB Recommended Minimum navigation info
- $GPRMC Recommended Minimum specific GPS/TRANSIT data
- $GPGGA Global Positioning System fix data
- $GPGSA GPS DOP and Satellites active
- $GPGLL Geographic position, Latitude, Longitude
- $GPGSV GPS Satellites in View
- $GPRTE Routes

La parte dopo il prefisso è suddivisa in campi, ognuno dei quali fornisce un tipo di informazione. I campi sono di lunghezza variabile e suddivisi tra loro tramite virgole. Qualora una sentence sia composta da n campi ma non si ha informazione su alcuni di essi, le posizioni corrispondenti a tali campi vengono lasciati vuoti, ma le virgole saranno comunque mantenute. Il checksum serve a verificare la correttezza dei dati ricevuti. Esso viene separato dal corpo della frase da un asterisco. Il calcolo del checksum viene effettuato sul payload della frase, che è la parte che va da dopo il carattere speciale di inizio frase $ a prima dell'asterisco: per ogni carattere del payload viene calcolato lo XOR col carattere precedente (il carattere precedente al primo carattere del payload è 0). Dopo di ciò il valore ottenuto viene convertito in esadecimale. Per verificare la correttezza del messaggio ricevuto è quindi sufficiente effettuare tale procedura e confrontare il checksum ottenuto con il checksum della frase: se sono uguali il messaggio è integro, altrimenti no. La comunicazione seriale RS232 è abbastanza stabile ed affidabile, ma per comunicazioni tramite altri canali come infrarossi, onde radio o GSM il checksum è uno strumento abbastanza utile se non fondamentale.

Alcuni esempi di frasi NMEA sono le seguenti:

*$GPRMC,062352.000,A,4507.15643,N,00738.21815,E,61.9,65.5,050811,0.0,W*7F*

*$GPGGA,062352.000,4507.15643,N,00738.21815,E,2,15,0.7,258.55,M,48.2,M,,*6D*

*$GNGSA,A,3,08,02,07,10,26,73,28,71,68,84,72,19,1.1,0.7,0.8*2C*

*$GNGSA,A,3,05,15,21,,,,,,,,,1.1,0.7,0.8*21*

*$GPGSV,3,1,11,02,04,218,24,05,66,233,36,07,32,054,41,08,60,048,50*77*
*$GPGSV,3,2,11,10,29,163,44,15,18,293,44,19,01,046,35,21,05,328,19*79*
*$GPGSV,3,3,11,26,53,303,48,27,07,242,,28,46,137,45,,,,*4D*
*$GLGSV,2,1,08,68,71,309,50,71,74,254,47,72,43,036,42,73,47,155,33*6B*
*$GLGSV,2,2,08,74,06,036,32,75,03,348,,78,42,036,,84,21,231,40*65*

3.5.2 Descrizione delle frasi di GPRMC e GPGGA

Ai fini del presente lavoro le frasi NMEA scelte sono quelle contrassegnate dai prefissi $GPRMC e $GPGGA. Da queste vengono estrapolati i dati di interesse, ovvero quelli relativi a posizione LLA, velocità, angolo di track, fix e numero dei satelliti.

- La stringa $GPRMC contiene informazioni sull'ora UTC (Universal Time Coordinated , tempo coordinato universale, che si basa sul fuso orario del meridiano di Greenwich, detto GMT, Greenwich Mean Time) in formato hhmmss.mmm (ad esempio 112343.985 indica l'ora GMT 11:23:43 e 985 millesimi di secondo), la data in formato ddmmyy (ad esempio 120413 indica la data del 12 Aprile 2013), Latitudine e Longitudine espresse in sessagesimali nel formato ggpp.ss (gradi primi.secondi, ad esempio 3744.655 indica una latitudine in sessagesimali di 37°44' 655"), l'angolo di Track, ovvero la direzione di movimento, espressa in gradi ggg.dd rispetto al nord), la velocità del veicolo espressa in nodi e diverse altre informazioni.

L'immagine seguente descrive ciascuno dei campi della frase.

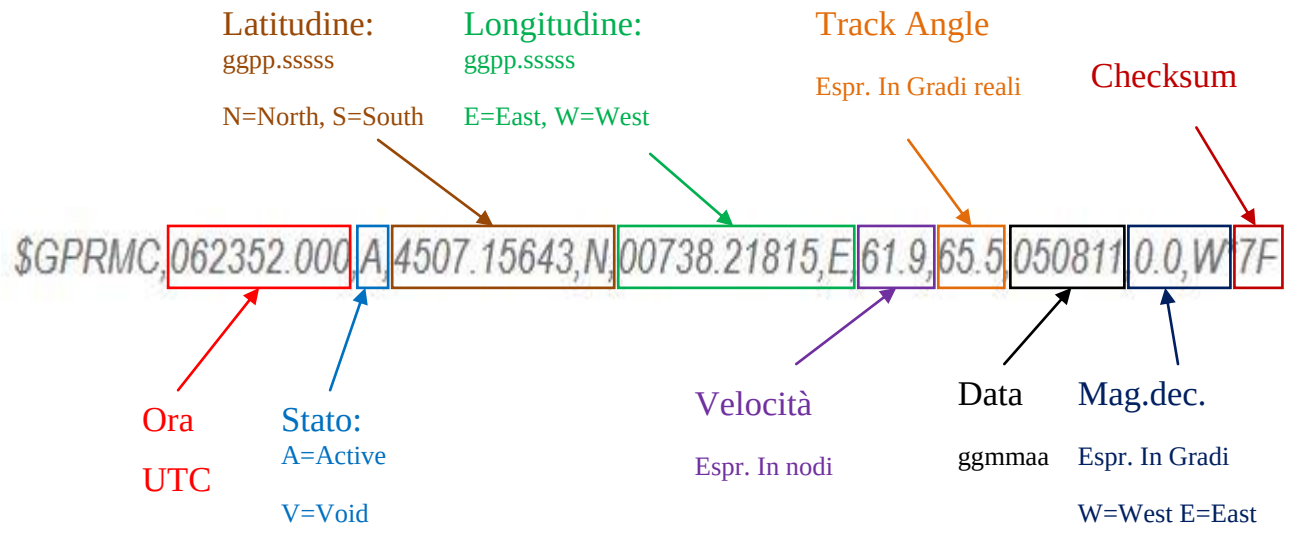

Figura 20 : Descrizione campi frase GPRMC

- La frase $GPGGA, oltre a comprendere i campi di latitudine, longitudine e ora UTC include l'altitudine (dell'antenna rispetto al livello del mare) e dati relativi alla qualità di tali informazioni, quali ad esempio il tipo di fix GPS, che può essere 0 (segnale assente, messaggio non valido), 1 (segnale ok, messaggio GPS valido) o 2 (segnale ok, qualità ottima, correzione differenziale DGPS), il numero di satelliti in vista, e così via.

L'immagine seguente descrive i campi della frase GGA.

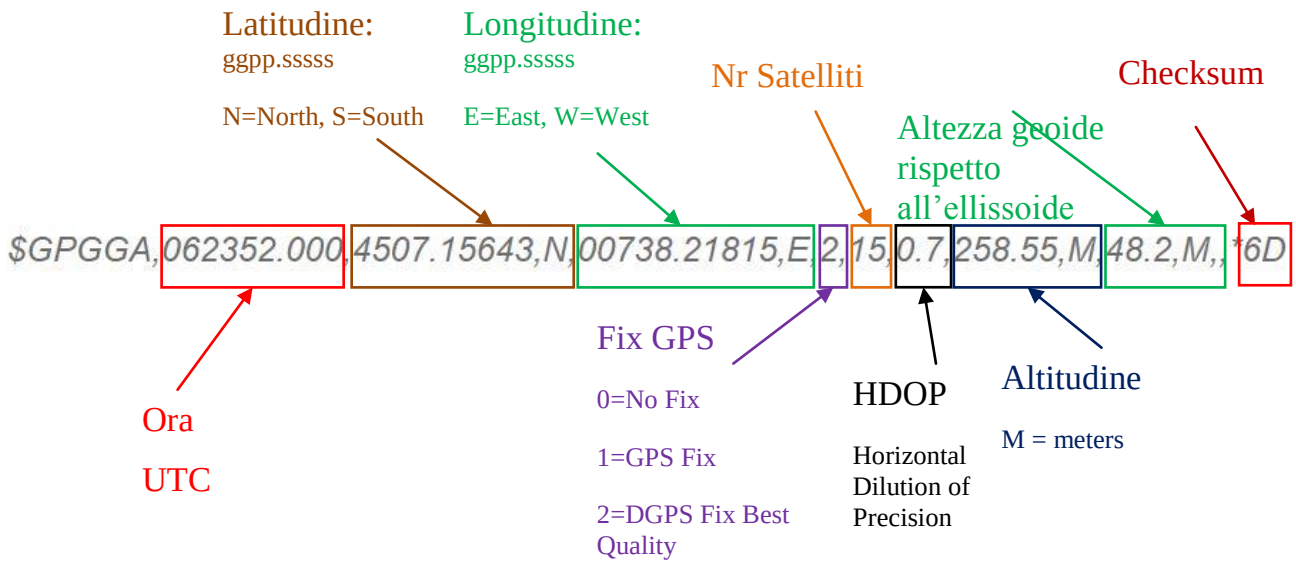

Figura 21 : Descrizione campi frase GGA

CAPITOLO 4. ALGORITMO DI NAVIGAZIONE INERZIALE

In questo capitolo descriveremo innanzitutto gli aspetti teorici dell'algoritmo, passando poi ai dettagli implementativi, elencando le piattaforme hardware e software utilizzate e la descrizione sul software implementato.

4.1 DESCRIZIONE DELL'ALGORITMO

L'algoritmo implementato prende in input le accelerazioni e gli angoli di Eulero dati dai sensori, e, data la posizione LLA, velocità e assetto all'istante t e restituisce in output il nuovo vettore di stato dell'istante t+dt (l'istante corrente), contenente le nuove coordinate e velocità.

L'algoritmo, che implementa il filtro di Kalman esteso (EKF, versione non lineare del filtro di kalman) è composto essenzialmente da due fasi: Predizione e Update. La fase di predizione riceve i valori inerziali da una IMU per "predire" gli stati della navigazione del veicolo (latitudine, longitudine, altezza ellissoidrica e velocità). La fase di update riceve i dati del GPS per "correggere" lo stato ottenuto nella fase di predizione.

4.1.1 Descrizione teorica

Le equazioni che descrivono il modello del sistema sono le seguenti:

$$\dot{\boldsymbol{x}}(\boldsymbol{t}) = \boldsymbol{f}\,(\boldsymbol{x}(\boldsymbol{t}), \boldsymbol{u}(\boldsymbol{t}), \boldsymbol{t}) + \boldsymbol{\omega}(\boldsymbol{t})$$

$\boldsymbol{x}(\boldsymbol{t})$ = **[p v b] : vettore di stato.** Composto da posizione, velocità e bias nelle accelerazioni al tempo t

p = $[\boldsymbol{\Lambda}\ \boldsymbol{\Phi}\ \mathbf{h}]^{\boldsymbol{T}}$ (longitudine , latitudine ed altezza ellissoidrica)

v = $[\boldsymbol{v_N}\ \boldsymbol{v_E}\ \boldsymbol{v_D}]^{\boldsymbol{T}}$ (velocità rispetto al frame navigazionale)

$\boldsymbol{b} = [\boldsymbol{b}_{a_N}\ \boldsymbol{b}_{a_E}\ \boldsymbol{b}_{a_D}]$ (bias nelle acc. rispetto al frame navigazionale)

u(t) : segnale noto o vettore di input. Nel nostro caso i dati sulle accelerazioni all'istante t

$$\boldsymbol{y}(\boldsymbol{t}) \;=\; \boldsymbol{h}(\boldsymbol{x}(\boldsymbol{t}), \boldsymbol{t}) \;+\; \boldsymbol{v}(\boldsymbol{t}) \qquad\qquad \boldsymbol{y} = [\,\boldsymbol{p}\;\;\boldsymbol{v}\,]^{\boldsymbol{T}}$$ vettore di output del sistema

La funzione *h* è rappresentata dalla matrice ***H*** che in questo caso sarà la matrice

$$\mathrm{H} \;=\; \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \;=\; [I^{6x6} \mid 0^{6x3}]^T$$

La funzione *f* è rappresentata dal seguente vettore:

$$\begin{bmatrix} \dfrac{v_N}{R_{NS}+h} \\ \dfrac{v_E}{(R_{EW}+h)\cos\Lambda} \\ -v_D \\ \dfrac{-v_E^2 \sin\Lambda}{(R_{EW}+h)\cos\Lambda} + \dfrac{v_N v_D}{R_{NS}+h} + \; a_N - \; b_{a_N} \\ \dfrac{v_E v_N \sin\Lambda}{(R_{EW}+h)\cos\Lambda} + \dfrac{v_E v_D}{R_{NS}+h} + \; a_E - \; b_{a_E} \\ -\dfrac{v_E^2}{(R_{EW}+h)} + \dfrac{v_N^2}{R_{NS}+h} + g + \; a_D - \; b_{a_D} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

4.1.2 Fase di predizione

- Viene costruita la matrice dei coseni direttori tramite gli angoli di eulero ricavati dalla piattaforma inerziale:

$$DCM = \begin{bmatrix} \cos\theta\cos\Psi & \cos\Phi\sin\Psi + \sin\Phi\sin\theta\cos\Psi & \sin\Phi\sin\Psi - \cos\Phi\sin\theta\cos\Psi \\ -\cos\theta\sin\Psi & \cos\Phi\cos\Psi - \sin\Phi\sin\theta\sin\Psi & \sin\Phi\cos\Psi + \cos\Phi\sin\theta\sin\Psi \\ \sin\theta & -\sin\Phi\cos\theta & \cos\Phi\cos\theta \end{bmatrix}$$

  Dove Φ,θ e Ψ indicano rispettivamente gli angoli di roll, pitch e yaw. Tale matrice servirà a rappresentare le accelerazioni dei sensori nel frame navigazionale.

