



UNIVERSITÀ
DEGLI STUDI
FIRENZE

Scuola di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea Magistrale in Informatica
Resilient and Secure Cyber-Physical System

UTILIZZO DI UN ALGORITMO SENSOR
FUSION NELL'AMBITO DELLA
LOCALIZZAZIONE FERROTRAMVIARIA

USE OF A SENSOR FUSION ALGORITHM IN
THE AREA OF TRAMWAY LOCALIZATION

ALEX FOGLIA

ANDREA BONDAVALLI

Anno Accademico 2018-2019

INDICE

| | | |
|-------|--|----|
| 1 | Stato dell'Arte | 7 |
| 1.1 | Sistemi Ferroviari e Ferrotramviari | 7 |
| 1.2 | Il Problema del Posizionamento | 8 |
| 1.2.1 | Possibili Sviluppi | 12 |
| 2 | Sensor Fusion | 15 |
| 2.1 | Panoramica | 15 |
| 2.1.1 | Sistemi Dinamici | 15 |
| 2.1.2 | Misure e Rumore | 16 |
| 2.2 | I Filtri di Kalman | 17 |
| 2.2.1 | Premesse statistiche | 18 |
| 2.2.2 | Classificazioni | 19 |
| 2.2.3 | Filtri di Kalman Estesi | 19 |
| 3 | Applicazione di SFA: La Tramvia di Firenze | 21 |
| 3.1 | Architettura di Sistema | 22 |
| 3.1.1 | Architettura Hardware | 22 |
| 3.1.2 | Architettura Software | 23 |
| 3.2 | Gestione della trasmissione dei dati | 26 |
| 3.2.1 | Trasmissione in entrata | 26 |
| 3.2.2 | Trasmissione in uscita | 28 |
| 3.3 | Scenario di Esempio | 30 |
| 3.3.1 | Nota sui numeri di sequenza | 33 |
| 3.4 | Possibili sviluppi | 34 |
| 3.4.1 | Problematiche legate alla security | 34 |
| 3.4.2 | Miglioramenti al servizio fornito | 37 |

ELENCO DELLE TABELLE

| | | |
|-----------|--|----|
| Tabella 1 | Segnalazioni semaforiche ferrotramviarie | 12 |
| Tabella 2 | Protocollo di comunicazione in entrata | 26 |
| Tabella 3 | Significato del campo <code>SENSOR_TYPE</code> | 27 |
| Tabella 4 | Protocollo di comunicazione in uscita | 29 |
| Tabella 5 | Formato del pacchetto di <i>ack</i> | 29 |
| Tabella 6 | Condizioni iniziali | 30 |

ELENCO DELLE FIGURE

| | | |
|-----------|---|----|
| Figura 1 | Treno in arrivo alla stazione ferroviaria di Firenze Santa Maria Novella | 8 |
| Figura 2 | Tramvia di Danhai, Taipan | 8 |
| Figura 3 | Schema di un tipico scenario tramviario | 9 |
| Figura 4 | UCS realizzato da Thales Italia SPA | 10 |
| Figura 5 | Conta Assi | 11 |
| Figura 6 | Esempio di <i>Point Machine</i> installata su una traccia ferrotramviaria | 11 |
| Figura 7 | Schema SFA | 13 |
| Figura 8 | Grafico dell' errore di stima della posizione con $\alpha = 10^0, \varepsilon = 10^{-3}$ | 17 |
| Figura 9 | Tramvia di Firenze - Linea T1 | 21 |
| Figura 10 | Architettura hardware bordo treno | 24 |
| Figura 11 | Architettura software bordo treno | 25 |

STATO DELL'ARTE

1.1 SISTEMI FERROVIARI E FERROTRAMVIARI

Il concetto di *treno* come comunemente percepito nasce con l'inizio della Rivoluzione Industriale, avvenuta tra il *XVIII* e il *XIX* secolo, a seguito della quale l'avvento della macchina a vapore ha permesso all'umanità di disporre di fonti di energia sufficienti a fare evolvere i primi rudimentali trasporti su binario negli odierni sistemi ferroviari.

È possibile schematizzare un Sistema Ferroviario, o Ferrotramviario, come un veicolo, il treno, vincolato a muoversi attraverso una propulsione, elettrica o a combustibile, lungo una traccia fissa, il binario.

Queste caratteristiche accomunano qualsiasi sistema di trasporto ferroviario o ferrotramviario a prescindere dalla sua scala in termini di veicoli transitanti ed estensione geografica. Ciò che invece differenzia un Sistema Ferroviario da un Sistema Ferrotramviario sono:

- Le caratteristiche fisiche del treno, come lunghezza e massa;
- Le caratteristiche geografiche dell'ambiente operativo;
- Gli scopi del trasporto.

In generale, nel trasporto ferroviario si utilizzano treni caratterizzati da grandi dimensioni, che trasportano persone o merci su lunghe percorrenze (regionali, nazionali o internazionali), operando pertanto prevalentemente in ambienti extra urbani. Un esempio di treno operante in un sistema ferroviario classico è quello in figura 1.

Il trasporto ferrotramviario, di contro, vede l'utilizzo di treni dalle ridotte dimensioni, più leggeri di quelli usati nei sistemi ferroviari, e che hanno lo scopo di rappresentare un'alternativa per il cittadino all'utilizzo di mezzi privati durante i suoi spostamenti all'interno di un'area metropolitana. Quest'ultima caratteristica implica che l'ambiente operativo di un



Figura 1: Treno in arrivo alla stazione ferroviaria di Firenze Santa Maria Novella

sistema ferrotramviario sia radicalmente diverso da quello di un sistema ferroviario: i treni si muovono lungo rotaie installate su strade urbane, quindi il traffico ferrotramviario è fuso con il traffico automobilistico, motociclistico, ciclistico e pedonale che caratterizza l'ambiente urbano, come mostrato nelle figure 2 e 3.



Figura 2: Tramvia di Danhai, Taipan

1.2 IL PROBLEMA DEL POSIZIONAMENTO

Per posizionamento ferroviario, si intende la stima della posizione di un treno all'interno di una traccia ferroviaria. Esso esiste tanto nel contesto ferrotramviario quanto nel contesto ferroviario classico.

Sovente questa stima viene espressa come progressiva chilometrica rispetto all'origine della linea oppure, più raramente, come coordinata



Figura 3: Schema di un tipico scenario tramviario

geografica.

Il problema del posizionamento sorge nel momento in cui, per ragioni di rotta, un treno ha necessità di spostarsi da una sezione di binario, anche detta traccia, ad un'altra. Questa operazione di scambio è offerta dal sistema di *interlocking*. Tale sistema è detto *safety-critical*, in quanto offre una funzionalità che deve rispettare adeguati standard di sicurezza. Gli odierni sistemi di posizionamento si basano principalmente sull'utilizzo di strumenti installati a terra, che hanno lo scopo di rilevare il passaggio di un treno, e quindi di interagire con il sistema di *interlocking* della traccia al fine di garantire, con un elevato livello di confidenza, un transito sicuro dei mezzi.

Odiere Tecniche di Posizionamento

I sistemi di posizionamento attualmente in uso sono basati su un'architettura distribuita composta dai seguenti blocchi:

- Sottosistema di *interlocking*;
- Sottosistema di comunicazione treno-traccia;
- Sottosistema semaforico.

