

UNIVERSITÀ DEGLI STUDI DI TRENTO

---

DIPARTIMENTO DI INGEGNERIA INDUSTRIALE  
CORSO DI LAUREA IN INGEGNERIA MECCATRONICA

**SVILUPPO E VALIDAZIONE  
Sperimentale di algoritmi di  
localizzazione e controllo di  
un veicolo autonomo**

*Relatore:*

Prof. FRANCESCO BIRAL

*Candidato:*

JACOPO MOLINAROLI

*Correlatore:*

MATTEO RAGNI

---

Anno Accademico 2015 – 2016



*“I’ve missed more than 9000 shots  
in my career. I’ve lost almost 300 games.  
26 times, I’ve been trusted to take  
the game winning shot and missed.  
I’ve failed over and over and over again  
in my life. And that is why I succeed.”*

*Michael Jeffrey Jordan*



# Abstract

I veicoli a guida autonoma si stanno affacciando sul mercato *consumer*, ma molti degli algoritmi di controllo più avanzati sono ancora in fase sperimentale e richiedono piattaforme di test realistiche, ma pratiche. In questa tesi svolta nei dei laboratori di Meccatronica del Dipartimento di Ingegneria Industriale dell’Università degli Studi di Trento, si realizza un veicolo autonomo e si approfondiscono i concetti di *sensor fusion* nel campo della localizzazione del veicolo. Le tecniche presentate prevedono di fondere i risultati forniti da un *Global Positioning System Indoor* (GPS Indoor), una *Inertial Measurement Unit* (IMU) e degli *Odometry Sensors* (Encoders) mediante un Filtro di Kalman Esteso (EKF). Gli algoritmi sviluppati sono validati sperimentalmente sottoponendo il veicolo a traiettorie rettilinee, circolari e di *lane change*. Per concludere viene presentato il comportamento del veicolo durante una prova su un circuito ovale.



# **Ringraziamenti**

Desidero ringraziare il mio relatore Prof. Francesco Biral per avermi dato l'opportunità di lavorare su questo progetto ed essere stato sempre disponibile e cordiale nei miei confronti. Inoltre ringrazio Matteo Ragni, correlatore, amico e collega di lunga data, per avermi seguito e aiutato anche in questo lavoro. Vorrei quindi esprimere la mia gratitudine a Son Tran e a Luca De Pascali con i quali ho avuto modo di confrontarmi, scambiare idee, opinioni e consigli nel corso di questo progetto di tesi.

Voglio ringraziare i miei genitori Gerardo ed Antonella che mi hanno permesso di raggiungere anche questo traguardo e mia sorella Morgana. Un ringraziamento speciale va a Giulia, per supportarmi e sopportarmi costantemente da tanto tempo, e a tutta la sua famiglia. Un grazie sincero anche a tutti gli amici che ho trovato qui a Trento, i miei colleghi universitari (su tutti un grazie di cuore a David, Giamma, Dela e Scardo), i miei compagni di squadra del Paganella Lavis (più forti nello spogliatoio che su un campo da basket) e a tutti i miei coinvolti presenti e passati dell'interno 7.

*Trento, Marzo 2017.*



# Indice

<b>Abstract</b>	<b>i</b>
<b>Ringraziamenti</b>	<b>iii</b>
<b>Introduzione</b>	<b>1</b>
Stato dell'arte . . . . .	1
Obbiettivi . . . . .	7
<b>1 Il veicolo eRumby</b>	<b>11</b>
1.1 Caratteristiche generali . . . . .	11
1.2 Attuatori . . . . .	14
1.2.1 Motore di trazione . . . . .	14
1.2.2 Servomotore . . . . .	16
1.2.3 Ricevente . . . . .	16
1.3 Sensori . . . . .	18
1.3.1 Encoder . . . . .	18
1.3.2 Inertial Measurement Unit . . . . .	20
1.3.3 Global Positioning System . . . . .	22
1.4 Schede di controllo . . . . .	25
1.4.1 Arduino Mega . . . . .	25

1.4.2	XBee	26
1.4.3	BeagleBone Black	28
<b>2</b>	<b>Architettura software del veicolo</b>	<b>31</b>
2.1	Architettura della rete	31
2.2	Moduli software	33
2.3	Arduino Mega	33
2.3.1	Lettura encoder	33
2.3.2	Lettura IMU	35
2.3.3	Lettura GPS Indoor	36
2.3.4	Lettura ricevente	37
2.3.5	Gestione dei motori	37
2.4	BeagleBone	38
2.4.1	Salvataggio e scambio dati	39
2.4.2	Data fusion	41
2.4.3	Controllore della velocità	42
2.4.4	Controllo	43
<b>3</b>	<b>Stato del veicolo</b>	<b>47</b>
3.1	Sensor Fusion	47
3.2	Filtro di Kalman discreto	52
3.3	Filtro di Kalman Esteso	55
3.4	Filtro di Kalman Unscented	57
3.5	Ricostruzione dello stato del veicolo	58
3.5.1	Definizione del sideslip angle	61

<b>4 Prove sperimental</b>	<b>63</b>
4.1 Analisi dei dati . . . . .	63
4.1.1 Prova in condizioni stazionarie . . . . .	64
4.1.2 Prova su traiettoria circolare . . . . .	66
4.1.3 Prova su traiettoria rettilinea . . . . .	73
4.1.4 Prova di “lane change” . . . . .	78
4.1.5 Prova su circuito . . . . .	83
4.2 Prove sul filtro di Kalman . . . . .	86
<b>Conclusioni e sviluppi futuri</b>	<b>95</b>
<b>Appendice</b>	<b>99</b>
Assemblaggio del veicolo . . . . .	99
<b>Bibliografia</b>	<b>111</b>



# Introduzione

## Stato dell'arte

L'idea di veicoli manovribili a distanza e di veicoli in grado di muoversi autonomamente solletica da molto la fantasia dell'uomo. Un primo prototipo si fa risalire già al genio di Leonardo da Vinci diversi secoli prima dell'invenzione dell'automobile. Sfruttando molle sottoposte a una elevata tensione, Leonardo progettò un carrello in grado di muoversi e sterzare lungo un percorso predeterminato senza essere spinto o tirato[5].

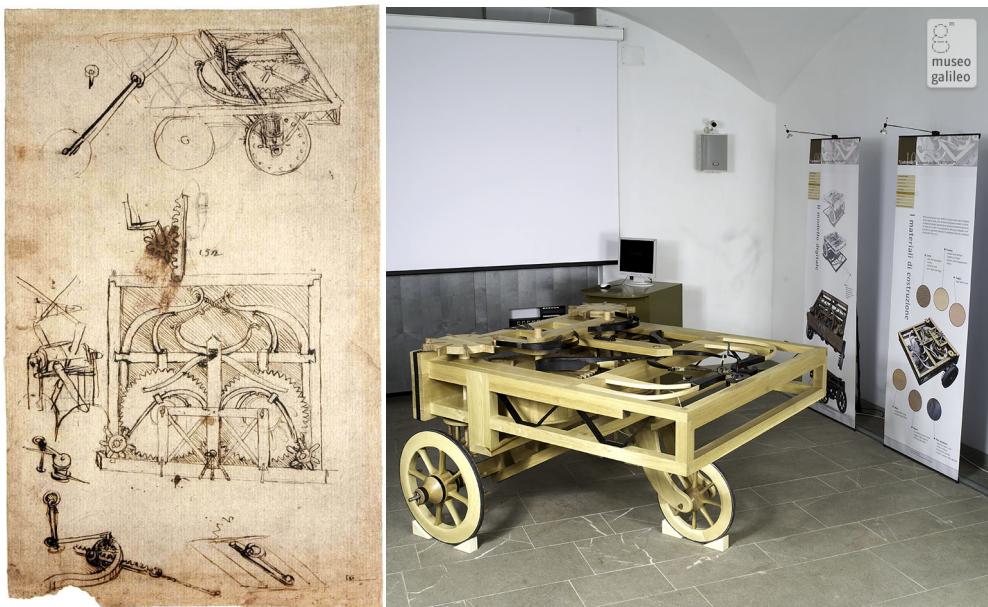


Figura 1: *Carrello di Leonardo da Vinci.*  
A sinistra il progetto originale e a destra la replica presente al Museo Galileo (FI),  
<http://www.museogalileo.it/>.

## Introduzione

---

In epoca moderna, una importante svolta si ebbe nella seconda metà del 1800 quando venne costruito il Whitehead Torpedo, un siluro in grado di muoversi sott'acqua mantenendo una profondità costante[20]. Risale invece al 1925 il primo tentativo di realizzare una auto senza conducente guidata da remoto. L'azienda statunitense Houdina Radio Control rielaborò una Chandler, chiamata Phantom Auto, dotandola di una antenna radio in grado di captare gli impulsi inviati da un operatore che controlla in remoto il veicolo<sup>1</sup>. Un altro passo importante si ebbe nel 1933 quando venne testato il primo sistema di autopilota per aerei a lungo raggio. Noti i piani di volo del velivolo, il sistema Mechanical Mike era in grado di mantenere una rotta costante e ben precisa facendo uso di semplici giroscopi[2]. Solo dopo la Seconda Guerra Mondiale, negli anni 50, si ebbe prima lo sviluppo del primo *cruise control*, capace di mantenere la velocità desiderata attraverso un acceleratore meccanico[17], e successivamente lo studio dei primi sistemi di guida automatizzata mediante sensori lungo le carreggiate delle strade, capaci di controllare acceleratore e freno dell'automobile[13]. Lo sviluppo del primo veicolo auto-guidato avvenne nel 1961 sotto il nome di The Stanford Cart. In piena corsa allo spazio i ricercatori iniziarono lo studio di mezzi da sbarco lunari radio-comandati dovendo affrontare però il problema del ritardo di comunicazione tra Terra e Luna di circa 2.5 secondi. Per questo motivo The Stanford Cart venne dotato di telecamera e programmato per predire e seguire una determinata traiettoria riferendosi alle ultime coordinate ricevute[6]. Avvicinandoci ai giorni nostri, negli anni '80 e '90 le precedenti soluzioni vennero perfezionate in veicoli sperimentali come VaMoRs (1984), un van Mercedes-Benz in grado di procedere senza pilota e rielaborare le informazioni sulla sua posizione e la traiettoria mediante l'ausilio di sensori e telecamere[7], oppure il prototipo Argo (1998), una Lancia Thema modificata che, mediante un sistema di analisi dei dati dell'ambiente esterno fu in grado di percorrere duemila chilometri in sei giorni in quasi totale autonomia[4]. Con l'ingresso nel nuovo millennio, l'interesse verso i veicoli a guida autonoma crebbe esponenzialmente. Tra il 2004 e il 2013 il *Defense Advanced Research Projects Agency* organizzò cinque edizioni del *DARPA Grand Challenge*, un concorso a premi per veicoli autonomi creato per stimolare lo sviluppo delle tecnologie necessarie per la realizzazione di veicoli completamente autonomi. La terza edizione è conosciuta anche con il nome di *DARPA Urban Challenge 2007*, la quarta come *DARPA Robotics Challenge 2012* e la quinta come *DARPA FANG Challenge 2013*, dove l'acronimo FANG si riferisce

---

<sup>1</sup> "Phantom Auto' to Be Operated Here", The Free-Lance Star, 17 giugno 1932. Google News Archive, 14 Settembre 2013.

a *Fast Adaptable Next-Generation Ground Vehicle*. Attualmente tutte le maggiori *stakeholder* nel campo *automotive* sono impegnate nello sviluppo di veicoli robotizzati. Nel 2015 l'azienda statunitense Tesla Motors Inc. ha lanciato sul mercato il Tesla Autopilot, un sistema di assistenza alla guida<sup>2</sup>, mentre nel maggio del 2016 Google ha siglato un accordo con il gruppo FCA secondo il quale, dopo una prima fase di test, produrrà il primo modello commerciale di auto a guida autonoma che sfrutta la tecnologia di Waymo (ex progetto Google *driverless car*)[16]. Questo livello di sviluppo è stato possibile grazie a sensori avanzati dediti alla raccolta di informazioni, ad algoritmi per la rielaborazione dei dati sempre più sofisticati e l'elevata potenza di calcolo utilizzabile al giorno d'oggi. Per percepire l'ambiente circostante i veicoli autonomi usano una combinazione di sensori quali radar - i.e. ad ultrasuoni, infrarossi o laser - e sistemi di acquisizione video - i.e. video camere e sensori di luminosità. La *sensor fusion* dei diversi ingressi sensoriali amplia le possibilità percettive del veicolo, sopperendo alle debolezze del singolo sensore. Tali sistemi riescono ad accumulare informazioni con robustezza, ma la analisi e la selezione delle azioni per lo sviluppo di un veicolo affidabile rimane tuttora un problema aperto a livello di ricerca.

---

<sup>2</sup>"Autopilot - Tesla", <https://www.tesla.com/presskit/autopilot> .

## Introduzione

---

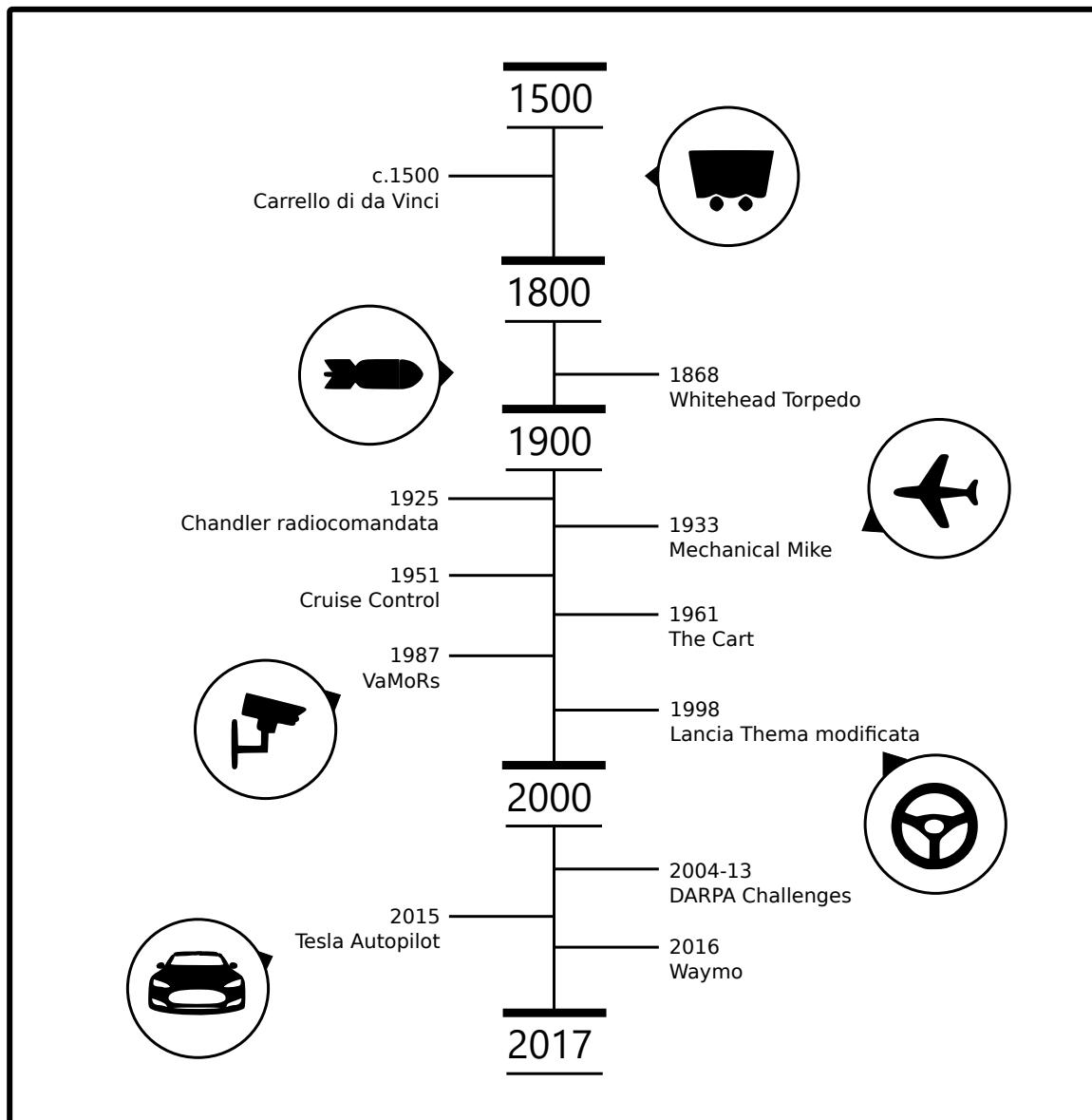


Figura 2: *Infografica dell'evoluzione dei veicoli a guida remota e autonoma.*  
«A brief history of autonomous vehicle technology», WIRED, marzo 2016.

I sistemi a guida autonoma hanno la possibilità di apportare un cambio fondamentale nel mondo del trasporto e proprio per questo è stato pubblicato nel 2014 dalla *Society of Automobile Engineers International* (SAE International) un sistema di classificazione basato su sei diversi livelli:

- Livello 0. Il sistema automatizzato non ha alcun controllo sul veicolo, ma può fornire avvertimenti al guidatore.
- Livello 1. Il conducente deve essere pronto a prendere il controllo in ogni istante. Il sistema automatizzato può includere caratteristiche come il sistema *Adaptive Cruise Control* (ACC), il sistema di *Parking Assistance* (PA) con sterzo automatizzato e il *Lane Keeping Assistance* (LKA).
- Livello 2. Il conducente è obbligato ad individuare oggetti e possibili eventi particolari e a intervenire se il sistema non risponde correttamente. Il sistema autonomo è in grado di accelerare, frenare e sterzare e si disattiva immediatamente nel momento in cui il conducente prende il controllo del veicolo.
- Livello 3. In ambienti limitati e noti il guidatore può distogliere in modo sicuro l'attenzione dalle comuni azioni di guida, ma deve essere comunque pronto ad intervenire qualora ce ne fosse bisogno.
- Livello 4. Il sistema autonomo è in grado di controllare il veicolo in quasi tutte le condizioni ambientali ed è cura del guidatore attivare il sistema solo nelle condizioni in cui è sicuro farlo e, quando attivo, non è richiesto alcun suo intervento.
- Livello 5. Non sono necessari interventi umani se non l'impostazione della destinazione e l'avvio del sistema. Il sistema autonomo può guidare il veicolo in completa autonomia, dove è legale farlo, e prendere decisioni in modo del tutto autonomo.

La tecnologia alla base della guida autonoma ha la potenzialità di influenzare in modo determinante la sicurezza, la congestione e l'uso delle strade. Ad esempio, ad ogni guidatore che si mette alla guida aumenta il traffico sulla strada per gli altri guidatori e aumenta la probabilità che un altro guidatore possa avere un incidente. Inoltre è noto come le introduzioni di sistemi di frenata al rilevamento di ostacoli, di sistemi di avviso di invasione della corsia o di sistemi di allerta per gli angoli di visuale ciechi abbiano permesso di ridurre in

modo drastico il numero di incidenti[15]. Un ultimo aspetto da non trascurare è la possibilità di aumentare la mobilità di quelle persone che non sono abili alla guida o che non si sentono sicure alla guida. Ad esempio, un sistema al livello 4 della scala SAE International permette il trasporto senza bisogno di accompagnatori di persone persone disabili o non abilitate alla guida. Negli ultimi anni la tecnologia alla base dei veicoli autonomi ha fatto passi da gigante, ma non è ancora matura poiché è impiegabile solo in situazioni note e usando sensori ancora molto costosi. A riguardo Elon Musk, CEO della Tesla Motors Inc., riassume la situazione con queste parole: *«It's much easier problem than people think it is. ... But it's not a one-guy-three-months problem. It's more like thousands of people for two year.»*<sup>3</sup>. I principali ostacoli alla diffusione di veicoli autonomi sono:

- La diffidenza dei consumatori. Sono molte le persone contrarie all'idea di non avere alcun controllo del veicolo durante la guida, come sono molte le persone che non hanno fiducia sull'attuale livello di affidabilità di questi strumenti.
- I tempi di transizione. Per parlare di sistema completamente autonomo il 100% dei veicoli deve essere in grado di muoversi sulle strade in modo autonomo per il 100% del tempo. Secondo la ricerca condotta dall'*Institute for Housing and Urban Development Studies* (IHS), la vendita dei primi veicoli autonomi avverrà nel attorno al 2025, nel 2035 circa il 10% delle nuove vetture saranno autonome e per il 2050 quasi tutti i nuovi veicoli venduti saranno autonomi, con però strade percorse da veicoli a guida manuale con una percentuale che oscillerà tra il 40% e il 60% [19].
- Il *mixed driving mode*. Visti i lunghi tempi di transizione, si assisterà ad un passaggio graduale da veicoli a guida manuale a veicoli autonomi. In questa fase intermedia si avrà l'interazione tra uomo e macchina su più livelli: quello tra il guidatore e veicolo più o meno intelligente e quello tra veicoli guidati esclusivamente dall'uomo e veicoli autonomi. L'interazione e la comprensione tra questi due mondi risulta poco sicura e difficile da affrontare dal punto di vista tecnologico.
- La necessità di avere sistemi di sicurezza adeguati. In caso di malfunzionamenti o di situazioni nelle quali il veicolo autonomo non è in grado di agire risulta fonda-

---

<sup>3</sup>«E' un problema molto più semplice di quello che la gente pensi. ...Ma non è un problema risolvibile da una persona in tre mesi. Si tratta di un problema che deve essere affrontato da migliaia di persone per due anni.» [3]

mentale la applicazione di nuove norme di sicurezza e la ricerca di nuove soluzioni per limitare i danni. Un tragico esempio è il primo incidente mortale avvenuto nel maggio 2016 con una Tesla S mentre il sistema autopilota era attivo<sup>4</sup>.

- I problemi etici in caso di sinistro. Il primo problema è legato alla responsabilità oggettiva in caso di incidente, se dei passeggeri o dei produttori, mentre il secondo è legato alla fase decisionale del veicolo autonomo costretto tra due eventi catastrofici. Alcuni esempi possono essere la scelta tra preservare la vita dei passeggeri e minimizzare gli effetti dello schianto, oppure dover preservare la sicurezza di una persona anziché di un'altra secondo qualche scala di valori.
- I problemi legali in caso di sinistro. Nel settore se ne discute da diversi anni, ma inevitabilmente l'avvento in grande massa dei veicoli senza conducente porterà ad una rivoluzione completa del mondo legislativo stradale e di quello assicurativo.

## Obbiettivi

A partire dal 2007, il Dipartimento di Ingegneria Industriale dell'Università degli Studi di Trento ha iniziato la realizzazione di un primo prototipo di veicolo radiocomandato denominato RUMBy, caratterizzato da un motore a combustione interna e successivamente un secondo prototipo di veicolo radiocomandato dotato di motore elettrico, denominato eRumby. L'obiettivo principale dei prototipi realizzati è quello di simulare in scala il comportamento di un veicolo reale e permettere così lo studio e lo sviluppo di nuovi algoritmi rivolti alla guida autonoma.

---

<sup>4</sup>"Tesla driver dies in first fatal crash while using autopilot mode ", The Guardian, 30 giugno 2016.



Figura 3: *Prototipo RUMBy*.

In questo lavoro di tesi si vuole dunque realizzare un sistema per la localizzazione e il tracciamento di eRumby dotandolo di molteplici sensori a basso costo e accuratezza simile a quella usata nei veicoli commerciali. Come per la maggior parte dei veicoli senza conducente, il prototipo vede l'utilizzo di un *Global Positioning System Indoor* (GPS Indoor) assieme ad una *Inertial Measurement Unit* (IMU) e degli *Odometry Sensors* (Encoders) in *sensor fusion*. Nello specifico:

- Nel primo capitolo verrà presentata l'architettura del veicolo eRumby ponendo l'attenzione sugli attuatori (motore elettrico e motore servo), i sensori (encoders, GPS e IMU) e i moduli hardware (Arduino Mega, BeagleBone e XBee) che equipaggiano il veicolo;
- Nel secondo capitolo verrà descritta l'architettura software progettata e sviluppata per questo veicolo e le relative interfacce di comunicazione;

- Nel terzo capitolo verrà posta l'attenzione sui requisiti necessari per la ricostruzione dello stato del veicolo, per poi passare allo studio degli algoritmi per la *sensor fusion* presenti in letteratura e alla conseguente derivazione dell'algoritmo impiegato su questo specifico veicolo;
- Nel quarto capitolo verranno utilizzati i dati ricostruiti per verificare il comportamento del veicolo per mezzo di varie prove sperimentali.

Per finire verranno presentate alcune applicazioni e alcuni possibili sviluppi futuri del progetto eRumby.