- La DCM viene moltiplicata per la matrice di rotazione dovuta alla declinazione magnetica (angolo tra il nord geografico e il nord magnetico). Questo poiché lo yaw calcolato dai sensori fa riferimento al nord magnetico.
- La matrice ottenuta viene moltiplicata per l'intervallo di tempo *dt* che intercorre tra la misurazione corrente e la precedente.
- Viene calcolata la matrice di transizione $\boldsymbol{F}$ nel modo seguente:

$$F = \begin{bmatrix} . & . & . & \frac{dt}{R_{NS} + Alt} & . & . & . & . & . \\ . & . & . & . & \frac{dt}{(R_{EW} + Alt)\cos Lat} & . & . & . & . \\ . & . & . & . & . & -dt & . & . & . \\ . & . & . & . & . & . & -dcm_{11} & -dcm_{12} & -dcm_{13} \\ . & . & . & . & . & . & -dcm_{21} & -dcm_{22} & -dcm_{23} \\ . & . & . & . & . & . & -dcm_{31} & -dcm_{32} & -dcm_{33} \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \end{bmatrix}$$

E da essa viene ricavata F_d , discretizzazione di $\boldsymbol{F}$: $F_d = F + I$ (I è la matrice identità di dimensione opportuna).

- Viene calcolata la matrice ***G*** nel seguente modo:

$$G = \begin{bmatrix} . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ dcm_{11} & dcm_{12} & dcm_{13} & dt & . & . \\ dcm_{21} & dcm_{22} & dcm_{23} & . & dt & . \\ dcm_{31} & dcm_{32} & dcm_{33} & . & . & dt \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \end{bmatrix}$$

Da questa viene calcolata G_d ovvero la discretizzazione di G: $G_d = \left(\frac{1}{2}F + I\right) * G$

Le matrici vengono discretizzate poiché i campionamenti vengono fatti in tempo discreto. La matrice *dcm* i cui valori sono inclusi in F e G è comprensiva delle operazioni sopra descritte: $dcm = R_{decmag}$ * DCM * dt

- Viene calcolato il vettore degli ingressi u_{k-1} , che contiene le accelerazioni ricavate dai sensori:

$$u = \begin{bmatrix} a_x \\ a_y \\ a_z \\ 0 \\ 0 \\ 9.81 \end{bmatrix}$$

- Viene calcolato lo stato a priori nel seguente modo:

$$x_k = F_d * x_{k-1} + G_d * u_{k-1}$$

Il calcolo esteso dello stato è quindi il seguente:

$$x_k = x_{k-1} + F * x_{k-1} + \frac{1}{2}F * G * u_{k-1} + G * u_{k-1}$$

Ovvero mediante F viene calcolato l'incremento di posizione dato dalla velocità allo stato x_{k-1} mentre mediante G viene calcolato l'incremento di posizione e velocità dato dalle accelerazioni osservate.

Tale formula corrisponde alla formulazione delle leggi classiche del moto:

- $x = x_0 + v * dt + \frac{1}{2} a * dt^2$
- $v = v_0 + a * dt$

- Viene calcolata la matrice di covarianza degli errori:

$$P_k = F_d * P_{k-1} * F_d^T + Q$$

4.1.3 Fase di Correzione

- Viene calcolato il guadagno ottimo di Kalman:

$$K_k = P_k * H^T * (H * P_k * H^T + R)^{-1}$$

In cui il valore tra parentesi rappresenta la matrice di innovazione della covarianza S_k. Esso corrisponde al valore di correzione da applicare a stato e covarianza.

- Viene corretto il vettore di stato:

$$x_k(+) = x_k(-) + K_k * (y - H * x_k(-))$$

- Viene corretta la matrice di covarianza:

$$P_k(+) = (I - K_k * H) * P_k(-)$$

4.2 STRUMENTI HARDWARE E SOFTWARE UTILIZZATI

Nel presente lavoro sono stati utilizzati diversi strumenti hardware e software. Come hardware sono stati utilizzati il ricevitore GPS Teseo II , la IMU XSens Mti, la IMU ST INemo M1. Il GPS è stato connesso al computer mediante dongle rs232, INemo mediante cavo miniUSB, mentre l'mti mediante il cavo XSens in dotazione. Come software sono state utilizzate le API per la gestione dei sensori, l'ambiente di sviluppo Visual Studio 2012 e l'ambiente MathWorks Matlab. Il software creato è stato sviluppato nel linguaggio di programmazione C#. Esso è composto dalla parte grafica, l'interfaccia che gestisce i parametri di sensori e algoritmo, e dalla parte algoritmica, che implementa la gestione delle IMU (mediante delle classi statiche wrapper) e l'algoritmo vero e proprio basato sul filtro di Kalman.

4.3 DESCRIZIONE DELL'HARDWARE

Figura 22 : ST iNemo M1 , XSens Mti, Teseo GPS board

4.3.1 ST iNemo M1

Unità di misura inerziale sviluppata da ST Microelectronics. Essa è la piattaforma successiva alla STEVAL-MKI062V2 nota come iNemoV2, che susseguì alla prima versione.

La board è composta da un giroscopio a 3 assi (roll,pitch,yaw) a scala di misura editabile (±250°/s, ±500°/s, ±2000°/s), un modulo geomagnetico (accelerometri e magnetometri) a 6

assi (scala degli accelerometri selezionabile tra ±2mg, ±4mg, ±8mg, ±16mg, scala dei magnetometri da ±1.3Gauss a ±8.1Gauss), un microcontrollore 32 bit basato su ARM, e sensori di pressione e temperatura. Essa viene alimentata e interfacciata mediante lo slot miniUSB integrato. Installato nel firmware è presento un algoritmo di fusione sensoriale AHRS che fornisce in output i dati ricavati dai sensori, e gli angoli di Eulero. Per iNemo sono state implementate le API (iNEMO_M1_SDK.dll) che forniscono le funzioni per la sua gestione e configurazione. Il frame di navigazione utilizzato da tale piattaforma ha configurazione NWU.

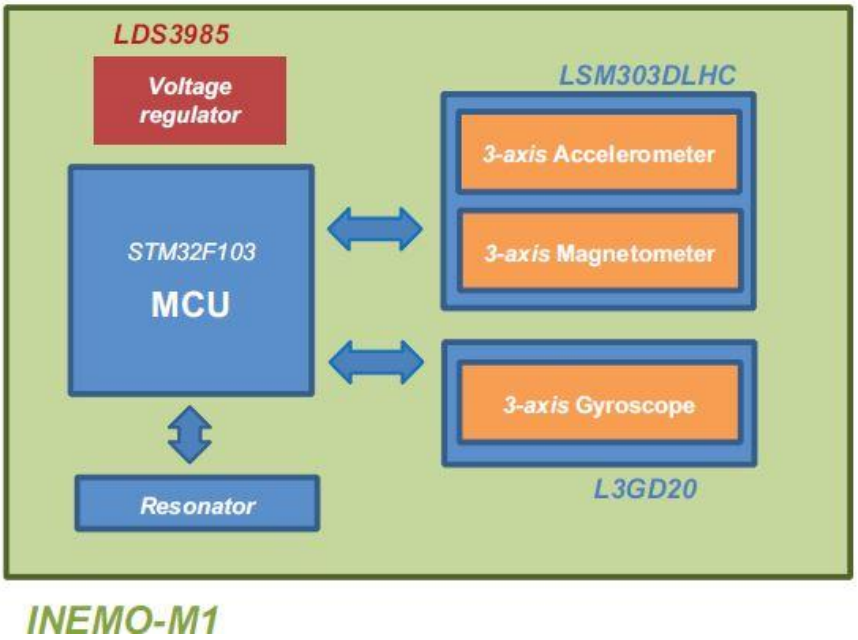

Figura 23: Schema tecnico ST iNemo M1

Per una descrizione tecnica completa si veda [13].

4.3.2 XSens Mti

Tale piattaforma è una IMU a 9 assi (Giroscopio 3D, Accelerometro 3D, magnetometro 3D) sviluppato dalla XSens Technologies B.V., la maggiore produttrice mondiale di dispositivi per Motion Tracking basati su tecnologia di sensori inerziali MEMS. Nella board vi è inoltre un sensore di temperatura. Le caratteristiche dei sensori in esso integrati sono descritte nella seguente tabella.

		rate of turn	acceleration	magnetic field	temperature
Unit		[deg/s]	[m/s^2]	[mGauss]	[°C]
Dimensions		3 axes	3 axes	3 axes	-
Full Scale	[units]	+/- 300*	+/- 50	+/- 750	-55...+125
Linearity	[% of FS]	0.1	0.2	0.2	<1
Bias stability	[units 1σ][11]	1	0.02	0.1	0.5[12]
Scale factor stability	[% 1σ][11]	-	0.03	0.5	-
Noise density	[units /√Hz]	0.05[13]	0.002	0.5 (1σ)[14]	-
Alignment error(15)	[deg]	0.1	0.1	0.1	-
Bandwidth	[Hz]	40	30	10	-
A/D resolution	[bits]	16	16	16	12

La piattaforma viene fornita in dotazione di cavo (convertitore USB/RS232), drivers e api (XSensCMT.dll) per la gestione e configurazione. E' integrato un algoritmo di sensor fusion basato su filtro di Kalman Esteso (EKF) che restituisce i dati dei sensori e gli angoli di Eulero. Il frame navigazionale utilizzato dall'mti può essere scelto tra le configurazioni NED e NWU.

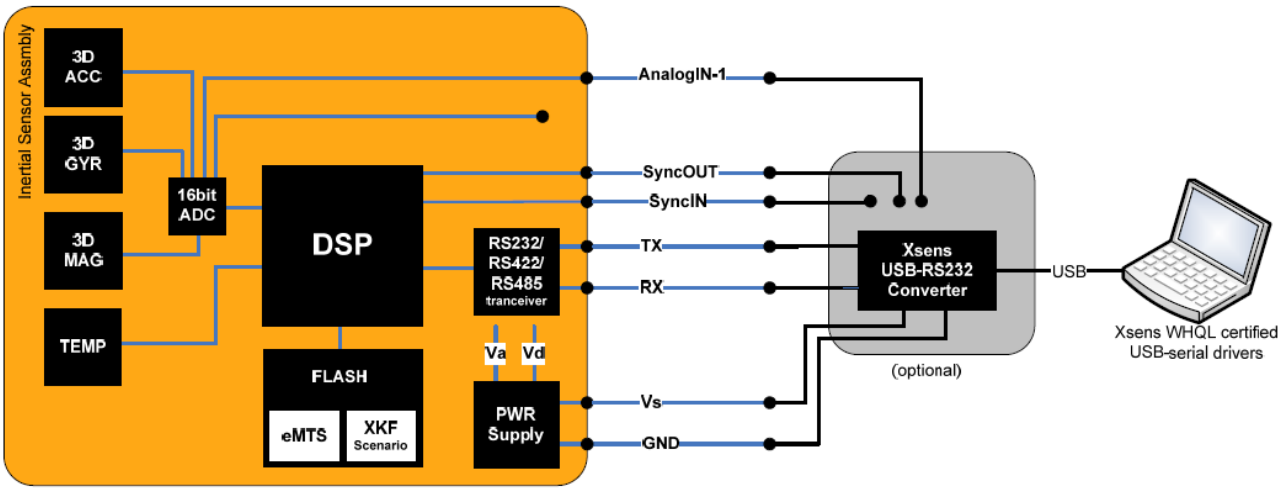

Figura 24: Schema Tecnico XSens Mti

4.3.3 GPS TESEO II

La board GPS utilizzata per questo lavoro è basata sul ricevitore STA8088, terza generazione di ricevitori GPS di ST Microelectronics. Nella board sono disponibili due porte seriali, una porta miniUSB, il connettore per l'alimentazione (2.5mm DC Power Plug), ed un led di accensione. Nel lavoro di tesi qui svolto è stata utilizzata la porta seriale (mediante un dongle USB-RS232 Converter) per la connessione al PC e la porta USB per l'alimentazione.

