



ABSTRACT

INDICE

- 1 Stato dell'Arte 9
 - 1.1 Sistemi Ferroviari e Ferrotramviari 9
 - 1.2 Il problema del posizionamento 10
 - 1.2.1 Criticità 13
- 2 Architettura di Sistema 15
 - 2.1 Descrizione generale 15
 - 2.2 Constituent Systems 16
 - 2.3 Specifica delle Interfacce 18
 - 2.3.1 Relied Upon Interfaces 18
 - 2.3.2 Altre Interfacce 20
 - 2.4 Interazioni 20
 - 2.4.1 Acquisizione dei dati 21
 - 2.4.2 Trasmissione della posizione 21
- 3 Specifiche Software 23
 - 3.1 Architettura Software 23
 - 3.2 Protocolli di Comunicazione 24
 - 3.2.1 Trasmissione in entrata 24
 - 3.2.2 Trasmissione in uscita 26
 - 3.3 Scenario di Esempio 27
- 4 Ambiente di Analisi 31
 - 4.1 Configurazioni 31
 - 4.1.1 Configurazione Online 31
 - 4.1.2 Configurazione Offline 32
- 5 Esperimenti e Risultati 35
- 6 Conclusioni 37

ELENCO DELLE TABELLE

| | | |
|-----------|----------------------------------------|----|
| Tabella 1 | Specifiche delle RUPI del sistema | 19 |
| Tabella 2 | Protocollo di comunicazione in entrata | 25 |
| Tabella 3 | Significato del campo SENSOR_TYPE | 25 |
| Tabella 4 | Protocollo di comunicazione in uscita | 26 |
| Tabella 5 | Formato del pacchetto di <i>ack</i> | 27 |
| Tabella 6 | Condizioni iniziali | 27 |

ELENCO DELLE FIGURE

| | | |
|-----------|-----------------------------------------|----|
| Figura 1 | Schema di un tipico scenario tramviario | 10 |
| Figura 2 | UCS | 11 |
| Figura 3 | Conta Assi | 12 |
| Figura 4 | <i>Point Machine</i> | 12 |
| Figura 5 | Schema SFA | 15 |
| Figura 6 | <i>Inertial Measurment Unit</i> | 17 |
| Figura 7 | Ricevitore GPS ublox EVK-M8T | 17 |
| Figura 8 | Nvidia TX-Jetson | 18 |
| Figura 9 | Modem TP-LINK M7350 LTE-4G | 19 |
| Figura 10 | RUMI | 20 |
| Figura 11 | Sequenza di acquisizione dati | 21 |
| Figura 12 | Architettura software bordo treno | 24 |
| Figura 13 | Configurazione <i>online</i> | 32 |
| Figura 14 | Configurazione <i>offline</i> | 32 |

1

STATO DELL'ARTE

Questa Tesi si colloca nell’ambito del posizionamento ferrotramviario. Il problema del posizionamento esiste tanto nel contesto ferroviario quanto nel contesto ferrotramviario, dal momento che il secondo nasce come derivazione dal primo. In questo capitolo vengono introdotte le principali caratteristiche dei sistemi ferrotramviari e si descrive lo stato dell’arte nell’ambito del posizionamento ferrotramviario, al fine di introdurre gli argomenti trattati nel seguito.

1.1 SISTEMI FERROVIARI E FERROTRAMVIARI

È possibile schematizzare un Sistema Ferroviario, o Ferrotramviario, come un insieme di vetture vincolate a muoversi lungo una traccia fissa. Questa schematizzazione vale per 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, e che operano pertanto prevalentemente in ambienti extra urbani. Nel trasporto ferrotramviario, di contro, si utilizzano vetture dalle ridotte dimensioni, più leggere di quelle usate 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

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 cittadino, come mostrato in figura 1.



Figura 1: Schema di un tipico scenario tramviario

1.2 IL PROBLEMA DEL POSIZIONAMENTO

Per posizionamento ferroviario, si intende la valutazione della posizione di un treno all'interno di una traccia ferroviaria. [1] Esso esiste tanto nel contesto ferrotramviario quanto nel contesto ferroviario classico.

Tale posizione viene espressa come progressiva chilometrica rispetto all'origine della linea.

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*. [2] [6]

Tale sistema è detto *safety-critical* [7], pertanto offre una funzionalità che deve rispettare adeguati standard di sicurezza. [4] [5]

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.[3]

Odierne Tecniche di Posizionamento

I sistemi di posizionamento attualmente in uso hanno come nucleo essenziale il sistema di *interlocking*, il quale si fa 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. [9] Un sistema di *interlocking* è composto dai seguenti elementi:

- *Switch Control Unit (UCS)*:

Piattaforma certificata SIL-3 [8] 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* verso gli elementi di *interlocking* installati a terra e ne consente un controllo sicuro in accordo allo standard SIL-3.