SOTTOSISTEMA DI INTERLOCKING: Il sottosistema di *interlocking* è la parte che si fa effettivamente carico di offrire al treno un attraversamento sicuro di una *Junction Area (JA)*. Una JA è un punto della linea ferroviaria in cui il treno può cambiare direzione, e occupare una nuova traccia di

binario.

La nuova traccia da occupare potrebbe avere particolari vincoli sul numero di treni contemporaneamente transitanti, ed in ogni caso lo scambio di rotaia deve essere corretto ed avvenire in sicurezza, in quanto occupare la traccia sbagliata potrebbe avere ripercussioni finanche catastrofiche.

Un sistema di *interlocking* è composto dai seguenti elementi:

- *Switch Control Unit (UCS):*

Piattaforma certificata SIL-3 che rappresenta il nucleo del sistema di *interlocking* e che implementa l'intera logica di gestione di una JA. Un UCS dispone di un'interfaccia di *Input/Output (I/O)* verso gli elementi di *interlocking* installati a terra che ne consente un controllo sicuro in accordo allo standard SIL-3.



Figura 4: UCS realizzato da Thales Italia SPA

- *Conta Assi:*

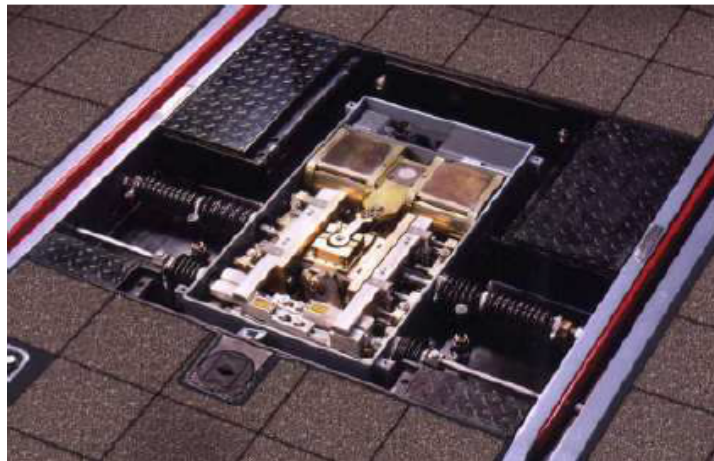
Il Conta Assi, o in inglese *Axle Counter (AC)*, è un sistema certificato SIL-3 che ha lo scopo di rilevare la presenza del treno e fornire quindi lo stato di occupazione della sezione di traccia in cui l'AC è installato.

- *Point Machines:*

Le *Point Machines* infine, sono degli strumenti certificati SIL-3 che hanno lo scopo di direzionare le rotaie verso una determinata sezione di traccia.



Figura 5: Conta Assi

Figura 6: Esempio di *Point Machine* installata su una traccia ferrotramviaria

L'intero sistema di *interlocking* viene attivato dai *Track Circuit*. Questi apparati sono installati a terra prima di ciascuna JA, e segnalano al sistema di *interlocking* l'avvicinamento di un treno alla successiva JA.

SOTTOSISTEMA DI COMUNICAZIONE TRENO-TRACCIA: Il sottosistema di comunicazione treno-traccia è gestito da un computer installato bordo treno, chiamato *On Board Control Unit* (OBCU), ed ha lo scopo di fornire funzionalità non legate alla *safety* e pertanto poco interessanti. OBCU viene principalmente utilizzato per monitorare lo stato del traffico ferrotramviario in una architettura di *monitoring* centralizzata. Il monitoring si basa su comunicazioni *wireless*. In alcune applicazioni può comprendere una comunicazione più o meno diretta con il sistema di *interlocking* allo scopo di segnalare l'avvicinamento del treno a una JA.

SOTTOSISTEMA SEMAFORICO: Il sottosistema semaforico prende in ingresso informazioni dal sistema di *interlocking* ed eventualmente, da OBCU, e gestisce i segnali luminosi da mostrare sui semafori a un mac-





| Segnale | Descrizione | Significato |
|--|-----------------------------------|---------------------------|
|  | Barra bianca orizzontale | Fermarsi |
|  | Barra bianca verticale | Procedere avanti |
|  | Barra bianca ruotata di 45 gradi | Procedere solo a destra |
|  | Barra bianca ruotata di -45 gradi | Procedere solo a sinistra |

Tabella 1: Segnalazioni semaforiche ferrotramviarie

chinista che si appresta a superare una JA.

In tabella 1 viene riportata la lista dei segnali semaforici utilizzati nel contesto ferrotramviario.

1.2.1 Possibili Sviluppi

Le attuali tecniche di posizionamento richiedono un intervento trascurabile di computer installati a bordo e una grande quantità di apparati installati a terra. Mentre i computer di bordo non forniscono in generale funzionalità legate alla *safety*, gli apparati installati a terra sono costosi e hanno un impatto ambientale non trascurabile.

È possibile considerare il treno e il computer di bordo come un unico sistema, ossia il treno viene modellato come un *Cyber-Physical System*.

Un *Cyber-Physical System* (CPS) è un sistema composto da una parte *fisica* e da una parte *cyber*. Il sottosistema fisico è composto da sensori e attuatori che hanno rispettivamente lo scopo di rilevare lo stato dell'ambiente circostante e di alterarlo se necessario. Il sottosistema *cyber* è essenzialmente un elaboratore, che dispone di processore, memoria, e interfacce I/O verso i sensori gli attuatori, ed eventuali operatori umani. Una tale

architettura di sistema, permette di sfruttare le capacità di calcolo dei moderni processori per implementare algoritmi anche molto complessi per il *processing* di grandi quantità di dati provenienti dai sensori.

Lo scopo della Tesi è quello di mostrare come può un CPS sostituire il complesso e costoso sistema di posizionamento tuttora operante, attraverso l'uso combinato di un insieme di sensori i cui dati rilevati vengono processati da un algoritmo noto come *Sensor Fusion Algorithm* (SFA).

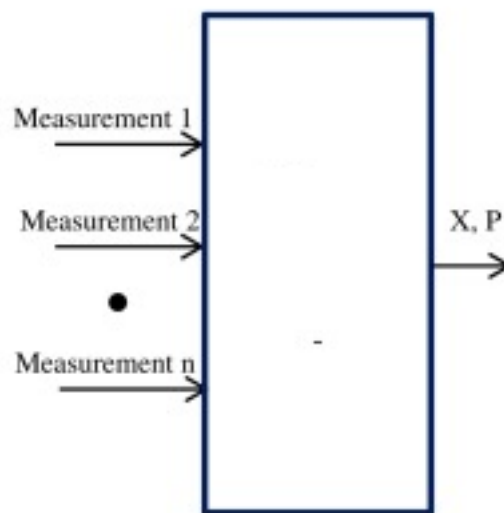


Figura 7: Schema SFA

Tale algoritmo è schematizzabile come una *black-box*: le misurazioni dei sensori sono l'ingresso, mentre l'uscita è la misura cercata, nella fattispecie, la posizione del treno lungo la traccia. Utilizzando SFA, il treno è in grado di auto-posizionarsi, capacità che minimizza la necessità di installare apparati di terra.