4.4 DESCRIZIONE DEL SOFTWARE

L'implementazione del software consta essenzialmente di 9 classi e 3 librerie. Oltre alle dll descritte sopra (per la gestione e configurazione delle IMU), è stata utilizzata la libreria ZedGraph.dll, un controllo .NET per la visualizzazione di grafici.

4.4.1 Descrizione delle classi

Le classi sono le seguenti:

- MainForm
- iNemoWrapper
- xSensWrapper
- GPSWrapper
- PosData
- MemsData
- GPSData
- MO
- Parameters

4.4.1.1 Classe MainForm

La classe **MainForm** è la classe principale. Essa eredita dalla classe System.Windows.Forms.Form e si occupa della gestione dell'interfaccia grafica, dell'istanziazione dei Wrapper dei dispositivi, dell'esecuzione del programma e dell'esecuzione dell'algoritmo. Il diagramma in figura mostra i principali campi e metodi.

MainForm
Classe
→ Form

Campi

- dcm : double[][]
- freqIN : long
- freqMainLoop : long
- freqXS : long
- IndIN : long
- IndXS : long
- ine : iNemoWrapper
- MainLoopTimer : Timer
- meanpitch : double[]
- meanroll : double[]
- oldTimeStMem : ulong[]
- Pns : double[][][]
- PositionData : PosData
- Qn : double[][][]
- R : double[][][]
- recordGPS : GPSData
- recordMems1 : MemData[]
- recordMems2 : MemData[]
- teseo : GPSWrapper
- useiNemo : bool
- useUpdate : CheckBox
- useXSens : bool
- usoGPS : bool
- xks : double[][][]
- xs : xSensWrapper

Proprietà

- PeriodIN : long
- PeriodMainLoop : long
- PeriodXS : long

Metodi

- clearButton_Click() : void
- ClearGraph() : void
- condiscBT_Click() : void
- CreateGraph() : void
- FilterPrediction() : void
- FilterUpdate() : void
- getDCM() : void
- InitFile() : void
- Initialization() : void
- InitializeComponent() : void
- loadDeviceConf() : void
- loadSettings() : void
- loadSettingsFromFile() : void
- MainForm()
- MainForm_Load() : void
- mainLoop() : void
- setDelay() : void
- setpcov() : void
- setpcov2() : void
- setvcov() : void
- setvcov2() : void
- start() : void
- stop() : void

Figura 25 : MainForm Class Diagram

Il mainForm implementa al suo interno un Timer (MainLoopTimer) impostato ad una frequenza di 10Hz che, quando il programma è avviato, richiama il metodo mainLoop. Tale metodo ha il compito di controllare la presenza di nuovi record dei MEMS, di chiamare il metodo filterPrediction() sui record trovati, di verificare la presenza de pacchetto GPS, di chiamare il metodo filterUpdate() su tale pacchetto e di aggiornare i dati sui controlli grafici. Essa inoltre implementa la gestione del tempo di convergenza dei sensori e del delay. La classe MainForm mantiene in memoria due record PosData, una per ogni piattaforma inerziale, in modo da contenere i valori uscenti dal filtro per ognuna delle due. Da notare che il metodo principale dell'applicazione gira a 10Hz, le IMU a 50Hz ed il ricevitore GPS a 1Hz. Per tale motivo è stata predisposta una comunicazione asincrona tra applicazione e dispositivi. Ad ogni giro nel mainLoop viene controllata la presenza di record nei buffer interni delle due piattaforme inerziali e nel buffer del wrapper GPS. Dato che la frequenza di campionamento dei sensori è superiore alla frequenza dell'applicazione, si tiene conto del fatto che nei buffer saranno presenti più record. In effetti il controllo sui buffer e la fase di predizione sul record prelevato vengono implementati in un ciclo while che gira finchè non ci sono più record nei buffer.

Di particolare interesse per quanto riguarda l'algoritmo sono le due funzioni *FilterPrediction()* e *FilterUpdate()* , che implementano la fase di predizione e di correzione del filtro di Kalman Esteso. Le funzioni di *Start()* e *Stop()* permettono di avviare e fermare l'applicazione (vengono richiamate alla pressione di un bottone dell'interfaccia, lo startButton). La funzione *Start()* richiama le funzioni di avvio delle IMU e del GPS, ed avvia il MainLoopTimer. La funzione Stop ferma il timer dell'applicazione e l'acquisizione dei dati da parte dei dispositivi. E' stato previsto il funzionamento simultaneo delle due board inerziali, in modo da poterne confrontare le prestazioni. Per tale motivo è stato necessario predisporre il software in modo da considerare i dati da entrambe le piattaforme. All'avvio dell'applicazione vengono caricati i valori di configurazione del filtro e i dati di configurazione per i sensori dai files di configurazione. Tali dati possono essere modificati

attraverso delle voci nei menu (Si veda la descrizione dell'interfaccia). La classe statica Parameters contiene i valori di default dell'applicazione che possono essere memorizzati sul file di configurazione del filtro: *filtersettings.conf*. Gli altri due files di configurazione sono devices.conf e settingfiles.ini. Nel primo sono contenuti i parametri di configurazione dei dispositivi, mentre nel secondo sono contenuti i nomi degli altri due files (nel caso si vogliano avere più files di configurazione del filtro o dei dispositivi basta modificare tali voci mediante la voce "Choose Setting Files" nel menu Edit.

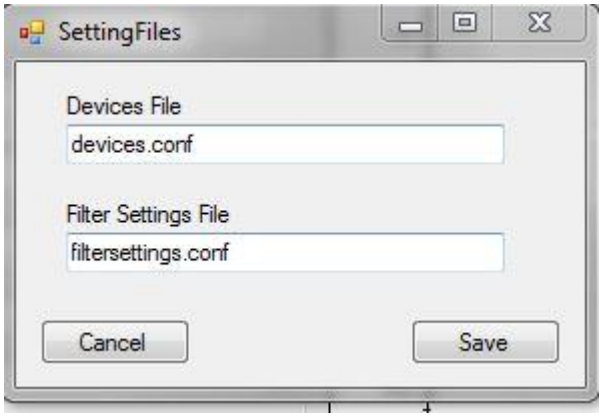

Figura 26 : Form per la scelta dei files di configurazione

Nel file di configurazione dei dispositivi sono contenute le preferenze riguardo i dispositivi da utilizzare, la relativa frequenza di campionamento e la porta COM su cui tale dispositivo è connesso. Tali parametri sono modificabili tramite un apposito Form accessibile dalla voce "Device Settings" nel menu Edit. Il ricevitore GPS non da la scelta della frequenza da utilizzare: Esso gira alla frequenza di 1Hz (1 campione al secondo).

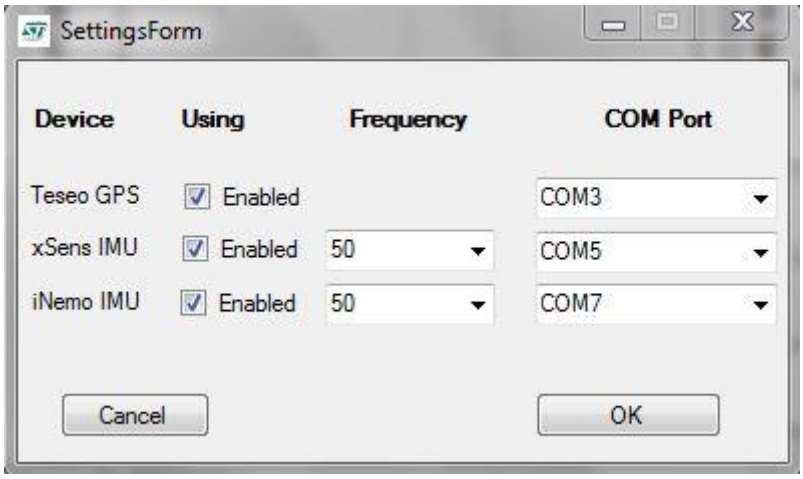

Figura 27 : Form per la configurazione dei dispositivi

Di default sono stati scelti dei valori di frequenza pari a 50Hz (50 campioni al secondo) per entrambi i dispositivi. XSens Mti permette una frequenza massima di sampling di 100Hz, mentre iNemo M1 per dati calibrati (ovvero i dati elaborati dal filtro AHRS interno) garantisce una frequenza massima di 50Hz.

I parametri del filtro che possono essere configurati sono le soglie di covarianza della posizione LL (Latitude Longitude) Pcov, di covarianza della velocità Vcov e di covarianza dell'altezza (Altitude) Hcov delle due piattaforme inerziali, il tempo di attesa per la convergenza dei sensori convt ed il tempo di delay. In particolare i parametri di covarianza vengono memorizzati all'interno della matrice R, ovvero la matrice di covarianza dei rumori di misura, utilizzata per calcolare il guadagno ottimo di Kalman (usato a sua volta per correggere stato e covarianza degli errori nella fase di update).

Le piattaforme inerziali al loro avvio hanno bisogno in media di un tempo di circa 10 secondi per la stabilizzazione dei valori. Il parametro convt permette di impostare tale ritardo nel programma in modo che i dati sensoriali ottenuti in quel range di tempo vengano scartati (tali valori risulterebbero poco significativi).

Subito dopo il tempo di convergenza dei sensori, alla prima ricezione di un record GPS viene effettuata l'inizializzazione dei parametri dell'algoritmo mediante la funzione Initialization() il cui codice è riportato qui sotto.

```
public void Initialization(GPSData record)
{
            //istanziazione dei vettori di stato, uno per piattaforma
            xks = new double[Parameters.FilterLogFiles.Length][][];
            for (int i = 0; i < xks.Length; i++)
            {
                MO.mat_creat(ref xks[i], 9, 1, 0.0);
                xks[i][0][0] = Convert.ToDouble(record.latitudine) * Parameters.D2R;
                xks[i][1][0] = Convert.ToDouble(record.longitudine) * Parameters.D2R;
                xks[i][2][0] = Convert.ToDouble(record.altitudine);
            }
            MO.mat_creat(ref y, 6, 1, 0.0);
            MO.mat_creat(ref Pn, 9, 9, 1.0);
            //istanziazione delle matrici di covarianza, una per piattaforma
            Pns = new double[Parameters.FilterLogFiles.Length][][];
            for (int i = 0; i < Pns.Length; i++)
                MO.mat_creat(ref Pns[i], 9, 9, 1);
            //istanziazione delle matrici utilizzate dal filtro
            Qn = new double[2][][];
            R = new double[2][][];
            MO.mat_creat(ref Qn[0], 9, 9, 0.0);
            MO.mat_creat(ref Qn[1], 9, 9, 0.0);
            MO.mat_creat(ref R[0], 6, 6, 0.0);
            MO.mat_creat(ref R[1], 6, 6, 0.0);
            double val = 0.18 * 0.18;
            double val2 = 0.18 * 0.18;
            //covarianza rumore di processo
            Qn[0][3][3] = Qn[0][4][4] = Qn[0][5][5] = val;
            Qn[1][3][3] = Qn[1][4][4] = Qn[1][5][5] = val2;
            val = 1.0e-8;
            val2 = 1.0e-8;
            Qn[0][6][6] = Qn[0][7][7] = Qn[0][8][8] = val;
            Qn[1][6][6] = Qn[1][7][7] = Qn[1][8][8] = val2;
            //covarianza rumore di misura
            R[0][0][0] = R[0][1][1] = pcov;
            R[1][0][0] = R[1][1][1] = pcov2;
            R[0][2][2] = hcov;
            R[1][2][2] = hcov;
            R[0][3][3] = R[0][4][4] = vcov;
            R[1][3][3] = R[1][4][4] = vcov2;
            R[0][5][5] = 0.2;
            R[1][5][5] = 0.2;
            //altre matrici
            MO.mat_creat(ref Uk, 6, 1, 0.0); //vettore ingressi
            MO.mat_creat(ref Kn, 9, 6, 0.0); //guadagno di kalman
            MO.mat_creat(ref F, 9, 9, 0.0); //matrice di transizione
            MO.mat_creat(ref G, 9, 6, 0.0); //matrice del modello
            MO.mat_creat(ref Fd, 9, 9, 0.0);   //discretizzazione di F
            MO.mat_creat(ref Gd, 9, 6, 0.0);    //discretizzazioe di G
            //variabili temporanee
            MO.mat_creat(ref temp99, 9, 9, 0.0);
            MO.mat_creat(ref temp99sec, 9, 9, 0.0);
            MO.mat_creat(ref temp96, 9, 6, 0.0);
            MO.mat_creat(ref temp91, 9, 1, 0.0);
            MO.mat_creat(ref temp91sec, 9, 1, 0.0);
            MO.mat_creat(ref temp66, 6, 6, 0.0);
            MO.mat_creat(ref t66, 6, 6, 0.0);
            MO.mat_creat(ref temp61, 6, 1, 0.0);
            MO.mat_creat(ref temp33, 3, 3, 0.0);
            MO.mat_creat(ref C, 3, 3, 0.0);
}
```