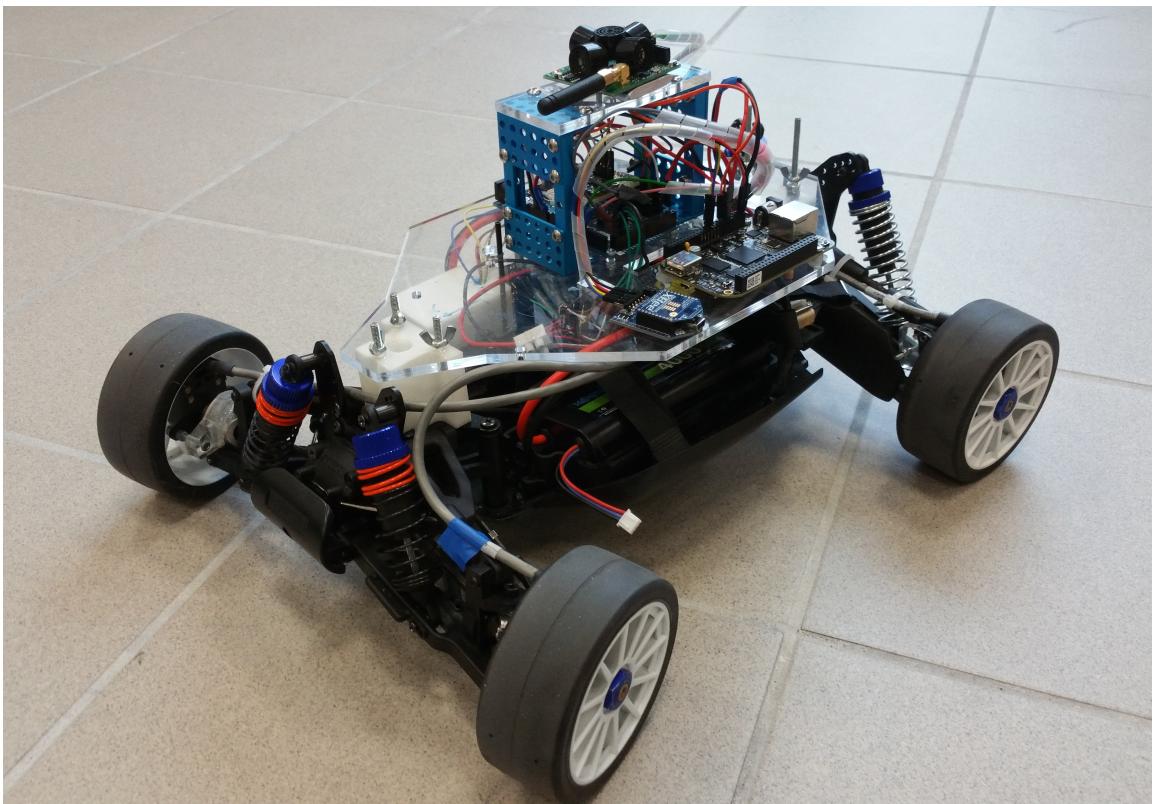


Figura 4: *Prototipo eRumby*.

## Introduzione

---

# Capitolo 1

## Il veicolo eRumby

### 1.1 Caratteristiche generali

Il veicolo eRumby del Dipartimento di Ingegneria Industriale è un veicolo elettrico radio-comandato in scala 1:8 e nasce da una rielaborazione del modello INFERNO VE Race prodotto dall'azienda giapponese di modellismo Kyosho. Di base il modello è composto da un pianale in alluminio, una barra anti rollio, quattro sospensioni regolabili e una trasmissione integrale ottenuta mediante l'uso di tre differenziali. Sul veicolo è inoltre presente una ricevente radio Syncro KT-200 che riceve il segnale proveniente dal radiocomando Syncro KT-201 2.4 GHz e lo trasmette al motore servo che controlla lo sterzo e alla *Electronic Speed Control* (ESC) connessa al motore *brushless* che si occupa della trazione. Per semplificare questa fase iniziale di studio si è modificata la struttura in veicolo a trazione posteriore, scollegando il differenziale anteriore.



Figura 1.1: *Kyosho Inferno VE Race*.

Per rendere il veicolo autonomo, il prototipo eRumby è stato dotato di sensori e moduli hardware. Come per la maggior parte dei veicoli senza conducente, al veicolo è stato aggiunto un *Global Positioning System* (GPS), una *Inertial Measurement Unit* (IMU) e degli *Odometry Sensors* (Encoder). Le informazioni provenienti dai sensori sono fuse tra loro per compensare la poca accuratezza delle singole letture sensoriali e per ricostruire precisamente lo stato del veicolo. Il GPS ha una accuratezza elevata ma una bassa frequenza di aggiornamento, mentre gli encoder e la IMU hanno una frequenza di aggiornamento elevata, ma presentano una accuratezza più bassa. La fusione dei due ingressi permette di ottenere una ricostruzione dello stato con una accuratezza e una banda molto maggiore. Sul veicolo è presente una scheda di controllo (Arduino Mega) che effettua la lettura e la trasmissione dei dati a un computer di bordo del veicolo (BeagleBone Black). Il compito principale del modulo BeagleBone è la raccolta delle informazioni provenienti dalla Arduino Mega, per poi occuparsi della rielaborazione dei dati acquisiti e del controllo del veicolo. Il modulo XBee, collegato alla BeagleBone, permette di incapsulare una connessione seriale in una connessione wireless.

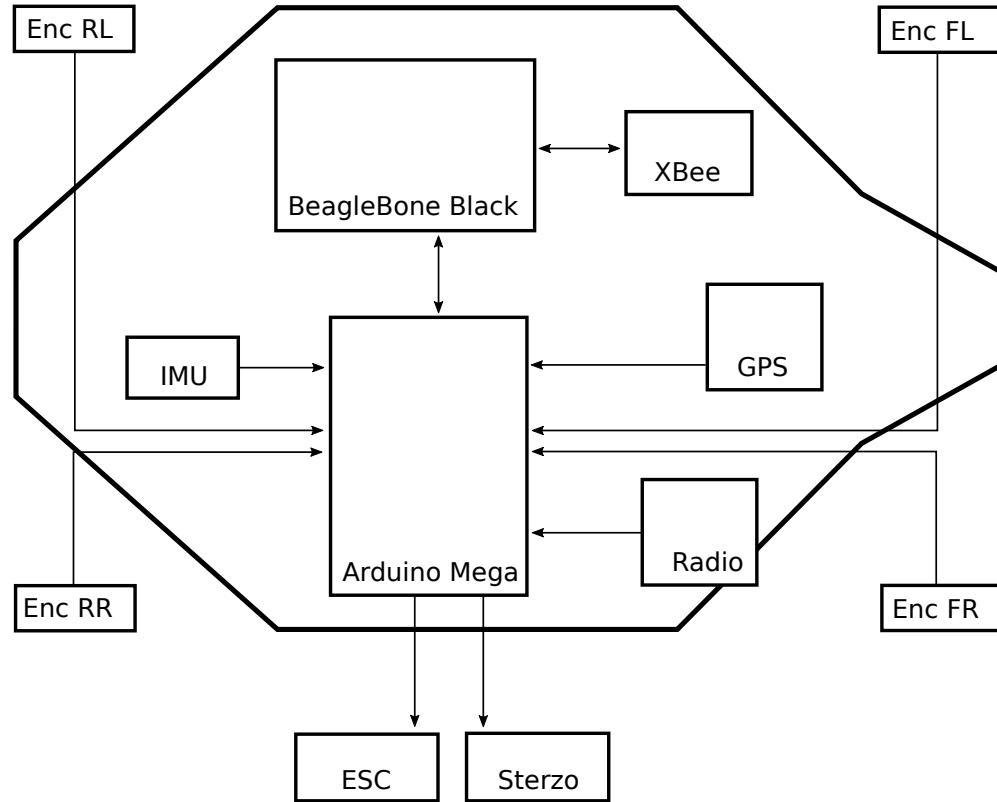


Figura 1.2: Componenti del veicolo eRumby e relative connessioni.

Il servo motore dello sterzo, il motore *brushless* per la trazione e la ricevente radio sono alimentati da una batteria a polimeri di litio (LiPo), 2S da 7.4 V. Una seconda batteria di uguali caratteristiche, affiancata da un regolatore di tensione a 5 V, viene impiegata per l'alimentazione dei sensori e delle schede di controllo. I quattro encoder sono posizionati ciascuno su una ruota, entrambi i motori e la ricevente radio sono collocati sullo *chassis* del veicolo, mentre per le due *board* e i restanti sensori è stata realizzata una piastra di supporto in plexiglas “ad hoc”.

Caratteristiche
Lunghezza 0.435 m
Larghezza 0.315 m
Altezza 0.206 m
Passo 0.320 m
Carreggiata anteriore 0.254 m
Carreggiata posteriore 0.254 m
Massa totale 4.500 kg
Massa ruota 0.111 kg

Tabella 1.1: *Caratteristiche del veicolo eRumby.*

## 1.2 Attuatori

### 1.2.1 Motore di trazione

Con motore elettrico si intende un sistema in cui la potenza di ingresso è di tipo elettrico e quella di uscita è di tipo meccanico e che assume la funzione di attuatore. Il motore elettrico è composto da uno statore e da un rotore. Il motore *brushless* è un motore elettrico in corrente continua in cui il rotore è formato da coppie di magneti permanenti e uno statore a campo magnetico rotante composto da tre avvolgimenti. A differenza del motore *brushed*, questa tipologia di motore non presenta contatti elettrici strisciati sull'albero motore. Gli avvolgimenti dello statore vengono alimentati con correnti alternate, opportunamente sfasate, che vanno a generare un campo magnetico. L'interazione di questo campo magnetico con quello generato dai magneti del rotore porta alla nascita di una coppia esercitata sul rotore che risulta massima nel momento in cui i due flussi magnetici sono perpendicolari tra loro. La commutazione della corrente circolante negli avvolgimenti dello statore, e quindi la variazione dell'orientamento del campo magnetico da essi generato, avviene elettronicamente. Alcuni dei vantaggi dei motori *brushless* rispetto ai *brushed* sono:

- un rapporto peso-peso potenza più favorevole;
- minori attriti e resistenze meccaniche;
- velocità di rotazioni maggiori.

Di contro, la elettronica necessaria al controllo del motore risulta essere estremamente più complessa.

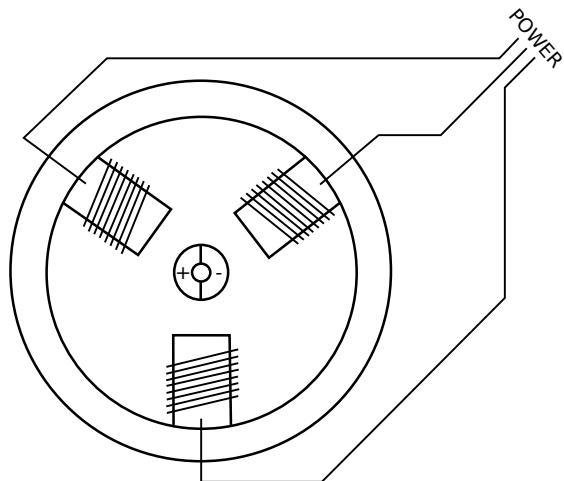


Figura 1.3: *Motore brushless*.

La propulsione del veicolo eRumby è fornita dal motore *brushless* Vortex 8 Evo prodotto dall'azienda Team Orion. Il processo di commutazione della corrente è affidata alla ESC Vortex R8 prodotta sempre da Team Orion. Il dispositivo ESC è un circuito elettrico che si occupa di convertire una corrente continua in ingresso in tre tensioni alternate sfasate di  $120^\circ$ , che controllano la velocità di rotazione del motore. La *Pulse-Width Modulation* (PWM) è una tecnica di modulazione digitale che consente di ottenere una tensione media variabile in funzione del rapporto tra la durata del segnale nello stato logico alto e la durata complessiva di un periodo, rapporto conosciuto con il nome di *duty cycle*. Il segnale PWM è un segnale in ingresso alla ESC per controllare le tensioni generate in uscita. Lo stesso ESC utilizza la tecnica PWM per ricostruire i fronti d'onda in uscita.



Figura 1.4: Motore Vortex 8 Evo e ESC Vortex R8 di Team Orion.

### 1.2.2 Servomotore

Lo sterzo è controllato da un servomotore, un motore a corrente continua che è dotato di un meccanismo di demoltiplica per aumentare la coppia in fase di rotazione e che è in grado di ruotare di un angolo limitato, generalmente nel range  $0\text{--}180^\circ$ , ed è in grado di mantenere la posizione raggiunta. La rotazione del motore avviene per mezzo di un circuito di controllo interno in grado di rilevare l'angolo di rotazione raggiunto tramite un potenziometro resistivo.

### 1.2.3 Ricevente



Figura 1.5: Radiocomando e ricevente Syncro KT.

La ricevente, una Syncro KT-200, genera un segnale ad onda quadra con un *duty cycle* proporzionale al comando da inviare alla ESC del motore e al servomotore dello sterzo. Per essere in grado di implementare un controllo attivo sul veicolo è stata eseguita una analisi dei segnali trasmessi tra radiocomando e ricevente ricavando la frequenza, l'ampiezza e il *duty cycle* del segnale.

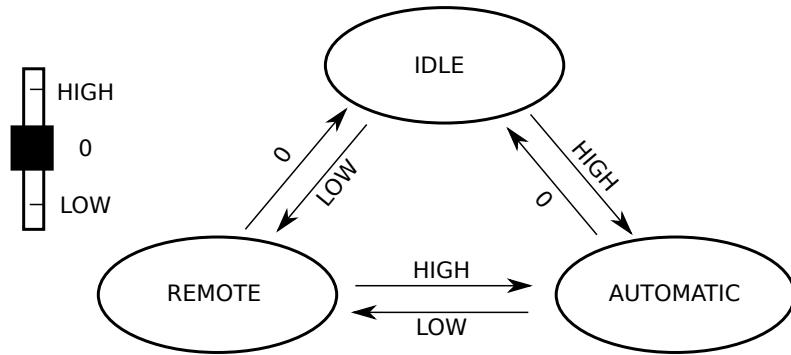


Figura 1.6: Modalità di funzionamento del veicolo.

Sul radiocomando, un Syncro KT-201 2.4 GHz, è presente un interruttore a tre canali che viene utilizzato per selezionare le tre diverse modalità di funzionamento del veicolo:

**Idle.** Quando l'interruttore è nella posizione 0 il veicolo viene messo in stato di sicurezza disabilitando qualsiasi tipo di input.

**Remote.** Quando l'interruttore è nella posizione low il veicolo viene controllato per mezzo del radiocomando.

**Automatic.** Quando l'interruttore è nella posizione high il veicolo viene guidato mediante l'uso di algoritmi di controllo.

## 1.3 Sensori

### 1.3.1 Encoder



Figura 1.7: *Encoder E5 US-Digital.*

In corrispondenza delle ruote del veicolo sono stati montati quattro encoder “E5” prodotti dalla azienda US-Digital. I modelli di encoder più utilizzati per la rilevazione dei segnali si basano sulla tecnologia ottica. Gli encoder ottici incrementali sono trasduttori in grado di rilevare lo spostamento angolare di un corpo in rotazione e si basano principalmente su due modalità di scansione.

- Scansione ottica trasmissiva. Il sistema di lettura si basa sulla rotazione di un disco graduato con un reticolo radiale, composto da tacche opache e tacche trasparenti alternate fra loro, che viene illuminato da una sorgente a raggi infrarossi. Il reticolo proietta l’immagine sui ricevitori mascherati anch’essi da un reticolo avente lo stesso passo. I ricevitori rilevano le variazioni di luce dovute allo spostamento del disco convertendole in un impulso elettrico.
- Sistema ottico riflessivo. Il sistema di lettura si basa sulla rotazione di un disco graduato, una sorgente luminosa illumina un disco graduato composto da tacche opache alternate da tacche riflettenti dove la luce si riflette e viene letta dal ricevitore che converte le variazioni in un impulso elettrico.

Gli encoder ottici E5 funzionano con la scansione ottica trasmissiva, sono dotati di 100 tacche opache e 100 tacche trasparenti e sono dotati un terminale di uscita a 5 vie di cui due per l'alimentazione e tre per i canali A, B e I.

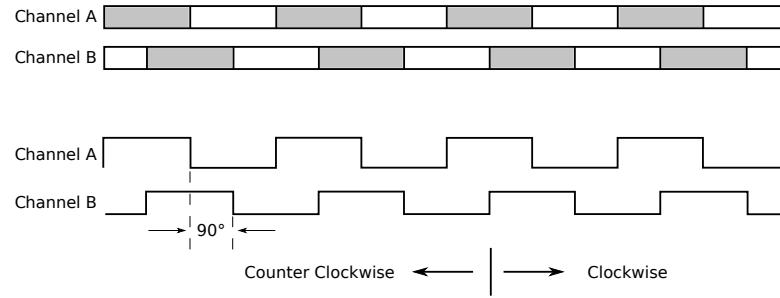


Figura 1.8: Schema di lettura dei canali A e B degli encoder.

Il canale I genera un impulso nel momento in cui viene compiuta una rotazione intera, mentre il canale A e il canale B sono sfasati tra di loro di  $90^\circ$  e generano un impulso al passaggio delle tacche. L'utilità di avere due canali è quella di poter aumentare la risoluzione e determinare il verso di rotazione. Per aumentare la risoluzione basta eseguire una operazione XOR tra il segnale del canale A e quello del canale B raddoppiando in questo modo le transizioni basso-alto del segnale con un conseguente incremento della risoluzione. Per quanto riguarda i versi di rotazione basta valutare lo sfasamento tra i segnali dei due canali. Esistono sostanzialmente due tipologie di algoritmi per elaborare il segnale ricevuto dagli encoder e valutare la velocità angolare delle ruote.

- *Line per period* (LPP). Prende in considerazione il numero di impulsi  $n_c$  ricevuti in un intervallo di tempo  $T_s$  rispetto al numero totale di tacche  $N$ . Aumentando il periodo di campionamento è possibile ridurre l'errore di troncamento a scapito di un maggiore ritardo nella stima.

$$\omega_{LLP} = \frac{n_c \frac{2\pi}{N}}{T_s}$$

- *Fixed position* (FP). Prende in considerazione l'intervallo di tempo  $T_t$  che intercorre tra due impulsi tenendo conto del numero totale di tacche  $N$  e la risoluzione  $\Delta t$ .

Aumentando il numero di tacche si ha un errore più grande, ma un ritardo minore.

$$\omega_{FP} = \frac{\frac{2\pi}{N}}{\left[\frac{T_t}{\Delta t}\right] \Delta t}$$

Caratteristiche
Temperatura di utilizzo da $-40$ a $100^{\circ}\text{C}$
Alimentazione 5 V
Output LOW 0.5 V
Output HIGH 2 V
Output corrente per canale 8 mA
Massimo gioco assiale dell'albero $\pm 0.010$ in.
Accelerazione massima 250 000 rad s $^{-2}$

Tabella 1.2: *Caratteristiche generali degli encoder “E5” della US-Digital* .

### 1.3.2 Inertial Measurement Unit

Le piattaforme *Inertial Measurement Unit* (IMU) sono dispositivi elettronici che utilizzano giroscopi, accelerometri e magnetometri montati su una piattaforma inerziale per misurare le velocità angolari, le forze inerziali e le forze dei campi magnetici rispetto a tre assi ortogonali tra di loro. L'IMU è il principale componente usato nella navigazione inerziale, tramite le sue misurazioni, opportunamente rielaborate, è possibile ricostruire velocità, posizione e orientamento del veicolo utilizzando il principio noto come *dead reckoning*. Ad esempio, dagli accelerometri è possibile ottenere velocità e posizione con una semplice integrazione nel tempo mentre dai giroscopi è possibile risalire agli angoli di rotazione del sistema. La necessità di integrare numericamente il segnale dell'accelerometro porta inevitabilmente all'introduzione di *bias* e fenomeni di *drift* nei segnali elaborati della IMU. In generale il campionamento risulta essere molto rumoroso e diventa fondamentale il filtraggio.

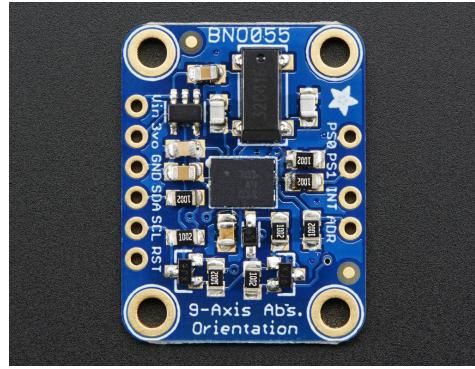


Figura 1.9: *Adafruit BNO055 Absolute Orientation Sensor.*

Il modello montato sul veicolo è una IMU “Adafruit BNO055 Absolute Orientation Sensor” a 9 gradi di libertà. La piattaforma viene utilizzata per caratterizzare l’orientazione del veicolo, le accelerazioni e le velocità di rotazione attorno ai tre assi a cui è sottoposta . Il sensore BNO055 può fornire come output:

- Orientazione assoluta dei tre assi mediante angoli di Eulero su una sfera di  $360^\circ$  (100 Hz);
- Orientazione assoluta mediante quaternioni (100 Hz);
- Vettore velocità angolare sui 3 assi in  $\text{rad s}^{-1}$  (100 Hz);
- Vettore delle accelerazioni sui 3 assi in  $\text{m s}^{-2}$  (100 Hz);
- Vettore della forza del campo magnetico sui 3 assi in micro Tesla  $\mu\text{T}$  (100 Hz);
- Vettore della accelerazione lineare sui 3 assi (viene eliminata la forza di gravità) in  $\text{m s}^{-2}$  (100 Hz);
- Vettore della accelerazione gravitazionale in  $\text{m s}^{-2}$  (100 Hz);
- Temperatura in  $^\circ\text{C}$  (1 Hz).

Caratteristiche		
Accelerometro	Power Mode	NORMAL
	Range	$\pm 4g$
	Bandwidth	62.5 Hz
	Resolution	14 bits
Giroscopio	Power Mode	NORMAL
	Range	$2000^\circ \text{ s}^{-1}$
	Bandwidth	32 Hz
	Resolution	16 bits
Magnetometro	Power Mode	FORCED
	ODR	20 Hz
	XY Repetition	15
	Z Repetition	16
	Resolution x/y/z	13/13/15 bits

Tabella 1.3: *Caratteristiche della IMU Adafruit BNO055.*

### 1.3.3 Global Positioning System

Il GPS fa parte del sistema di navigazione satellitare del programma *NAVigation System with Time and Ranging* ( NAVSTAR) del Department of Defense (DoD) degli Stati Uniti d’America. Nato nel 1978 per scopi militari, ora è largamente diffuso anche in ambito civile. Il GPS consente di ottenere la posizione sulla superficie terrestre rilevando la latitudine, la longitudine e la altitudine sfruttando il ritardo di trasmissione tra i satelliti artificiali in orbita. La posizione individuata dal sistema, senza l’ausilio di altri strumenti, ha una precisione dell’ordine dei metri anche se fino al 2000 il DoD negò tale accuratezza ad uso civile, aggiungendo un rumore chiamato *Selective Availability* per allargare il raggio di precisione all’ordine di grandezza dei 100 metri.



Figura 1.10: *GPS indoor Marvelmind Robotics.*

Per essere in grado di utilizzare il nostro veicolo nei laboratori, si è scelto di utilizzare un GPS indoor, altrimenti noto come *Indoor Positioning System* (IPS). Il GPS classico, utilizzato in ambiente chiuso, non ha sempre una ricezione sufficientemente buona dai satelliti ed inoltre la precisione sulla posizione è dell'ordine dei metri, mentre con un sistema IPS si è in grado di ottenere una precisione dell'ordine dei centimetri. In commercio ci sono diversi sistemi di navigazione indoor e per il nostro progetto la scelta è ricaduta sul sistema prodotto dalla azienda Marvelmind Robotics. Si tratta di un sistema di navigazione interna che utilizza una antenna *router* e dei *beacon* che comunicano tra di loro attraverso una trasmissione radio su una banda libera. Ogni faro è dotato di 5 sensori ultrasuoni e di una batteria LiPo. Nello specifico il sistema è composto da:

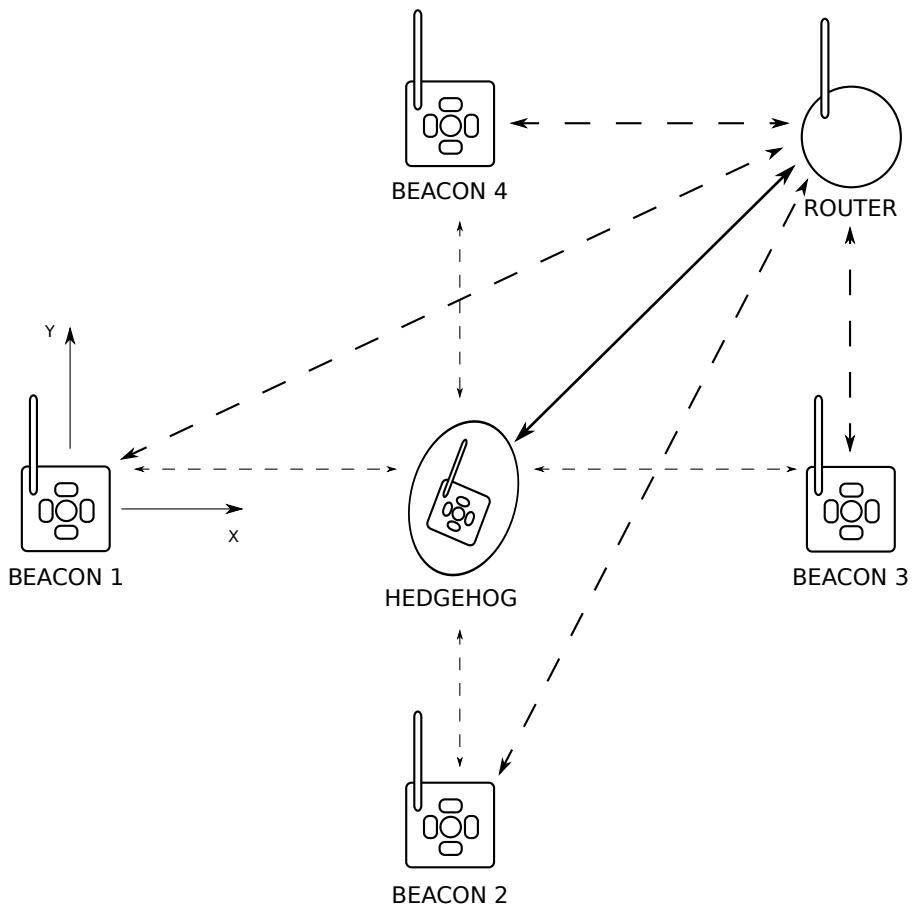


Figura 1.11: *Schema del sistema IPS.*

- 1 *beacon* mobile, detto *hedgehog*, montato sul veicolo. Connesso alla scheda di controllo Arduino Mega attraverso UART, riceve aggiornamenti sulla posizione dal router con una frequenza di 16 Hz. Deve essere disposto orizzontalmente per migliorare la copertura dei segnali ultrasuoni e non deve essere posto vicino a qualsiasi cosa che possa limitare il segnale a ultrasuoni.
- 4 *beacon* stazionari. Ognuno di essi è montato su un apposito supporto, misura la distanza rispetto gli altri fari attraverso impulsi ad ultrasuono e comunica con il router via wireless. La distanza massima tra i fari è di 50 metri.