Un algoritmo che tiene conto delle misurazioni di un *set* di sensori, usato in luogo di un semplice *processing* di insiemi di misure provenienti da sorgenti omologhe, permette al sistema di correggere il rumore che disturba le singole misurazioni, realizzando così una nuova misura più accurata di quella che si avrebbe considerando i sensori in maniera mutuamente esclusiva.

SENSOR FUSION

Nei sistemi in cui è richiesta un'alta *reliability* delle misure, l'informazione fornita dai singoli sensori non è sufficiente. In questi casi è raccomandato l'utilizzo di un insieme di sensori in contemporanea.

2.1 PANORAMICA

In generale, un algoritmo SFA viene utilizzato per stimare lo stato di un sistema dinamico in un ambiente caratterizzato da *rumore*.

2.1.1 Sistemi Dinamici

Un sistema dinamico è una modellazione matematica di un processo che evolve nel tempo, la cui evoluzione è descritta attraverso un sistema di equazioni differenziali o alle differenze, nel caso esso si evolva rispettivamente a tempo continuo o a tempo discreto.

Sia S l'insieme dei possibili stati che il sistema può assumere, e sia $m = |S|$ la dimensione dello spazio degli stati.

Senza perdere in generalità, si possono formalizzare questi due tipi di sistemi dinamici come:

$$y'(t) = f(t, y(t)), \quad t \geq 0$$

Con $y(0) \in \mathbb{R}^m$ condizione iniziale nota, e:

$$y_{n+1} = f(n, y_n), \quad n = 0, 1, \dots$$

con al solito $y_0 \in \mathbb{R}^m$ condizione iniziale nota.

Ricavare lo stato del sistema dinamico per un certo istante t , o n , equivale a risolvere le equazioni cui sopra e valutarne la traiettoria soluzione in t

o in n .

Un semplice sistema dinamico è rappresentato da un punto materiale che si muove con una accelerazione costante

$$\vec{a} = a\vec{k}$$

dove \vec{k} è un qualunque versore della base canonica di \mathbb{R}^3 .

Supponendo che il punto si muova con velocità iniziale $\vec{z}'(0) = v_0\vec{k}$ nota e inizi il moto da una coordinata $\vec{z}(0) = z_0\vec{k}$ nota, si ha:

$$z''(t) = a$$

$$z'(t) = \int a dt = at + v_0$$

$$z(t) = \int (at + v_0) dt = \frac{1}{2}at^2 + v_0t + z_0$$

L'equazione $z(t)$ descrive completamente la traiettoria di moto del punto materiale, mentre $z'(t)$ descrive completamente la traiettoria della velocità del punto durante il suo moto.

2.1.2 Misure e Rumore

In questo semplice esempio, viene fatta l'assunzione di conoscere a priori il valore esatto di a , di v_0 e di z_0 .

Nella pratica, per misurare l'accelerazione a è necessario uno strumento denominato *accelerometro*, il quale produrrà delle misure giocoforza affette da errori casuali. Si supponga di sostituire a nell'equazione $z(t)$ con una sua perturbazione $\tilde{a} = a + \varepsilon$ dove ε è una variazione casuale della misura data dal *rumore* che caratterizza qualsiasi processo di misura. Si può supporre $\text{Var}(\varepsilon) = 0$ e considerare, ai fini di questa trattazione, ε come un valore costante; in realtà ε è una variabile casuale a varianza generalmente non nulla. Si suppongano inoltre $v_0 = z_0 = 0$ per comodità di calcolo:

$$z(t) = \frac{1}{2}\tilde{a}t^2 = \frac{1}{2}(a + \varepsilon)t^2 = \frac{1}{2}(at^2 + \varepsilon t^2)$$

Si nota immediatamente che la variazione della misura $z(t)$ data da ε aumenta con il quadrato del tempo.

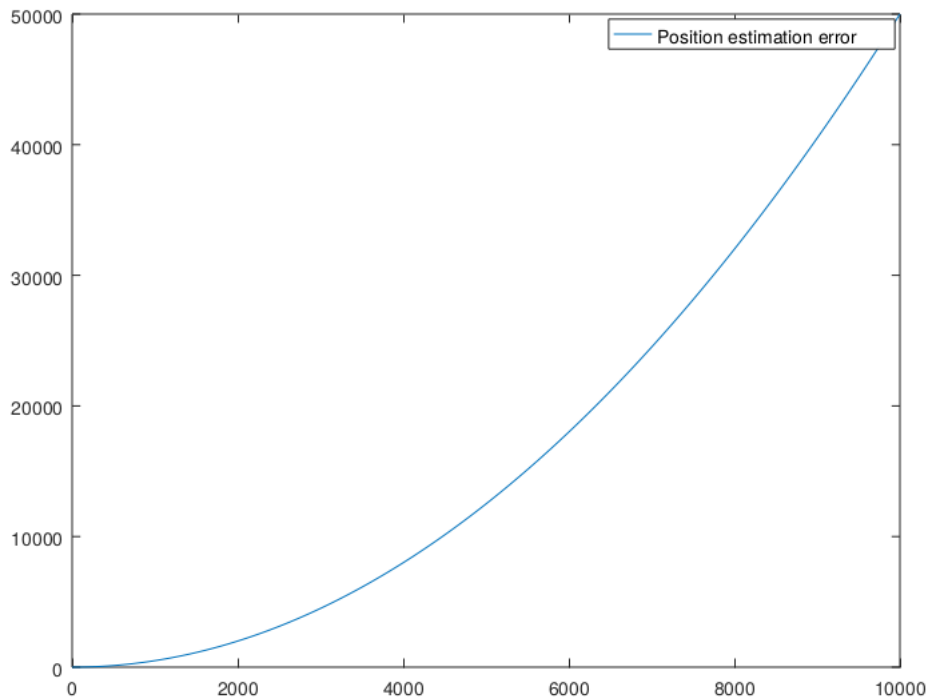


Figura 8: Grafico dell' errore di stima della posizione con $\alpha = 10^0$, $\varepsilon = 10^{-3}$

2.2 I FILTRI DI KALMAN

Un Filtro di Kalman, o in inglese *Kalman Filter* (KF), è un modello di SFA progettato appositamente per risolvere o rendere trascurabile il problema del rumore nei processi di misura.

Siano X_1, \dots, X_N N sorgenti distinte di dati.

Si definisce *misurazione* del valore j dall' i -esima sorgente l'osservazione dell'evento:

$$X_i = j$$

Per un opportuno j e per $i = 1, \dots, N$.

Modellando ciascuna X_i come una variabile casuale, si ha che ciascuna X_i è caratterizzata da una distribuzione di probabilità:

$$p_{X_i} = \{p_j : P(X_i = j) = p_j\}, \quad i = 1, \dots, N$$

Un KF è essenzialmente un algoritmo che utilizza una serie di osservazioni $X_i = j$, e cerca di produrre la stima di una *distribuzione di probabilità congiunta* delle variabili casuali X_i .

2.2.1 Premesse statistiche

È opportuno fissare la notazione che sarà usata nelle discussioni che seguiranno.