Il parametro di delay imposta un ritardo in millisecondi tra l'elaborazione dei dati del ricevitore GPS e i dati delle IMU. Dopo diverse misurazioni fatte in movimento è stato notato un disallineamento dei risultati del filtro in predizione (fase in cui vengono utilizzati i dati delle IMU) rispetto ai risultati in update (fase in cui vengono utilizzati i dati del GPS per la correzione). Questo poiché i valori ricavati dai sensori vengono ricevuti "immediatamente", mentre i valori del GPS sono ricevuti dai satelliti. Il valore di delay permette appunto di colmare questo disallineamento, fornendo uno shift temporale dopo il quale il filtro comincia a processare. Il valore di delay che è stato ritenuto più vicino all'effettivo gap temporale è 500ms. Per fare ciò vengono mantenute in memoria due liste di record MemsData, una per ogni piattaforma, nelle quali vengono memorizzati i campioni restituiti dai Wrapper . Appena superato il delay impostato, la fase di predizione del filtro viene eseguita non sugli ultimi record ricevuti, bensì sui più vecchi presenti nelle liste (che vengono quindi estratti).

4.4.1.2 Classe xSensWrapper

La classe xSensWrapper[18] ha il compito di fornire al programma un'interfaccia semplice per la gestione dell'Mti. Al suo interno richiama le funzioni di libreria fornite nelle API. I metodi più importanti sono Connect() e Disconnect() che rispettivamente istanziano e cancellano l'oggetto mt della classe MotionTracker (classe di libreria), GetBufferLength(), che fornisce il numero di record del buffer interno utilizzati, isBufferEmpty() che restituisce un valore booleano indicante se sono presenti o no dei record non ancora elaborati nel buffer interno, e il metodo ReadData(). Tale metodo viene richiamato nel MainLoop e restituisce un record di tipo MemsData se tale record è presente nel buffer interno, valore null altrimenti. In ogni record viene anche restituito il numero progressivo del campione come timestamp.

[18] Per Wrapper si intende un dispositivo che interfacci il software ad alto livello con l'hardware/firmware del dispositivo

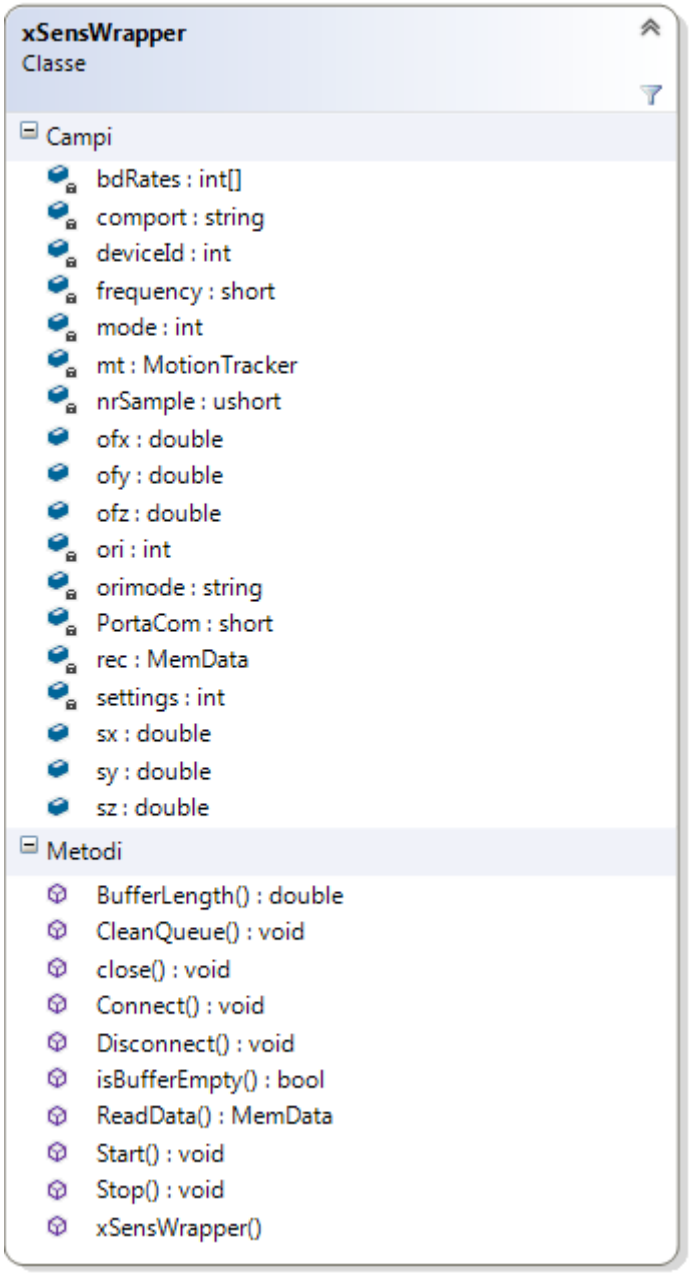

Figura 28 : XSensWrapper Class diagram

4.4.1.3 Classe iNemoWrapper

La classe iNemoWrapper viene implementata estendendo un wrapper di iNemo M1 fornito insieme alle API come esempio d'uso. Le funzioni più importanti sono Connect() e Disconnect() che rispettivamente stabiliscono e chiudono una connessione tra il dispositivo e il programma, start() e stop() che avviano e fermano l'acquisizione e la funzione readData() che per conformità con la readData() dell'altra unità inerziale restituisce un record di tipo MemsData se presente nel buffer. Ad ogni campionamento viene restituito nel record anche il numero progressivo del campione, che funge da timestamp .

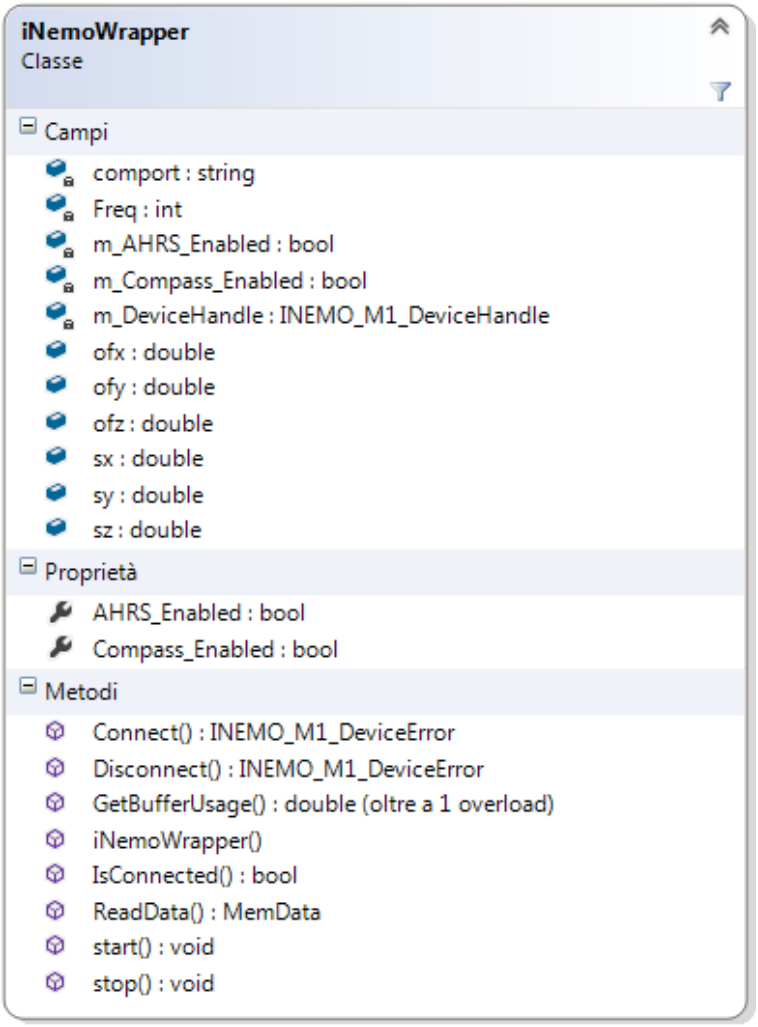

Figura 29 : iNemoWrapper Class diagram

4.4.1.4 Classe MemsData

La classe MemsData rappresenta la struttura dati del record restituito dai sensori. Esso ha i seguenti campi:

- vettore delle accelerazioni (in x,y e z del sensore)
- vettore delle velocità angolari (ricavati dai giroscopi, sempre secondo al body frame)
- vettore dei valori del magnetometro
- angoli di eulero (roll, pitch e yaw)
- il campo source, utilizzato per distinguere da quale piattaforma tale record è calcolato: 0 per xSens, 1 per iNemo.

Tale struttura dati prevede anche la restituzione dei quaternioni invece che gli angoli di eulero. In questo progetto però non sono stati utilizzati.

Essa inoltre contiene le funzioni ToString() e ToFile(). La prima restituisce il record in una stringa in formato *tsv* (tabbed separated values), in modo da essere facilmente integrabili in matlab. La seconda memorizza il record in formato tsv in un file specificato come parametro.

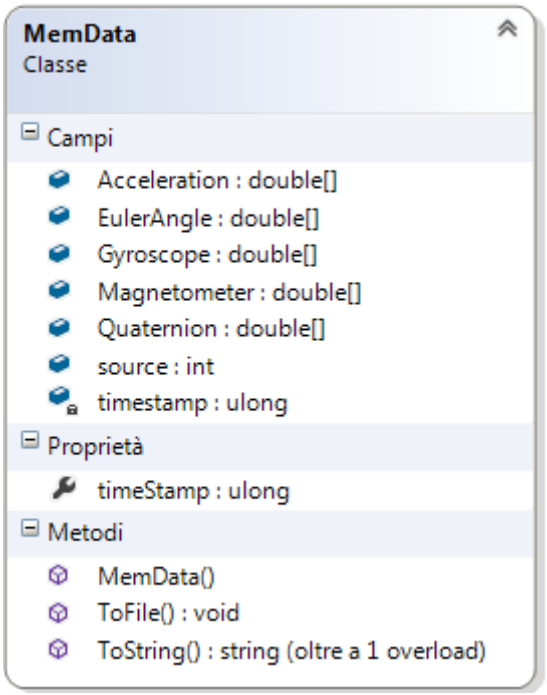

Figura 30 : MemsData Class diagram

4.4.1.5 Classe GPSWrapper

La classe GPSWrapper implementa la gestione del dispositivo di ricezione GPS. Mentre gli altri due dispositivi avevano a disposizione delle API per la comunicazione a basso livello con l'hardware ed un buffer interno per contenere i dati acquisiti, per il GPS è stato necessario implementare il protocollo di comunicazione. Essendo il ricevitore GPS connesso al PC mediante cavo USB-RS232 converter, è stata implementata la gestione della comunicazione seriale. Il ricevitore, quando connesso, trasmette ogni secondo sulla porta seriale una stringa comprendente le frasi NMEA del pacchetto. Ogni volta che viene trasmesso qualcosa sulla porta, viene generato un evento che viene gestito all'interno del wrapper, richiamando il metodo ReadLine() della porta seriale, che raccoglie una singola frase NMEA.