Figura 2: UCS

- 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.



Figura 3: Conta Assi

- *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.

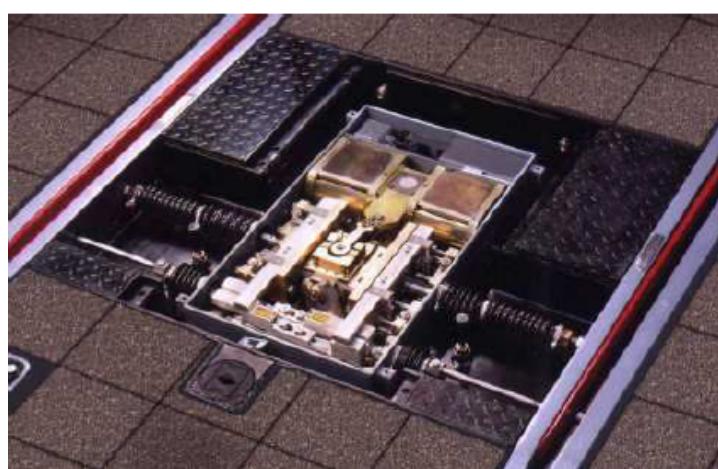


Figura 4: *Point Machine*

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.

1.2.1 Criticità

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. [10][20]

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 i risultati sperimentali delle campagne di analisi condotte su di un innovativo sottosistema di posizionamento. Tale sottosistema ha lo scopo di stimare la posizione di un treno attraverso l'uso combinato di un insieme di sensori installati bordo treno, e al fine di integrarne le misure, queste dovranno essere processate da un algoritmo noto come *Sensor Fusion Algorithm* (SFA). [11]

2

ARCHITETTURA DI SISTEMA

In questo capitolo viene descritta l'architettura *hardware* del CPS in esame, in particolare, ne vengono evidenziati i *Constituent System* (CS) e le loro interfacce di comunicazione. Vengono infine descritte le interazioni alle particolari interfacce del sistema.

2.1 DESCRIZIONE GENERALE

Lo scopo del sistema è quello di implementare un meccanismo di posizionamento basato su SFA.

Il software che esegue tale algoritmo è schematizzabile come una *black-box* (figura 5), la quale prende in ingresso un certo insieme di misure, e fornisce in uscita una stima affidabile della posizione del treno lungo la traccia. [12]

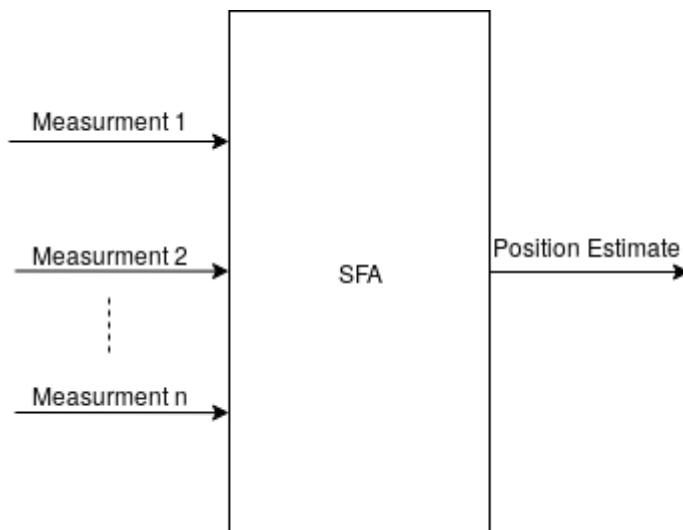


Figura 5: Schema SFA

Questo algoritmo verrà eseguito su di un hardware installato a bordo treno, ed ha lo scopo di monitorare costantemente il moto dello stesso. Le grandezze fisiche che dovranno essere misurate e fornite a SFA sono:

- Vettore accelerazione;
- Vettore velocità angolare;
- Coordinate geografiche;
- Velocità lineare (scalare).