Per una data distribuzione di probabilità p_X si definisce la *media*, o *valore atteso*, di p_X come:

$$\mathbb{E}(p_X) = \sum_{i=1}^{+\infty} X_i p_{X_i} = \mu_X$$

Si definisce *varianza* di p_X , la quantità:

$$\text{Var}(p_X) = \mathbb{E}[(p_X - \mu_X)]^2 = \sum_{i=1}^{+\infty} (X_i - \mu_X)^2 p_{X_i} = \sigma_X^2$$

Date due variabili casuali, siano esse X e Y , definite sullo stesso insieme di cardinalità M , si chiama *covarianza* di x, y la seguente quantità:

$$\text{cov}(X, Y) = \frac{1}{M} \sum_{i=1}^M (X - \mathbb{E}(X))(Y - \mathbb{E}(Y)) = \sigma_{XY}$$

Si osservi che

$$\text{cov}(X, X) = \frac{1}{M} \sum_{i=1}^M (X - \mathbb{E}(X))(X - \mathbb{E}(X)) = \frac{1}{M} \sum_{i=1}^M (X - \mathbb{E}(X))^2 = \sigma_X^2$$

Per una variabile aleatoria multivariata, quale una distribuzione di probabilità congiunta su N variabili casuali, siano esse X_1, \dots, X_N , si definisce la matrice di *varianza-covarianza*, o semplicemente matrice di *covarianza*, la seguente matrice quadrata $N \times N$:

$$\begin{aligned} \Sigma &= \begin{pmatrix} \text{cov}(X_1, X_1) & \text{cov}(X_1, X_2) & \dots & \text{cov}(X_1, X_N) \\ \vdots & \vdots & \vdots & \\ \text{cov}(X_N, X_1) & \text{cov}(X_N, X_2) & \dots & \text{cov}(X_N, X_N) \end{pmatrix} \\ &= \begin{pmatrix} \sigma_{X_1}^2 & \sigma_{X_1, X_2} & \dots & \sigma_{X_1, X_N} \\ \vdots & \vdots & \vdots & \\ \sigma_{X_N, X_1} & \sigma_{X_N, X_2} & \dots & \sigma_{X_N}^2 \end{pmatrix} \end{aligned}$$

2.2.2 Classificazioni

I KF sono comunemente basati su sistemi dinamici *lineari* a tempo discreto, tuttavia i fenomeni reali sono raramente lineari. Un modello lineare è spesso un'approssimazione di un modello più complesso.

Nel dominio applicativo in cui si colloca la Tesi, ossia quello del posizionamento ferroviario, occorre basare il Filtro di Kalman su un modello non-lineare.

2.2.3 Filtri di Kalman Estesi

I Filtri di Kalman Estesi, in inglese *Extended Kalman Filter* (EKF), sono una generalizzazione al caso non-lineare dei classici Filtri di Kalman lineari. Il funzionamento base di un EKF coinvolge la stima dello stato di un sistema dinamico *discreto* non lineare:

$$\mathbf{x}_{k+1} = F(\mathbf{x}_k, \mathbf{v}_k)$$

$$\mathbf{y}_k = H(\mathbf{x}_k, \mathbf{n}_k)$$

Dove

- \mathbf{x}_k rappresenta lo stato non osservato del sistema;
- \mathbf{y}_k rappresenta l'unica misura osservata;
- \mathbf{v}_k rappresenta il *rumore di processo*;
- \mathbf{n}_k rappresenta il *rumore di misura*.

Si suppongono infine note le funzioni F, H che descrivono la dinamica del modello.

Filtri di Kalman Unscented

APPLICAZIONE DI SFA: LA TRAMVIA DI FIRENZE

In questo capitolo verrà analizzata una particolare applicazione di SFA al problema del posizionamento ferrotramviario.

Nell'ambito di un progetto di ricerca finanziato dall'Unione Europea, si è voluto studiare l'usabilità di SFA come sistema di posizionamento ferrotramviario alternativo a quello descritto nel Capitolo 1, il quale fa un largo uso di apparati installati a terra, fatto che si vorrebbe minimizzare. La linea ferrotramviaria scelta come ambiente di prova è la linea T1 della Tramvia di Firenze, che collega la stazione di *Villa Costanza*, sita nel comune di Scandicci all'altezza dell'omonimo parcheggio di interscambio dell'autostrada A1, all'ospedale di *Careggi*, sito quest'ultimo nel comune di Firenze.

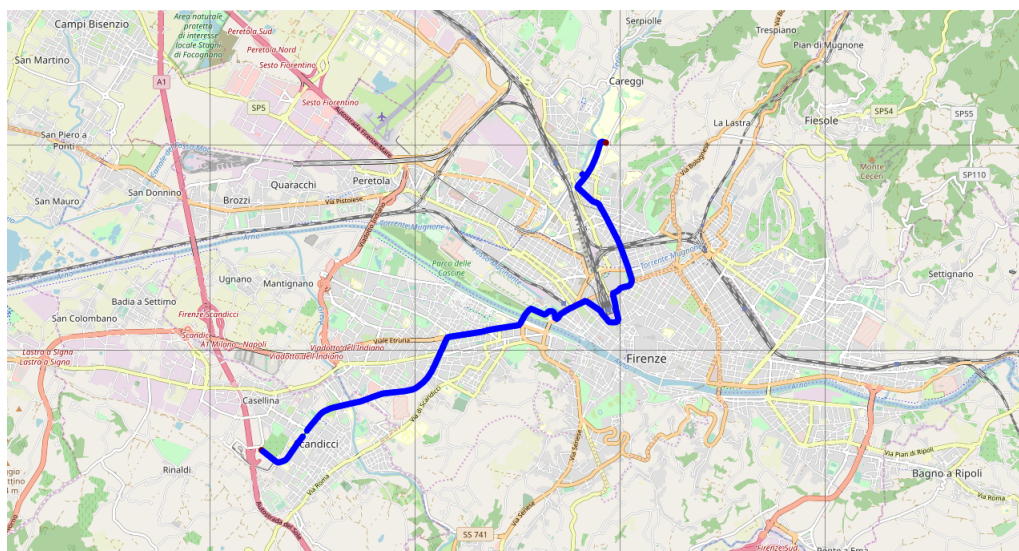


Figura 9: Tramvia di Firenze - Linea T1

3.1 ARCHITETTURA DI SISTEMA

Il sistema progettato ha lo scopo di eseguire SFA su una piattaforma hardware installata bordo treno, la quale riceve i dati *raw* dai sensori e li elabora al fine di stimare la progressiva chilometrica del treno in ciascun istante di tempo.

Tale posizione sarà inviata, attraverso un modem LTE:

- All'OBCU, per essere utilizzata attivamente all'interno del sistema di *interlocking*
- Ad un arbitrario host che esegue un software grafico di tracciamento del treno: il RailTrackTool (RTT)

È possibile descrivere l'architettura di sistema a due differenti livelli: architettura a livello *hardware* e architettura a livello *software*.

3.1.1 Architettura Hardware

Sul treno è stata installata una scheda Nvidia TX-Jetson quale piattaforma di elaborazione dei dati. I sensori atti a campionare le misurazioni sono stati collegati alla scheda mediante appositi bus dati.