La stringa è divisa in "linee" ognuna delle quali rappresenta una frase NMEA contrassegnata all'inizio dal codice della frase: "$ccccc".

Da queste vengono estratte automaticamente le frasi di interesse, che sono:

- $GPRMC, da cui vengono ricavati data/ora del GPS, Velocità e Angolo Track
- $GPGGA , da cui vengono ricavati Latitudine, Longitudine, Altitudine, Numero di satelliti e Fix del segnale GPS.

Le frasi suddette vengono ricavate mediante un automa a stati finiti descritto dalla seguente figura.

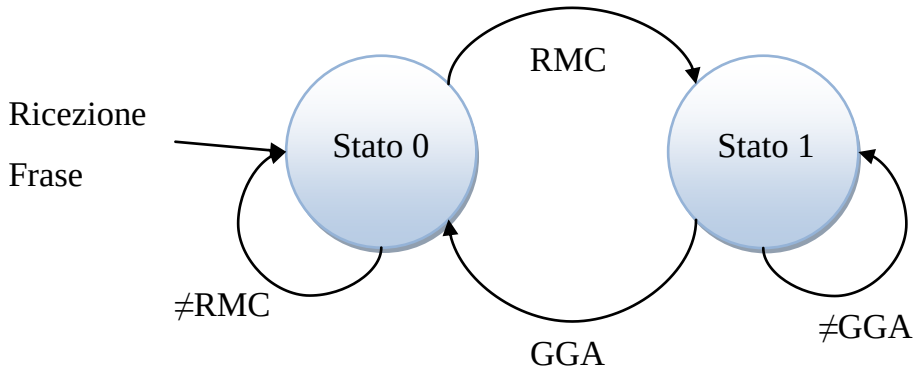

Figura 31 : Automa a stati utilizzato per ricavare le frasi NMEA di interesse

1. Nello stato 0 si aspetta la frase contrassegnata col codice $GPRMC. Tutte le altre frasi vengono scartate. Quando si riceve una frase RMC, oltre a passare allo stato 1 si elabora la frase. Ogni frase è suddivisa in campi da delle virgole. Utilizzando il comando Split della classe String, impostando la virgola come carattere di separazione, la stringa viene suddivisa in un array di stringhe in cui ogni stringa è un campo della frase. Da esse vengono prese ed elaborati campi di velocità, angolo di track e la data del gps. In particolare la velocità, espressa in nodi (miglia all'ora, MPH) viene convertita nel formato metrico internazionale, ovvero in m/s (metri al secondo).
2. Nello stato 1 ci si trova qualora si sia già ricevuta una frase RMC. Quindi si attende una frase GGA al fine di completare il record di informazioni che deve essere poi restituito. Se in tale stato si riceve una nuova frase RMC, quella precedentemente elaborata viene scartata e quella nuova viene elaborata come descritto sopra. Se invece viene ricevuta una frase GGA , essa viene elaborata con lo stesso metodo descritto prima, ma per prelevare i dati riguardo latitudine, longitudine e altitudine, numero di satelliti e fix del segnale. In particolare i valori di latitudine e longitudine vengono restituiti dal GPS in formato sessagesimale. Viene effettuata quindi una conversione di tali valori per portarli in formato decimale, più comodo per i calcoli che devono essere svolti.

I record così ricavati vengono inseriti in un buffer implementato nel wrapper mediante la classe Queue<GPSData>. I record sono quindi di tipo GPSData. Il programma principale controlla la presenza di dati nel buffer (mediante il metodo isBufferEmpty()) e se ve ne sono, preleva il primo record nella coda mediante la funzione ReadData() (che al suo interno effettua un'operazione di dequeue), al fine di effettuare la fase di correzione del filtro.

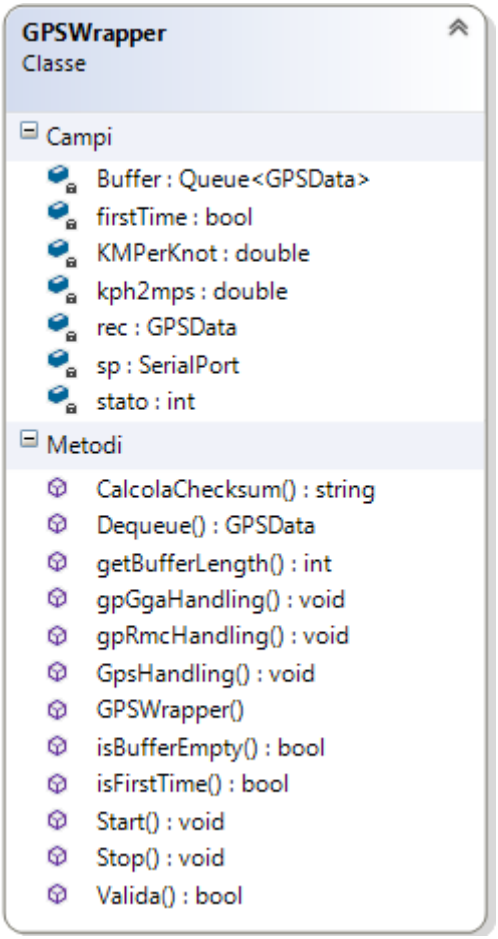

Figura 32 : GPSWrapper Class diagram

4.4.1.6 Classe GPSData

La classe GPSData rappresenta la struttura dati con cui vengono memorizzati i parametri del GPS. L'elenco dei parametri e dei metodi è descritto nella figura seguente.

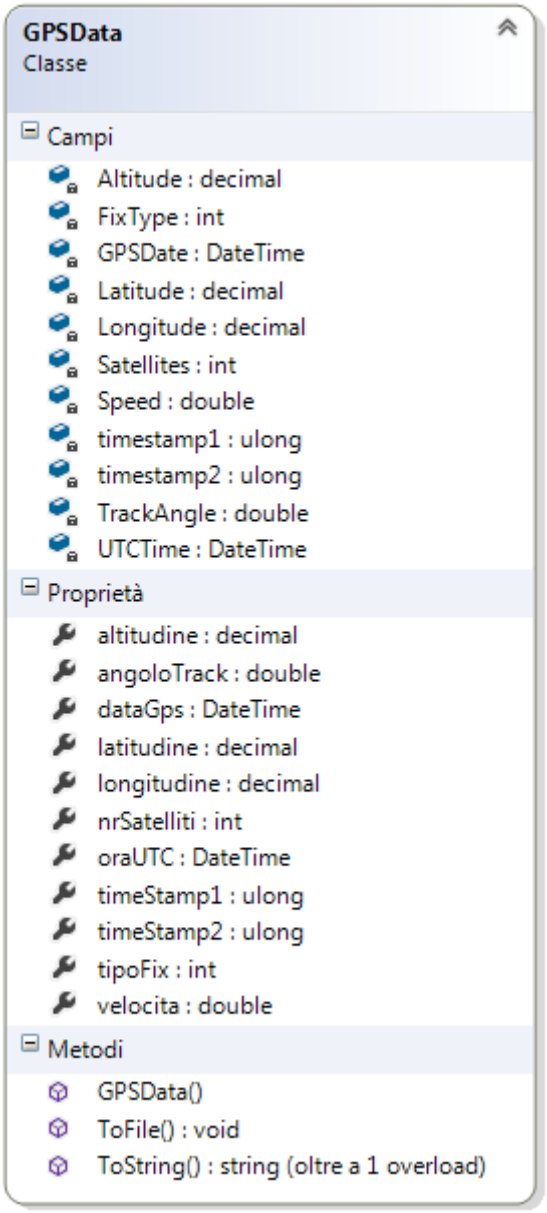

Figura 33 : GPSData Class diagram

Il metodo *ToString()* restituisce il record in formato stringa *tsv*. Il metodo *ToFile()* salva il record sul file specificato nell'argomento. I campi timestamp1 e timestamp2 memorizzano i valori di timestamp dell'ultimo campione elaborato della piattaforma xSens e della piattaforma iNemo rispettivamente.

4.4.1.7 Classe PosData

I dati elaborati dal filtro di Kalman esteso vengono memorizzati nella struttura dati PosData. I dati più importanti contenuti dalla classe sono:

- Latitudine
- Longitudine
- Altitudine
- Velocità Nord
- Velocità Est
- Velocità Down
- Angoli di Eulero ricavati dagli accelerometri
- Timestamp della piattaforma inerziale
- Flag binario *info* indicante se tale record è stato ottenuto da una fase di prediction o da una fase di update: 0 indica la predizione, 1 la correzione.

4.4.1.8 Altre classi

La classe Parameters, come anticipato, contiene i parametri di default del filtro. Possiamo dividere tali parametri in due categorie: parametri costanti e parametri variabili. I parametri costanti sono R_{ns} , che rappresenta la lunghezza del raggio polare, R_{ew}, che rappresenta la lunghezza del raggio equatoriale, D2R che è il fattore di conversione da gradi a radianti (Degrees To Radians), e MAG_DEC che rappresenta la declinazione magnetica. Tale parametro è impostato sulla declinazione magnetica rispetto alle coordinate di Catania (pari a circa 2.7°), ovvero dove è stato svolto il lavoro. Per ottenere una diversa declinazione magnetica è possibile consultare il sito http://www.ngdc.noaa.gov/geomag-web/ [14] dell'ente NOAA (National Oceanic and Atmospheric Administration).

Trai campi variabili più importanti vi sono Pcov, Vcov e Hcov. Pcov è la covarianza della posizione, che determina l'intervallo di correzione di latitudine e longitudine. Vcov è la covarianza della velocità ed Hcov è la covarianza dell'errore in altezza.

I loro valori di default sono:

$Pcov = (1.0e - 5 * D2R)^2$	Tolleranza di ±1 grado in latitudine e longitudine
$Vcov = 9$	Tolleranza di ± 3 m/s nelle velocità
$Hcov = 4$	Tolleranza di 2 m in altitudine

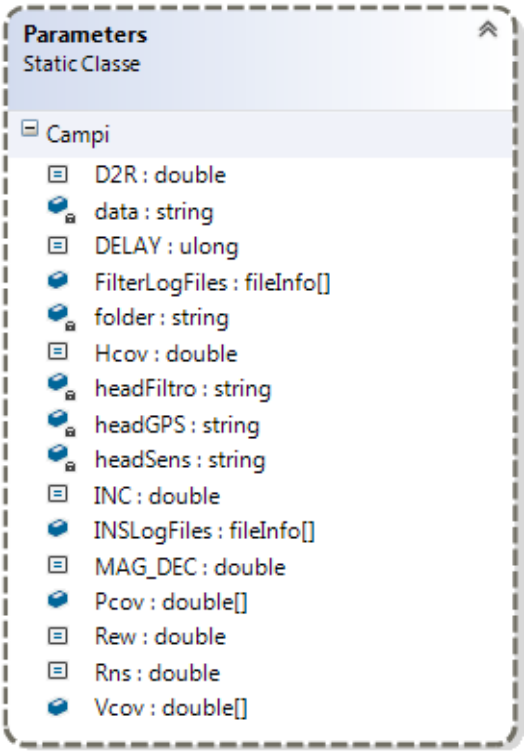

Figura 34 : Parameters Class diagram

Infine abbiamo la classe statica MO che è l'acronimo di *Matrix Operations.* In essa sono contenute le operazioni matriciali come prodotto vettoriale, prodotto scalare, somma, operatore di trasposizione, calcolo dell'inversa e così via.