- 1 router connesso al PC. Cuore centrale del sistema, serve per configurare e monitorare il sistema attraverso il computer e inoltre si occupa della comunicazione con il faro mobile.

Caratteristiche
Precisione al cm con una incertezza pari a $\pm 1$ cm
Distanza massima tra i fari di 50 m
Area di copertura dei segnali di $100 \text{ m}^2$
Frequenza di aggiornamento dei dati pari a 16 Hz

Tabella 1.4: *Caratteristiche generali del sistema GPS Indor Marvelmind Robotics.*

## 1.4 Schede di controllo

### 1.4.1 Arduino Mega

Arduino è una piattaforma hardware basata su microcontrollore ATMEL, con regolatore di tensione, *General Purpose Input Output* (GPIO) e interfaccia seriale che permette la comunicazione con il computer utilizzato per programmare. L'idea venne sviluppata in un istituto di formazione post-dottorale con sede a Ivrea, la Interaction Design Institute, con lo scopo di realizzare uno strumento per la prototipazione rapida e per scopi didattici, hobbyistici e professionali. Il nome Arduino deriva dal nome di un bar di Ivrea frequentato dai fondatori del progetto, a sua volta derivato da Arduino d'Ivrea, Re di Italia dal 1002 al 1014. In questo caso, sul veicolo eRumby è stata utilizzata una Arduino Mega 2560, una scheda basata su ATmega2560. La scheda lavora con una frequenza di 16 MHz, è dotata di 4 porte seriali UART, 16 ingressi analogici e 54 pin digitali di input e di output, di cui 15 possono essere usati come output PWM. Inoltre è dotata di un jack di ingresso per la alimentazione e di un ingresso USB che permette la connessione della scheda con un computer. L'Arduino Mega può essere alimentata sia tramite connessione USB che tramite alimentazione esterna e in caso di più fonti sceglie quale utilizzare in modo automatico.

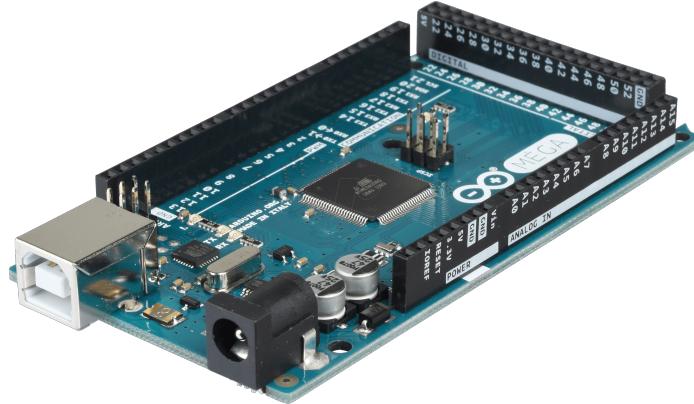


Figura 1.12: *Arduino Mega*.

Caratteristiche
Microcontrollore ATmega1280
Tensione di funzionamento 5 V
Tensione in ingresso (raccomandata) 712 V
Tensione in ingresso (limite) 620 V
Pin I/O digitali 54
Pin I/O analogici 16
Corrente DC per i pin I/O 40 mA
Corrente DC per i pin a 3.3 V a 40 mA
Flash memory 128KB
SRAM 8KB
EEPROM 4KB
Clock speed 16 MHz

Tabella 1.5: *Caratteristiche della scheda Arduino Mega 2560*.

## 1.4.2 XBee

Il modulo radio Digi XBee è prodotto dalla azienda Digi International e consente il trasferimento dati via frequenza radio sfruttando una comunicazione seriale. In breve, tutto ciò che il microcontrollore invia sul pin RX di un modulo XBee giunge ad un altro XBee sul pin TX e può dunque essere ricevuto dal secondo dispositivo. I moduli XBee più diffusi operano alla frequenza di 2.4 GHz utilizzando il protocollo di comunicazione 802.15 e ZigBee. Le principali caratteristiche sono:

- Bidirezionalità. Un singolo modulo XBee è in grado sia di ricevere che di trasmettere.
- Indirizzi unici. Ogni XBee ha un proprio numero seriale che permette di indirizzare in maniera univoca la comunicazione tra due moduli.
- Il controllo di errore e altri algoritmi di trasmissione robusta sono implementati in hardware, rendendo la comunicazione affidabile
- Possibilità di assegnare diversi canali ai moduli per minimizzare eventuali interferenze.

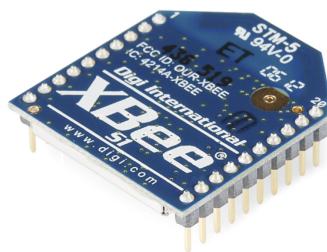


Figura 1.13: *Digi XBee*.

Il protocollo di comunicazione su cui lavorano i moduli XBee è l'802.15.4. Questo è uno standard per le comunicazioni wireless definito dalla Institute of Electrical and Electronics Engineers (IEEE). Questo standard nasce per soddisfare requisiti di basso consumo per velocità di trasferimento basse, connettività semplice e possibilità di operare con batteria. Lo standard prevede una comunicazione di tipo *point-to-point* o di tipo *point-to-multipoint*, noto anche come “a stella”. Il primo prevede che due dispositivi A e B comunichino unicamente tra di loro, il secondo prevedere che un coordinatore, detto nodo centrale, possa ricevere e trasmettere da e verso tutti i nodi periferici, i quali possono comunicare solo con il coordinatore.

### 1.4.3 BeagleBone Black

La scheda BeagleBone Black è un *System on Chip* (SoC) su una singola scheda a basso consumo caratterizzato da un circuito open-hardware e un SoC ARM-based con prestazioni e connettività prossime a quelle di un laptop. Viene prodotto da Texas Instruments in associazione con Digi-Key e Newark Element14. La scheda è stata sviluppato da un piccolo team di ingegneri con l'intento di realizzare una *board* educativa ed è venduta al pubblico sotto licenza *Creative Commons* (CC). La *board* garantisce compatibilità con sistemi Debian, Ubuntu, Android. Per la realizzazione di questo progetto, a bordo della BeagleBone è stato utilizzato il sistema operativo Debian Jessie, un OS multi-architettura per computer composto interamente da software libero. Il nome Debian è stato coniato dal fondatore del progetto, Ian Murdock, mentre i nomi in codice delle distribuzioni sono presi dai personaggi del film di animazione “Toy Story - Il mondo dei giocattoli”. L'attuale distribuzione stabile è la versione 8.0, conosciuta con il nome di Jessie uscita nell'aprile 2015 e con supporto garantito fino ad aprile 2020.



Figura 1.14: *BeagleBone Black*.

Caratteristiche
Processore AM335x 1GHz ARM® Cortex-A8
RAM 512MB DDR3
4GB 8-bit eMMC on-board flash storage
Acceleratore grafico 3D
2x PRU 32-bit microcontrollers
Porta USB per alimentazione e comunicazione
Ingresso Ethernet
Ingresso HDMI
2x 46 pin headers

Tabella 1.6: *Caratteristiche della scheda BeagleBone Black.*



# Capitolo 2

## Architettura software del veicolo

### 2.1 Architettura della rete

L'architettura del veicolo eRumby è composta da una Arduino Mega, una BeagleBone e i sensori IMU, GPS e encoder e i motori di trazione e sterzo. L'Arduino Mega si occupa di raccogliere i dati provenienti dai sensori, inviarli alla BeagleBone, gestire i segnali provenienti dalla ricevente e controllare i motori. Il cuore centrale del sistema è la BeagleBone e si occupa del salvataggio dei dati, della *sensor fusion* e del controllo del veicolo fornendo alla Arduino Mega i segnali di input per la gestione dei motori. Le due schede di controllo sono connesse tra loro mediante comunicazione seriale e il trasferimento dei dati avviene per mezzo di *union*.

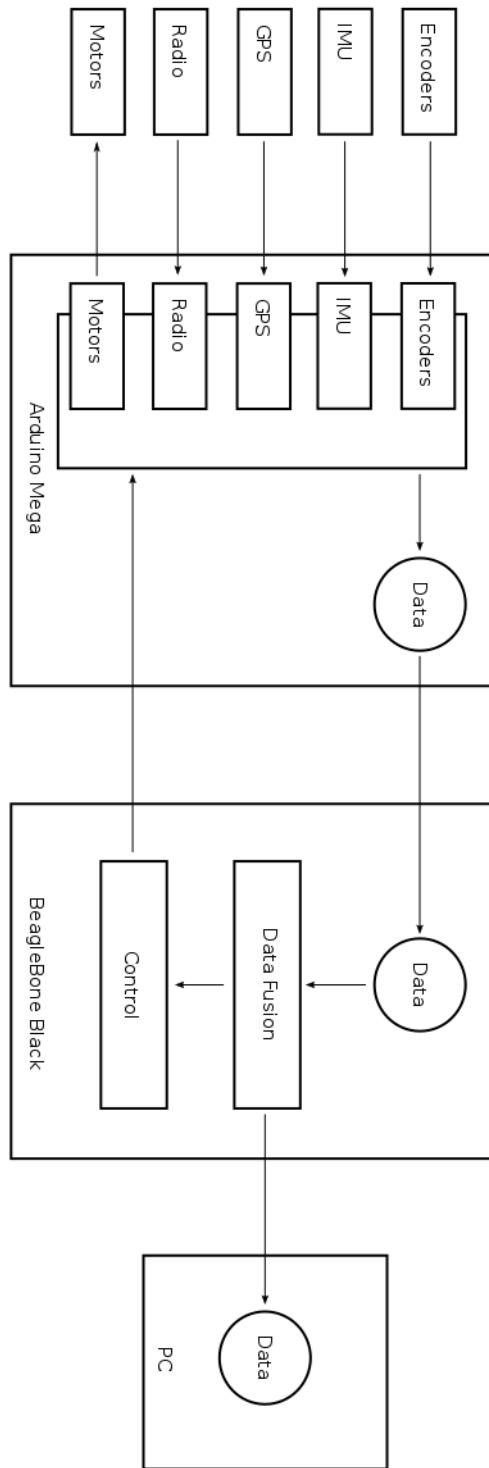


Figura 2.1: Architettura software del veicolo

## 2.2 Moduli software

Nella Arduino Mega e nella BeagleBone Black sono stati implementati diversi moduli software:

- Lettura encoder;
- Lettura IMU;
- Lettura GPS;
- Lettura ricevente;
- Gestione dei motori;
- Salvataggio e scambio dati;
- Data fusion;
- Controllore della velocità;
- Controllo.

I moduli sono indipendenti l'uno dall'altro al fine di garantire la modularità sia hardware (sensori) che software (algoritmi). La sincronizzazione delle informazioni raccolte avviene ad una frequenza di circa  $50Hz$ . La raccolta dei dati provenienti dai vari moduli consente di tenere traccia istante per istante dello stato del veicolo e, attraverso l'utilizzo di modelli matematici e della fusione dei dati, consente di avere una indicazione dello stato del veicolo con una affidabilità maggiore.

## 2.3 Arduino Mega

### 2.3.1 Lettura encoder

I segnali provenienti dagli encoder sono onde quadre con frequenza variabile in funzione della velocità di rotazione. Presentano un valore di *HIGH* in corrispondenza della tacca

opaca e un valore di *LOW* in corrispondenza di una tacca trasparente. Il modulo si basa sull'algoritmo *Fixed Position* e calcola l'intervallo di tempo che intercorre tra il passaggio di due tacche trasparenti. Il microcontrollore riceve un *interrupt* ogni volta che il fotoricevitore rileva lo spostamento da una tacca trasparente ad una opaca la *Interrupt Service Routine* (ISR) associata tiene conto del numero di volte che è stata richiamata dall'ultimo invio dati (*counter*) e calcola la durata della permanenza nello stato alto mediante i timer del sistema (*pulseWidth*) e l'ultimo stato temporale in cui si ha avuto un fronte di salita (*edgeTime*). La struttura dei dati è:

```
typedef struct Dati {  
    uint 16_t pulseWidth;  
    uint 16_t edgeTime;  
    byte counter ;  
}
```

La ISR si occupa di gestire un'operazione *bitwise or-esclusiva* (XOR) per determinare le variazioni rispetto all'ultima lettura, confrontando i valori relativi ai bit della porta A con quelli che erano stati memorizzati nella lettura precedente e nel caso in cui ci siano variazioni viene aggiornata la struttura dati dell'encoder corrispondente. La velocità di rotazione delle singole ruote è calcolata come:

$$\omega = \begin{cases} \frac{2\pi}{N\Delta t} & c \geq 1 \\ 0 & c < 1 \end{cases}$$

con  $N$  risoluzione dell'encoder,  $c$  il numero di volte in cui la ISR è stata invocata e  $\Delta t$  l'intervallo di tempo in cui il segnale è rimasto alto.

Il modulo attualmente gestisce solo uno dei tre canali (A, B, I) del sensore. Infatti ogni canale richiederebbe un *interrupt* rendendo difficile garantire la frequenza di campionamento. L'aggiunta del canale B porterebbe ad un miglioramento della risoluzione nella lettura degli encoder e la possibilità di determinare il verso di rotazione.

### 2.3.2 Lettura IMU

All'accensione del sistema, il modulo IMU richiede la calibrazione dei sensori per poi occuparsi della lettura dei giroscopi, degli accelerometri e dei magnetometri. La comunicazione con la Arduino Mega avviene su protocollo I2C e la ricezione del canale di dati è discriminato da un indirizzo associato alla tipologia di sensore.

Sensore	Indirizzo
Accelerometro	((int) 0x53)
Magnetometro	((int) 0x1E)
Giroscopio	((int) 0x68)

Tabella 2.1: *Indirizzi I2C IMU.*

Le accelerazioni sono normalizzate rispetto la gravità e il modulo è in grado di compensare errori di offset mediante calibrazione. La direzione del campo magnetico è determinata per mezzo della relazione:

$$B_{Heading} = \arctan \left( \frac{-B_{y0}}{B_{x0}} \right)$$

dove  $B_{x0}$  e  $B_{y0}$  sono i vettori del campo magnetico rispetto gli assi  $x_0$  e  $y_0$  della terna assoluta, ottenuti dalla lettura del magnetometro valutati rispetto la terna mobile solidale al veicolo, opportunamente ruotata. La frequenza di aggiornamento dei dati forniti dal modulo IMU è di 50 Hz, fornisce le accelerazioni lungo gli assi x, y e z, gli angoli di *roll*, *pitch* e *yaw* e i relativi *roll rate*, *pitch rate* e *yaw rate*. Posto  $Ox_v y_v z_v$  il sistema di riferimento solidale al veicolo e  $Ox_0 y_0 z_0$  il sistema di riferimento solidale al terreno, la sequenza delle rotazioni che definisce la matrice di rotazione  $R$  è data da:

- Rotazione dell'angolo di *yaw*  $\psi$  attorno all'asse  $z_v$ .
- Rotazione dell'angolo di *roll*  $\theta$  attorno all'asse  $y_v$ .
- Rotazione dell'angolo di *pitch*  $\phi$  attorno all'asse  $x_v$ .

Angolo	Range di rotazione degli angoli
Pitch	da $+180^\circ$ a $-180^\circ$ , decresce ruotando in senso orario
Roll	da $+90^\circ$ a $-90^\circ$ , cresce aumentando l'inclinazione
Yaw	da $0^\circ$ a $360^\circ$ , cresce ruotando in senso orario

Tabella 2.2: *Convenzioni sulla rotazione degli angoli.*

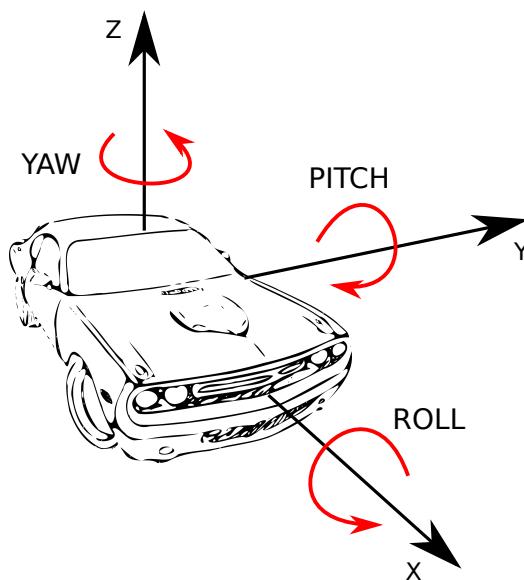


Figura 2.2: *Sistema di riferimento della IMU solidale al veicolo.*

### 2.3.3 Lettura GPS Indoor

Il modulo GPS Indoor si occupa di leggere le coordinate x e y fornite dal *hedgehog*, le confronta con le precedenti e le aggiorna in caso di cambiamenti. La comunicazione tra Arduino Mega e *hedgehog* avviene tramite connessione seriale ad una velocità di trasmissione

di 500 kbps. La frequenza di aggiornamento della posizione è di circa 10 Hz, paragonabile ad un sensore GPS commerciale.

### 2.3.4 Lettura ricevente

La ricevente genera tre segnali ad onda quadra, uno per ogni canale che riceve dal radio-comando: il grilletto per l'acceleratore, la manopola dello sterzo e il selettore modalità di utilizzo del veicolo. Il segnale è ricevuto ad una frequenza costante di 71.4 Hz, con un *duty cycle* variabile tra il 7.5% e il 14.5% proporzionale al comando da inviare alla ESC del motore e al servomotore dello sterzo. Questo modulo si occupa di calcolare la durata della permanenza nello stato alto del segnale per ricostruire successivamente la posizione delle leve del radiocomando. Le informazioni ricavate vengono utilizzate dal modulo dei motori per costruire il segnale di output.

### 2.3.5 Gestione dei motori

Lo scopo di questo modulo è quella di generare i segnali da inviare alla ESC e al servomotore dello sterzo. Entrambi i motori necessitano di un segnale ad onda quadra a frequenza costante, il diverso comportamento è dettato dal *duty cycle* del segnale stesso. Per la generazione dei segnali si è scelto di utilizzare la libreria “PWM.h” che permette di configurare uno dei timer a 16 bit in dotazione alla scheda di controllo, in modo che il segnale d’uscita segua il comportamento desiderato. Le funzioni della libreria necessitano di soli due parametri in input:

**Frequenza:** è la frequenza della onda quadra da generare.

**Duty cycle:** è la percentuale del periodo della onda quadra in cui il segnale deve rimanere nello stato HIGH.

Entrambe le grandezze sono note dalla analisi eseguita sulla ricevente. Si definisce la proporzione:

$$(2^{16} - 1) : 100 = \Delta : \delta^*$$

dove  $\Delta$  è il valore desiderato dalla libreria,  $\delta^*$  è il valore percentuale del *duty cycle* del segnale e  $2^{16} - 1$  è il numero di bit del timer utilizzato dalla libreria. A questo punto è facile ricavare il valore desiderato dalla libreria:

$$\Delta = \frac{(2^{16} - 1)\delta^*}{100}$$

e ricostruire i valori di accelerazione e sterzo del radiocomando. Per non forzare troppo i motori e limitare la velocità massima del veicolo si è scelto di ridurre la banda disponibile del 20%.

Label	Acceleratore	Input	Valore
DUTY_ESC_MIN	Minimo	7.5%	4915
DUTY_ESC_NEUTRO	Neutro	10.5%	6881
DUTY_ESC_MAX	Massimo	14.5%	9502

Label	Sterzo	Input	Valore
DUTY_SERVO_DX	Destra	7.5%	4915
DUTY_SERVO_NEUTRO	Neutro	10.5%	6881
DUTY_SERVO_SX	Sinistra	14.5%	9502

Tabella 2.3: *Valori per la ricostruzione dei segnali*

## 2.4 BeagleBone

La BeagleBone non è dotata di interfaccia grafica e l'utilizzo avviene da remoto attraverso un prompt di comandi di tipo testuale (*Shell*) mediante un protocollo *Secure SHell* (SSH). L'accesso avviene conoscendo l'*hostname* della macchina, in questo caso `beaglebone`, oppure conoscendo il suo indirizzo IP. Ad esempio, se si connette la BeagleBone al proprio PC tramite cavo USB l'indirizzo IP di default è 192.168.7.2.

All'interno della BeagleBone è presente la cartella `/script/` dove sono raccolti tutti i codici utilizzati per il funzionamento del veicolo. In particolare:

- `enable_uart.sh`: serve per abilitare le porte UART utilizzate nelle comunicazioni BeagleBone - Arduino Mega e BeagleBone - XBee.
- `start_eRumby.sh`: script utilizzato per avviare il sistema eRumby.

### 2.4.1 Salvataggio e scambio dati

Le informazioni lette dai sensori vengono indirizzate verso strutture di dati:

```
typedef struct {
    float encoder[4];                      /* FL,FR,RL,RR */
    float acc[3];                           /* x,y,z */
    float gyro[3];                          /* x,y,z */
    float orientation[3];                   /* x,y,z */
    float gps[2];                           /* x,y */
    float time_gps;
    float time_imu;
    short int command_traction_motor;
    short int command_steering_motor;
} sensor_t;

typedef union {
    sensor_t sensor;
    char sensor_data_byte[sizeof(sensor)];
} sensor_data_t;
```

In `float encoder[4]` ci sono i valori dagli encoder nell'ordine anteriore sinistro (FL), anteriore destro (FR), posteriore sinistro (RL) e posteriore destro (RR). Per i dati provenienti dalla IMU, in `float acc[3]` ci sono le accelerazioni rispetto gli assi x, y e z, in `float gyro[3]` ci sono le velocità di rotazione degli assi x, y e z e in `float orientation[3]` ci sono gli angoli di rotazione rispetto gli assi x, y e z. In `float gps[2]` ci sono i valori delle coordinate x e y forniti dal GPS. `Float time_gps` e `float time_imu` contengono rispettivamente i tempi di campionamento del GPS e della IMU mentre `short int command_traction_motor` e `short int command_steering_motor` contengono i segnali di input del motore di trazione e del servo motore. La struttura dei dati viene inviata dalla Arduino Mega alla BeagleBone attraverso una comunicazione seriale. Per la comunicazione la BeagleBone sfrutta la libreria “serialib.h”. Una volta ricevuti, i dati vengono salvati in file di *log* e utilizzati nell'esecuzione degli algoritmi di *sensor fusion* e di controllo del veicolo.

Label	Significato	Unità di misura
enc_fl	velocità angolare proveniente dall'encoder anteriore sinistro	$\text{rad s}^{-1}$
enc_fr	velocità angolare proveniente dall'encoder anteriore destro	$\text{rad s}^{-1}$
enc_rl	velocità angolare proveniente dall'encoder posteriore sinistro	$\text{rad s}^{-1}$
enc_rr	velocità angolare proveniente dall'encoder posteriore destro	$\text{rad s}^{-1}$
acc_x	accelerazione lineare lungo l'asse "x" rispetto al sistema di riferimento solidale alla IMU	$\text{m s}^{-2}$
acc_y	accelerazione lineare lungo l'asse "y" rispetto al sistema di riferimento solidale alla IMU	$\text{m s}^{-2}$
acc_z	accelerazione lineare lungo l'asse "z" rispetto al sistema di riferimento solidale alla IMU	$\text{m s}^{-2}$
gyro_x	velocità di rollio rispetto al sistema di riferimento solidale alla IMU	$\text{rad s}^{-1}$
gyro_y	velocità di beccheggio rispetto al sistema di riferimento solidale alla IMU	$\text{rad s}^{-1}$
gyro_z	velocità di imbardata rispetto al sistema di riferimento solidale alla IMU	$\text{rad s}^{-1}$
roll	angolo di rollio rispetto al sistema di riferimento solidale alla IMU	$^\circ$
pitch	angolo di beccheggio rispetto al sistema di riferimento solidale alla IMU	$^\circ$
yaw	angolo di imbardata rispetto al sistema di riferimento solidale alla IMU	$^\circ$

Tabella 2.4: *Struttura del file di log*

Label	Significato	Unità di misura
gps_x	posizione del veicolo lungo l'asse "x" rispetto al sistema di riferimento solidale al GPS	m
gps_y	posizione del veicolo lungo l'asse "y" rispetto al sistema di riferimento solidale al GPS	m
time	istante in cui avviene l'acquisizione dei dati da parte della BeagleBone	s
time_imu	Intervallo di campionamento della IMU	s
time_gps	Intervallo di campionamento del GPS	s
traction	percentuale del segnale di comando al motore di trazione	%
steering	percentuale del segnale di comando al motore di sterzo	%

Tabella 2.5: *Struttura del file di log*

### 2.4.2 Data fusion

Il modulo *data fusion* rielabora i dati per migliorare la stima delle grandezze ricavate dai sensori. L'obbiettivo è quello di avere una indicazione affidabile sullo stato del veicolo. L'algoritmo implementato è un Filtro di Kalman Esteso (EKF) e verrà descritto nel dettaglio nel capitolo successivo 3. Le informazioni ricavate dalla *sensor fusion* fanno riferimento al modello del veicolo 2D descritto nella sezione 3.5. Anche in questo caso i valori ottenuti vengono indirizzati verso strutture di dati:

```
typedef struct {
    float pos[2];           /* x, y */
    float ang[2];           /* psi, omega */
    float vel[2];           /* vx, vy */
```

```

        float acc[2];           /* ax , ay */
} fusion_output_t;

typedef union {
    fusion_output_t fusion_output;
    char fusion_output_data_byte[sizeof(fusion_output)];
} fusion_output_data_t;

```

Per ora le informazioni ricavate dalla *sensor fusion* sono:

- *float pos[1]*: descrive la coordinata *x*.
- *float pos[2]*: descrive la coordinata *y*.
- *float ang[1]*: descrive l'angolo di *yaw*.
- *float ang[2]*: descrive lo *yaw rate*.
- *float vel[1]*: descrive la velocità *v<sub>x</sub>*.
- *float vel[2]*: descrive la velocità *v<sub>y</sub>*.
- *float acc[1]*: descrive l'accelerazione *a<sub>x</sub>*.
- *float acc[2]*: descrive l'accelerazione *a<sub>y</sub>*.