Il *sensor set* utilizzato in quest'applicazione è composto dai seguenti sensori:

- *Inertial Measurement Unit* (IMU):
Sensore incaricato di misurare i vettori accelerazione (**a**) e velocità angolare (**v_{ang}**) attraverso l'uso combinato di un accelerometro e un giroscopio. Le misure di IMU sono prese rispetto a un sistema inerziale solidale con il binario e sono espresse in unità stabilite dallo standard internazionale (SI):

$$\mathbf{a} \left[\frac{\text{m}}{\text{s}^2} \right] \quad \mathbf{v}_{\text{ang}} \left[\frac{\text{rad}}{\text{s}} \right]$$

Si tratta del sensore principale su cui si basa l'esecuzione di SFA, ed è caratterizzato da un *drift*, il quale fa discostare il valore di accelerazione misurato da quello reale, in una quantità che è funzione del tempo.

- Odometro:

Per realizzare l'odometro è stato installato un rilevatore radar su una ruota del treno. Il radar misura il tempo impiegato dalla ruota a compiere un giro completo, e determina la velocità angolare della ruota $\varphi'(t) = \frac{2\pi}{\text{tempo}} \left[\frac{\text{rad}}{\text{s}} \right]$.

Noto il raggio r [m] della ruota, è possibile determinare la velocità lineare alla circonferenza della ruota $x'(t)$ attraverso la relazione cinematica $x'(t) = r\varphi'(t) \left[\frac{\text{m rad}}{\text{s}} \right] = r\varphi'(t) \left[\frac{\text{m}}{\text{s}} \right]$.

Approssimando il treno come un *corpo rigido*, questa sarà la velocità lineare con cui il treno si sta muovendo.

- Global Positioning System (GPS):

Sensore che riceve i dati di posizione attraverso il sistema satellitare GPS.

Le misure di GPS sono riportate in formato standard come tripla di coordinate (latitudine, longitudine, altitudine), rispettivamente espresse in gradi N-S, in gradi E-O e in metri.

In generale queste misure sono le meno affidabili in quanto la *varianza* della variabile aleatoria che modella tale sorgente è la più significativa.

Ad una data frequenza, i sensori inviano dati verso la scheda; quest'ultima, dopo aver eseguito un'iterazione di SFA, invia a OBCU (e/o a RTT) la stima della posizione del treno attraverso apposita modulazione di segnale elettromagnetico, in accordo con il protocollo LTE. Lo schema riportato in figura 10 mostra un diagramma dell'architettura hardware appena descritta.

3.1.2 Architettura Software

Sulla scheda è installato il sistema operativo Ubuntu 16.04 LTS, basato su kernel Linux.

Qualunque software menzionato in questa Tesi è stato sviluppato in linguaggio C++.

Un set di tre moduli software, denominati interface-modules, sono in esecuzione sulla scheda.

Sia MOD_i l'i-esimo modulo software del set e SERIAL_i l'i-esima interfaccia seriale della scheda, per $i = 1, 2, 3$.

Il funzionamento di interface-modules è il seguente:

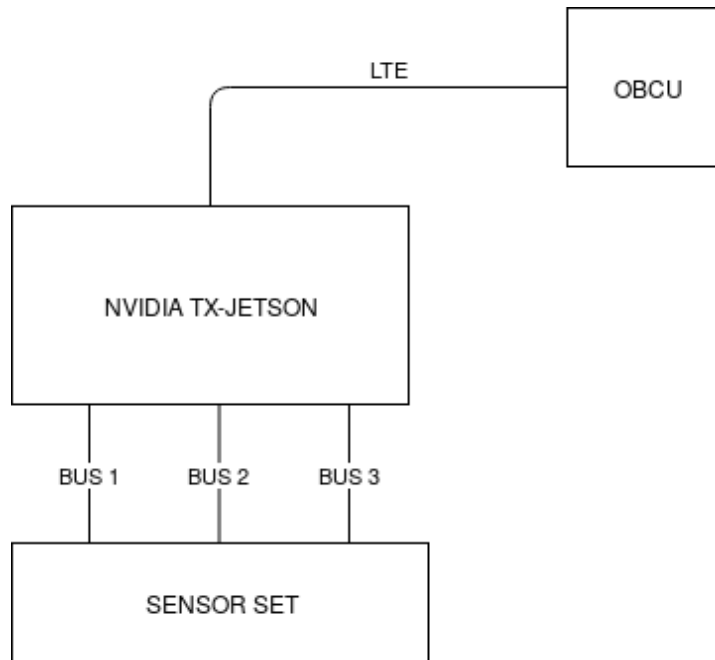


Figura 10: Architettura hardware bordo treno

- IMU invia la coppia (accelerazione, velocità angolare) a SERIAL_1, MOD_1 legge i valori da SERIAL_1 e li invia a un secondo modulo software, denominato listener, attraverso l'interfaccia di rete loopback, in quanto listener esegue anch'esso sulla scheda;
- Odometro invia il valore di velocità lineare a SERIAL_2, MOD_2 legge i valori da SERIAL_2 e li invia a listener;
- GPS invia i valori di (latitudine, longitudine, altitudine) a SERIAL_3, MOD_3 legge i valori da SERIAL_3 e li invia a listener.

La comunicazione fra interface-modules e listener avviene attraverso un protocollo applicazione stabilito arbitrariamente, sia esso INPUT_PROTOCOL, mentre a livello di trasporto si utilizza UDP.

I valori ricevuti da listener vengono salvati in apposite *strutture dati* rappresentanti misure della stessa sorgente:

- I vettori accelerazione e velocità angolare rilevati da IMU vengono convertiti nella struttura dati IMU_POD;
- La velocità rilevata dal Radar/Odometro viene convertita nella struttura dati ODO_POD;

- La posizione rilevata dal GPS viene infine convertita nella struttura dati GPS_POD.

Il software che esegue effettivamente SFA è compilato come una libreria, *FusionLib*, utilizzata da *listener*. *FusionLib* dispone di interfacce software in entrata e in uscita, ossia *listener* è in grado di inviare le misurazioni a SFA, quali variabili di tipo IMU_POD, ODO_POD, GPS_POD ed altresì di ricevere la stima della posizione del treno, essendo questo l'output dell'algoritmo, quale variabile di tipo SFA_OUTPUT_POD.

Ogniqualvolta *listener* riceva un'uscita da SFA, si fa carico della comunicazione tra scheda e OBCU/RTT. Questa comunicazione, fisicamente possibile attraverso l'utilizzo del modem LTE, avviene utilizzando un protocollo di rete arbitrario a livello applicazione, sia esso OUTPUT_PROTOCOL, mentre al livello di trasporto la scelta è nuovamente ricaduta su UDP per ragioni di efficienza.

Uno schema dell'architettura software è quello mostrato in figura 11.