In quest'applicazione, SFA utilizza queste informazioni in combinazione con un'apposita digitalizzazione della traccia tramviaria su cui si trova il treno monitorato.[13][14][15]

Queste informazioni si suppongono note a priori ed accedibili tramite un *database* caricato in memoria centrale. [16]

2.2 CONSTITUENT SYSTEMS

Il sistema studiato si compone dei seguenti CS:

- *Sensor Set*, ossia un insieme di sensori atto a campionare le misure di interesse per il sistema. Il *Sensor Set* è composto dai seguenti moduli:
 - *Inertial Measurement Unit* (IMU):
Unità incaricata di trasmettere al sistema i vettori accelerazione (\mathbf{a}) e velocità angolare (\mathbf{v}_{ang}). Le misure di IMU sono prese rispetto alla Terra e sono espresse in unità stabilite dallo standard internazionale (SI):

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

Esso è il sensore principale. Date le caratteristiche intrinseche del particolare SFA utilizzato, ossia un *Filtro di Kalman*, il sistema potrebbe funzionare anche senza i rimanenti sensori. Si osserverebbe tuttavia un calo delle performance in termini di errore commesso sulla stima della posizione del treno. [17] [18]

- Odometro:
Unità incaricata di fornire al sistema i campionamenti dei valori di velocità lineare del treno, espressi in $\frac{\text{m}}{\text{s}}$.

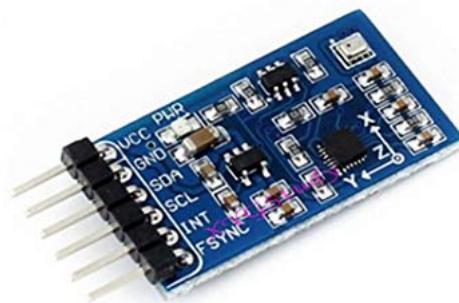


Figura 6: *Inertial Measurement Unit*

- GPS:

Unità che fornisce al sistema le misure di posizione del treno. 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 sul livello del mare.



Figura 7: Ricevitore GPS ublox EVK-M8T

- Piattaforma di elaborazione dati. Consiste di una scheda Nvidia TX-Jetson su cui viene eseguito SFA.



Figura 8: Nvidia TX-Jetson

- *On Board Control Unit* (OBCU). Computer di bordo del treno. Esso non svolge alcun ruolo attivo nel sistema di posizionamento, tuttavia la progressiva chilometrica, stimata da SFA, dovrà essere trasmessa a OBCU al fine di poter utilizzare questa informazione all'interno del sistema di *interlocking* della traccia.

2.3 SPECIFICA DELLE INTERFACCE

2.3.1 Relied Upon Interfaces

Le interfacce sono definite come punti di interazione, tra un CS e l'ambiente oppure tra un CS e un altro.

In questa sezione si evidenziano le principali interfacce del sistema, alle quali si osservano le interazioni fondamentali che avvengono al suo interno.

Tali interfacce prendono il nome di *Relied Upon Interfaces* (RUI). Le RUI si dividono in:

- *Relied Upon Physical Interfaces* (RUPI), in cui l'interazione avviene tramite osservazione diretta di una grandezza fisica;
- *Relied Upon Message Interfaces* (RUMI), dove l'interazione è rappresentata da uno scambio di messaggi a livello *cyber*.

La specifica delle RUI è di particolare importanza poiché qualunque struttura del sistema, responsabile del comportamento osservato, può essere ridotta alla specifica delle interfacce del sistema. [19].

Il CPS interagisce con l'ambiente attraverso le RUPI del *Sensor Set*, ossia gli strumenti di misura che esso integra. Queste interfacce acquisiscono, a diverse frequenze, i dati sul moto del treno che verranno elaborati dal resto del sistema di posizionamento (tabella 1).

Per quanto concerne le RUMI, se ne osservano di due tipi:

| RUPI | Grandezza Campionata | Parti interagenti |
|----------------|------------------------|---------------------|
| Accelerometro | Accelerazione | Ambiente - IMU |
| Giroscopio | Velocità angolare | Ambiente - IMU |
| Radar | Velocità lineare | Ambiente - Odometro |
| Ricevitore GPS | Coordinate geografiche | Ambiente - GPS |

Tabella 1: Specifica delle RUPI del sistema

- Tre bus dati, che collegano il *Sensor Set* alla scheda Nvidia TX-Jetson. Su ciascuno di essi, *Sensor Set* invia rispettivamente messaggi contenenti i dati campionati da IMU, Odometro e GPS.
- Interfaccia LTE. Essa permette di realizzare una *rete wireless ad hoc* fra la scheda e OBCU.
All'interno di tale rete vengono instradati datagrammi IP contenenti le informazioni sulla progressiva chilometrica stimata da SFA, ed eventualmente messaggi di *acknowledgment* di OBCU verso la scheda.