I dati ricavati dalla *sensor fusion* vengono poi inviati al PC sfruttando i moduli radio XBee.

### 2.4.3 Controllore della velocità

Questo modulo si occupa di regolare la velocità di avanzamento longitudinale del veicolo mediante una legge di controllo proporzionale integrale (PI) non lineare, dove l'azione proporzionale (P) regola la variazione istantanea del processo, mentre l'azione integrale (I) permette la convergenza a zero dell'errore a regime. La legge nello spazio del tempo è:

$$\hat{h}(t) = K_P e(t) + K_I \int_0^t e(\tau) d\tau$$

dove:

- $\hat{h}$  è la variabile in uscita dal controllore;
- $e(t)$  è l'errore tra la velocità di riferimento e quella media assunta dal veicolo;
- $K_p$  coefficiente dell'azione proporzionale;
- $K_I$  coefficiente dell'azione integrale.

La funzione di trasferimento in frequenza è:

$$R(s) = \frac{\hat{H}(s)}{E(s)} = K_p + \frac{K_I}{s} = K_p \frac{1 + T_I s}{T_I s}$$

con  $K_I = K_p / T_I$ , con  $T_I = \tau_p$  tempo integrale.

#### 2.4.4 Controllo

In questo modulo sono implementati tre distinti algoritmi di controllo del veicolo.

**Preview Controller.** Poichè in molti sistemi di controllo è possibile modellare la evoluzione futura dello stato e dei disturbi, è possibile migliorare le prestazioni del controllore sfruttando queste informazioni. Ad esempio, conoscendo la destinazione del veicolo è possibile pianificare in anticipo la traiettoria che deve seguire entro un orizzonte temporale limitato. Questo tipo di controllo prende il nome di Preview Controller e venne sviluppato per la prima volta negli anni '60 da Masayoshi Tomizuka.

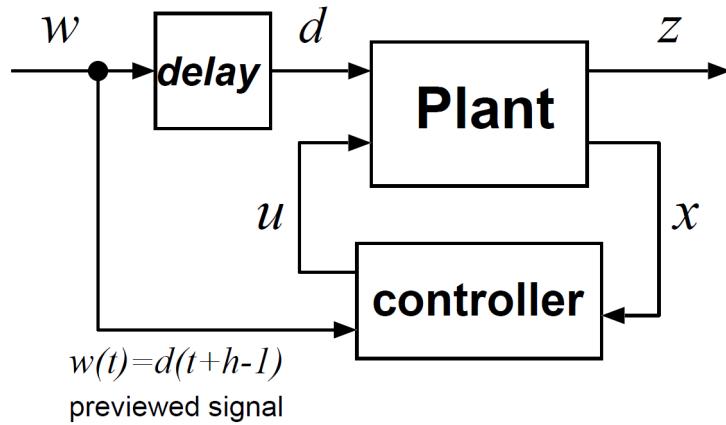


Figura 2.3: Problema del Preview Control [21].

**Pure Pursuit.** Questo algoritmo si occupa di calcolare geometricamente l’arco di traiettoria necessario per muovere un veicolo dalla posizione attuale a una posizione obiettivo. La posizione obiettivo è il punto successivo del percorso che il veicolo deve compiere rispetto alla sua posa attuale. Il Pure Pursuit venne sviluppato all’inizio degli anni ’80 da Wallace et al. durante la realizzazione di Terragator, un robot a sei ruote sterzanti usato per sperimentare algoritmi di computer vision outdoor.

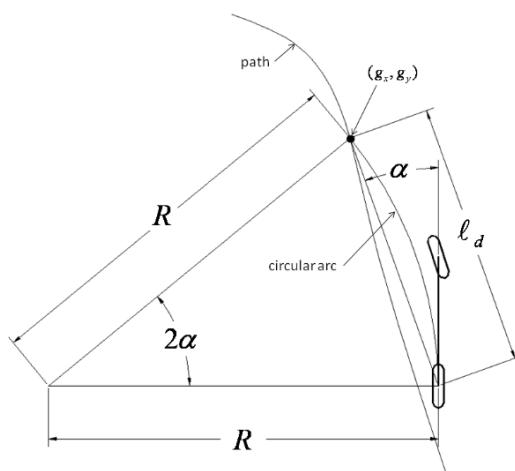


Figura 2.4: Geometria alla base del Pure Pursuit [22].

**Stanley Method.** E' una legge di controllo di feedback non lineare dell'errore di deriva del veicolo ed è usata per determinare in modo autonomo una traiettoria in real-time. Questo controllo è utilizzato per determinare la distanza tra il centro dell'asse anteriore del veicolo e fornisce come output il punto più vicino del percorso e fornisce una correzione dell'angolo di sterzo. Il Stanley Method venne utilizzato per la prima volta sul veicolo autonomo Stanley realizzato dalla Stanford University e che vinse il DARPA Grand Challenge 2005.

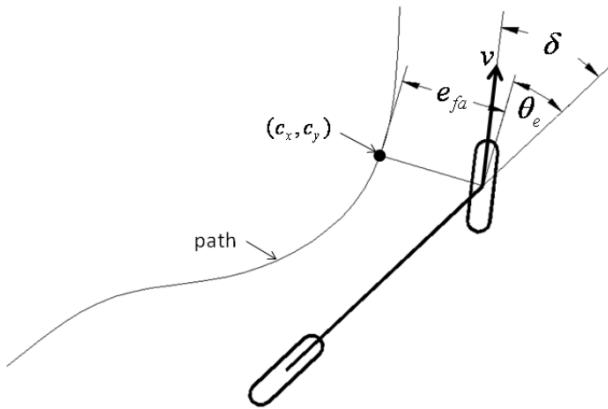


Figura 2.5: Geometria alla base del Stanley Method [12].



# Capitolo 3

## Stato del veicolo

### 3.1 Sensor Fusion

Uno degli aforismi più conosciuti del fisico tedesco Max Planck recita: «La fisica si fonda su misure, e poiché ogni misura è legata a una sua percezione sensoriale, tutti i concetti della fisica provengono dal mondo sensibile, ai cui fenomeni in fondo ogni legge fisica si riferisce.»<sup>1</sup> L'idea alla base della *sensor fusion* è quella di organizzare e gestire in modo intelligente una mole elevata di dati ridondanti prodotti da diversi sensori in modo da ridurre la incertezza sulle informazioni dello stato del sistema e ottenere stime più accurate, più complete e più affidabili di quelle ottenibili dai sensori presi singolarmente. Uno dei maggiori vantaggi è la possibilità di avere in questo modo una conoscenza più completa del sistema avendo una visione d'insieme fornita da tutte le misure gestite. Un altro punto importante è la possibilità di creare sistemi robusti, ovvero sistemi meno sensibili alla rottura o al malfunzionamento di uno o più sensori, oppure di essere in grado di unire le informazioni da più sensori per ottenerne una più accurata e meno soggetta al rumore. In questo caso l'operazione di filtraggio può essere eseguita per mezzo di opportuni modelli e algoritmi. La struttura di un sistema di *sensor fusion* è la composizione di una rete di sensori i cui output sono fusi per mezzo di un algoritmo di *sensor fusion*. I sensori all'interno di un sistema possono interagire tra di loro in modo diverso e generalmente si fa riferimento a:

---

<sup>1</sup>Planck M., *The philosophy of physics*, 1936

- **Sensori Complementari.** Sensori indipendenti che forniscono informazioni relative a diverse grandezze fisiche, ma complementari per l'osservazione di uno stesso fenomeno.
- **Sensori Concorrenti.** Sensori distinti che forniscono informazioni indipendenti relativamente alla stessa grandezza fisica, utili per aumentare l'affidabilità del sistema di misura.
- **Sensori Cooperanti.** Sensori indipendenti la cui osservazione è combinata per ottenere informazioni non deducibili da un solo sensore.

In generale il concetto di fusione dell'informazione è anche conosciuto con il nome di processo di *Data Fusion* e l'elaborazione può avvenire attraverso una:

- **Struttura Centralizzata.** Le misure acquisite da ogni singolo sensore vengono inviate ad una unità centrale dove sono rielaborate per ottenere una stima dello stato del sistema. Il vantaggio di questa struttura è una perdita ridotta della informazione dei dati, ma richiede elevate risorse computazionale. I possibili problemi di comunicazione influiscono sulle prestazioni del sistema.
- **Struttura Distribuita.** Ogni sensore è dotato di risorse computazionali proprie e quindi rielabora la misurazioni prima del successivo invio all'unità centrale dove vengono combinate assieme alle altre. La struttura distribuita è molto più robusta di quella centralizzata poiché ogni pacchetto dati inviato condensa in sé tutta l'informazione relativa al passato e quindi i problemi di comunicazione influiscono meno.

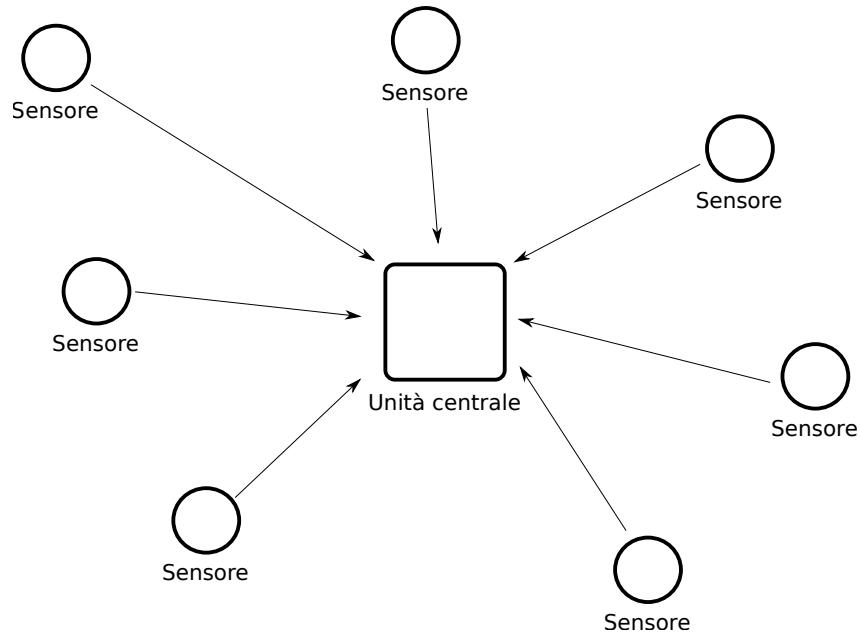


Figura 3.1: Struttura centralizzata.

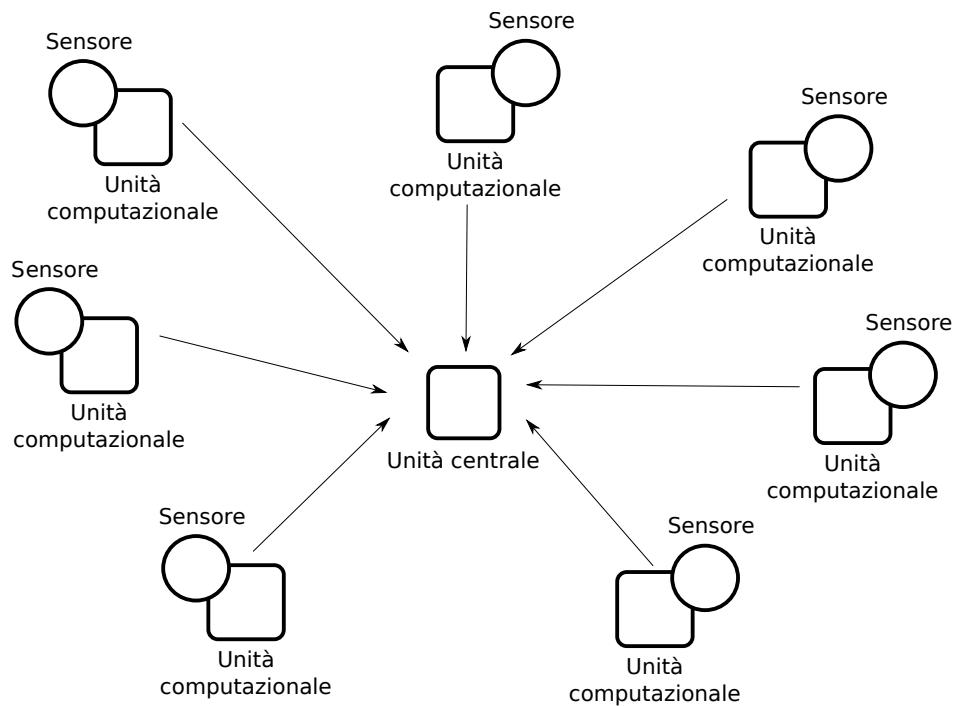


Figura 3.2: Struttura distribuita.

In letteratura si possono trovare diverse tecniche per la applicazione della *sensor fusion*. Quelle più note e utilizzate sono essenzialmente 4: le reti neurali, le logiche *fuzzy*, l’osservatore di stato e il filtro di Kalman.

- **Reti Neurali.** Una *Artificial Neural Network* (ANN) è un modello matematico informatico di calcolo che si ispira alle reti neurali biologiche. Il modello si basa sulla interconnessione delle informazioni costituite da un gruppo di nodi, neuroni artificiali, e dei processi. Le reti neurali sono strutture non lineari di dati statistici che vengono utilizzati per modellare il sistema. In altre parole, sono delle tecniche di approssimazione utilizzabili quando si ha una scarsa conoscenza del legame esistente tra le variabili indipendenti e quelle dipendenti. In questo modo è possibile trattare il sistema come una *black box* alla quale fornire degli input per ottenere degli output. In altre parole i risultati ottenuti hanno una elevata probabilità di essere accettabili, ma non è noto come questi risultati siano stati ottenuti. Le ANN sono generalmente impiegate in applicazioni dove i dati possono essere parzialmente errati oppure dove non è possibile sviluppare modelli analitici del problema e sono largamente utilizzate anche per analisi meteorologiche e analisi finanziarie. Il vantaggio delle reti neurali è la possibilità di lavorare in parallelo e rielaborare una notevole quantità di dati, permettendo di avere solo un calo delle prestazioni in caso di malfunzionamento scongiurando il blocco del sistema.

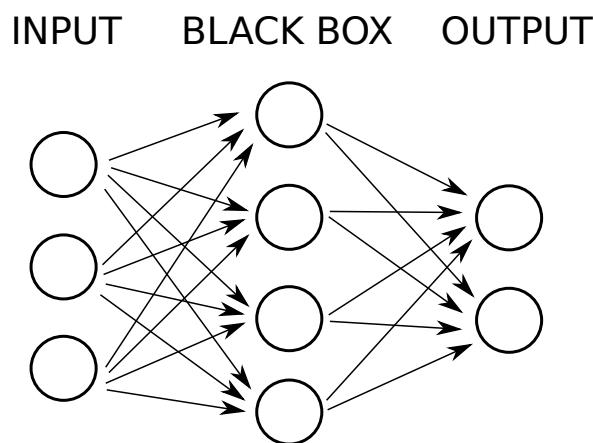


Figura 3.3: *Logica alla base delle reti neurali.*

- Logiche *Fuzzy*. Si basano sulla logica classica, ma in questo caso sono presenti più gradi di verità e vengono generalmente definiti in un intervallo compreso tra 0 e 1. Quando il grado di verità è 0 assume il significato di totalmente falso, mentre quando è 1 assume il significato di totalmente vero. Il primo passo vede la conversione dei dati forniti in valori linguistici con un grado di verità associato. Nel secondo passo i valori linguistici vengono valutati secondo regole prestabilite e la funzione di appartenenza finale è fornita dalla inferenza delle regole. A questo punto, nel terzo passo, i valori linguistici ottenuti vengono riconvertiti in output del sistema.

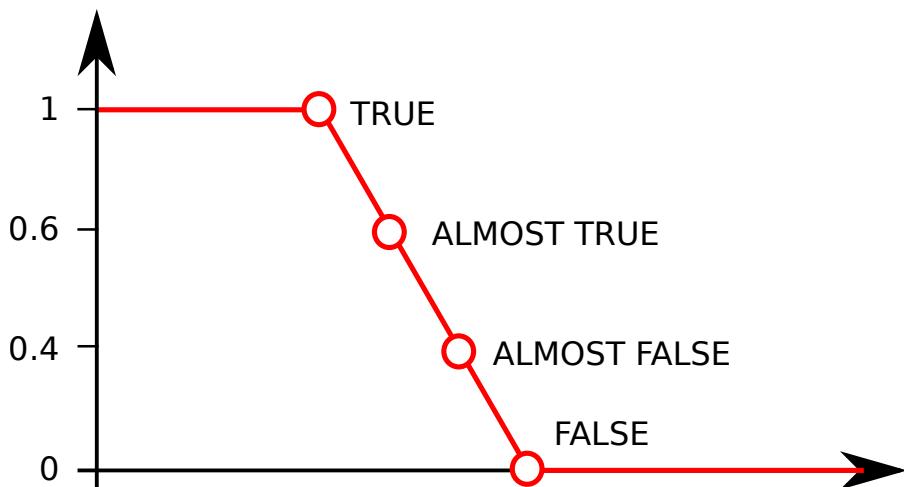


Figura 3.4: Logica Fuzzy.

- Osservatore di Luenberger. Si tratta di uno stimatore dello stato per sistemi lineari. Dato il sistema dinamico:

$$\begin{cases} x_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k \end{cases}$$

l'output del sistema  $y_k$  è utilizzato per stimare lo stato del sistema  $x_k$  per mezzo della relazione:

$$\begin{cases} x_{k+1} = Ax_k + L[y_k - \hat{y}_k] Bu_k \\ \hat{y}_k = C\hat{x}_k \end{cases}$$

con  $L$  matrice di guadagno e  $\hat{x}$  e  $\hat{y}$  le variabili stimate. L'osservatore è asintoticamente stabile se l'errore soddisfa la relazione:

$$e_{k+1} = (A - LC) e_k$$

- Filtro di Kalman. Elabora una stima dello stato di un sistema e viene utilizzato per avere una indicazione più corretta di un dato ricevendo informazioni da diversi sensori ognuno con un proprio rumore di misura. Il filtro di Kalman è composto da una fase di predizione e una di correzione. Nella fase di predizione al tempo  $t$  viene fatta una stima del valore delle variabili di interesse al tempo  $t + \Delta t$ , mentre la fase di correzione confronta i dati predetti con quelli effettivamente misurati ed effettua una opportuna correzione.

In questo caso, volendo attuare la *sensor fusion* a bordo di un veicolo dotato di molteplici sensori e vista la massiccia presenza in letteratura di modelli dinamici per i veicoli, la scelta è ricaduta sullo sviluppo di un filtro di Kalman.

	RETE NEURALE	LOGICA FUZZY	FILTRO DI KALMAN
Conoscenza del sistema	Sistema come una Black Box	Non richiede un modello dettagliato	Richiede un modello dinamico del sistema
Tipologie di sistemi	Adatto per sistemi non lineari	Adatto per diverse tipologie di sistemi	Adatto per sistemi lineari
Sviluppo iniziale	Richiede ampi campioni per l'apprendimento	Non presenta particolari problemi	Settaggio sperimentale
Relazione tra input e output	Sconosciuto	Facilmente comprensibile	Comprensibile

Tabella 3.1: *Comparazione tra le tecniche di sensor fusion prese in esame.*

## 3.2 Filtro di Kalman discreto

Il filtro di Kalman venne introdotto per la prima volta nel 1960 da R. E. Kalman, un ingegnere e matematico statunitense di origini ungheresi. Il filtro da lui ideato è una tecnica

tuttora molto utilizzata nei sistemi di controllo e nelle avionica per estrarre una informazione accurata da una serie di dati discreti affetti da rumore. L'algoritmo alla base del filtro si occupa del filtraggio dei dati prendendo in considerazione il successivo valore stimato e il successivo valore predetto utilizzando un controllo in feedback. Nel campo della navigazione autonoma consente di ottenere una stima migliore dello stato del veicolo mediante la lettura di più sensori. Le equazioni del filtro di Kalman vengono generalmente distinte in *prediction equations* e in *update equations*. Le *prediction equations* forniscono una previsione dello stato attuale e della covarianza dell'errore e vengono valutate per ottenere una stima a priori per il passo successivo, mentre le *update equations* si occupano del feedback unendo una nuova misura con la stima a priori per avere una migliore stima a posteriori.

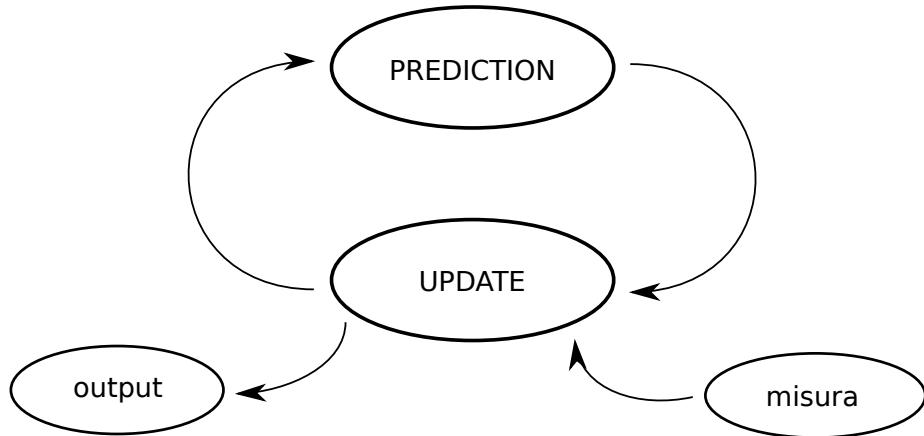


Figura 3.5: Schema generale del filtro di Kalman.

Nella fase di predizione viene fatta una stima dello stato in accordo con il modello del sistema. Questo modello può contenere qualsiasi equazione in grado di descrivere gli stati. L'errore prodotto dall'uso del modello per la determinazione degli stati viene aggiunto alla matrice di covarianza. Se non ci fosse questa correzione fornita dal modello, l'errore si propagherebbe fornendo un output inutilizzabile. Le equazioni per la fase di predizione sono:

$$\hat{x}_k = A_k \hat{x}_{k-1}$$

$$P_k = A_k P_{k-1} A_k^T + Q_k$$

dove:

- $\hat{x}_k$  è il vettore di stato.
- $A_k$  è la matrice di stato, si occupa di descrivere l'evoluzione della variabile di stato rispetto al suo valore attuale.
- $P_k$  è la matrice di varianza dell'errore sullo stato e rappresenta la variabilità dell'errore sulla stima dello stato a causa dei disturbi dati dall'errore di misura e dal disturbo dello stato.
- $Q_k$  è la matrice di covarianza del disturbo sullo stato, rappresenta la variabilità statistica del vettore dei disturbi sullo stato.

La fase di correzione si occupa di correggere le predizioni con le nuove misure acquisite. Queste correzioni sono pesate utilizzando il guadagno di Kalman  $K$ , basato sulle incertezze del modello di predizione e sulle misure disponibili. A regime, la riduzione nella matrice di covarianza nel passo di correzione corrisponde all'errore aggiunto nella fase di predizione. A questo punto il filtro si può considerare convergente. Le equazioni per la fase di correzione sono dunque:

$$K_k = P_k H_k^T (H_k P_k H_k^T + R_k)^{-1}$$

$$\hat{x}_k = \hat{x}_k + K(z_k - H_k \hat{x}_k)$$

$$P_k = (I - K_k H_k) P_k$$

dove:

- $z_k$  vettore delle misure.
- $K_k$  è la matrice di correzione della stima e indica la confidenza della misura rispetto alla stima dello stato in base al valore precedente.

- $H_k$  è la matrice delle uscite e descrive il valore assunto dalle variabili misurate in funzione del valore attuale delle variabili di stato.
- $R_k$  è la matrice di covarianza del rumore sulle misure, ovvero descrive la variabilità statistica del vettore dei disturbi di misura e rappresenta la potenza del disturbo sulla misura introdotta su ciascuna delle misure accessibili.

Nello sviluppo del filtro di Kalman, le matrici  $Q_k$  di covarianza del rumore del processo e  $R_k$  di covarianza dell'errore di misurazione dovrebbero essere misurate prima che le operazioni di filtraggio vengano eseguite. Generalmente  $Q_k$  è impiegata per rappresentare l'incertezza nel modello del processo, in questo modo è possibile utilizzare modelli semplificati inserendo un determinato grado di incertezza attraverso la matrice  $Q_k$ . Tuttavia, che si abbia o meno un fondamento razionale per la scelta dei parametri, si possono ottenere prestazioni migliori regolando i parametri delle due matrici. Inoltre è utile sapere che durante la fase di filtraggio sia la covarianza dell'errore di stima  $P_k$  che il guadagno di Kalman  $K_t$  si stabilizzano rapidamente e poi rimangono costanti.