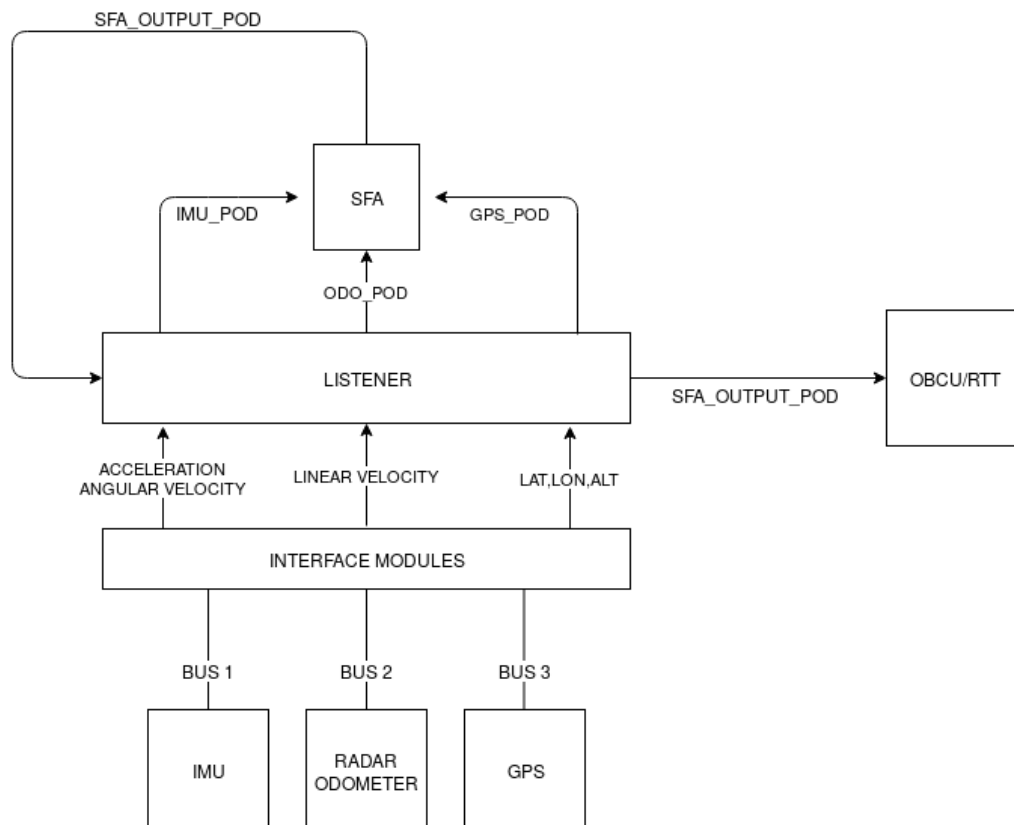


Figura 11: Architettura software bordo treno

3.2 GESTIONE DELLA TRASMISSIONE DEI DATI

Nella precedente sezione sono stati brevemente introdotti i protocolli di comunicazione implementati per gestire la comunicazione UDP:

- In entrata, tra interface-modules e listener (INPUT_PROTOCOL);
- In uscita, tra listener e OBCU/RTT (OUTPUT_PROTOCOL).

3.2.1 Trasmissione in entrata

Per trasmettere i dati da interface-modules a listener, e dunque dai sensori al modulo software che implementa SFA, è stato realizzato un protocollo di comunicazione denominato INPUT_PROTOCOL.

Tale protocollo fa affidamento a livello trasporto su UDP per massimizzare la velocità di trasmissione senza dover necessariamente rinunciare all'integrità dei messaggi trasmessi, in quanto la comunicazione avviene tra processi in esecuzione sulla stessa macchina, e la probabilità che un messaggio venga perso o che questo venga ricevuto con errori, è assolutamente trascurabile.

Il protocollo definisce il formato del *payload* del pacchetto UDP che contiene le informazioni di IMU, Radar/Odometro, o GPS, ed è descritto in tabella 2.

| Campo | Descrizione | Indici di bit | Tipo |
|-------------|----------------------------|---------------|----------|
| SENSOR_TYPE | ID Sensore Sorgente | 0-7 | uint8_t |
| Seq.NO | Numero di sequenza | 8-23 | uint16_t |
| N_INT | Numero di interi trasmessi | 24-31 | uint8_t |
| N_DOUBLE | Numero di double trasmessi | 31-38 | uint8_t |

Tabella 2: Protocollo di comunicazione in entrata

A discrezione del valore del campo `SENSOR_TYPE` si distingue il tipo di informazione trasportata dal pacchetto, come descritto in tabella 3.

I pacchetti `GROUND TRUTH` sono pacchetti di inizializzazione dell'algo-

| Valore di <code>SENSOR_TYPE</code> | Sorgente del pacchetto |
|------------------------------------|------------------------|
| 1 | IMU |
| 2 | ODOMETRO |
| 3 | GPS |
| 8 | GROUND TRUTH |
| 9 | STROBE |
| 10 | STOP |

Tabella 3: Significato del campo `SENSOR_TYPE`

ritmo: alla ricezione del pacchetto `GROUND TRUTH` l'algoritmo si avvia leggendo i valori trasmessi in coda al pacchetto, in accordo al valore dei campi `N_INT` e `N_DOUBLE`. Tali valori forniscono informazioni come progressiva chilometrica e velocità iniziali del treno.

I pacchetti `STROBE` sono inviati ogni secondo e forniscono un solo valore double, ossia un timestamp che l'algoritmo utilizza per sincronizzarsi.

Il pacchetto `STOP` non contiene alcuna informazione utile: indica soltanto all'algoritmo di terminare l'esecuzione.

Alla ricezione di un pacchetto, `listener` legge il valore del campo `SENSOR_TYPE`, e costruisce, in accordo alla relazione sorgente-struttura dati, la variabile da inviare a SFA.

Il corretto ordinamento dei pacchetti trasmessi a SFA è garantito attraverso l'esplicito utilizzo di un buffer, codificato all'interno di `listener`, in cui i pacchetti vengono temporaneamente salvati prima di essere inviati a SFA, ed eventualmente ordinati sulla base del valore del campo `Seq.NO`. Si osservi che se l'integrità non è minacciata dall'utilizzo di UDP quale protocollo di trasporto fra processi all'interno della stessa macchina fisica, altrettanto non si può dire dell'ordinamento dei messaggi. Questi potrebbero subire dei ritardi casuali in base allo stato del sistema operativo, in particolare lo *scheduling* dei processi può avere influenze determinanti sullo scorretto ordinamento dei messaggi trasmessi. Utilizzando TCP si ovvierebbe a questa problematica, ma l'overhead insito nel protocollo stesso causerebbe un notevole degrado delle performance di SFA.

3.2.2 *Trasmissione in uscita*

La trasmissione dei dati in uscita da SFA avviene, in accordo al protocollo OUTPUT_PROTOCOL tra listener e OBCU, o comunque, tra listener e qualunque host arbitrario che intenda ricevere le informazioni in uscita, come ad esempio un PC sul quale viene eseguito RTT.

Come specificato, la comunicazione è posta in essere, a livello fisico, attraverso il protocollo LTE, ossia un un protocollo *wireless*; mentre a livello trasporto si è scelto di continuare a usare UDP in luogo di TCP, col fine di massimizzare le *performance* del sistema.

Il rischio di ricevere alcune informazioni in maniera errata, o non riceverle del tutto, è nettamente più elevato rispetto allo scenario precedente, nel caso in cui lo spazio fisico attraverso cui si propaga il segnale LTE è tale per cui quest'ultimo venga disturbato da sorgenti esterne.

Gli effetti deleteri di questa condizione sono particolarmente osservabili in alcuni tratti della linea ferrotramviaria, dove possono essere presenti numerose abitazioni e mezzi di trasporto in strada che si interpongono fisicamente tra la scheda NVidia TX-Jetson su cui esegue SFA e l'arbitrario host su cui viene eseguito RTT.

Occorre tuttavia osservare che il tracciamento del treno tramite RTT non è in alcun modo legato alla *safety* del sistema, in quanto le funzionalità *safety-critical* riguardano la comunicazione tra la scheda e OBCU, ossia tra la scheda e il sistema di *interlocking*.