4.4.2 Descrizione dell'interfaccia

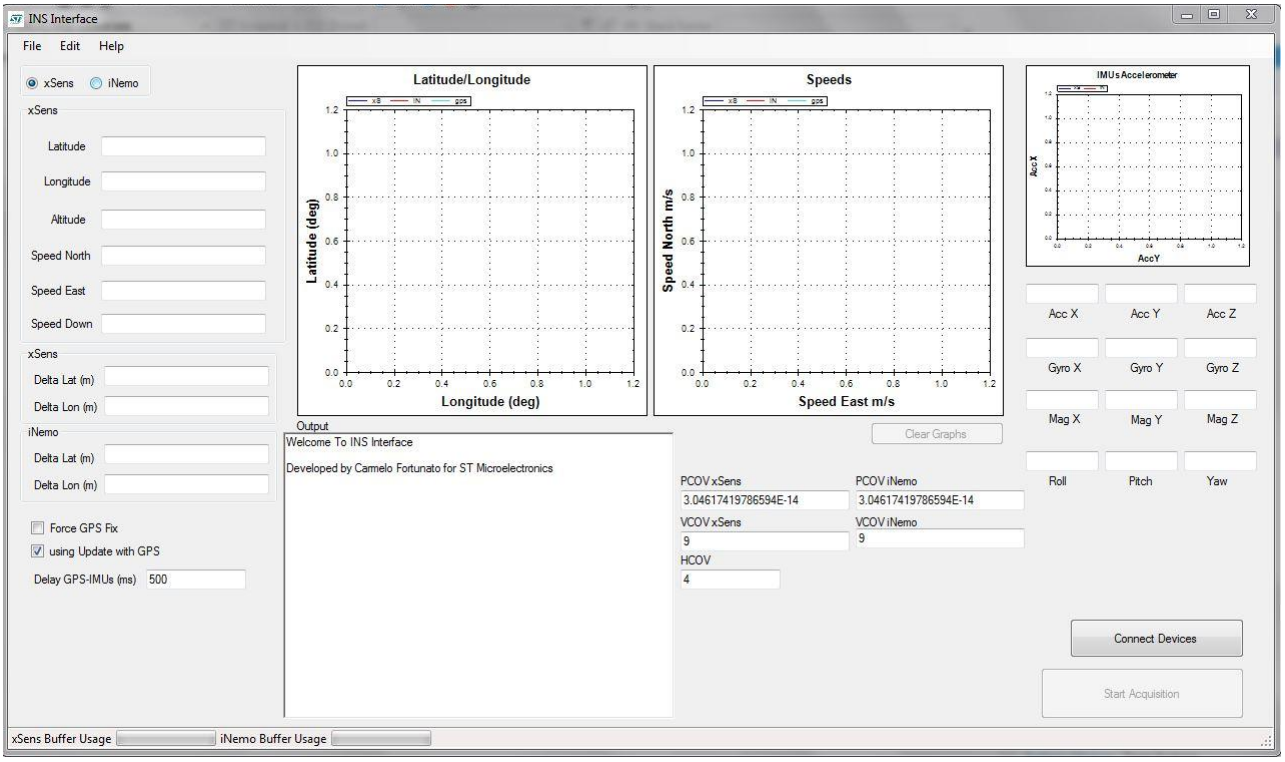

Figura 35 : Interfaccia grafica dell'applicazione

L'interfaccia implementa diversi controlli grafici per la visualizzazione dei dati ed il controllo su esecuzione e comportamento dell'appliczione.

Le TextBox a sinistra mostrano i valori di posizione LLA, Velocità (NED) e la differenza (delta) di posizione ricavata dal filtro rispetto alla posizione del GPS (espressa in metri), sia per xSens che per iNemo.

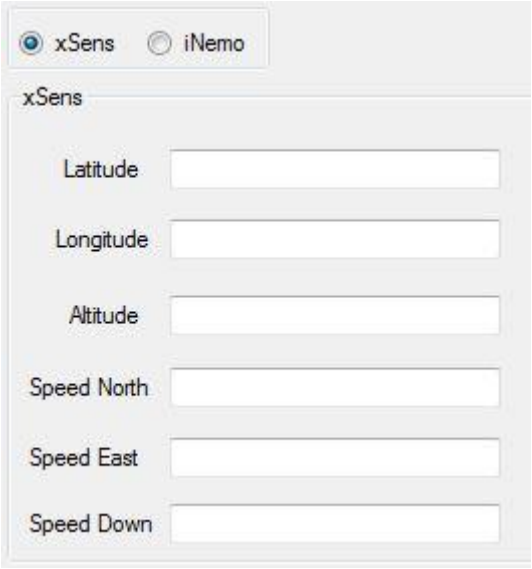

Figura 36 : Posizione e velocità calcolate dal filtro dai valori di una IMU. Dal radiobutton in alto è possibile scegliere quali valori visualizzare.

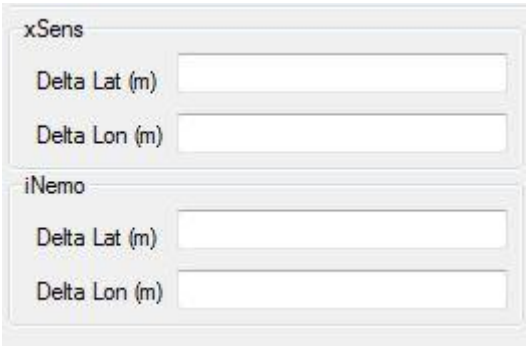

Figura 37 : Textbox per la visualizzazione degli errori in latitudine e in longitudine

Per convertire il delta di latitudine e longitudine in metri sono state utilizzate le formule, inverse a quelle usate nel filtro (inserite nella matrice F):

$$dLat = (R_{ns} + Alt) * \sin(Lat - gpsLat)$$

$$dLon = (R_{ew} + Alt) * \cos gpsLat * \sin(Lon - gpsLon)$$

Tali funzioni saranno poi usate per calcolare i risultati rispetto al tempo di esecuzione.

In basso a sinistra sono presenti due CheckBox e un'altra TextBox, raffigurate nella seguente immagine.

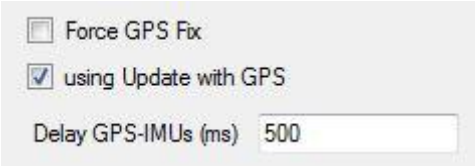

Figura 38 : Controlli per la gestione di GPS, Update e ritardo

Le Checkbox *"Force GPS Fix"* permette di fissare il valore di Fix GPS a 1, nel caso si vogliono fare delle prove coi sensori anche in assenza di segnale GPS. La Checkbox con etichetta *"using Update with GPS"* invece permette di abilitare/disabilitare la fase di correzione del filtro, utile per valutare il comportamento dell'algoritmo in Dead Reckoning con le due piattaforme inerziali. La TextBox invece permette di scegliere il ritardo (Delay)[19] sull'elaborazione dei valori dati dai sensori rispetto a quelli forniti dal GPS.

La RichTextBox in basso al centro con etichetta *"Output"* permette di visualizzare messaggi del sistema e, quando in esecuzione, visualizza i dati del GPS ed i tempi di esecuzione delle due piattaforme. Subito a destra di questa vi sono le caselle di testo per modificare i valori di Pcov, Vcov ed Hcov da utilizzare nell'algoritmo.

[19] Per comprendere il motivo della casella di ritardo si veda il capitolo 5.

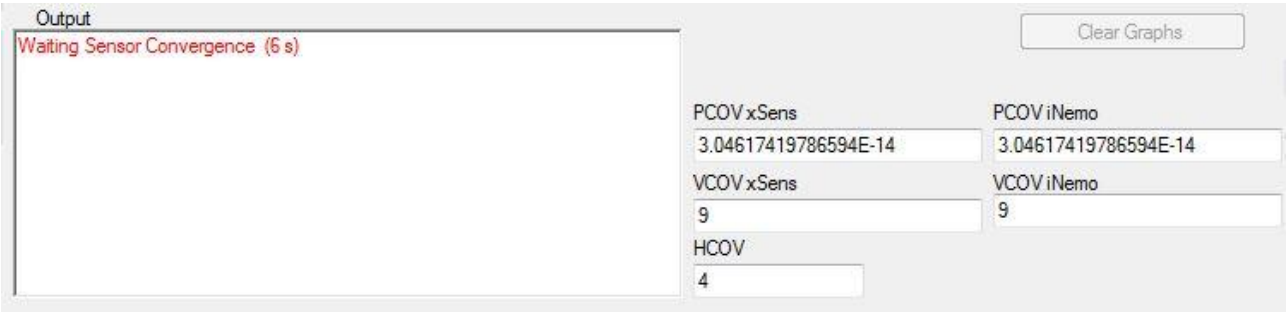

Figura 39 : Textbox di output e delle covarianze

Sopra di essi abbiamo i grafici per la visualizzazione di Latitudine e Longitudine, delle velocità Nord ed Est e dei parametri dei sensori. I primi due grafici visualizzeranno i percorsi e le velocità calcolate dal filtro per entrambe le piattaforme, ma anche percorsi e velocità ricavate dal GPS. La distinzione tra i differenti percorsi avviene tramite colorazione: La linea rossa rappresenta iNemo, la linea blu xSens, mentre la linea turchese i valori del GPS. Il terzo grafico mostrerà i valori delle accelerazioni Nord ed Est di entrambe le piattaforme inerziali. Tale strumento è stato integrato per scopi di test dei sensori, soprattutto in condizioni statiche, ma potrebbe essere utile anche in fase di acquisizione in movimento per confrontare i valori uscenti dal filtro con le accelerazioni percepite dai sensori.

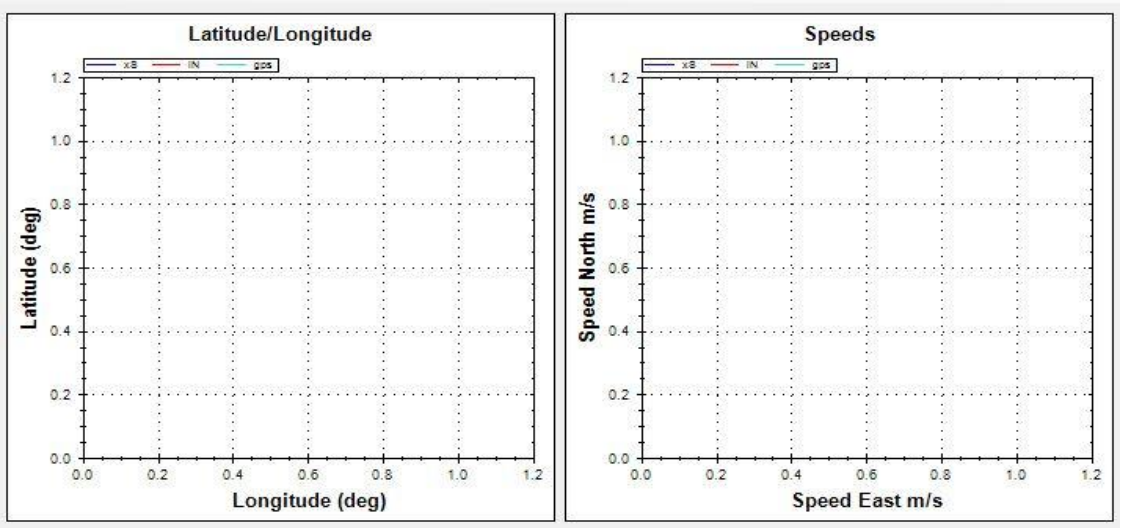

Figura 40 : Grafici per la visualizzazione di Posizioni L-L e Velocità N-E

Tali grafici sono realizzati mediante un controllo grafico su libreria dll di nome ZedGraph. Esso è risultato molto utile al fine di visualizzare i risultati e le differenze di percorso calcolato dal filtro per i tre dispositivi. Tramite delle semplici funzioni è possibile:

- Creare i grafici (funzione CreateGraph())
- Inserire valori (UpdateGraph())
- Ripulire la vista del controllo (ClearGraph())

Inoltre il controllo permette la gestione dello zoom dalla rotella del mouse o mediante selezione rettangolare (click-move-releaseclick) e la possibilità di salvare la vista della window su file, in diversi formati.

I radio Button in alto a sinistra permettono di scegliere di quale piattaforma (xSens o iNemo) si vogliono osservare i valori dei sensori e i valori calcolati. Le TextBox che visualizzano i valori di Latitudine, Longitudine, Altitudine, Velocità nei tre assi e i dati dei sensori, visualizzano i dati di xSens o di iNemo in base alla scelta fatta.

Infine, i bottoni Connect e Start servono appunto per connettere i dispositivi secondo le impostazioni scelte dal form SettingsForm e per avviare l'applicazione secondo la configurazione del filtro impostata dalle TextBox relative ai valori del filtro.