Figura 9: Modem TP-LINK M7350 LTE-4G

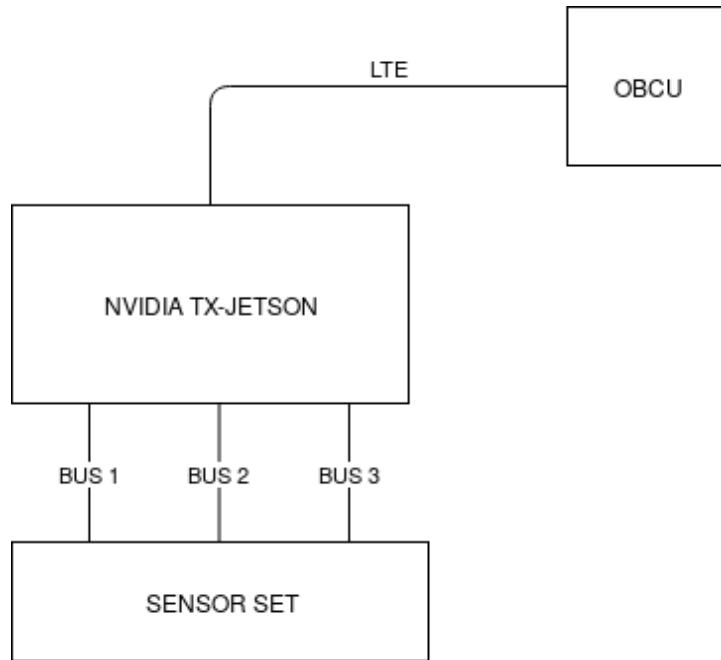


Figura 10: RUMI

2.3.2 Altre Interfacce

Oltre alle RUI, descritte in 2.3.1, esistono altre interfacce che hanno lo scopo di rendere il sistema osservabile e manutenibile, e sono le seguenti: [20]

- *Time Synchronization Interfaces* (TSI). Le TSI permettono al CPS di effettuare una sincronizzazione col tempo fisico al fine di stabilire una *global timebase* [21].
- *Utility Interfaces* (UI). Interfacce dei CS che ne consentono la configurazione, il controllo, e l'osservazione non intrusiva del suo comportamento [22].

Come verrà approfondito nel successivo capitolo, sia le TSI che le UI sono nella fattispecie interfacce *software*.

2.4 INTERAZIONI

In questa sezione vengono descritte le interazioni osservabili alle interfacce del sistema.

2.4.1 Acquisizione dei dati

L'acquisizione dei dati si divide in due differenti interazioni: la prima, con l'ambiente, avviene alle RUPI del *Sensor Set*, mentre la seconda avviene alle RUMI di tipo bus dati che collegano il *Sensor Set* alla piattaforma di elaborazione dati. I moduli che compongono il *Sensor Set* campionano ad una data frequenza le grandezze fisiche che descrivono il moto del treno. Ciascun campionamento fisico è seguito dall'invio dei valori letti alla piattaforma di elaborazione dati. I moduli del *Sensor Set* sono tra di loro indipendenti.

In figura 11 viene riportata una sequenza esempio di campionamento e

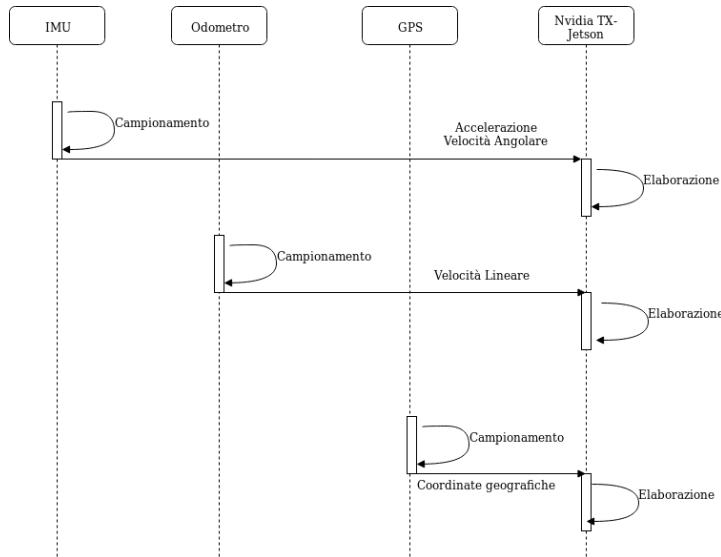


Figura 11: Sequenza di acquisizione dati

invio dei dati. I protocolli di comunicazione tra *Sensor Set* e la piattaforma di elaborazione dati sono definiti a livello *software*, e saranno descritti nel prossimo capitolo. Questa tipologia di interazione è detta *time-triggered*, in quanto è determinata unicamente dallo scorrere del tempo. [23] [24]

2.4.2 Trasmissione della posizione

La piattaforma di elaborazione dati esegue SFA durante l'intero moto del treno. Le misure fornite dai sensori vengono elaborate al fine di aggiornare continuamente la stima della posizione del treno.