Questa problematica è risolta attraverso l'esplicito utilizzo di un meccanismo di acknowledgment simile a quello utilizzato da TCP: ciascun pacchetto in uscita da SFA viene indicizzato con un *sequence number* e, in ricezione, viene inviato ogni secondo un *ack* replicante l'ultimo numero di sequenza correttamente ricevuto. Solo quando il mittente riceve l'*ack* i dal destinatario invierà il messaggio contenente l'uscita indicizzata con *sequence number* $i + 1$.

Anche in questo caso, il protocollo definisce il formato del *payload* del pacchetto UDP inviato da listener, ed è riportato in tabella 4.

| Campo | Descrizione | Indici di bit | Tipo |
|------------|--------------------------|---------------|----------|
| Seq.NO | Numero di sequenza | 0-15 | uint16_t |
| ECEF_X | Coordinata X del treno | 16-79 | double |
| ECEF_Y | Coordinata Y del treno | 80-143 | double |
| ECEF_Z | Coordinata Z del treno | 144-207 | double |
| FU_ARC_LEN | Progressiva chilometrica | 208-271 | double |

Tabella 4: Protocollo di comunicazione in uscita

In ricezione dovrà essere inviato il pacchetto *ack* al mittente, ed il suo formato è descritto in tabella 5. Si osserva che SFA produce la stima

| Campo | Descrizione | Indici di bit | Tipo |
|-------|---------------|---------------|----------|
| ACK | Ultimo Seq.NO | 0-15 | uint16_t |

Tabella 5: Formato del pacchetto di *ack*

della posizione del treno sia in termini di progressiva chilometrica che di coordinate ECEF.

ECEF è acronimo di *Earth Centered Earth Fixed* ed è uno standard che misura le coordinate geografiche di un oggetto come la terna $P = (x, y, z)$. Ciascuna coordinata viene espressa considerando la *proiezione su piano* della Terra, e prendendo come origine O l'intersezione fra l'equatore e il meridiano di *Greenwich*.

Le coordinate ECEF misurano tre lunghezze, pertanto in accordo a SI, esse sono espresse in metri.

Nella prossima sezione, viene descritto uno scenario di esempio del comportamento del sistema a *runtime*.

3.3 SCENARIO DI ESEMPIO

Si suppongano le condizioni iniziali riportate in tabella 6.

| Velocità | ECEF | Progressiva | IMU Sample Rate | ODO Sample Rate |
|-------------------|---------------|-------------|-----------------|-----------------|
| 0ms^{-1} | $(0, 0, 0)$ m | 0 km | 100 Hz | 20 Hz |

Tabella 6: Condizioni iniziali

1. $t = 0$:

- interface-modules invia a listener il seguente pacchetto GROUND TRUTH:

| SENSOR_TYPE | Seq. NO | N_INT | N_DOUBLE |
|-------------|---------|-------|----------|
| 0x08 | 0x00 | 0 | 5 |

E vi accoda i seguenti tre valori double: 0.0, 0.0, 0.0 ossia le coordinate ECEF iniziali, il seguente valore double: 0.0, ossia la velocità lineare iniziale, e infine il valore double: 0.0 che rappresenta la progressiva chilometrica iniziale.

- listener riceve il pacchetto e inizializza SFA con:
 - ECEF iniziali: $(0, 0, 0)$
 - Velocità lineare iniziale: 0.0
 - Progressiva chilometrica iniziale: 0.0

2. $t = t_0$:

- IMU campiona il seguente vettore accelerazione:

$$\mathbf{a} = (0.0001, -0.0001, -9.8100)$$

Assieme al seguente vettore velocità angolare:

$$\mathbf{v}_{\text{ang}} = (0.0003, -0.0001, 0.0002)$$

E lo invia, tramite SERIAL_1, a MOD_1 di interface-modules.

- MOD_1 invia a listener il seguente pacchetto IMU:

| SENSOR_TYPE | Seq. NO | N_INT | N_DOUBLE |
|-------------|---------|-------|----------|
| 0x01 | 0x01 | 0 | 6 |

Accodandovi nell'ordine il vettore accelerazione, e il vettore velocità angolare.

- listener riceve il pacchetto, crea e invia a SFA la seguente variabile IMU_POD:

- Seq.NO = 1
- Epoch = t_0
- ACC_X = 0.0001
- ACC_Y = -0.0001
- ACC_Z = -9.8100
- GYRO_X = 0.0003
- GYRO_Y = -0.0001
- GYRO_Z = 0.0002

- SFA elabora il pacchetto e inizia una computazione parallela per fornire a listener una variabile SFA_OUTPUT_POD della forma:

- Seq.NO = 0
- ECEF_X = E_X
- ECEF_Y = E_Y
- ECEF_Z = E_Z
- FU_ARC_LEN = P_{KM}

$$3. \quad t_0 < t < t_0 + \frac{1}{\text{ODO_SAMPLE_RATE}} = t_0 + \frac{1}{20}$$

Fintantoché l'odometro non campiona il suo primo valore di velocità, si ripetono le operazioni viste al passo precedente per ogni campionamento di IMU.

$$4. \quad t = t_0 + \frac{1}{20}$$

- Odometro campiona il seguente valore di velocità:

$$\mathbf{a} = (1.0010)$$

E lo invia, tramite SERIAL_2, a MOD_2 di interface-modules.

- MOD_2 invia a listener il seguente pacchetto ODOMETRO:

| SENSOR_TYPE | Seq. NO | N_INT | N_DOUBLE |
|-------------|---------|-------|----------|
| 0x02 | Seq_NO | 0 | 2 |

Accodandovi nell'ordine il valore di velocità rilevato, e il valore dello scarto quadratico medio della sorgente, noto a priori, in quanto caratteristica tecnica intrinseca dello strumento di misura, il radar; sia esso SIGMA_RADAR.

- listener riceve il pacchetto, crea e invia a SFA la seguente variabile ODO_POD:
 - Seq.NO = Seq_NO
 - Epoch = $t_0 + \frac{1}{20}$
 - vel = 1.0010
 - sigma = SIGMA_RADAR
- SFA elabora il pacchetto e utilizza la rilevazione di velocità in maniera utile a correggere il *drift* di IMU, al fine di produrre una stima della posizione più accurata.

5. $t = n t_0 \quad n \in \mathbb{N}^+$

Ogni secondo, il modulo STROBE di interface-modules, invia a listener un pacchetto della forma:

| SENSOR_TYPE | Seq. NO | N_INT | N_DOUBLE |
|-------------|---------|-------|----------|
| 0x09 | Seq_NO | 0 | 1 |

Accodandovi un *timestamp* che listener inoltra a SFA per scopi di sincronizzazione.

Quanto elencato viene ripetuto per ciascun campionamento successivo di IMU e odometro.

Non appena un'uscita di SFA si rende disponibile a listener questo si comporta come segue:

- listener riceve la variabile SFA_OUTPUT_POD, da SFA;
- listener costruisce il seguente pacchetto da inviare a OBCU, o a RTT:
 - Seq.NO = 0x00

- $ECEF_X = SFA_OUTPUT_POD.E_X$
 - $ECEF_Y = SFA_OUTPUT_POD.E_Y$
 - $ECEF_Z = SFA_OUTPUT_POD.E_Z$
 - $FU_ARC_LEN = SFA_OUTPUT_POD.P_{KM}$
- OBCU, o RTT, riceve il pacchetto e invia a listener l'*ack* 0x00.