CAPITOLO 5. RISULTATI

Prima di effettuare prove in movimento sono stati fatti diversi test in condizioni statiche. Mediante tali test è stata riscontrata una certa imprecisione degli accelerometri dovuta a bias e fattori di scala. Per correggere tali problemi e ricavare i fattori con cui aggiustare i valori uscenti dai sensori è stato eseguito il cosiddetto *test statico delle sei facce*, proposto dallo Standard IEEE 517. Tale metodo si basa sul misurare l'accelerazione di gravità agente sugli accelerometri della piattaforma inerziale in condizioni statiche (ovvero non in movimento, da fermo), disposta in sei diverse posizioni. Indicando con (roll,pitch,yaw) la posizione della piattaforma inerziale, le sei posizioni sono:

- (0,0,0) Accelerazione di gravità negativa sull'asse *z*
- (180,0,0) Accelerazione di gravità positiva sull'asse *z*
- (90,0,0) Accelerazione di gravità negativa sull'asse *y*
- (-90,0,0) Accelerazione di gravità positiva sull'asse *y*
- (0,90,0) Accelerazione di gravità positiva sull'asse *x*
- (0,-90,0) Accelerazione di gravità negativa sull'asse *x*

Per ogni posizione sono stati acquisiti 10000 campioni ed è stata fatta la media per ottenere un valore più accurato. Una volta ottenuti tali valori sono stati ricavati i fattori di scala e di offset secondo le seguenti formule:

$$Offset = \frac{\ddot{z}_{up} + \ddot{z}_{down}}{2}$$

$$SF = \frac{(\ddot{z}_{up} - \ddot{z}_{down} - 2g)}{2g}$$

Dove con $\ddot{z}_{up}$ e $\ddot{z}_{down}$ si intendono i valori di accelerazione ricavati nello stesso asse, una positiva e l'altra negativa, mentre con g si intende l'accelerazione di gravità ($\sim 9.81 m/s^2$). Tali fattori sono poi stati applicati per tutto il resto delle prove ai valori restituiti dagli accelerometri di entrambe le piattaforme inerziali. La procedura di correzione viene eseguita direttamente dai wrapper prima di restituire i valori.

5.1 Prove effettuate e risultati

Sono stati effettuate diverse prove su vari percorsi al fine di valutare la soluzione adottata. Le IMU sono stati fissati su un supporto in legno (uno accanto all'altro al fine di far percepire le stesse forze) disposto ben saldo nella base dell'automobile. L'antenna GPS, dotata di magnete, è stata posta sul tetto dell'autovettura. Il percorso è stato effettuato per metà integrando valori di sensori e gps (fase di Update attiva), mentre per metà in Dead Reckoning (disattivando la fase di update). Alla fine del percorso (quando si torna alla posizione di partenza) viene riattivata la fase di Update per correggere la posizione calcolata nella fase precedente.

Figura 41 : Mappa Satellitare del percorso effettuato

La figura qui sopra mostra il percorso effettuato per le ultime misure. Tale percorso ha una lunghezza di circa 540 metri. Le cifre in blu rappresentano le lunghezze dei singoli segmenti. Mentre le coordinate in rosso rappresentano latitudine e longitudine dei punti di incontro dei segmenti. Le distanze vengono calcolate tramite esse e conoscendo l'altitudine (pressoché uguale per tutto il percorso essendo quest'ultimo abbastanza pianeggiante) mediante le formule descritte al paragrafo 4.4.2. Il percorso è stato effettuato più volte, ed i tempi di percorrenza si sono aggirati intorno a 90-97 secondi.

L'asterisco rosso indica la posizione di partenza, mentre l'asterisco verde l'inizio del percorso in DR.

La frequenza con cui si effettua la correzione con i dati del gps dipende dalla frequenza di campionamento del GPS. Per tale motivo tale frequenza è di 1 Hz (una fase di correzione al secondo). Negli istanti di tempo che intercorrono tra due fasi di correzione viene comunque effettuato il Dead Reckoning (i dati dei sensori sono ricevuti ogni 20 millisecondi, quindi in un secondo saranno effettuate 50 fasi di predizione per piattaforma inerziale). Sono stati scelti percorsi rettangolari al fine di poter valutare la forma della traiettoria calcolata dall'algoritmo, nonché il comportamento dello stesso in rettilinei e curve, e pianeggianti, in modo da avere un riscontro sugli angoli percepiti. La velocità di regime avuta nel percorso in tutte le prove effettuate è di circa 35Km/h, con una media di 20Km/h (5.56m/s) .

In tali condizioni è stato notato che l'algoritmo era poco sensibile alle curve. Dopo aver indagato sui file di log in post processing (mediante matlab) è stato notato un comportamento inatteso dell'algoritmo AHRS interno delle piattaforme: durante una curva tale algoritmo "compensa" le accelerazioni percepite con un aumento degli angoli di roll e pitch (con picchi superiori a 20°).

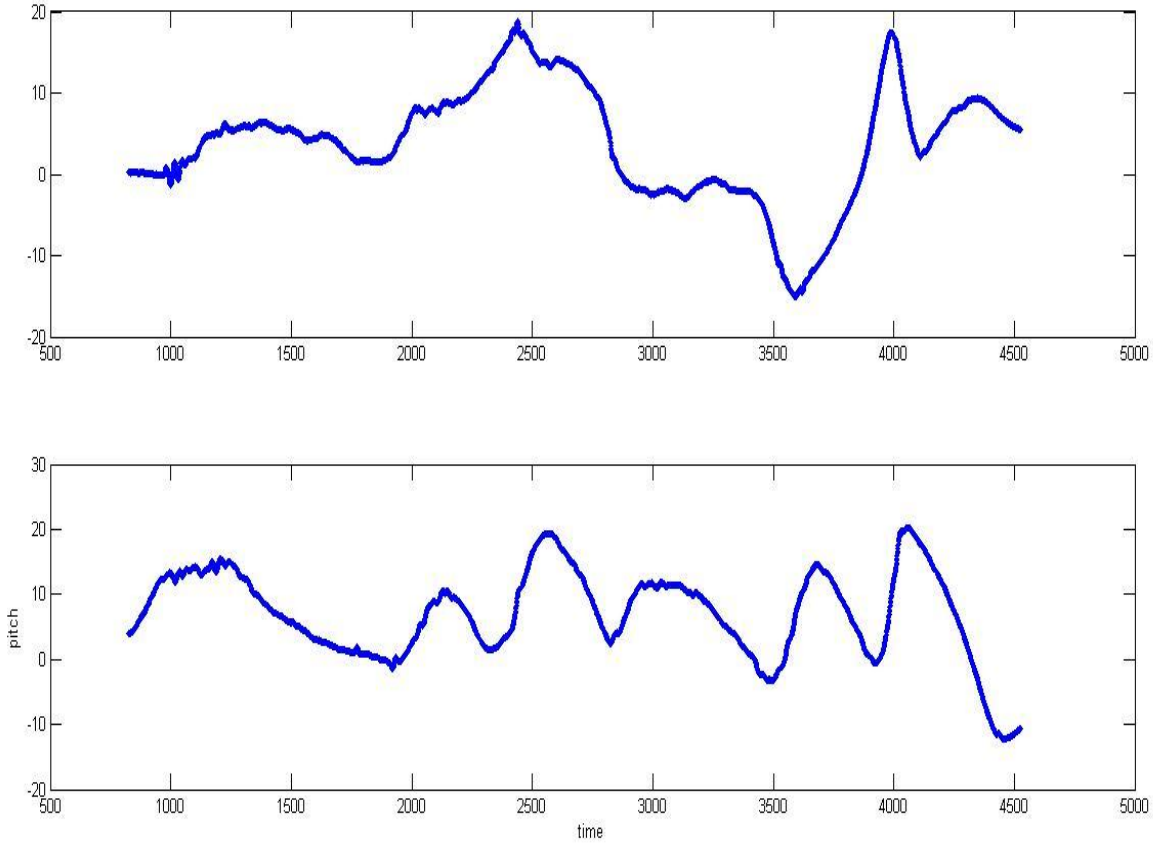

Figura 42 : Grafici di variazione di roll e pitch

Come è possibile notare dai grafici, in presenza delle curve i valori di roll e pitch cambiano valore, spostandosi da vicino allo zero a valori che vanno da -20 a +20. Ovviamente 20 gradi di rollio sono difficili da raggiungere con un'automobile, così come 20 gradi di beccheggio in un percorso pianeggiante.

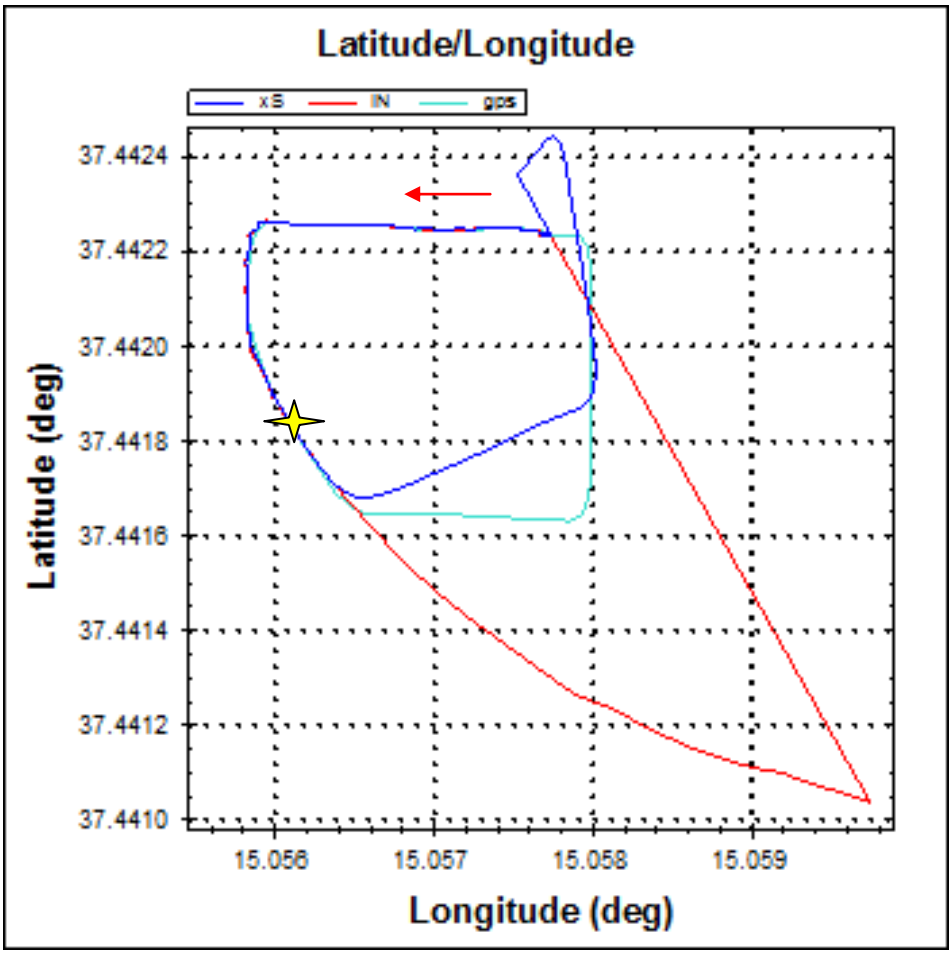

Figura 43 : Differenza tra xSens con angoli di roll e pitch corretti e iNemo senza correzione

Per sopperire a tale comportamento è stato modificato l'algoritmo in modo che, subito dopo il tempo di convergenza dei sensori, per 10 secondi campioni i valori di pitch e roll delle due piattaforme e di questi ne faccia la media. Tali valori medi saranno poi fissati nei successivi campioni acquisiti in movimento, sostituendo quelli forniti dalle piattaforme. Questa soluzione, essendo effettivamente un'approssimazione rudimentale, è applicabile quando si suppone di muoversi su un percorso pianeggiante. In presenza di salite o discese ripide si potrebbe considerare di correggere solo l'angolo di rollìo. Mediante questa modifica si sono riscontrati notevoli miglioramenti nelle curve, come mostra la figura sopra.

Il comportamento nel rettilineo è piuttosto accettabile: l'errore di deriva laterale con iNemo M1 si aggira, in dead reckoning, sui 3 metri dopo 10 secondi di esecuzione (tale errore cresce poi più rapidamente in quanto esso viene integrato e quindi accresciuto nei calcoli del filtro di kalman, con errori di circa 15 metri dopo 20 secondi), mentre con xSens l'errore è minore, pressoché nullo: infatti, il percorso ricavato nel rettilineo è abbastanza lineare, indice che l'errore laterale dipende per lo più dall'angolo di curvatura che rispetto a quello calcolato dal GPS risulta più accentuato. Differenze di pochi gradi possono causare errori di diversi metri.

Per quanto riguarda l'errore lungo la traiettoria di percorrenza, anch'esso tende a crescere dopo diversi secondi di esecuzione: intorno ai 25-30 secondi si ottiene un errore di circa 26 metri, con entrambe le piattaforme.

Nelle curve, come anticipato, l'algoritmo tende a non avere un angolo di curvatura perfetto rispetto a quello rilevato dal ricevitore GPS.

Tali risultati, per entrambe le piattaforme, sono comunque abbastanza naturali per un applicazione di questo tipo.