### 3.3 Filtro di Kalman Esteso

Nel caso in cui il sistema preso in esame non sia descritto da equazioni lineari si può fare uso del *Filtro di Kalman Esteso* (EKF). In questo caso le previsioni vengono fatte tramite equazioni non lineari, mentre la matrice di covarianza viene linearizzata nell'intorno della stima corrente usando le derivate parziali delle funzioni di stato e di misura. Anche in questo caso l'algoritmo è composto da due fasi, una prima di predizione e una seconda di correzione. Nella fase di predizione vengono generate le proiezioni dello stato del sistema in esame sulla base del modello adottato e delle precedenti stime e vengono aggiornati i parametri legati agli errori di misura dello stato. Nella fase di correzione vengono corrette le stime precedentemente calcolate con delle nuove misure e vengono aggiornati i parametri legati agli errori delle misurazioni. In questo caso si ha:

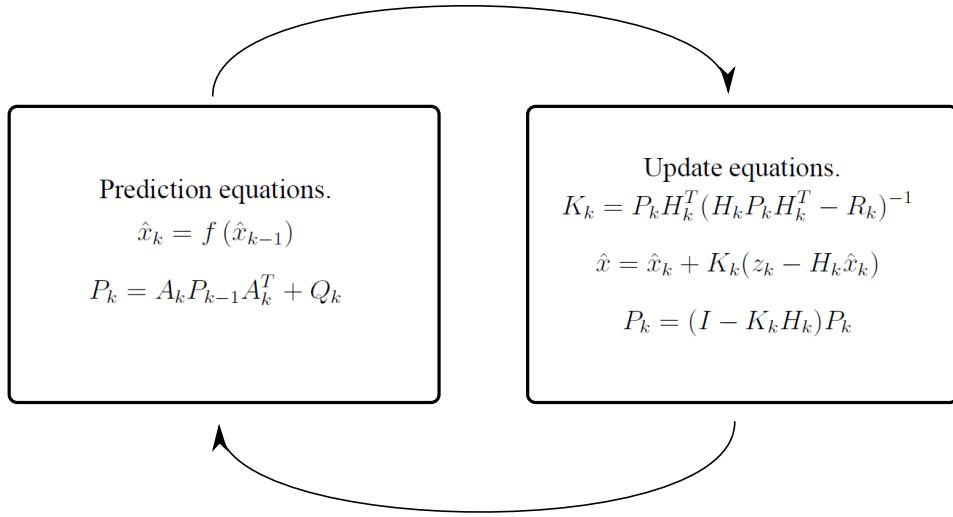


Figura 3.6: Schema generale dell'algoritmo di EKF

- *Prediction equations.*

$$\hat{x}_k = f(\hat{x}_{k-1})$$

$$P_k = A_k P_{k-1} A_k^T + Q_k$$

- *Update equations.*

$$K_k = P_k H_k^T (H_k P_k H_k^T - R_k)^{-1}$$

$$\hat{x}_k = \hat{x}_k + K_k (z_k - H_k \hat{x}_k)$$

$$P_k = (I - K_k H_k) P_k$$

- Dove:

$\hat{x}_{k-1}$  è lo stato stimato precedentemente;

$\hat{x}_k$  è lo stato stimato;

$K_k$  è il guadagno del filtro al passo  $k$ ;

$z_k$  sono le misure;

$A_k$  è lo Jacobiano di  $f(\hat{x}_{k-1})$  rispetto  $\hat{x}_{k-1}$ ;  
 $H_k$  è lo Jacobiano delle uscite predette rispetto  $\hat{x}_{k-1}$ .

### 3.4 Filtro di Kalman Unscented

Il *Filtro di Kalman Unscented* (UKF) rappresenta una alternativa al EKF e viene generalmente impiegato quando le equazioni di *prediction* e *update* sono altamente non lineari. In queste condizioni UKF ha una performance migliore del EKF mantenendo una complessità computazionale ridotta. Il UKF utilizzala la *Unscented Transform* (UT), tecnica di campionamento deterministico, per una variabile random che è sottoposta ad una trasformazione non lineare e permette di determinare una serie minima di punti di campionamento distribuiti attorno alla media. Questi punti di campionamento sono detti *sigma points*. Si consideri una variabile random  $x$ , di dimensione  $L$ , che si propaga attraverso una funzione non lineare  $y = f(x)$ , con media  $\bar{x}$  e covarianza  $P_x$ . Per calcolare le statistiche di  $y$  costruiamo una matrice  $\chi$  di  $2L + 1$  vettori sigma  $\chi_i$ :

$$\begin{aligned}\chi_0 &= \bar{x} \\ \chi_i &= \bar{x} + \left( \sqrt{(L + \lambda) P_x} \right)_i \quad i = 1 \dots L \\ \chi_{i+L} &= \bar{x} - \left( \sqrt{(L + \lambda) P_x} \right)_i \quad i = L + 1 \dots 2L\end{aligned}$$

con  $\lambda = \alpha^2 (L + k) - L$  parametro di scala ,  $\alpha$  diffusione dei *sigma points* attorno a  $\bar{x}$  e  $k$  è un secondo parametro di scala. Questi vettori *sigma* si propagano attraverso la funzione non lineare:

$$\gamma_i = f(\chi_i) \quad i = 1 \dots 2L$$

e la media e la covarianza di  $y$  sono approssimate usando una media e una covarianza pesata dei precedenti *sigma points*:

$$\bar{y} \approx \sum_{i=0}^{2L} W_i^{(m)} \gamma_i$$

$$P_y \approx \sum_{i=0}^{2L} W_i^{(c)} \{ \mathcal{Y}_i - \bar{y} \} \{ \mathcal{Y}_i - \bar{y} \}^T$$

dove i pesi  $W_i$  sono:

$$\begin{aligned} W_0^{(m)} &= \lambda / (L + \lambda) \\ W_0^{(c)} &= \lambda / (L + \lambda) + (1 - \alpha^2 + \beta) \\ W_i^{(m)} = W_i^{(c)} &= 1 / [2(L + \lambda)] \quad i = 1 \dots 2L \end{aligned}$$

### 3.5 Ricostruzione dello stato del veicolo

Considerando il veicolo eRumby, lo stato è formalizzato sulla posizione, la velocità e l'angolo di imbardata. Per questo caso, il modello più semplice possibile è quello di un corpo rigido che si muove nel piano. Nell'assumere tale modello trascuriamo i moti verticali, di beccheggio e di rollio, la deformabilità del telaio e le variazioni di altitudine della strada. Valgono dunque le seguenti relazioni:

$$\begin{cases} \dot{v}_x = a_x + v_y \Omega \\ \dot{v}_y = a_y - v_x \Omega \\ X_G = \int v_x \cos(\psi) - v_y \sin(\psi) dt \\ Y_G = \int v_x \sin(\psi) + v_y \cos(\psi) dt \\ \psi = \int \Omega dt \end{cases}$$

Indicando con  $\sim$  le misure di cui ci serviamo, abbiamo la posizione  $\tilde{X}$  e  $\tilde{Y}$  fornita dal GPS, le accelerazioni  $\tilde{a}_x$  e  $\tilde{a}_y$ , l'angolo di imbardata  $\tilde{\psi}$  e la velocità di imbardata  $\tilde{\Omega}$  misurate dalla IMU e la velocità  $\tilde{v}_x$  fornita dagli encoder per mezzo della relazione:

$$\tilde{v}_x(t) = \frac{\omega_{FL}(t) + \omega_{FR}(t)}{2} r_\omega$$

dove  $\omega_{FL}(t)$  e  $\omega_{FR}(t)$  sono le velocità angolari della ruota anteriore sinistra e di quella anteriore destra, mentre  $r_\omega$  è il raggio di rotolamento delle ruote. Si è scelto di valutare la velocità solo sulle ruote anteriori perché nel veicolo assemblato sono quelle non sono

attuate e quindi lo slip è trascurabile. Puoi dire che usiamo le ruote anteriori perché non sono attuate e quindi lo slip è trascurabile. Nel momento in cui dovesse essere introdotto un *Traction Control System* (TCS) sarebbe possibile configurare il veicolo con una trazione integrale, rendendo il sistema più robusto, e valutare la velocità rispetto tutti e quattro gli encoder :

$$\tilde{v}_x(t) = \frac{\omega_{FL}(t) + \omega_{FR}(t) + \omega_{RL}(t) + \omega_{RR}(t)}{4} r_\omega$$

con  $\omega_{RL}(t)$  e  $\omega_{RR}(t)$  le velocità angolari della ruota posteriore sinistra e di quella anteriore destra. Il sistema di equazioni in forma discreta è:

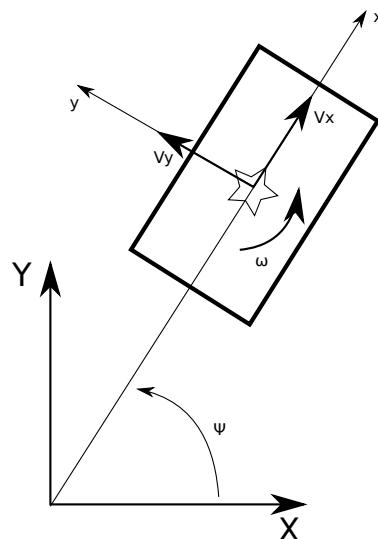


Figura 3.7: Modellazione del veicolo eRumby.

$$\begin{cases} X_{G_{k+1}} = X_{G_k} + (v_{x_k} \cos(\psi_k) - v_{y_k} \sin(\psi_k)) \Delta t \\ v_{x_{k+1}} = v_{x_k} + (a_{x_k} + \Omega_k v_{y_k}) \Delta t \\ a_{x_{k+1}} = a_{x_k} \\ Y_{G_{k+1}} = Y_{G_k} + (v_{x_k} \sin(\psi_k) + v_{y_k} \cos(\psi_k)) \Delta t \\ v_{y_{k+1}} = v_{y_k} + (a_{y_k} - \Omega_k v_{x_k}) \Delta t \\ a_{y_{k+1}} = a_{y_k} \\ \psi_{k+1} = \psi_k + \Omega_k \Delta t \\ \Omega_{k+1} = \Omega_k \end{cases}$$

che risulta essere non lineare, forzando l'uso del filtro di Kalman nella forma estesa. La matrice di stato  $A_k$  assume la forma:

$$A_k = \begin{bmatrix} 1 & \cos(\psi_k) \Delta t & 0 & 0 & -\sin(\psi_k) \Delta t & 0 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & \Omega_k \Delta t & 0 & 0 & v_{y_k} \Delta t \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sin(\psi_k) \Delta t & 0 & 1 & \cos(\psi_k) \Delta t & 0 & 0 & 0 \\ 0 & -\Omega_k \Delta t & 0 & 0 & 1 & \Delta t & 0 & -v_{x_k} \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

A questo punto siamo in grado di esplicitare il legame espresso tra il vettore delle variabili di stato  $\hat{x}$  e il vettore delle misure  $z$  mediante la matrice delle uscite  $H$ :

$$z = H \hat{x}$$

$$\begin{Bmatrix} \tilde{X} \\ \tilde{v}_x \\ \tilde{a}_x \\ \tilde{Y} \\ \tilde{a}_y \\ \tilde{\psi} \\ \tilde{\omega} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} X_G \\ v_x \\ a_x \\ Y_G \\ v_y \\ a_y \\ \psi \\ \omega \end{Bmatrix}$$

La matrice di covarianza dell'errore di misurazione  $R_k$  e la matrice covarianza del rumore del processo  $Q_k$  sono state misurate prima della applicazione del filtro e successivamente ottimizzate regolando i parametri.

### 3.5.1 Definizione del sideslip angle

Noto lo stato del veicolo, il *sideslip angle*, angolo definito tra la direzione della velocità longitudinale e quella effettiva del veicolo, può essere valutato tramite la relazione:

$$\beta_{k+1}(t) = \arctan \left( \frac{v_{y_{k+1}}(t)}{v_{x_{k+1}}(t)} \right)$$

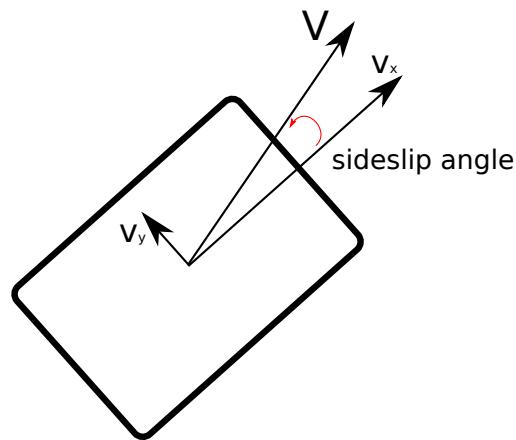


Figura 3.8: *Sideslip angle*.

La velocità longitudinale e quella laterale sono legate al *sideslip angle* per mezzo delle relazioni:

$$v_{x_{k+1}}(t) = V_{k+1}(t) \cos(\beta_{k+1}(t))$$

$$v_{y_{k+1}}(t) = V_{k+1}(t) \sin(\beta_{k+1}(t))$$

dove  $V_{k+1}$  è la velocità del veicolo. Sotto l'assunzione di angolo di *sideslip* piccolo si ha:

$$v_x(t) \simeq V(t)$$

$$v_y(t) \simeq V(t)\beta(t)$$

Data il modello cinematico:

$$a_x(t) = \dot{v}_x(t) - \Omega(t)v_y(t)$$

$$a_y(t) = \dot{v}_y(t) + \Omega(t)v_x(t)$$

le equazioni precedenti si possono riscrivere come:

$$\dot{V}(t) = V(t)\beta(t)\Omega(t) + a_x(t)$$

$$\dot{V}(t)\beta(t) + \dot{\beta}(t)V(t) = -V(t)\Omega(t) + a_y(t)$$

dalle quali è possibile determinare il *sideslip rate*  $\dot{\beta}(t)$ :

$$\dot{\beta}(t) = \frac{a_y(t) - \Omega(t)V(t)}{V(t)}$$

# Capitolo 4

## Prove sperimentali

### 4.1 Analisi dei dati

In questo capitolo si affronta la validazione del sistema meccanico e di acquisizione dati sulla base delle seguenti 5 prove:

1. Prova in condizioni stazionarie.
2. Prova su una traiettoria circolare.
3. Prova su una traiettoria rettilinea.
4. Prova di *lane change*.
5. Prova su un circuito.

In particolare, il primo test andrà a evidenziare la qualità dei dati raccolti dal sistema, i successivi tre andranno ad evidenziare le prestazioni del filtro di Kalman esteso presentato nella sezione 3.5 e l'ultima prova andrà a mostrare il comportamento del veicolo nel corso di una prova su un circuito.

### 4.1.1 Prova in condizioni stazionarie

Per valutare il corretto funzionamento della IMU è stata fatta una prima prova in condizioni stazionarie. I dati letti dagli accelerometri sono normalizzati rispetto l'accelerazione gravitazionale, quindi ci si aspetta un valore nullo in tutti e tre i casi.

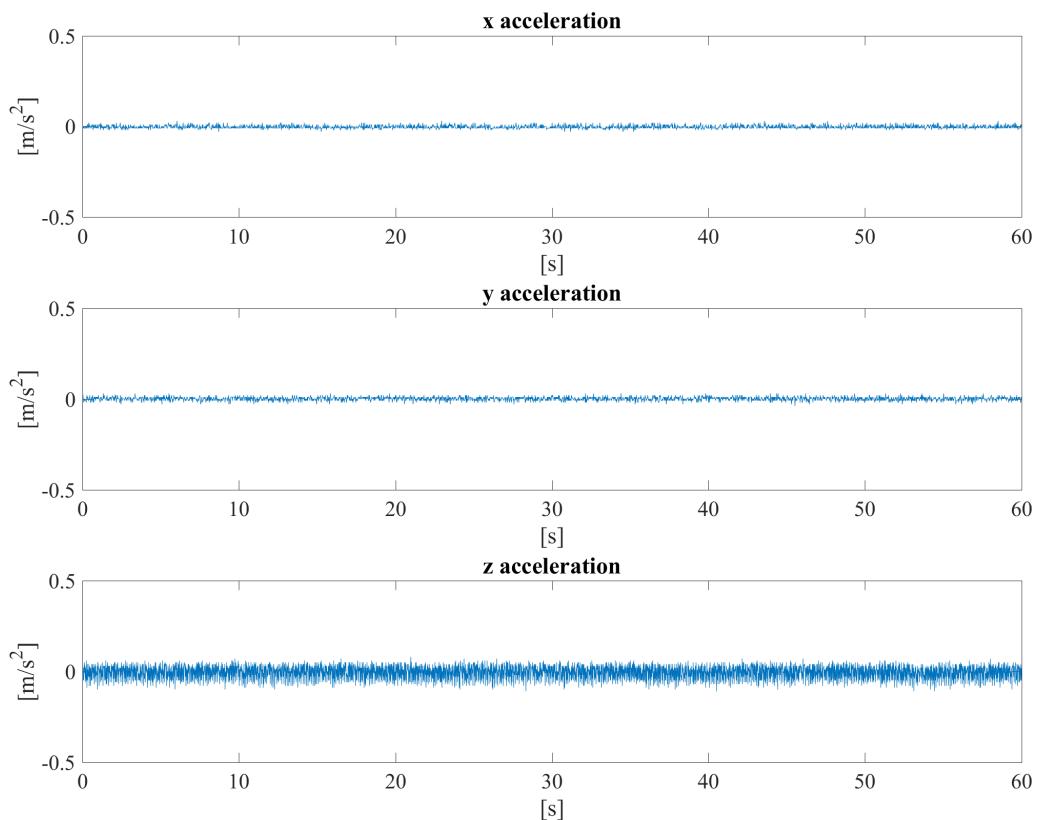


Figura 4.1: Accelerazioni in condizioni statiche.

Osservando i dati provenienti dagli accelerometri (figura 4.1) si può notare che il comportamento sperimentale rispecchia quello teorico previsto. Il segnale non risulta pulito, ma sono presenti oscillazioni dovute a possibili interferenze, disturbi ed approssimazioni.

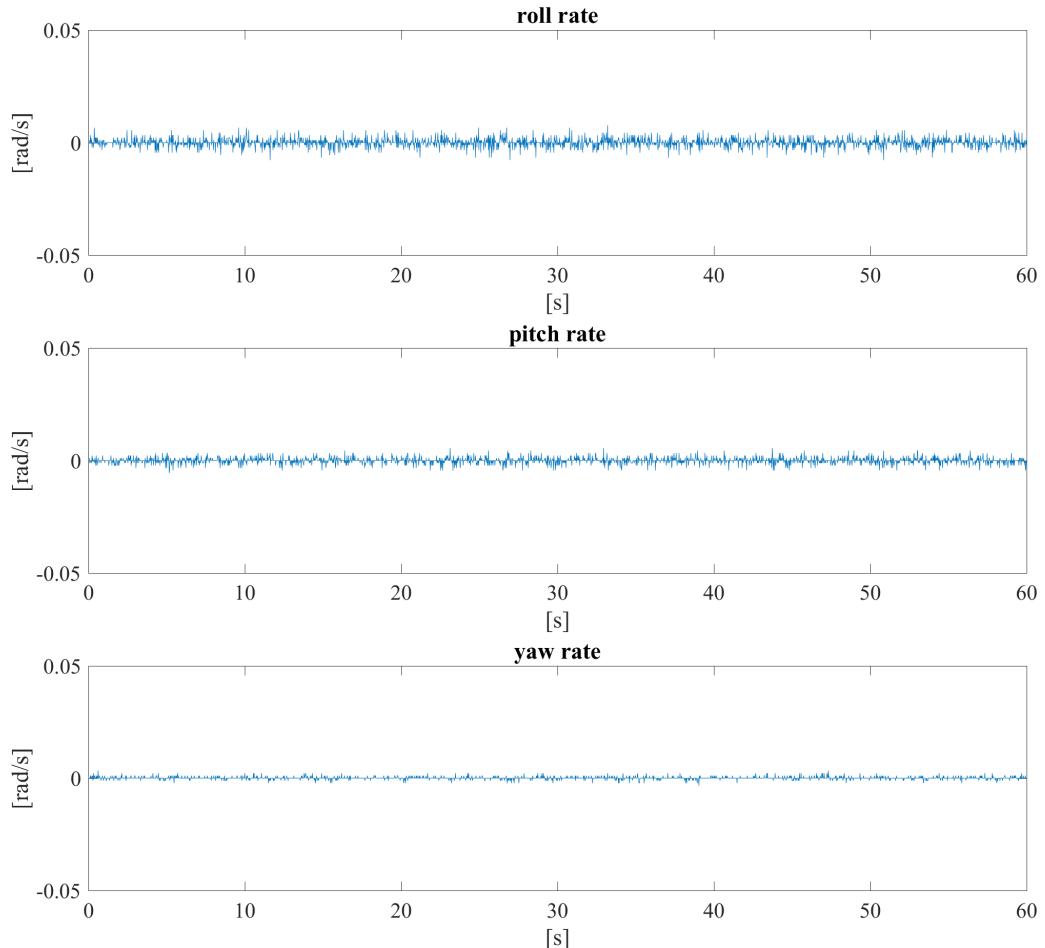


Figura 4.2: Velocità angolari in condizioni statiche.

Discorso analogo si può fare per le velocità angolari (figura 4.2). Poiché il veicolo è in condizioni statiche i dati ottenuti hanno valore nullo come ci si aspettava. Anche in questo caso si può notare che il segnale è affetto da rumore.

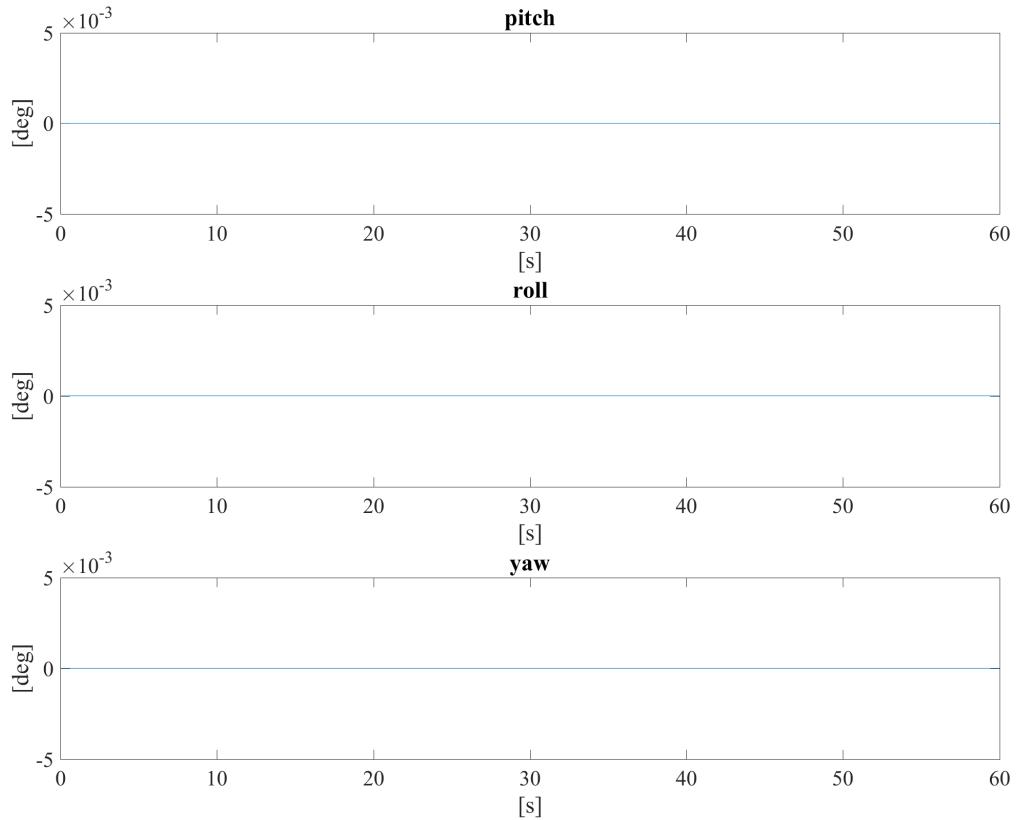


Figura 4.3: *Orientazione in condizioni statiche.*

Il veicolo è orientato verso il nord magnetico e posizionato su un banco di prova e anche in questo caso i risultati rispecchiano quelli aspettati con tutti i valori a zero (figura 4.3). A questo punto si può affermare che la IMU non ha *bias* rispetto alle dimensioni campionate.

#### 4.1.2 Prova su traiettoria circolare

Nella prova il veicolo esegue una traiettoria circolare, mantenendo lo sterzo ad un angolo fisso e velocità costante. La velocità è controllata in feedback mediante un PI. Con questa configurazione sono state eseguite diverse prove variando la velocità da  $0.50$  a  $2.00$   $m\ s^{-1}$ . I dati riportati nella figura 4.4 mostrano i risultati per velocità pari a  $1.75\ m\ s^{-1}$ , circa  $6.3\ km\ h^{-1}$ .

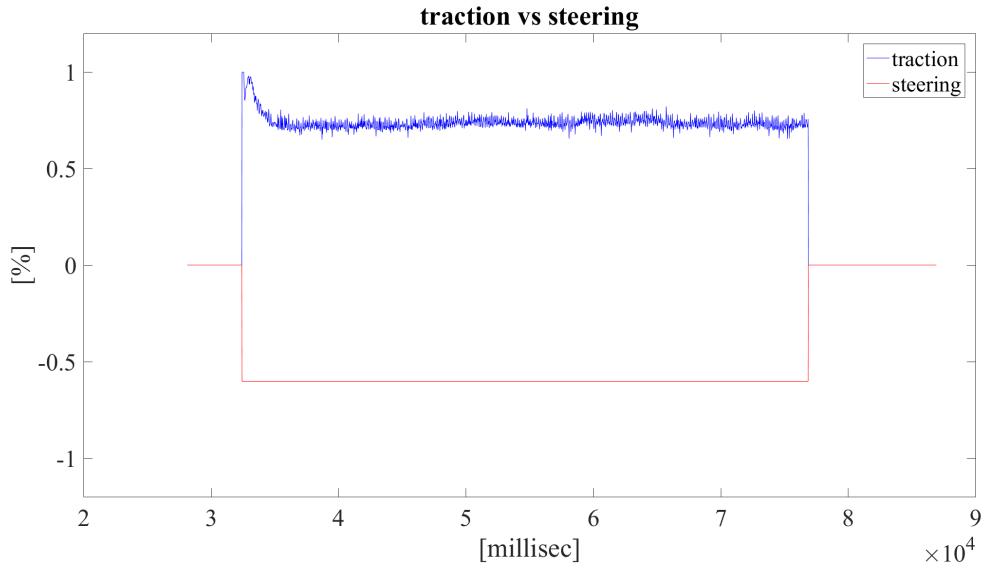


Figura 4.4: Valori in ingresso ai motori di sterzo e trazione durante la traiettoria circolare.