Ogniqualvolta un'aggiornamento di SFA viene completato, avviene un'interazione all'interfaccia LTE. Tale interazione consiste nell'invio di un

messaggio contenente la posizione del treno, dalla piattaforma di elaborazione dati verso OBCU, e nella trasmissione di un messaggio di *acknowledgment* nel senso opposto.

La tipologia di scambio dei messaggi esposta è detta *event-triggered* [24] in quanto le tempistiche di interazione non sono note a priori, ma dipendono dal tempo impiegato da SFA a compiere un'iterazione per aggiornare la stima prodotta.

LTE è a tutti gli effetti una regolare interfaccia di rete. A livello di trasporto, il messaggio trasmesso è contenuto nel *payload* di un datagramma UDP; in accordo al modello di rete ISO-OSI. [25]

La specifica del messaggio a livello applicazione sarà descritta nel prossimo capitolo.

3

SPECIFICHE SOFTWARE

In questo capitolo vengono descritte le specifiche software del sistema di posizionamento SFA. Si evidenziano le caratteristiche architetturali dei software che concorrono al raggiungimento dello scopo del sistema, e se ne descrivono i protocolli di comunicazione.

3.1 ARCHITETTURA SOFTWARE

Sulla piattaforma di elaborazione dati è installato il sistema operativo Ubuntu 16.04 LTS, basato su kernel Linux.

Su tale piattaforma viene eseguito il modulo SFA, opportunamente encapsulato in un eseguibile, denominato *listener*.

Un set di moduli software, denominati *interface-modules*, sono in esecuzione sulla scheda. Lo scopo di questo insieme di programmi è quello di funzionare come interfaccia interna verso i sensori del CPS. 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 *listener* è in grado di inviare le misurazioni ricevute da *interface-modules* a SFA, quali variabili di tipo IMU_POD, ODO_POD,

GPS_POD ed altresí 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, utilizzando un protocollo arbitrario a livello applicazione, sia esso OUTPUT_PROTOCOL.

Uno schema dell'architettura software è quello mostrato in figura 12.

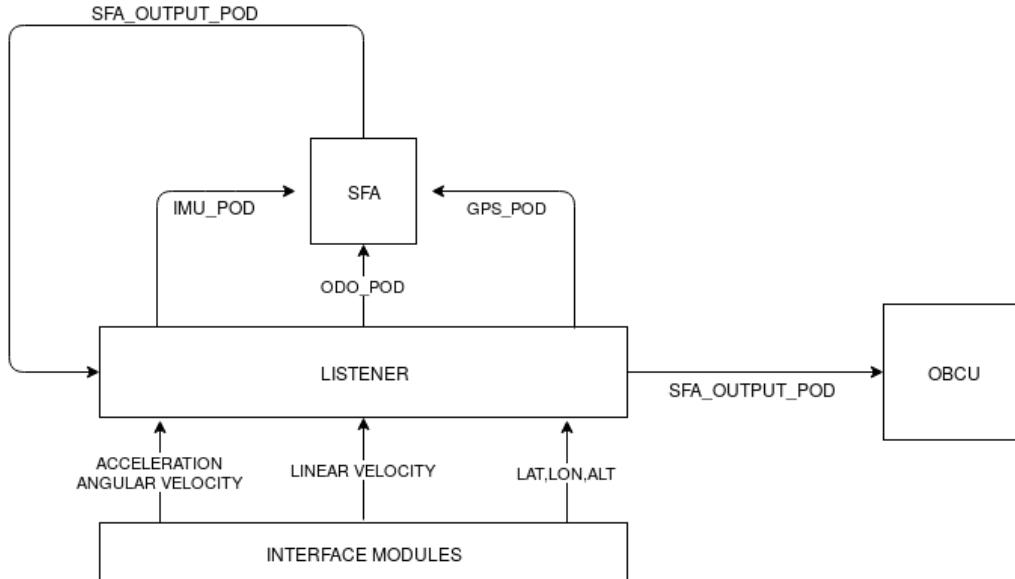


Figura 12: Architettura software bordo treno

3.2 PROTOCOLLI DI COMUNICAZIONE

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 (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.

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.

A discrezione del valore del campo SENSOR_TYPE si distingue il tipo di

| Campo | Descrizione | Indici di bit | Tipo |
|-------------|----------------------------|---------------|----------|
| SENSOR_TYPE | ID Sensore Sorgente | 0-7 | uint8_t |
| Seq.N0 | 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

informazione trasportata dal pacchetto, come descritto in tabella 3.