I risultati sopra considerati possono essere notati nella seguente figura, dove la freccia rossa indica il verso di percorrenza, mentre la stella indica l'inizio del percorso con la sola fase di predizione, ovvero il percorso in DR.

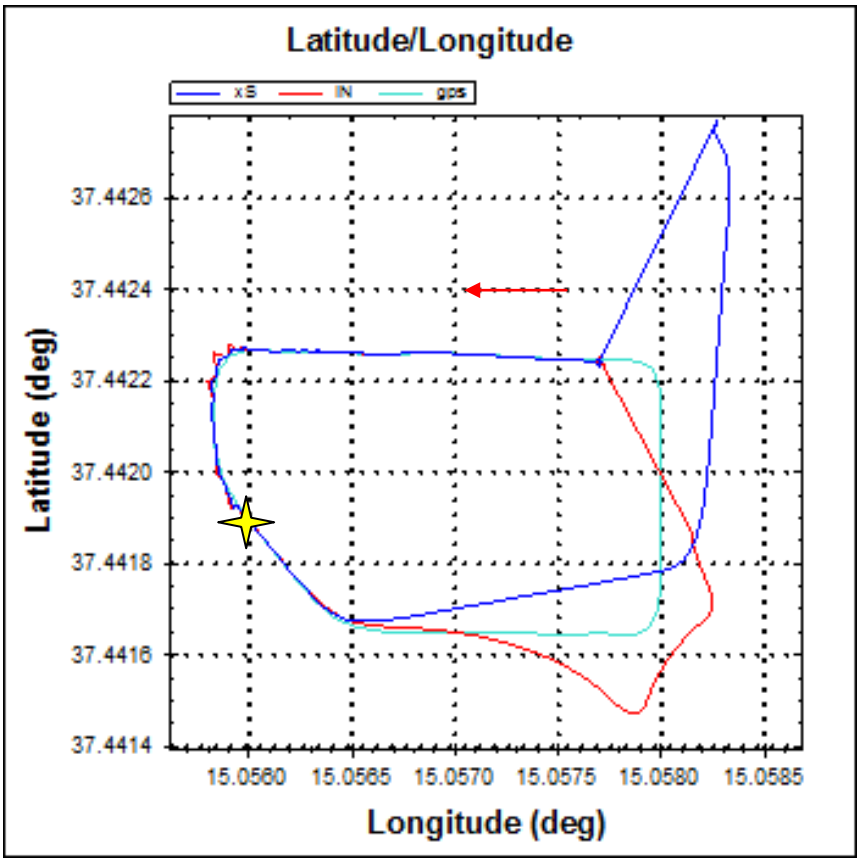

Figura 44 : Percorso con GPS - DR

In sostanza, dalle prove effettuate su tale percorso e dai calcoli effettuati sui dati di log in post processing si possono considerare i seguenti risultati:

- L'algoritmo svolge bene il suo compito nella parte in cui la fase di update è attiva, fase in cui si aggiornano i valori ricavati dai segnali inerziali con quello del GPS. Di fatto le linee di percorso calcolate in base alle uscite dei tre dispositivi sono pressoché sovrapposte.
- Negli intervalli di tempo tra una fase di update e l'altra il Dead Reckoning viene svolto in maniera abbastanza lineare, mostrando una certa stabilità dell'algoritmo in brevi intervalli di tempo.
- Alla disattivazione della fase di Update si può notare una deriva dell'errore da parte di entrambe le piattaforme a medio termine.

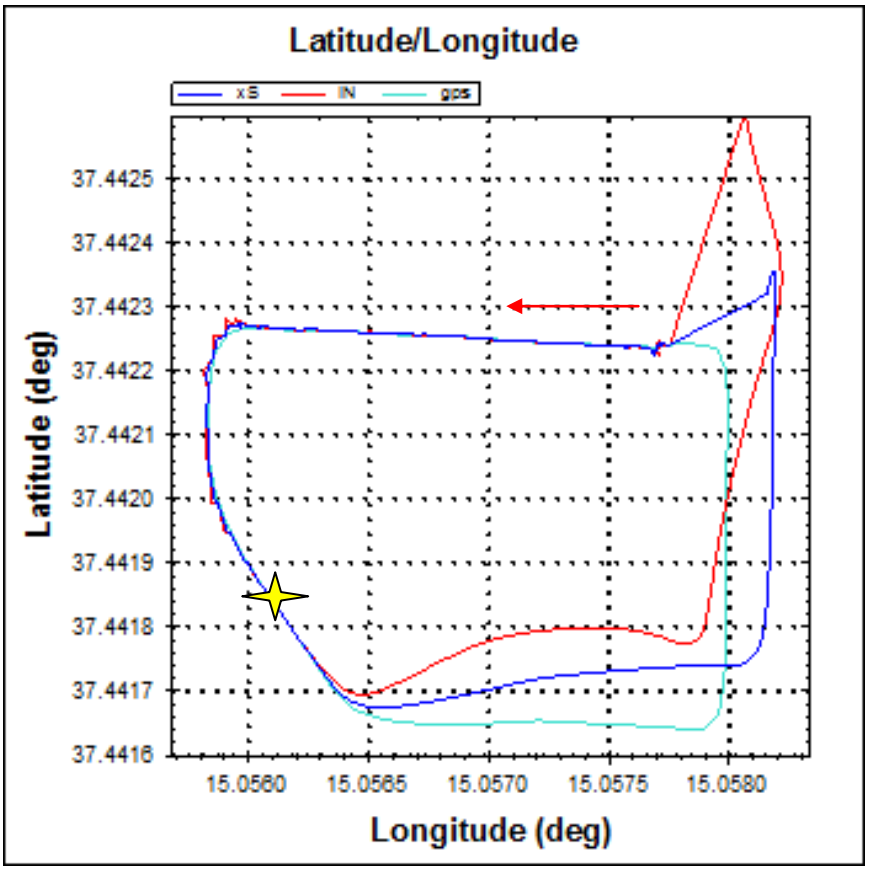

Figura 45 : Altro Percorso con GPS - DR

- Nelle curve il problema della deriva è ancora più sentito, ma è stato compensato dall'algoritmo.
- L'errore in metri dell'algoritmo è comunque piccolo per un sistema di Dead Reckoning, e quindi accettabile. I calcoli effettuati dall'algoritmo in assenza di segnale GPS permettono di avere una buona stima della posizione esatta in modo che al ritorno del segnale il GPS ricalcoli la posizione più rapidamente.
- Sebbene i risultati con la piattaforma iNemo M1 siano qualitativamente inferiori a quelli osservabili con xSens Mti, il risultato rimane comunque accettabile, dimostrando la maturità dei dispositivi inerziali ST.

Altre prove su un percorso differente sono raffigurate nelle immagini seguenti:

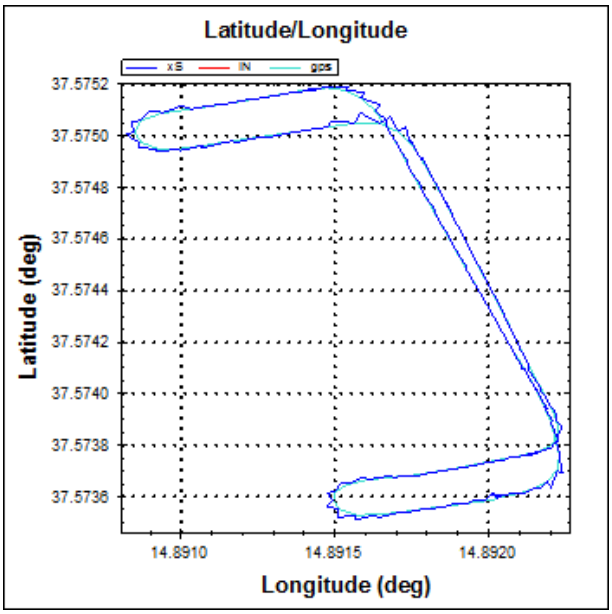

Figura 46 : Percorso realizzato con xSens Mti

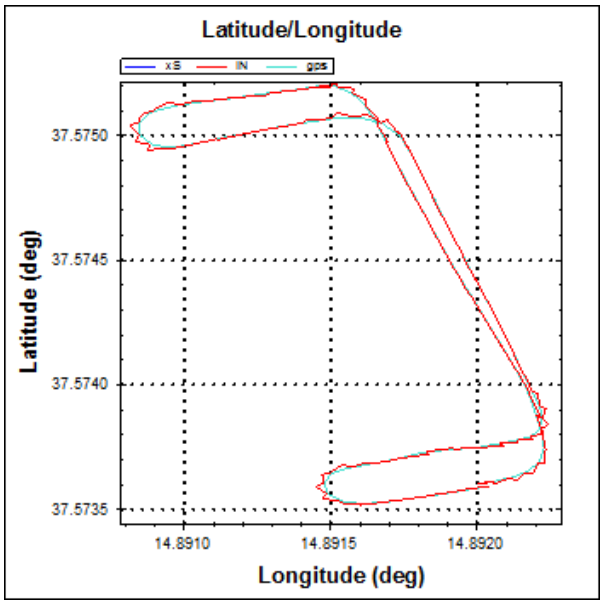

Figura 47 : Percorso con iNemo M1

Figura 48 : Percorso in mappa

CAPITOLO 6. CONCLUSIONI

Nel presente lavoro di tesi è stato sviluppato uno strumento software per rilevare la posizione di una piattaforma in movimento mediante tecniche di dead reckoning. Il software è basato su un filtro di Kalman esteso (EKF) , ovvero un algoritmo di fusione sensoriale in grado di predire e correggere le misure. Tale algoritmo è stato testato sul campo utilizzando contemporaneamente le due piattaforme inerziali xSens Mti e ST iNemo M1, al fine di valutare la sua prestazione sia con dispositivi commerciali che con board di valutazione. I risultati mostrano come questa sia una soluzione attuabile in caso di assenza di segnale GPS per brevi periodi. Ciò può suggerirne l'applicazione in contesti come la robotica e l'automotive al fine di approssimare la posizione della piattaforma mobile in contesti di segnale GPS scarso o assente. Dalle prove qui effettuate si può inoltre evincere come le piattaforme inerziali ST siano a livelli di qualità tali da poter essere utilizzate in applicazioni reali come ad esempio la navigazione inerziale.

BIBLIOGRAFIA

[1] : Kraus,Karl. Fotogrammetria – Vol.1 – Teoria ed applicazioni. Torino: Levrotto e Bella, 2004.

[2] : Shin,Eun-Hwan. “Accuracy Improvement of low-cost INS/GPS for land applications”. Tesi di dottorato, Department of Geomatics Engineering, University of Calgary, 2001.

[3] : http://itrf.ensg.ign.fr/ITRF_solutions/2005/ITRF2005.php

[4] : Sansò, Fernando. “Navigazione Geodetica e rilevamento cinematico”. Milano : Polipress, 2006.

[5] : Titterton, D.H., and J.L. Weston. “Strapdown inertial navigation technology”. Seconda edizione. American Institute of Aeronautics and Astronautics IEE, 2004.

[6] : Jekeli, Christopher. “Inertial Navigation Systems with Geodetic Applications”. Berlino – New York: de Gruyter, 2001

[7] : Cazzaniga,Noemi, “Sviluppo e implementazione di algoritmi per la navigazione inerziale assistita”, Tesi di dottorato, DIIAR – Sezione Rilevamento, Politecnico di Milano, 2007

[8] : Grewal, Andrews, “Kalman Filtering , Theory And Practice Using Matlab”, Wiley, 2001

[9] : STMicroelectronics, “GPS Evaluation Kit – User Manual CRM003, Revision 1.0”, 2008

[10] : El-Sheimy, Naser, “Inertial Techniques and INS/DGPS Integration”, Calgary, 2003

[11] : Bishop, R.H.. “The Mechatronics Handbook”. CRC Press, 2002

[12] : http://www.egnos-portal.eu

[13] : STMicroelectronics, “iNemo M1 – iNEMO System-on-board, Data Brief , Revision 2.”, 2012

[14] : http://www.ngdc.noaa.gov/geomag-web/

[15] : D. M. Young e R. T. Gregory. “A survey of numerical mathematics” Dover, New York, 1988

[16] : Fox , L. “The numerical solution of two-point boundary problems in ordinary differential equations”. Oxford University Press, 1957.

[17] : De Agostino, Mattia. “I Sensori Inerziali di basso costo per la Navigazione Geodetica”. Tesi di Dottorato. Politecnico di Torino, 2009.

[18] : Biagi, Ludovico. “ I fondamentali del GPS”, Geomatics Workbooks, Vol. 8, 2009

Indice delle Figure