Una indicazione della velocità del veicolo è fornita dagli encoder. Noto il raggio della ruota (0.05cm) e sotto l'ipotesi che questa non si deformi nel corso della prova, assunzione abbastanza realistica vista la notevole rigidezza dello pneumatico, la velocità media del veicolo rispecchia quella imposta dal controllore (figura 4.5).

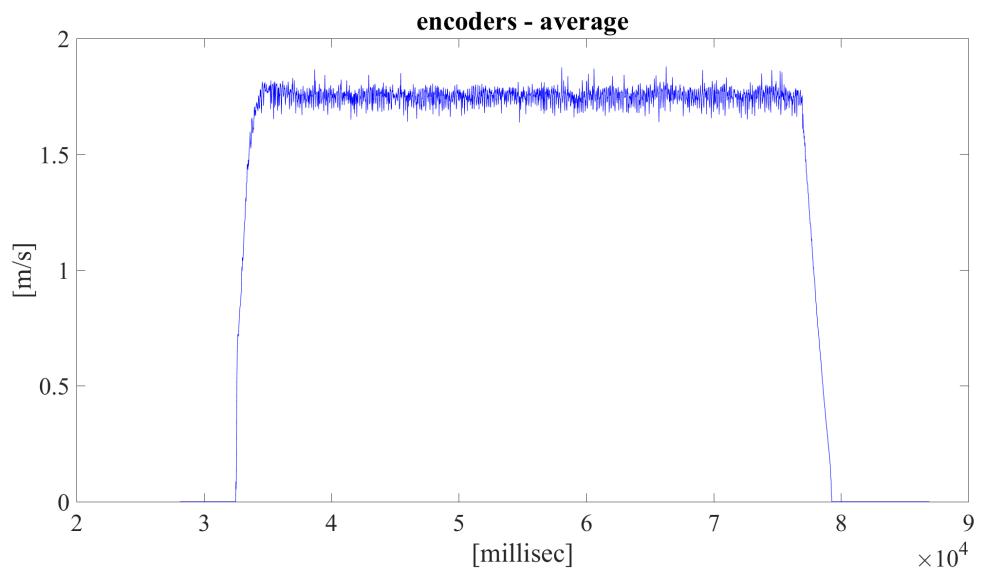


Figura 4.5: Velocità di rotazione media delle ruote durante la traiettoria circolare.

Il verso di rotazione del veicolo si può capire osservando le diverse velocità fornite dagli encoder (figura 4.6). Le ruote a sinistra hanno una velocità di rotazione maggiore di quelle di destra, da cui si può dedurre che il veicolo sta ruotando in senso orario. Un'ultima osservazione è legata alla trazione posteriore del veicolo. Gli encoder posteriori presentano un picco iniziale dovuto alla accelerazione imposta dal motore, con conseguente slip delle ruote prima di arrivare a regime, situazione in cui hanno perfetta aderenza con il terreno. La ruota posteriore destra mostra uno slittamento maggiore della ruota sinistra e questo potrebbe dovuto a una diversa aderenza tra le due ruote nella fase iniziale di accelerazione del veicolo.

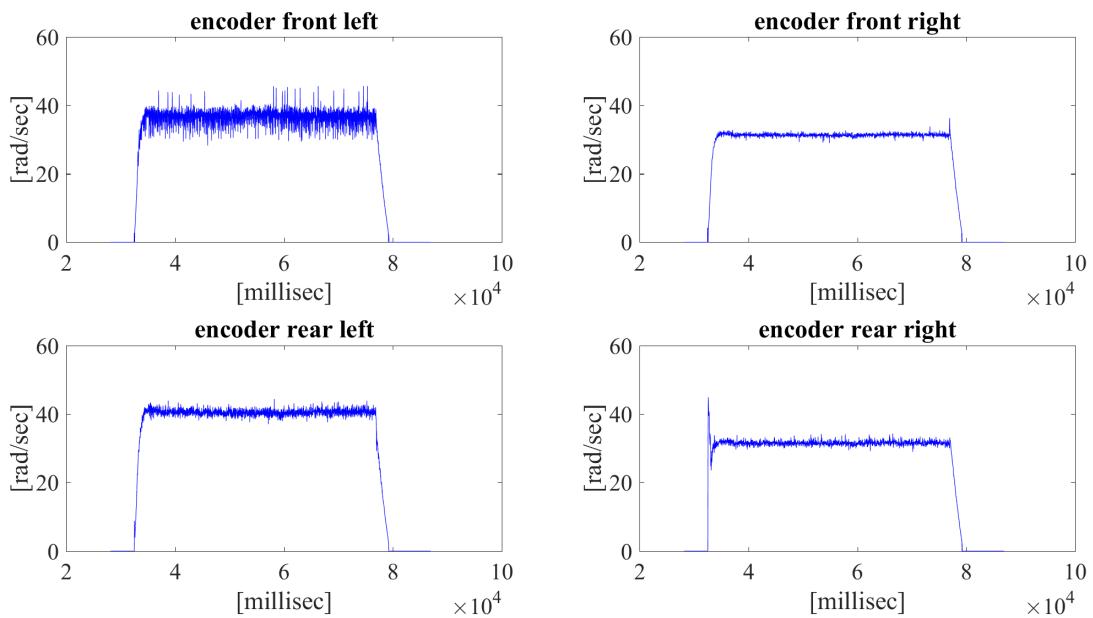


Figura 4.6: *Velocità di rotazione delle ruote durante la traiettoria circolare.*

I dati forniti dal GPS mostrano la traiettoria percorsa dal veicolo durante la prova (figura 4.7).

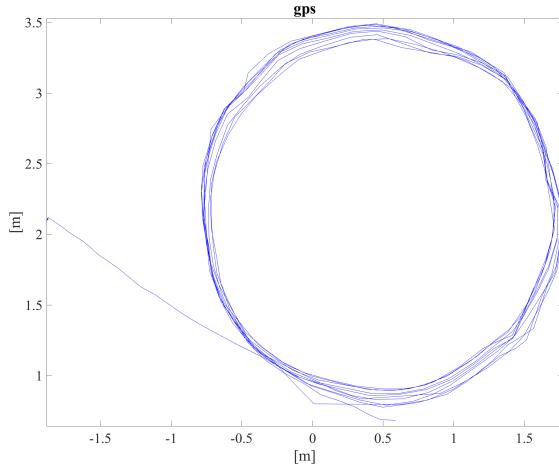


Figura 4.7: *Traiettoria percorsa dal veicolo durante la traiettoria circolare.*

Le accelerazioni fornite dalla IMU sono rumorose, ma osservando i valori ottenuti dal Filtro di Kalman Esteso (figura 4.8) si può notare che l’accelerazione rispetto l’asse  $x$  ha un picco iniziale dovuto alla messa in moto del veicolo, una fase centrale dove l’accelerazione oscilla attorno allo zero e una decelerazione finale in concomitanza con l’arresto del veicolo.

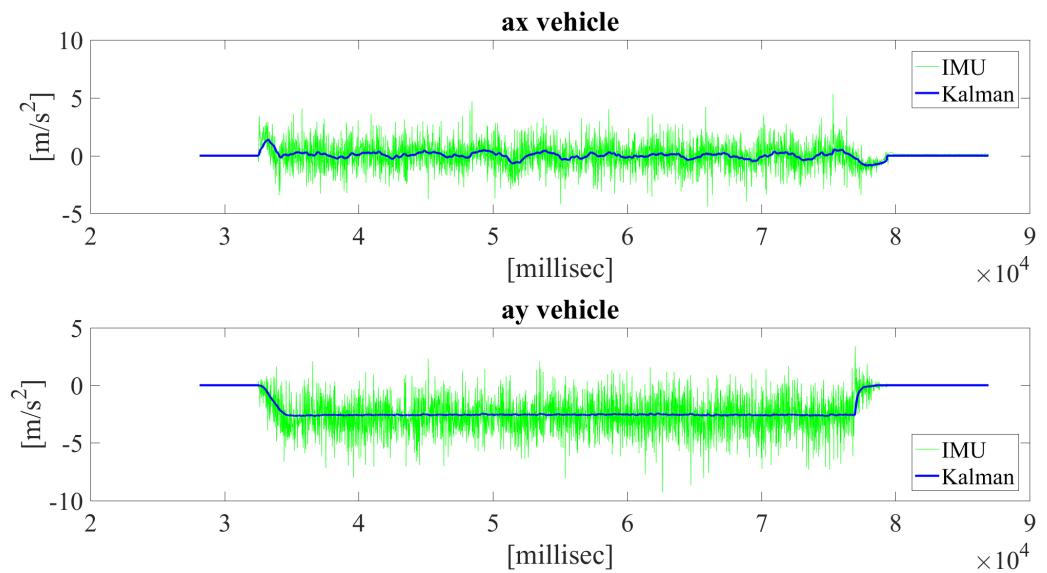


Figura 4.8: *Accelerazioni durante la traiettoria circolare.*

La rotazione del veicolo è evidente osservando l'accelerazione lungo l'asse y (figura 4.8), l'angolo di yaw (figura 4.9) e lo yaw rate (figura 4.10). I segnali provenienti dalla IMU e dagli encoder sono affetti da rumore che, oltre alle cause già citate, sono probabilmente dovute a vibrazioni del pianale su cui sono montati i sensori e ad una pavimentazione non perfetta. Il *sideslip angle* ricavato si assesta attorno ai 5° (figura 4.11).

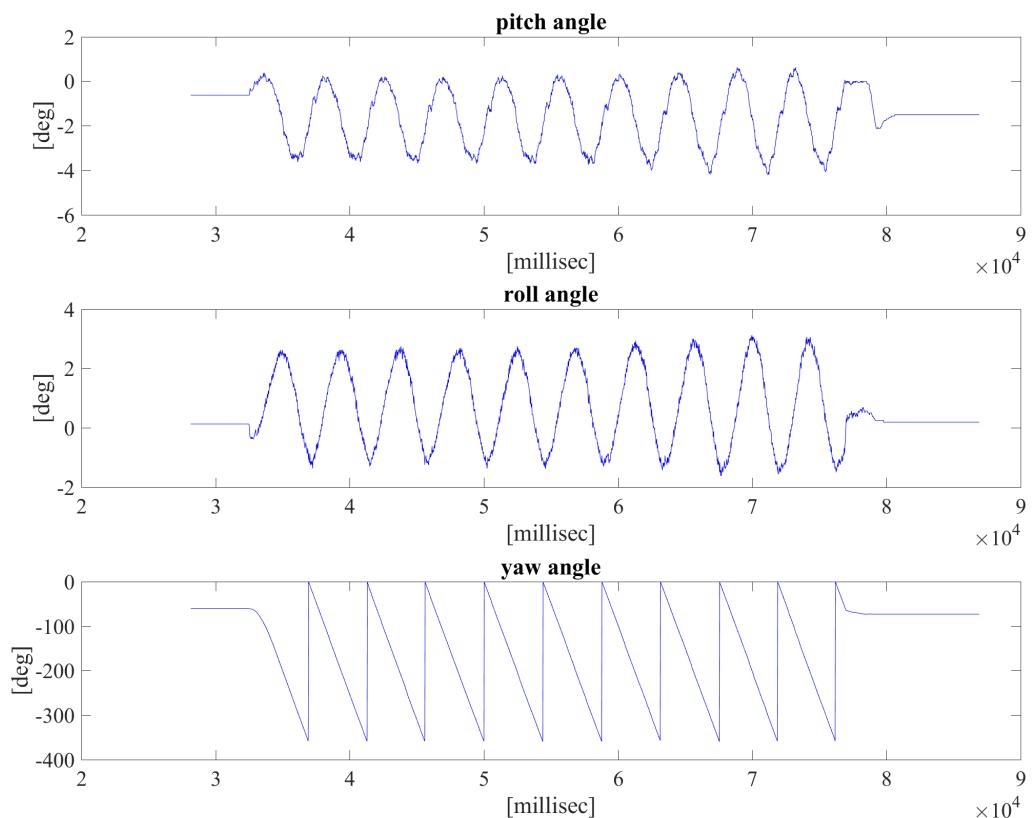


Figura 4.9: Angoli in condizioni durante la traiettoria circolare.

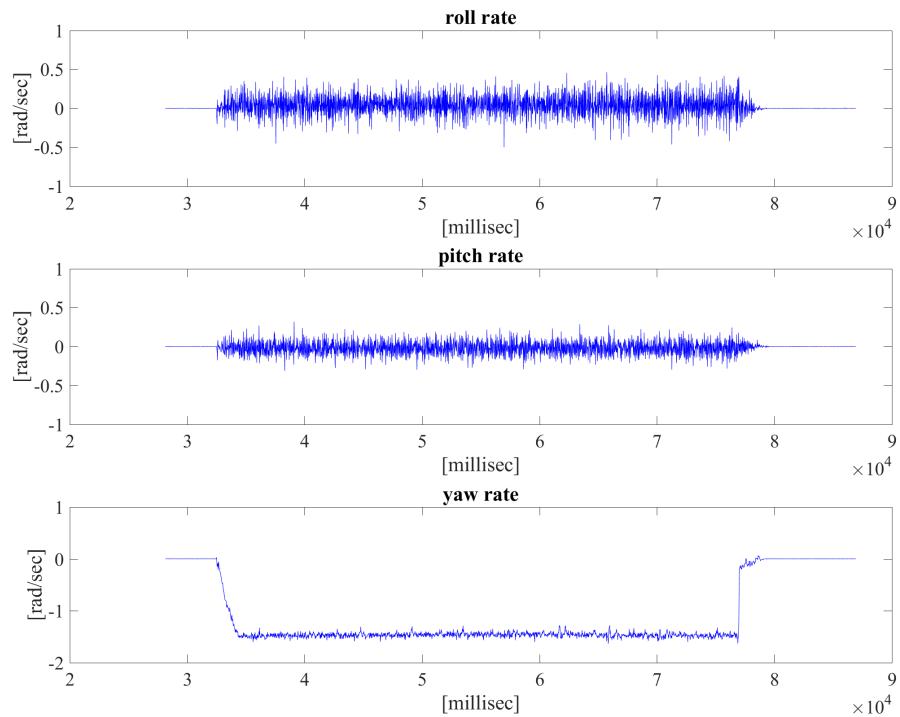


Figura 4.10: *Velocità angolari durante la traiettoria circolare.*

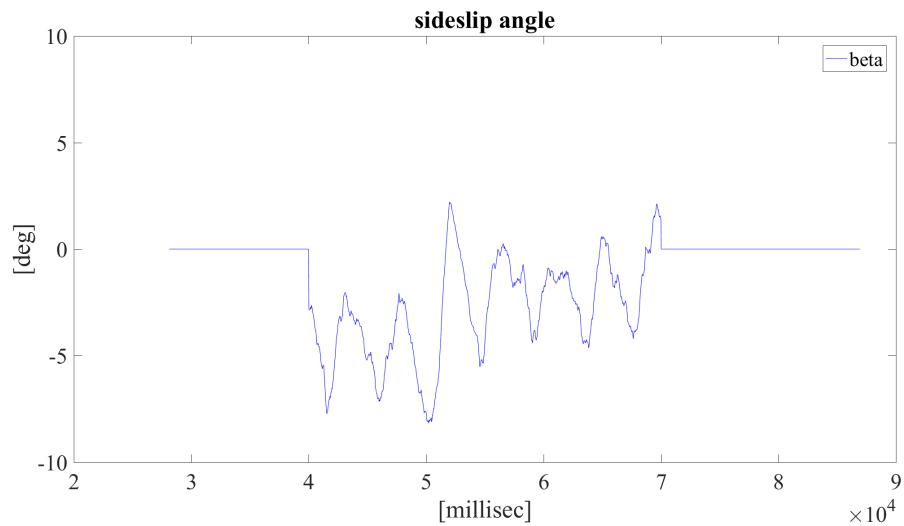


Figura 4.11: *Sideslip angle durante la traiettoria circolare.*

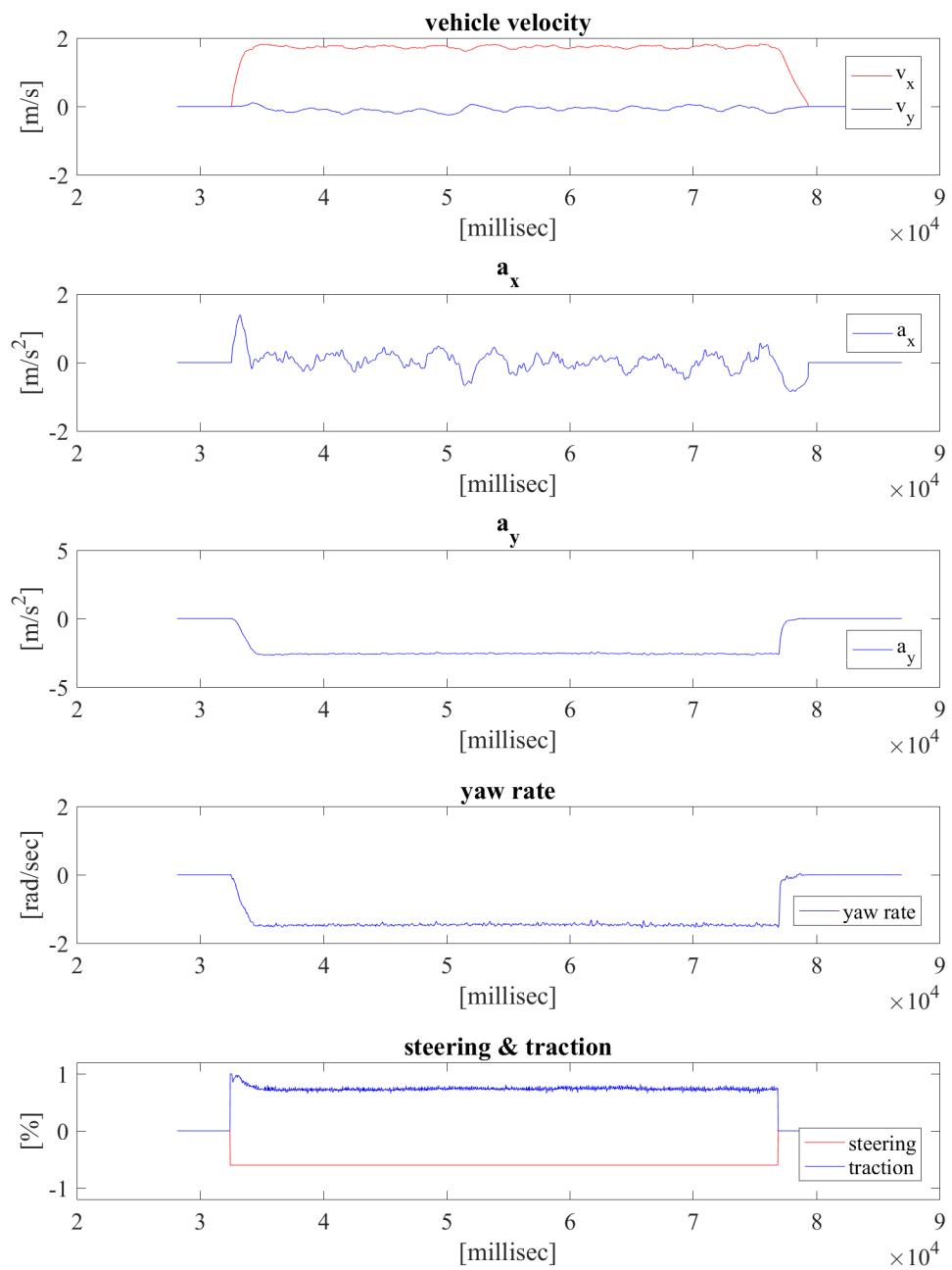


Figura 4.12: *Stato del veicolo durante la traiettoria circolare.*

Il Filtro di Kalman Esteso permette di ricostruire la traiettoria percorsa dal veicolo anche nei punti di assenza del segnale del GPS (figura 4.13).

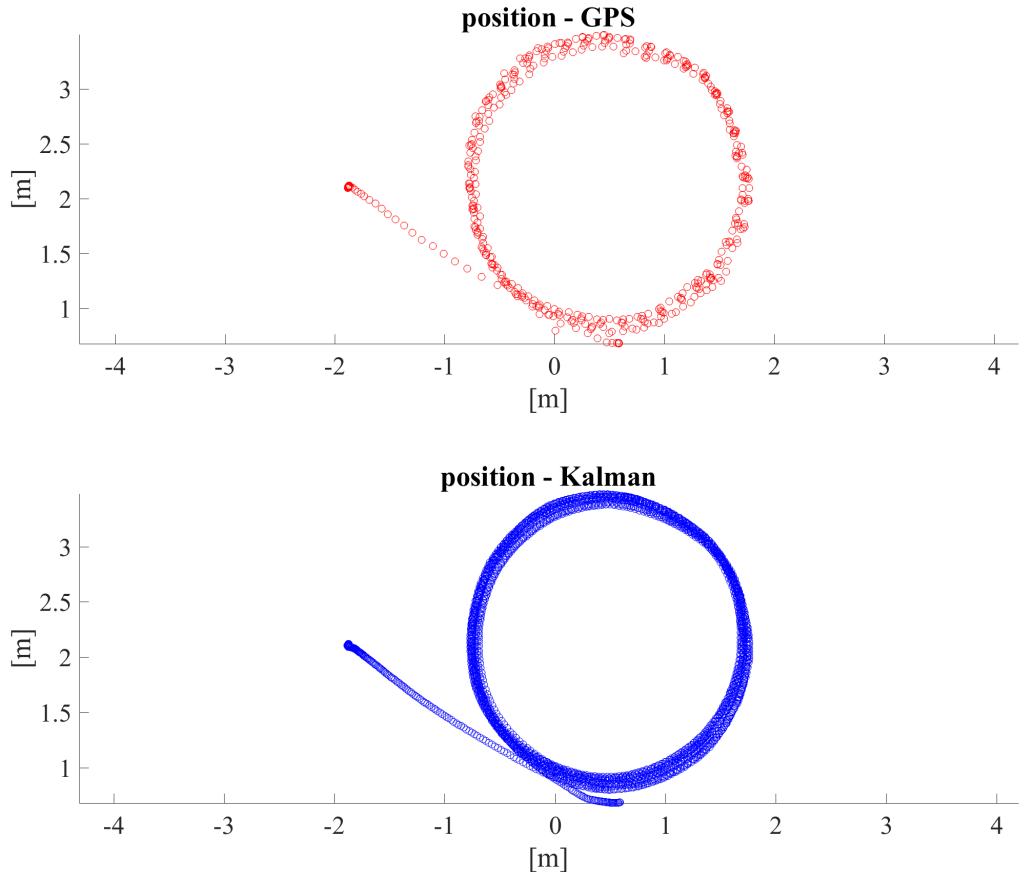


Figura 4.13: *Traiettoria ricavata con EKF confrontata con quella fornita dal GPS durante la prova su traiettoria circolare.*

### 4.1.3 Prova su traiettoria rettilinea

Questa prova vede il veicolo seguire una traiettoria rettilinea imponendo l'angolo di sterzo nullo e una velocità costante imposta tramite controllore PI fino allo spegnimento del motore. Anche in questo caso sono state eseguite prove a velocità diverse, per brevità si riportano graficati i valori della prova eseguita a  $1.50 \text{ m s}^{-1}$ , circa  $5.4 \text{ km h}^{-1}$ .

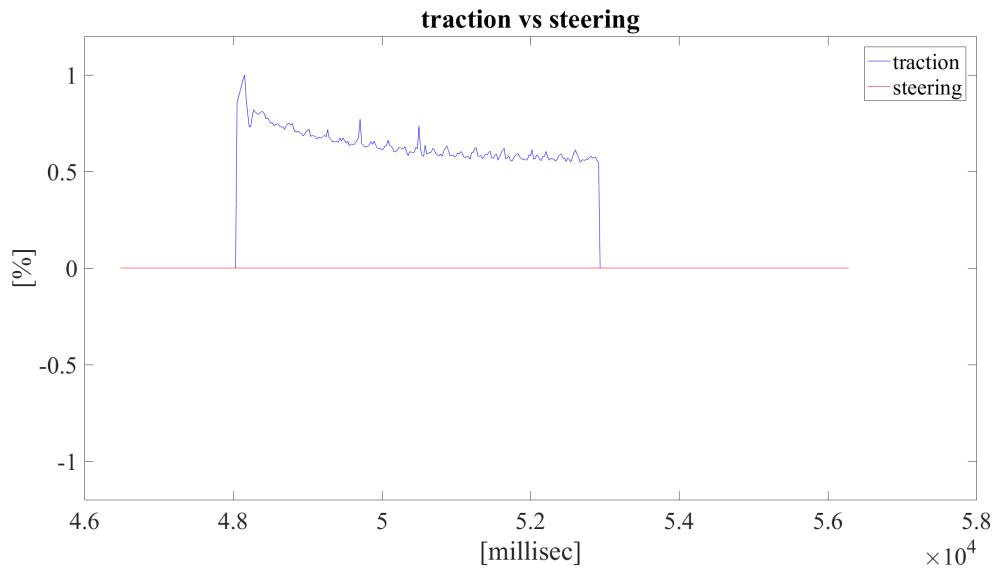


Figura 4.14: Valori in ingresso ai motori di sterzo e trazione durante la traiettoria rettilinea.