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

| Valore di SENSOR_TYPE | 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. Per quanto esposto nel Capitolo 2, l'interfaccia UDP attraverso cui comunicano interface-modules e listener funge da TSI per il sistema.

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.N0`. Si osservi che UDP non garantisce che l'ordine dei messaggi ricevuti sia lo stesso con i quali essi sono stati inviati. I messaggi potrebbero essere soggetti a ritardi casuali in base allo stato del sistema operativo. 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`.

Come specificato, la comunicazione è posta in essere, a livello fisico, attraverso il protocollo `LTE`, 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 conseguente rischio di ricevere alcuni messaggi in maniera errata, o non riceverli del tutto, è tanto più elevato quanto l'ambiente entro cui si propaga fisicamente la radiazione elettromagnetica è più disturbato da sorgenti esterne e corpi fisici interposti.

Questa problematica è risolta a livello software, 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. 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 |
|-------------------------|--------------------------|---------------|-----------------------|
| <code>Seq.N0</code> | Numero di sequenza | 0-15 | <code>uint16_t</code> |
| <code>Epoch</code> | Timestamp | 16-79 | <code>double</code> |
| <code>FU_ARC_LEN</code> | Progressiva chilometrica | 80-143 | <code>double</code> |

Tabella 4: Protocollo di comunicazione in uscita

In ricezione, OBCU dovrà inviare un pacchetto *ack* al mittente, ed il suo formato è descritto in tabella 5. Si osservi che, ai fini del posizionamento,

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

Tabella 5: Formato del pacchetto di *ack*

l'ordinamento dei pacchetti ricevuti non è fondamentale. A differenza di quanto esposto nel caso della trasmissione dei dati in entrata a SFA, è sufficiente che OBCU faccia riferimento al messaggio con il *timestamp* più recente.

3.3 SCENARIO DI ESEMPIO

Si suppongano le condizioni iniziali riportate in tabella 6.

| Vettore Velocità | Progressiva | IMU Sample Rate | ODO Sample Rate |
|---------------------------------|-------------|-----------------|-----------------|
| (0.0, 0.0, 0.0)ms ⁻¹ | 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 | 4 |

E vi accoda i seguenti tre valori double: 0.0, 0.0, 0.0, ossia il vettore velocità iniziale, ed il valore double 0.0, che rappresenta la progressiva chilometrica iniziale.

- `listener` riceve il pacchetto e inizializza SFA con:
 - Velocità iniziale: (0, 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:

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

Il pacchetto viene inviato a `interface-modules` attraverso i bus dati, e conseguentemente:

- `interface-modules` invia a `listener`, attraverso la socket UDP, il seguente pacchetto IMU:

| <code>SENSOR_TYPE</code> | <code>Seq. NO</code> | <code>N_INT</code> | <code>N_DOUBLE</code> |
|--------------------------|----------------------|--------------------|-----------------------|
| 0x01 | 0x01 | 0 | 6 |

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

- `listener` riceve il pacchetto, crea e inoltra 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 i dati ricevuti e inizia una computazione parallela per fornire a `listener` una variabile `SFA_OUTPUT_POD` della forma:

- `Seq.NO` = 0
- `FU_ARC_LEN` = P_{KM}

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

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

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

- Odometro campiona, e invia a `interface-modules`, il seguente valore di velocità:

$$v = (1.0010)$$

`interface-modules` invia a `listener` il seguente pacchetto ODOMETRO:

| SENSOR_TYPE | Seq. NO | N_INT | N_DOUBLE |
|-------------|---------|-------|----------|
| 0x02 | Seq_N0 | 0 | 2 |

Accordandovi 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 i dati ricevuti e utilizza la rilevazione di velocità in maniera utile a migliorare la stima della posizione.

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

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_N0 | 0 | 1 |

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

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

Nonappena la prima uscita di SFA si rende disponibile a `listener` questo si comporta come segue:

- `listener` riceve la variabile `SFA_OUTPUT_POD`, da SFA;
- Supponendo di trovarsi al tempo T , `listener` costruisce ed invia ad OBCU il seguente pacchetto:

| Seq. NO | Epoch | FU_ARC_LEN |
|---------|-------|------------|
| 0x00 | T | P_{KM_T} |

- OBCU riceve il pacchetto e invia a `listener` l'*ack* 0x00.