3.3.1 Nota sui numeri di sequenza

Il conteggio del numero di sequenza dei pacchetti in entrata è gestito dai moduli `interface_modules`. Questi moduli sono processi concorrenti che condividono un'area di memoria in cui viene caricata una variabile intera. Questa variabile viene incrementata in maniera mutuamente esclusiva ogniqualvolta un modulo riceva un pacchetto, e il suo valore sarà il valore del campo `Seq.NO` inviato a listener.

Per quanto concerne l'indicizzazione dei pacchetti in uscita, questa è invece gestita da listener, attraverso una variabile contatore incrementata ogni qualvolta SFA invii a listener un'uscita.

Il valore di tale contatore sarà pertanto il numero di sequenza del pacchetto `SFA_OUTPUT_POD` da inviare a OBCU o a RTT.

3.4 POSSIBILI SVILUPPI

Il sistema, così come è stato descritto, rappresenta essenzialmente un *core* minimale di un sistema di posizionamento basato su SFA, limitato rispetto alle potenzialità dell'algoritmo e comunque non esente da vulnerabilità legate alla *security*. In questa sezione verranno discusse le principali problematiche della soluzione descritta, in che modo queste possono essere risolte, e quali tecniche possono essere usate per migliorare l'usabilità del sistema.

3.4.1 Problematiche legate alla security

Per *security* si intende un insieme di tecniche che hanno come scopo la protezione dei dati, siano essi stoccati in un sistema informatico, oppure transitanti attraverso un sistema di telecomunicazione.

Tale protezione viene assicurata contro specifiche *minacce*, le quali sfruttano opportune *vulnerabilità*.

La *security* viene garantita attraverso l'uso di appropriate *tecniche preventive*, oppure *contromisure* applicabili in caso di violazioni alle principali *misure della security*:

- Integrità
- Confidenzialità
- Autenticazione

In un sistema *safety-critical* come quello descritto, una violazione di *security* potrebbe portare a una violazione di *safety*, pertanto è fondamentale ridurre al massimo le vulnerabilità del sistema. Nella fattispecie descritta in questa Tesi, tuttavia, la confidenzialità non è una misura fondamentale, mentre lo sono l'integrità e l'autenticazione.

Minacce all'integrità

È stato già discusso che l'utilizzo del protocollo UDP a livello di trasporto, non garantisce affatto che i messaggi ricevuti da OBCU siano corretti e ordinati.

Per ovviare al problema dell'ordinamento è stato implementato il già descritto meccanismo di acknowledgment, tuttavia esso fa l'implicita assunzione che se si è in grado di leggere correttamente il numero di

sequenza del pacchetto ricevuto, questo non sia stato alterato.
Si consideri il seguente scenario:

- listener invia a OBCU il seguente pacchetto:
 - Seq.NO = 0x17
 - ECEF_X = SFA_OUTPUT_POD.E_X
 - ECEF_Y = SFA_OUTPUT_POD.E_Y
 - ECEF_Z = SFA_OUTPUT_POD.E_Z
 - FU_ARC_LEN = SFA_OUTPUT_POD.P_{KM}
- OBCU riceve il seguente pacchetto:
 - Seq.NO = 0x25
 - ECEF_X = SFA_OUTPUT_POD.E_X
 - ECEF_Y = SFA_OUTPUT_POD.E_Y
 - ECEF_Z = SFA_OUTPUT_POD.E_Z
 - FU_ARC_LEN = SFA_OUTPUT_POD.P_{KM}

Per come è stato descritto il protocollo, OBCU accetta passivamente che il numero di sequenza ricevuto sia 0x25, anche se prima di questo era stato letto il valore 0x16, ed invierà a listener l'*ack* 0x25.

In questo caso, OBCU dovrebbe essere progettato in maniera tale da controllare sempre di ricevere un numero di sequenza pari all'ultimo ricevuto +1. Dal momento che, viste le caratteristiche intrinseche del protocollo, è impossibile che listener abbia inviato il pacchetto con numero di sequenza 0x25 se l'ultimo ack ricevuto non era 0x24, è probabile che, attraversando il canale, il pacchetto abbia subito alterazioni casuali in tutti i suoi bit, e quindi anche l'informazione di posizione potrebbe essere alterata.

Per ovviare definitivamente alla problematica dell'integrità, è opportuno integrare nel protocollo l'uso di una *funzione hash*. Il protocollo verrebbe modificato come segue:

- listener prepara il pacchetto contenente le informazioni di SFA_OUTPUT_POD;
- listener calcola $H(\text{SFA_OUTPUT_POD}) = y$;
- listener invia la coppia (SFA_OUTPUT_POD, y)

In ricezione, OBCU ricalcola $H(\text{SFA_OUTPUT_POD}) = y'$, e accetta il messaggio solo se $y' = y$. Infatti, grazie alla proprietà delle funzioni *hash*, una minima variazione del messaggio m causa una grande variazione del *digest* $H(m)$, quindi è altamente improbabile che un'alterazione casuale dei bit trasmessi, sia essa $(\text{SFA_OUTPUT_POD_WRONG}, Y_WRONG)$, mantenga la proprietà $H(\text{SFA_OUTPUT_POD_WRONG}) = Y_WRONG$.

Minacce all'autenticazione

Si consideri il caso in cui un malintenzionato sia in grado di inviare messaggi a OBCU e abbia interesse nel non segnalare al sistema di *interlocking* l'avvicinamento del treno alla JA.

L'attaccante si comporta come segue:

- Intercetta il messaggio $(\text{SFA_OUTPUT_POD}, H(\text{SFA_OUTPUT_POD}))$
- Modifica la posizione del treno ponendola lontano da una JA, forgiando un nuovo messaggio, sia esso $\text{SFA_OUTPUT_POD_DANGEROUS}$
- Calcola $H(\text{SFA_OUTPUT_POD_DANGEROUS}) = Y_DANGEROUS$
- Invia a OBCU la coppia $(\text{SFA_OUTPUT_POD_DANGEROUS}, Y_DANGEROUS)$

Per ovviare a questa problematica si potrebbero usare le seguenti tecniche:

1. Cifratura del *digest* della funzione *hash* con una chiave simmetrica condivisa tra listener e OBCU;
2. Cifratura del *digest* della funzione *hash* con la chiave privata di listener (Firma Digitale DSA);
3. Uso di una funzione *hash* che prende in ingresso sia il messaggio che una chiave simmetrica condivisa tra listener e OBCU (HMAC);
4. Accodare un segreto condiviso tra listener e OBCU al messaggio prima di calcolarne il *digest*.

In ciascuno di questi scenari, fatta assunzione di proprietà di *strong collision resistance* della funzione *hash* utilizzata, si garantisce che il messaggio può essere stato inviato solo da listener, in quanto un attaccante non avrebbe modo di modificare il messaggio e calcolare un *digest* valido. La soluzione meno dispendiosa in termini di complessità computazionale e più adatta a un simile scenario è la soluzione 4, in quanto non è necessario garantire anche la *non-ripudiabilità* ma solo l'autenticazione e l'integrità.

3.4.2 *Miglioramenti al servizio fornito*