Osservando le accelerazioni (figura 4.15) si può notare il picco della fase iniziale e la decelerazione nella fase finale lungo l'asse  $x$ , mentre lungo l'asse  $y$  l'accelerazione è praticamente a zero per tutta la durata della prova.

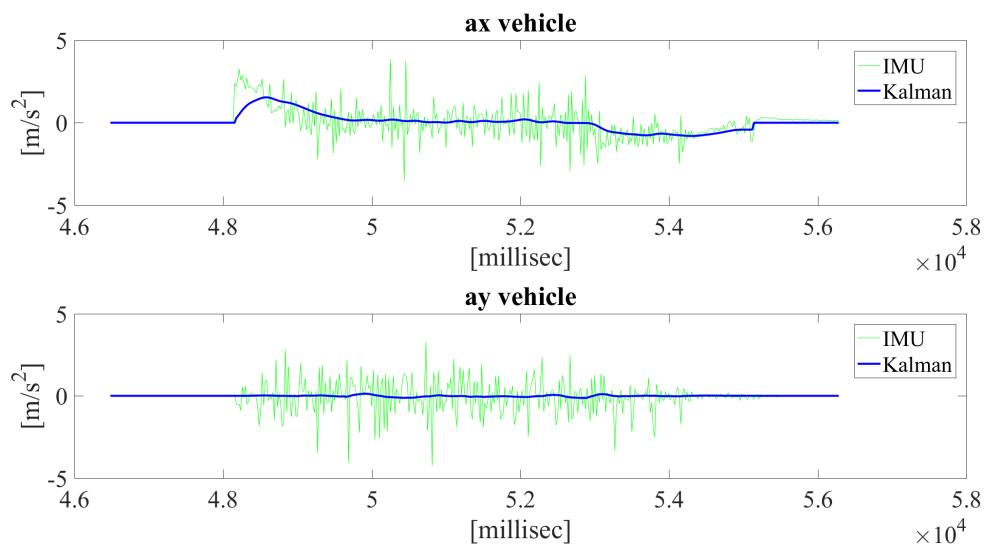


Figura 4.15: Accelerazioni durante la traiettoria rettilinea.

Osservando l'angolo di *yaw* e lo *yaw rate* (figura 4.16) si può capire che il veicolo non si muove perfettamente su una linea retta, ma presenta una leggera deriva e questo potrebbe essere dato da:

- un terreno non perfettamente uniforme sul quale è stata eseguita la prova;
- una non perfetta aderenza delle ruote;
- una distribuzione di coppia non uniforme tra le due ruote da parte del differenziale a causa di attriti interni che generano una piccola coppia di imbardata.

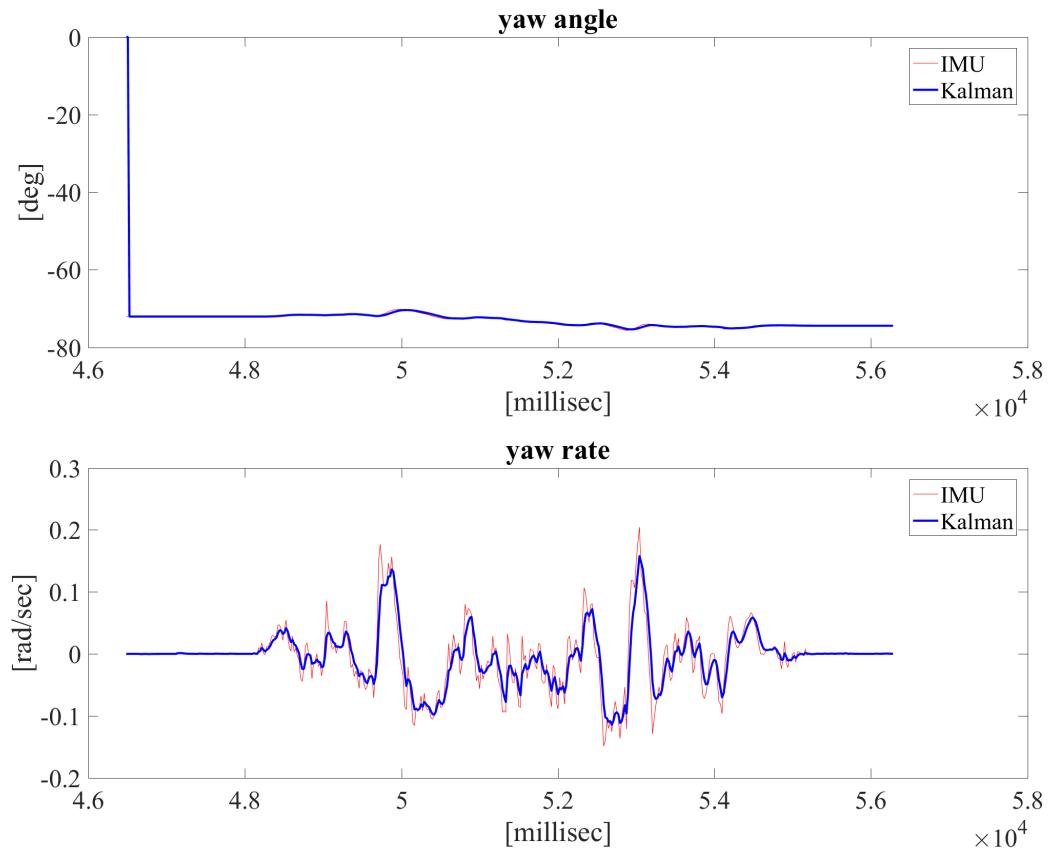


Figura 4.16: Angolo di yaw e yaw rate durante la traiettoria rettilinea.

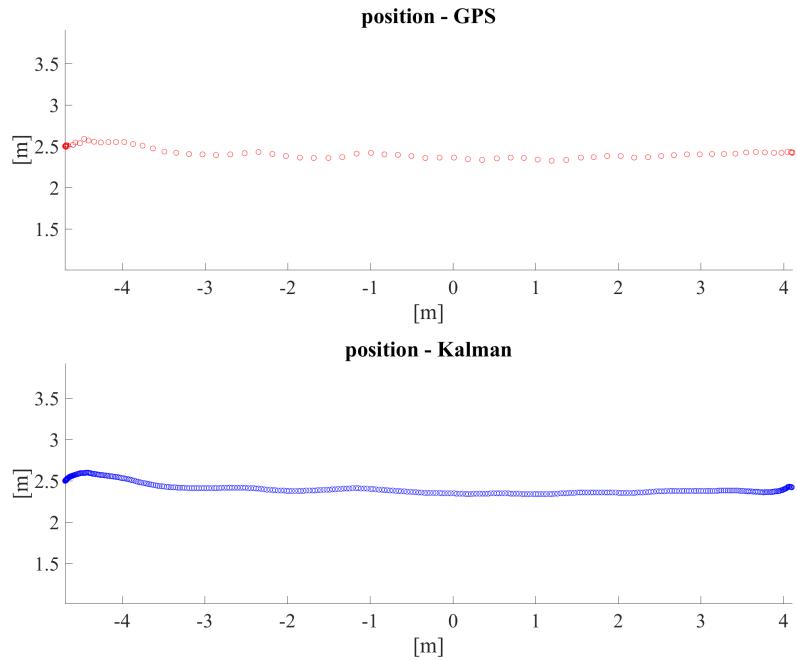


Figura 4.17: *Traiettoria ricavata con EKF confrontata con quella fornita dal GPS durante la prova su traiettoria circolare.*

L'angolo di *sideslip* è quasi nullo (figura 4.18), come ci si aspetta da un veicolo in traiettoria rettilinea.

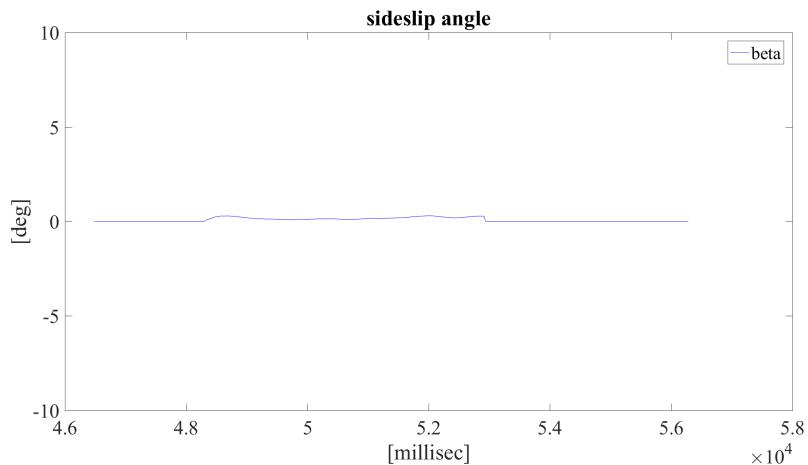


Figura 4.18: *Sideslip angle durante la traiettoria rettilinea.*

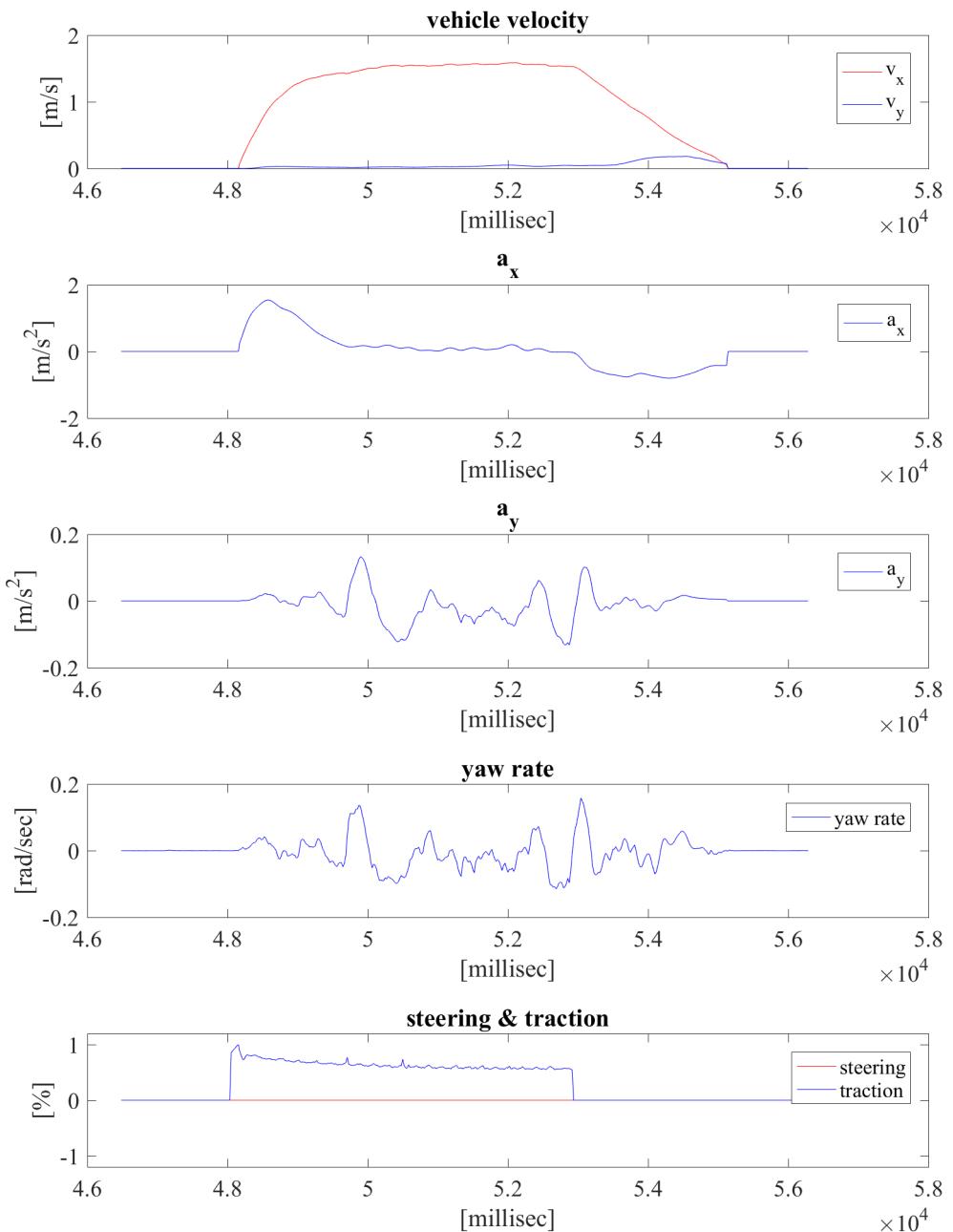


Figura 4.19: *Stato del veicolo durante la traiettoria rettilinea.*

#### 4.1.4 Prova di “lane change”

Utilizzando il *preview control* è stata imposta al veicolo una manovra di *lane change* (4.20).

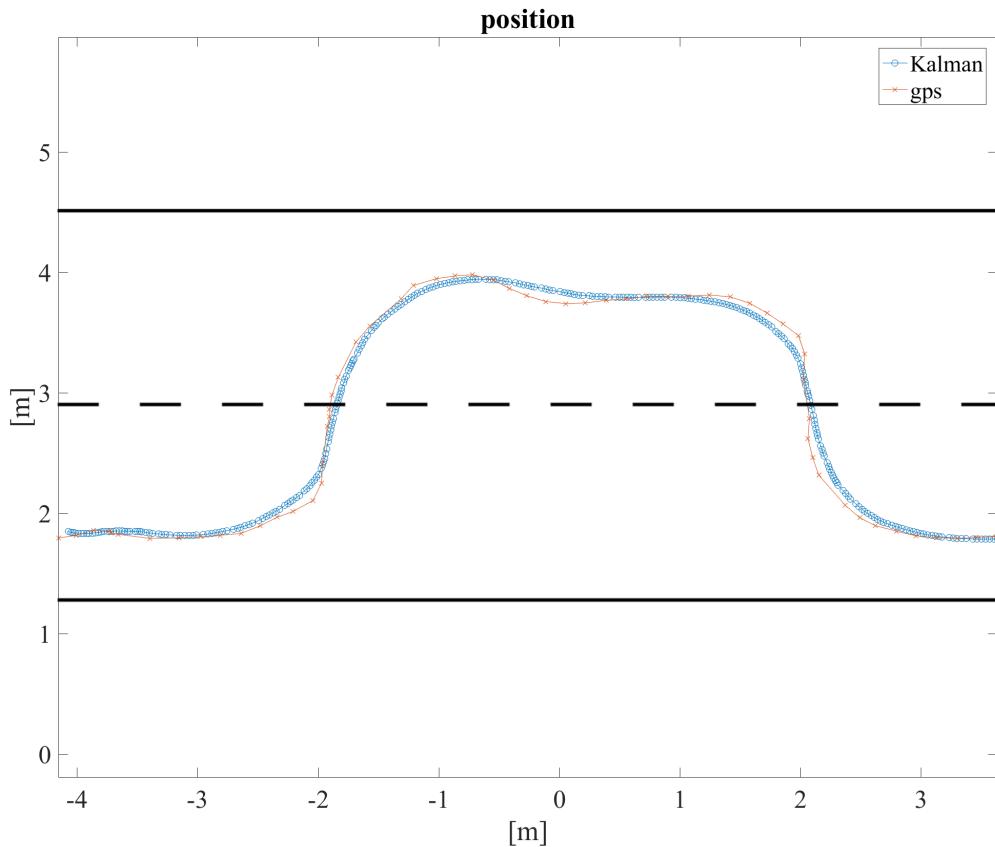


Figura 4.20: *Manovra di lane change*.

Mentre il veicolo in moto segue una traiettoria rettilinea gli viene imposto un cambio di corsia passando su una corsia di sinistra e successivamente rientrando nella corsia iniziale. La manovra viene eseguita mantenendo una velocità costante. Osservando la figura 4.21 si nota che, per compiere questa manovra, lo sterzo satura.

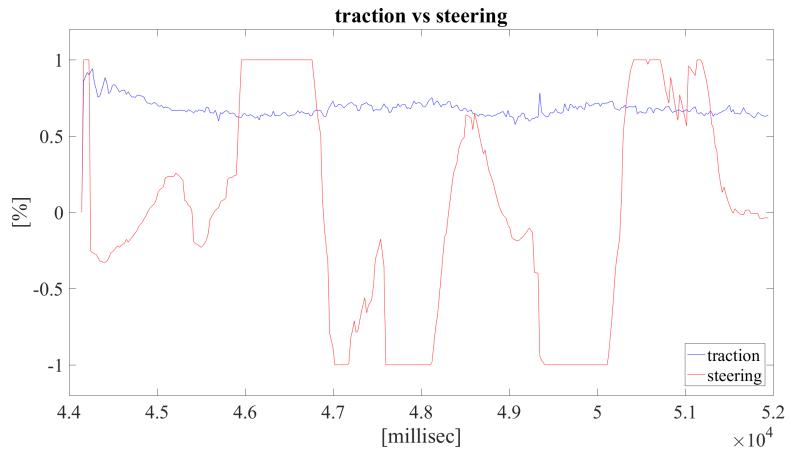


Figura 4.21: Valori in ingresso ai motori di sterzo e trazione durante il lane change.

Osservando l’accelerazione (figura 4.22), lungo l’asse  $x$  è possibile osservare il picco iniziale dovuto alla messa in moto del veicolo, che si stabilizza su valori quasi nulli quando il veicolo raggiunge la velocità di crociera. Spostando l’attenzione sull’accelerazione lungo l’asse  $y$  è possibile osservare i quattro picchi dovuti alle sterzate eseguite dal veicolo durante la prova.

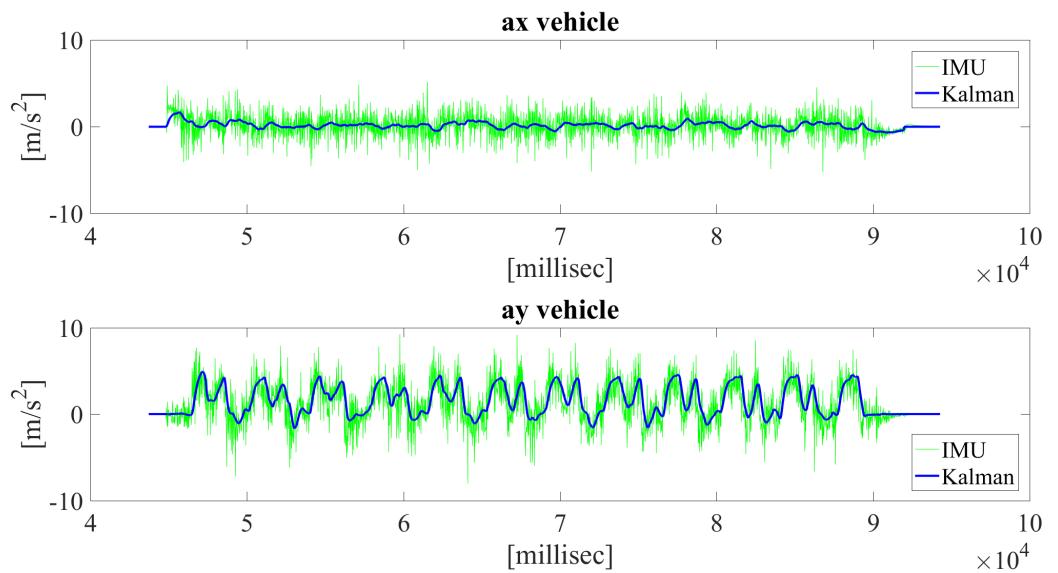


Figura 4.22: Accelerazioni durante il lane change.

Discorso analogo si può fare per lo *yaw angle* e lo *yaw rate* (figura 4.23) che mettono in luce le manovre eseguite dal veicolo.

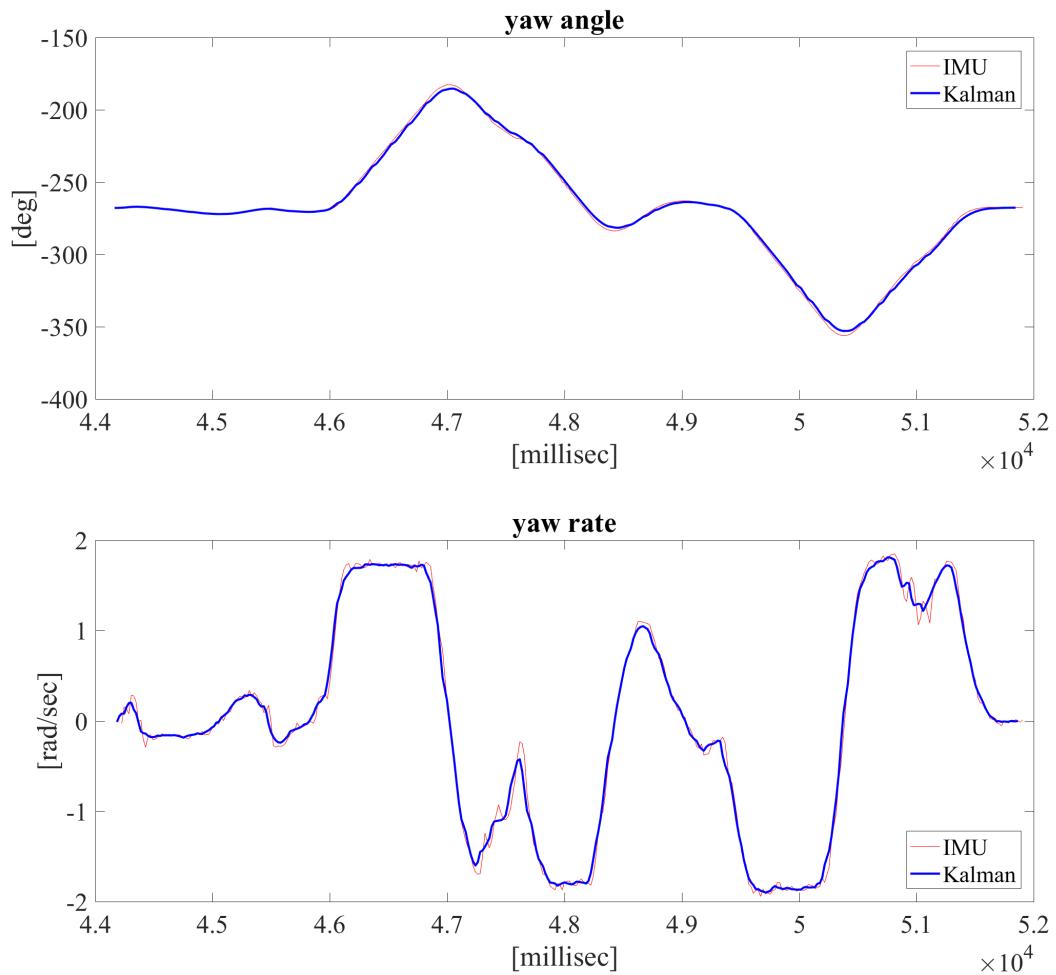


Figura 4.23: Valori in ingresso ai motori di sterzo e trazione durante il lane change.

La traiettoria data dal GPS e quella ricostruita sono mostrate nella figura 4.24:

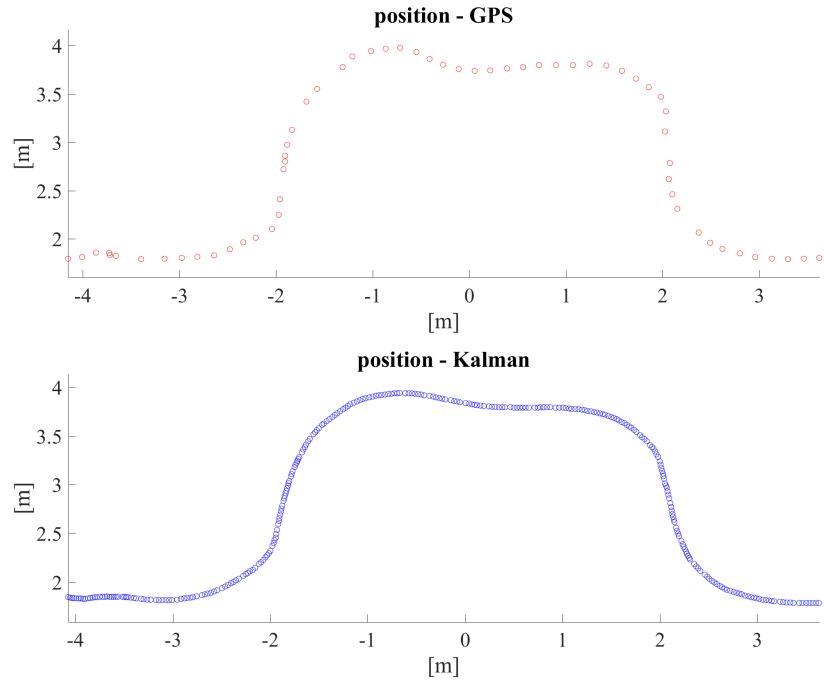


Figura 4.24: *Traiettoria ricostruita con EKF confrontata con quella fornita dal GPS durante il lane change.*

Il *sideslip angle* ricavato è mostrato nella figura 4.25 e Lo stato del veicolo durante la prova di *lane change* viene schematizzato dalla figura 4.26.