4

AMBIENTE DI ANALISI

Al fine di effettuare adeguate campagne di analisi, è stato progettato un ambiente che permetta l’osservazione non intrusiva del sistema descritto. In questo capitolo, si individuano e descrivono due differenti configurazioni di analisi:

- Configurazione *online*, atta a osservare il comportamento del sistema alle RUI;
- Configurazione *offline*, atta a osservare esclusivamente il comportamento del modulo SFA con dati simulati o riprodotti dal campo.

4.1 CONFIGURAZIONI

4.1.1 *Configurazione Online*

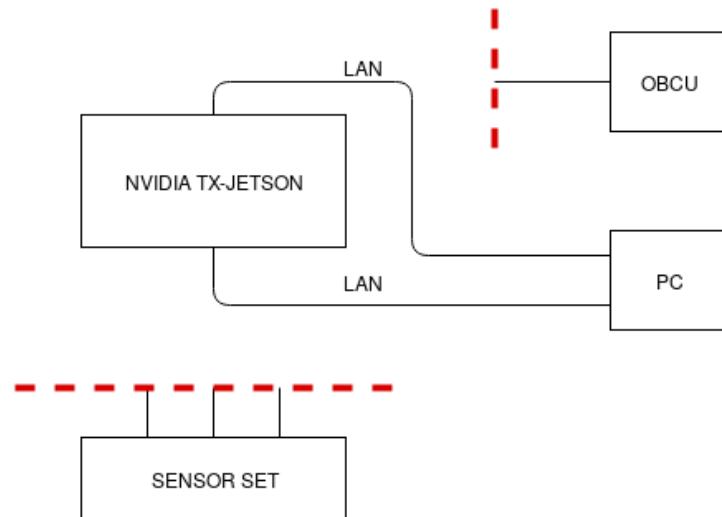
Partendo dall’architettura nominale descritta nel capitolo 2, si osservano le seguenti modifiche:

- *Sensor Set* viene rimpiazzato da un PC in grado di interfacciarsi con la piattaforma di elaborazione dati attraverso una LAN;
- OBCU viene rimpiazzato dallo stesso PC.

Lo schema dell’architettura descritta è riportato in figura 13.

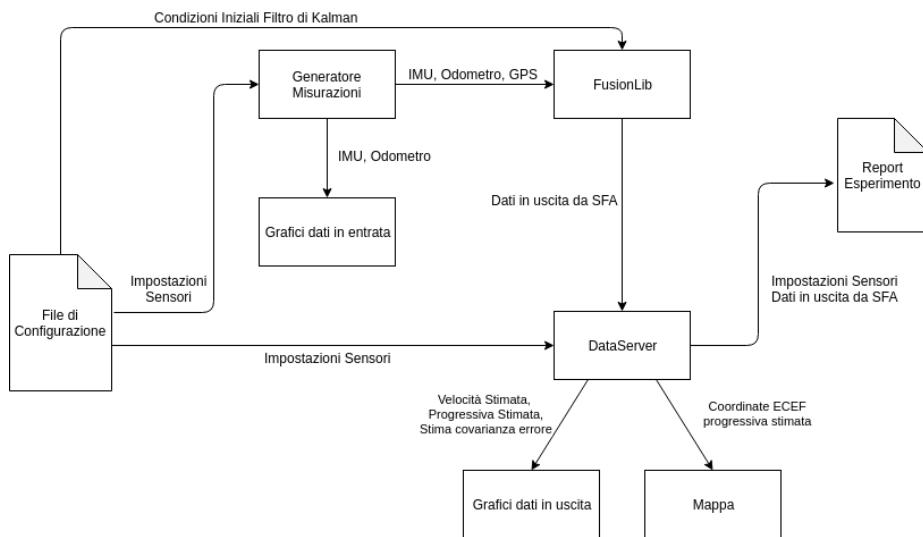
A livello di RUMI, i bus dati e il collegamento LTE sono stati rimpiazzati da una connessione LAN, ma permangono inalterate le interazioni ivi osservabili.

Le modifiche descritte permettono di osservare variazioni nel comportamento del sistema quando sottoposto a campagne di *fault injection*, simulando ad esempio guasti hardware nei bus dati, oppure perdite di segnale LTE.

Figura 13: Configurazione *online*

4.1.2 Configurazione *Offline*

Questa modalità di analisi prevede l'esecuzione del modulo SFA all'interno di un eseguibile Windows. Tale eseguibile prende il nome di Rail Track Tool (RTT), che include il modulo SFA come una classica dipendenza verso una libreria C++, sia essa FusionLib.

Figura 14: Configurazione *offline*

Questa configurazione permette di valutare il comportamento di SFA al variare delle caratteristiche dei dati in ingresso, come ad esempio il numero di sensori abilitati o la covarianza del rumore insito nel processo di misura. [26]

5

ESPERIMENTI E RISULTATI

6

CONCLUSIONI

BIBLIOGRAFIA

- [1] J. Otegui, *A Survey of Train Positioning Solutions*, *IEEE Sensors Journal*, Vol. 17, No. 20 (2017) (Cited on page 10.)
- [2] S. Busard et al, *Verification of railway interlocking systems*, Université catholique de Louvain, Louvain-La-Neuve, Belgium (Cited on page 10.)
- [3] T. Albrecht et al, *A precise and reliable train positioning system and its use for automation of train operation*, Proc. IEEE Int. Conf. Intell. Rail Transp. (2013) (Cited on page 10.)
- [4] CENELEC European Committee for Electrotechnical Standardization. *EN 50128:2011 - Railway Applications - Communications, signalling and processing systems - Software for railway control and protection systems* (2011) (Cited on page 10.)
- [5] MISRA, *MISRA-C:2004, Guidelines for the use of the C language in critical systems* (2004) (Cited on page 10.)
- [6] A. Bonacchi et al, *Validation Process for Railway Interlocking Systems*, DINFO, University of Florence, Via S. Marta 3, Firenze, Italy (Cited on page 10.)
- [7] N.R. Storey, *Safety Critical Computer Systems*, Addison-Wesley, Boston, MA, USA (1996) (Cited on page 10.)
- [8] International Electrotechnical Commission, *61508-1: Functional safety of electrical/electronic/programmable electronic safety-related systems, edition 2.0* (2010) (Cited on page 11.)
- [9] N.A. Zafar et al, *Towards the safety properties of moving block railway interlocking system*, *International Journal of Innovative Computing Information and Control* (2012) (Cited on page 11.)
- [10] D. Basile et al, *A Refinement Approach to Analyse Critical Cyber-Physical Systems*, *Software Engineering and Formal Methods* (2017) (Cited on page 13.)
- [11] A. Mirabadi et al, *Application of sensor fusion to railway systems*, *IEEE* (1996) (Cited on page 13.)

- [12] F. Bohringer, A. Geistler, *Adaptation of the kinematic train model using the interacting multiple model estimator*, *Advances in Transport*, vol. 74, no. 7. Southampton, (2004) (Cited on page 15.)
- [13] B. Cai, X. Wang, *Train positioning via integration and fusion of GPS and inertial sensors*, *WIT Transactions on the Built Environment*, Southampton (2000) (Cited on page 16.)
- [14] M. Malvezzi et al, *A localization algorithm for railway vehicles based on sensor fusion between tachometers and inertial measurement units*, *Proc. Inst. Mech.* (Cited on page 16.)
- [15] C. Reimer et al, *INS/GNSS/odometer data fusion in railway applications*, *Proc. DGON Intertial Sensors Syst. (ISS)*, vol. 2. Karlsruhe, Germany (2016) (Cited on page 16.)
- [16] L. Junyan et al, *Application Research of Embedded Database SQLite*, IEEE (2009) (Cited on page 16.)
- [17] X. Liu, A. Goldsmith, *Kalman Filtering with Partial Observation Losses*, Department of Electrical Engineering, Stanford University, Stanford, USA (Cited on page 16.)
- [18] R. Mazl, L. Preucil, *Sensor Data Fusion for Inertial Navigation of Trains in GPS Dark Areas (Mathematics in Science and Engineering)*, San Diego, CA, USA (2003) (Cited on page 16.)
- [19] H. Kopetz, *Real-Time Systems: Design Principles for Distributed Embedded Applications 2nd edn*, Springer, New York (2011) (Cited on page 19.)
- [20] A. Ceccarelli et al, *Basic Concepts on Systems of Systems, Cyber-Physical Systems of Systems*, Springer (2017) (Cited on pages 13 and 20.)
- [21] H. Kopetz, W. Ochsenreiter, *Clock synchronization in distributed real-time systems*, IEEE (1987) (Cited on page 20.)
- [22] K. Wolter et al, *Resilience Assessment and Evaluation of Computing Systems*, Springer (2012) (Cited on page 20.)
- [23] M.J. Pont, *Patterns for Time-Triggered Embedded Systems*, Addison-Wesley (2001) (Cited on page 21.)
- [24] H. Kopetz, *Event-Triggered versus Time-Triggered Real-Time Systems*, Springer (1991) (Cited on pages 21 and 22.)

- [25] J.F Kurose, K.W Ross, *Computer Networking: A Top-Down Approach*, Pearson (2013) (Cited on page 22.)
- [26] S. Dilhaire, D. Maillet, *Dealing with the measurement noise of a sensor* (2015) (Cited on page 33.)