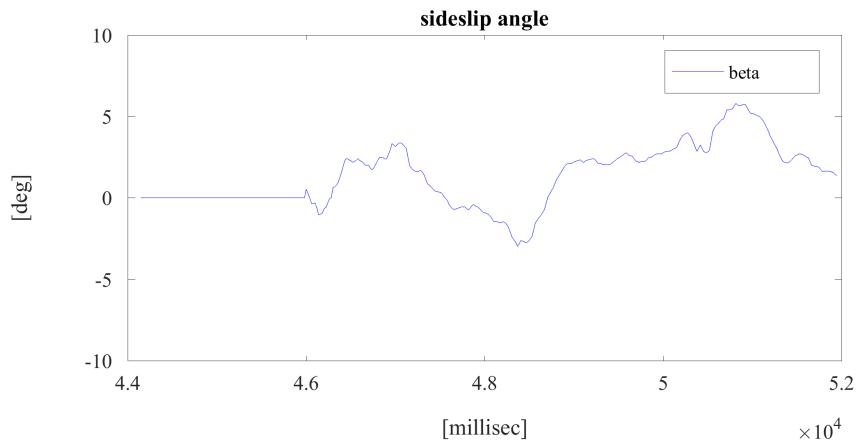


Figura 4.25: *Sideslip angle durante il lane change.*

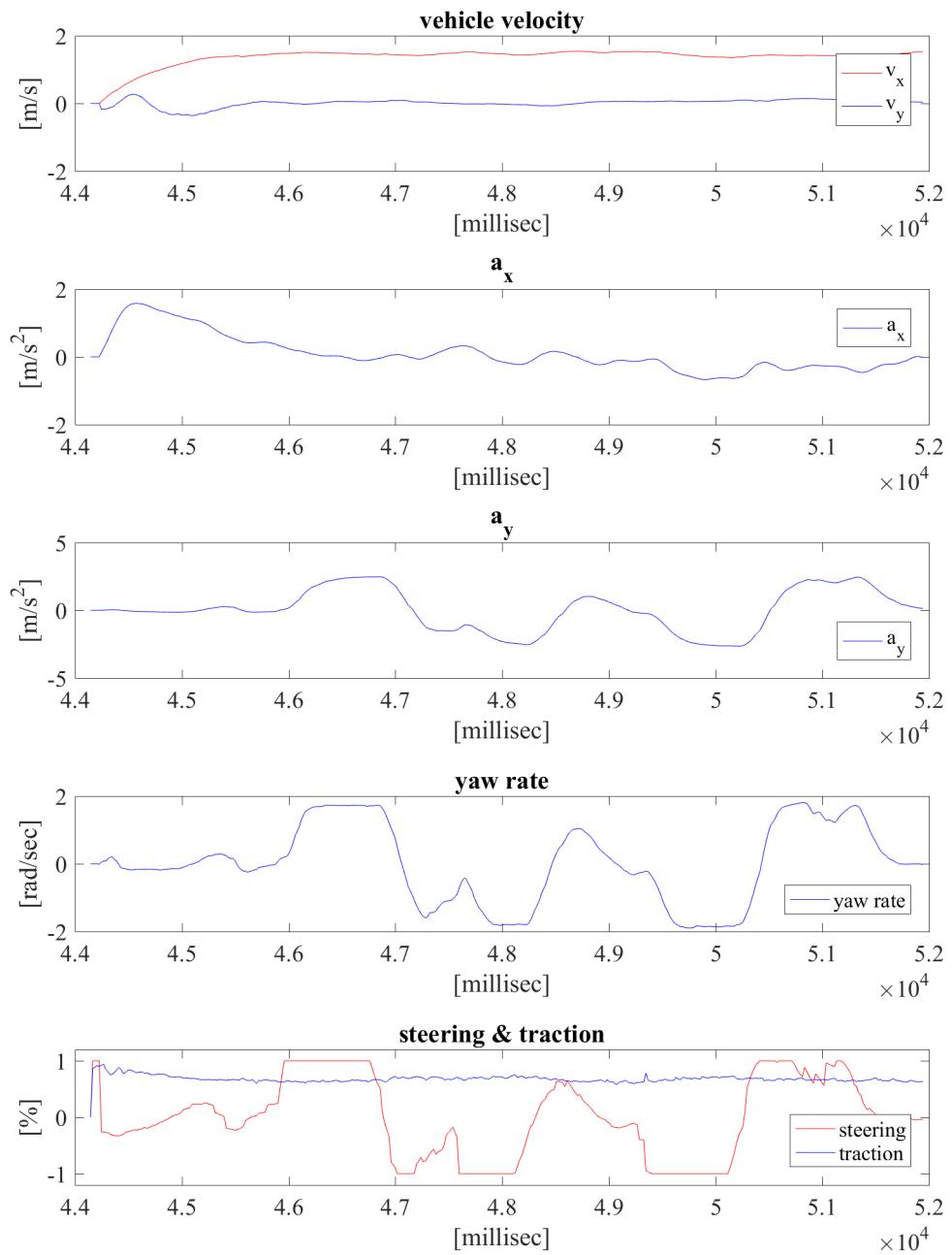


Figura 4.26: Stato del veicolo durante il lane change.

### 4.1.5 Prova su circuito

Viene qui di seguito mostrato il comportamento del veicolo durante una prova eseguita su un circuito ellittico. La prova è stata eseguita in laboratorio tracciando il circuito a terra e imponendo al veicolo la traiettoria da seguire utilizzando il *preview controller*.

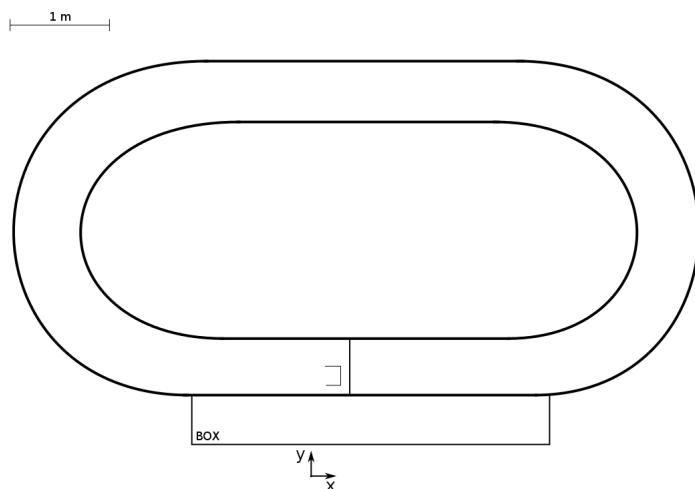


Figura 4.27: Circuito nel quale è stata eseguita la prova.

Nelle figure 4.28 e 4.29 è possibile osservare la traiettoria fornita dal GPS confrontata con quella ricostruita mediante il Filtro di Kalman Esteso e il comportamento del veicolo nel corso della prova rispetto al tracciato.

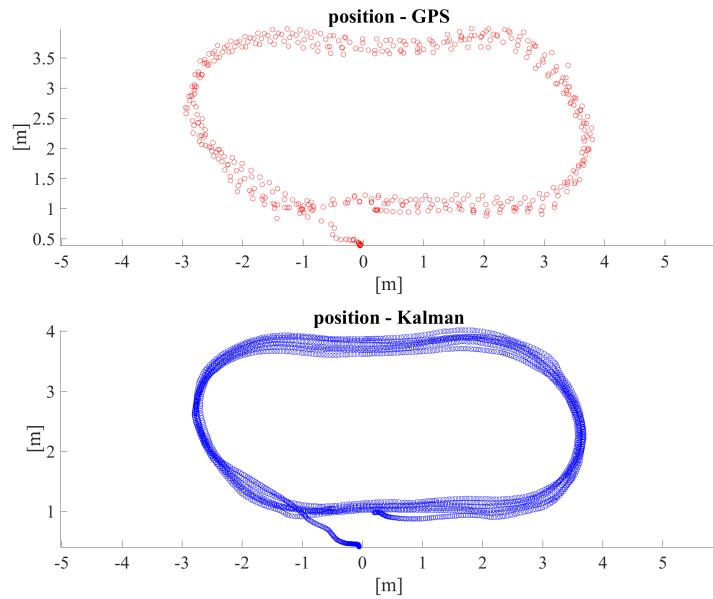


Figura 4.28: *Traiettoria ricostruita con EKF confrontata con quella fornita dal GPS durante la prova nel circuito.*

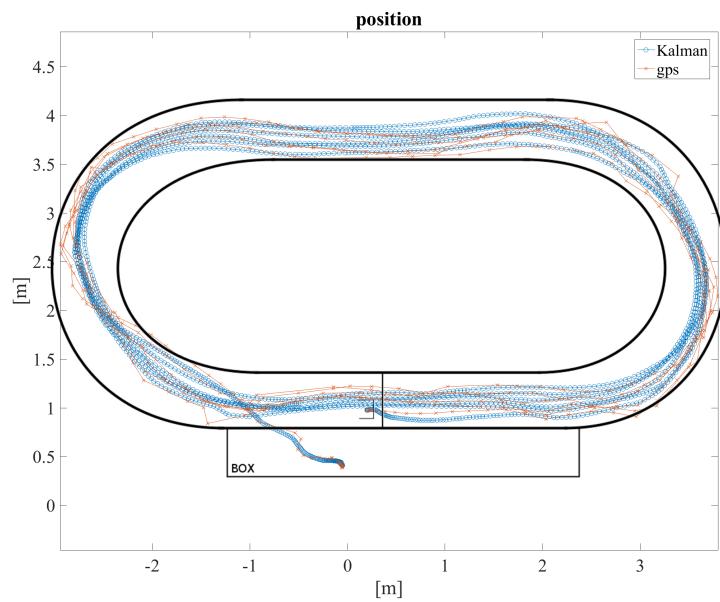


Figura 4.29: *Traiettoria percorsa dal veicolo durante la prova nel circuito.*

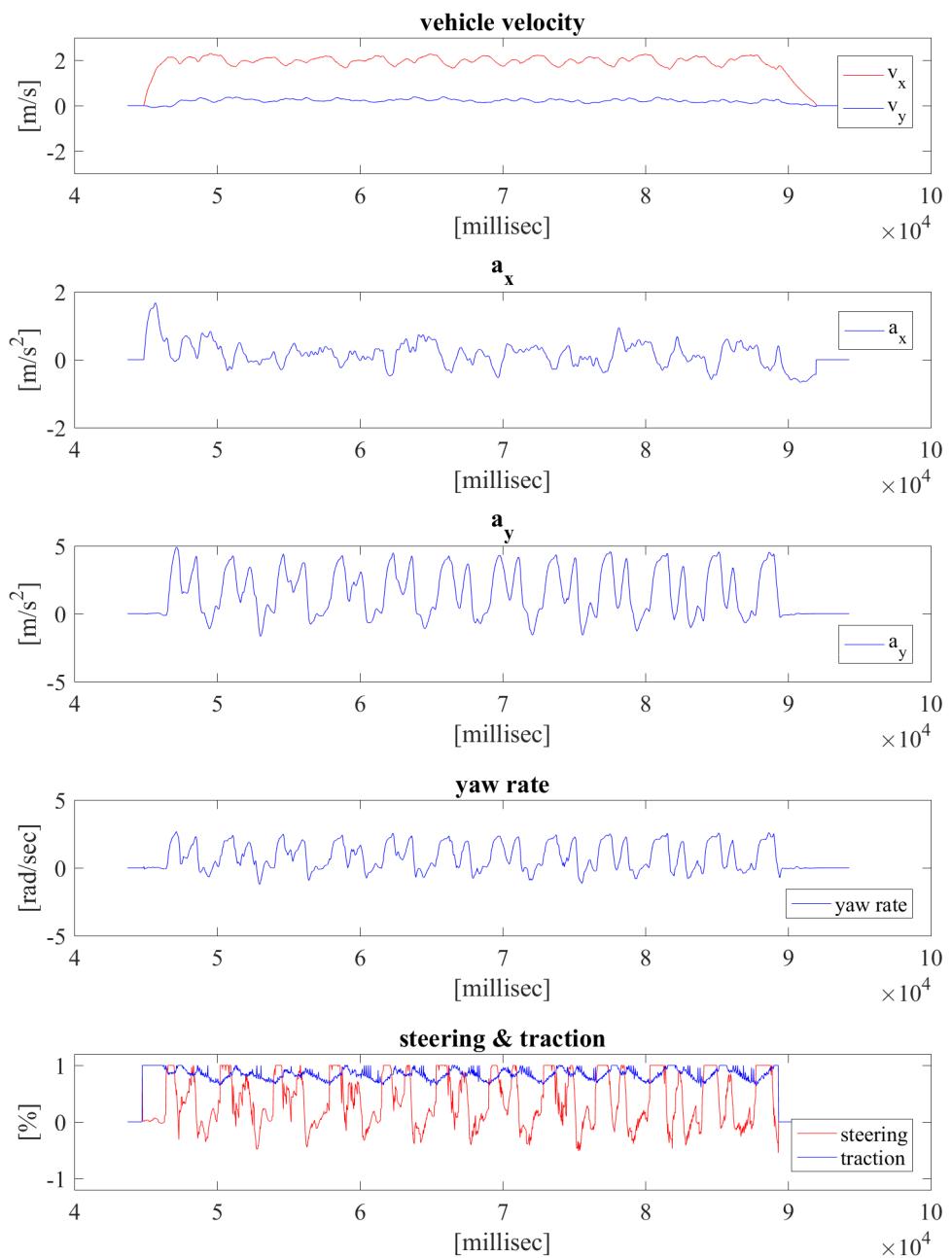


Figura 4.30: Stato del veicolo durante la prova nel circuito.

Il *sideslip angle* (figura 4.31) oscilla attorno ai  $5^\circ$  quando il veicolo è impegnato a percorrere la traiettoria curvilinea e si assesta su valori più bassi nei tratti rettilinei.

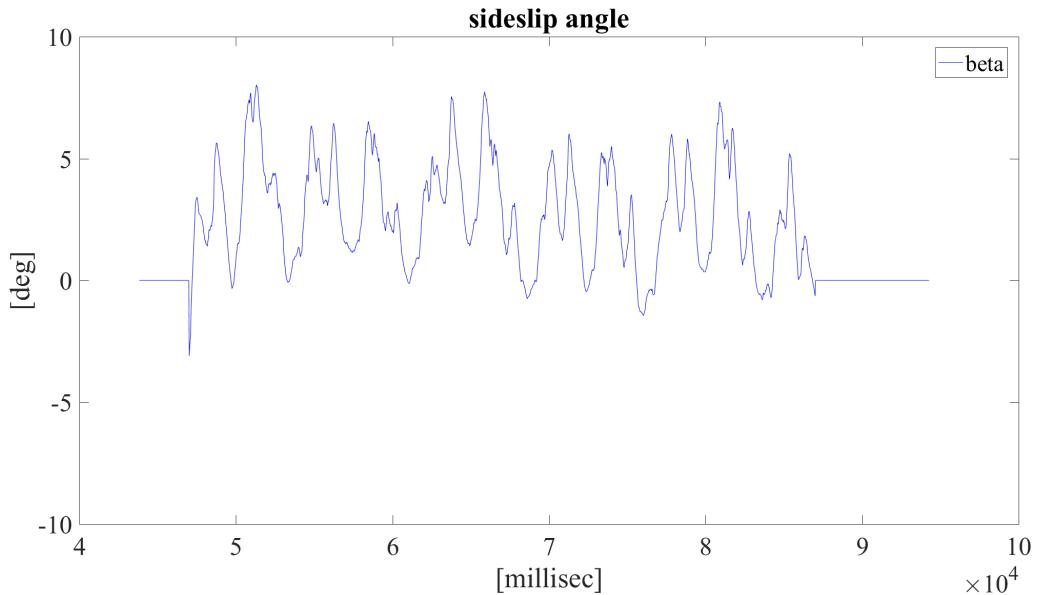


Figura 4.31: *Sideslip angle* durante la traiettoria rettilinea.

Lo stato del veicolo (figura 4.30) nel corso della prova è conforme a quanto previsto:

- il veicolo riesce a mantenere la velocità longitudinale imposta di  $2.00 \text{ m s}^{-1}$ , circa  $7.2 \text{ km h}^{-1}$ , per tutta la durata della prova;
- l'accelerazione lungo l'asse  $x$ , dopo il picco iniziale, si stabilizza su valori molto bassi e oscilla in corrispondenza delle curve compiute dal veicolo;
- l'accelerazione lungo l'asse  $y$ , confrontata con lo *yaw rate*, presenta dei picchi in corrispondenza del cambio dell'angolo di imbardata del veicolo.

## 4.2 Prove sul filtro di Kalman

Come descritto nella sezione 3.5, la matrice di covarianza del disturbo sullo stato è stata inizialmente ricavata sperimentalmente e successivamente pesata e aggiustata per ottimizzare il Filtro di Kalman. Il processo di ottimizzazione dell'algoritmo è stato eseguito

prendendo in considerazione inizialmente solo le accelerazioni, poiché sono misure ottenute direttamente dalla IMU, per poi passare alle velocità, misure ricavate dalla fusione delle informazioni provenienti dagli encoder e dalla IMU, e infine le coordinate del veicolo, ottenute dalla fusione delle informazioni provenienti anche dal GPS. Qui di seguito è riportato come variano le accelerazioni  $a_x$  (figura 4.32) e  $a_y$  (figura 4.33), le velocità  $v_x$  (figura 4.34) e  $v_y$  (figura 4.35), il *sideslip angle* (figura 4.36) e le traiettorie ricostruite (figura 4.37) al variare della matrice  $Q$ . Nello specifico è stata presa in esame la prova su traiettoria circolare descritta nella sezione 4.1.2. Per vedere come varia il comportamento del filtro in funzione di  $Q$ , si è scelto di confrontare la matrice  $Q_{ottima}$  con la medesima matrice scalata di un fattore  $10^{-1}$ ,  $10$ ,  $10^2$ , dove con  $Q_{ottima}$  si intende la matrice  $Q$  implementata nel filtro realizzato. Osservando le figure 4.32 e 4.33 relative alle accelerazioni  $a_x$  e  $a_y$  si può osservare che aumentando il fattore di scala ( $Q_o10$  e  $Q_o10^2$ ) si ha un peggioramento della prestazione del filtro ottenendo output più rumorosi. Diminuendo il fattore di scala ( $Q_o10^{-1}$ ) si ottengono output meno soggetti a rumore, ma che presentano un *delay*. Considerazioni analoghe si possono fare osservando i risultati mostrati nelle figure 4.34, 4.35, 4.36 e 4.37.

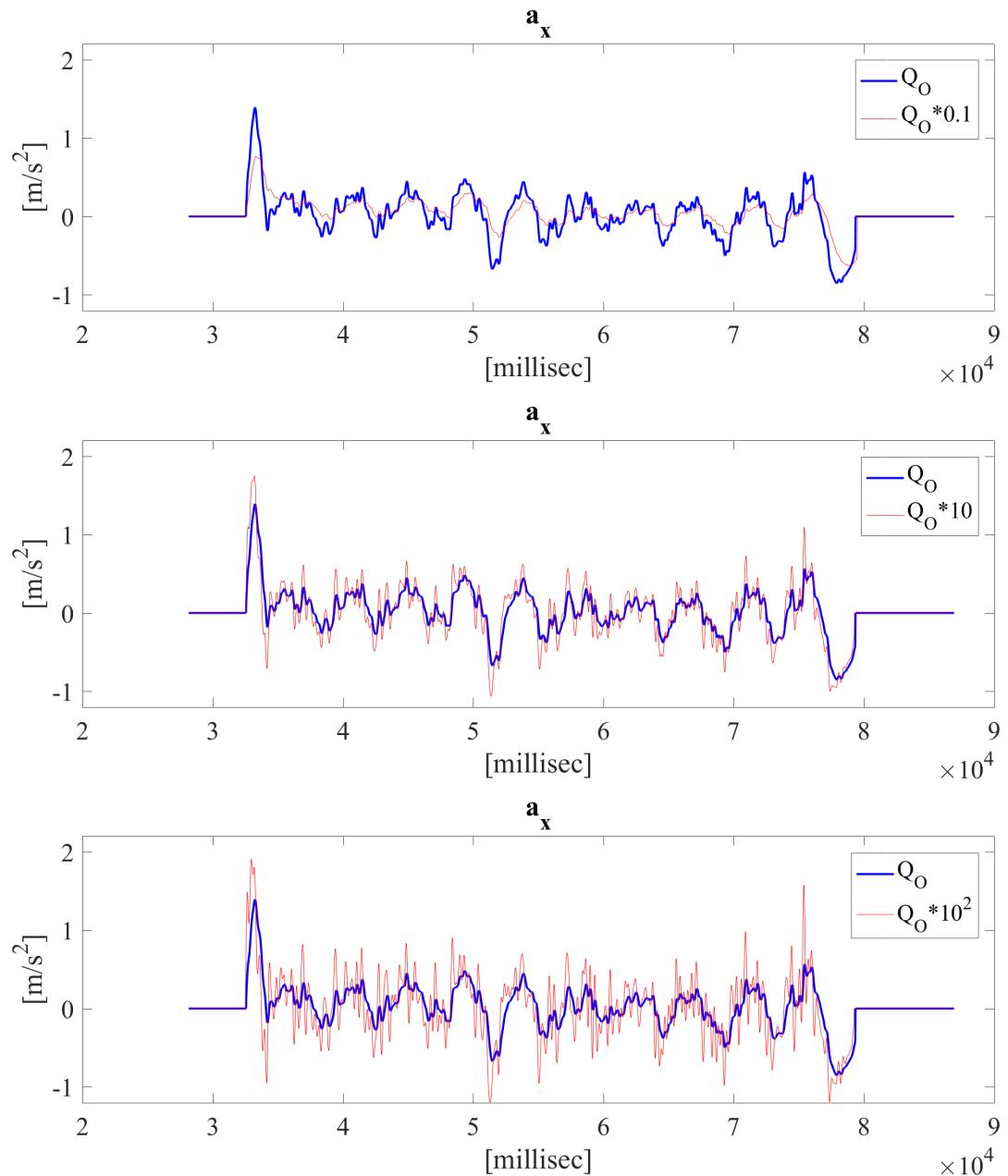


Figura 4.32: Valori di  $a_x$  ottenuti al variare di  $Q$ .

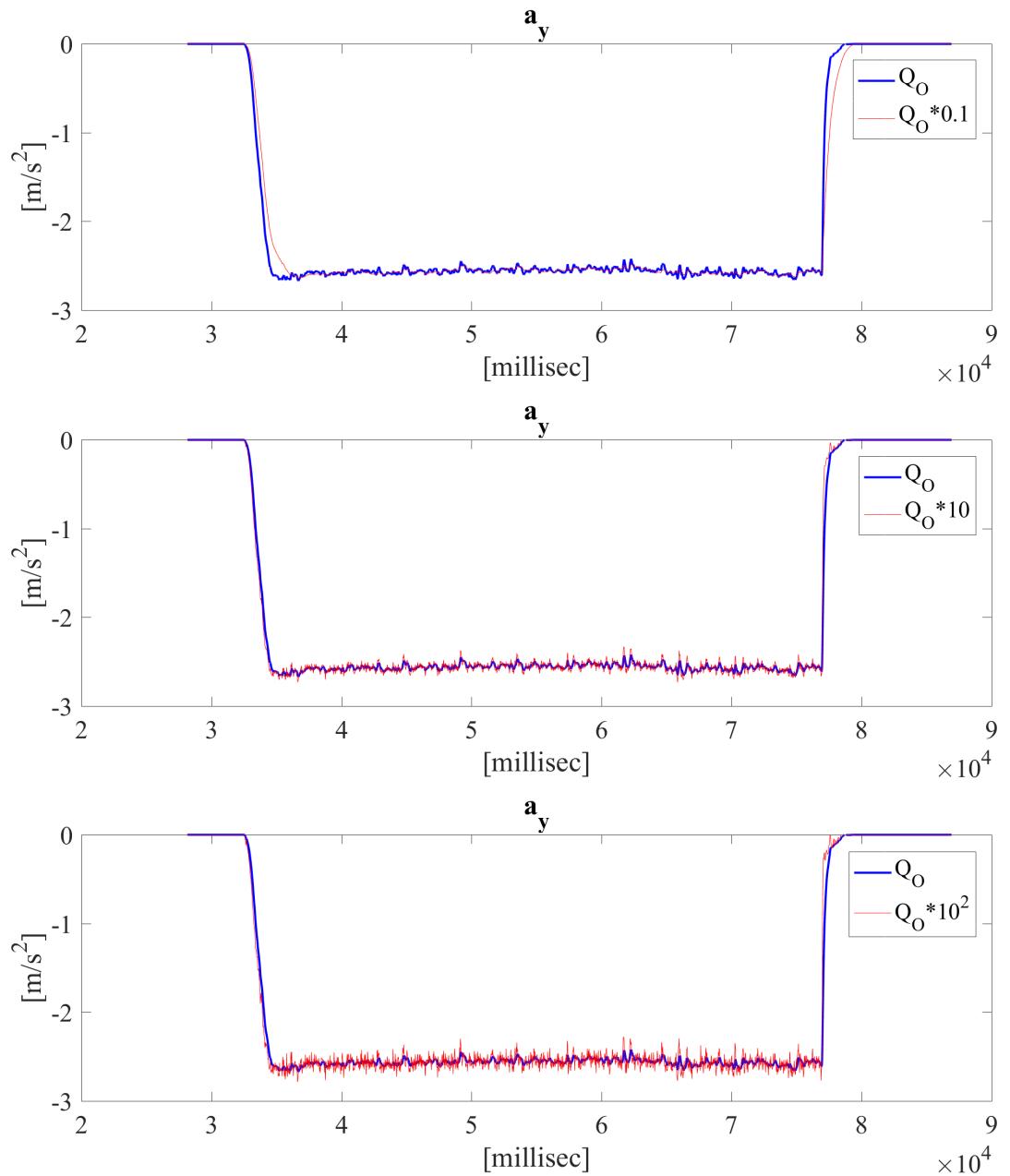


Figura 4.33: Valori di  $a_y$  ottenuti al variare di  $Q$ .

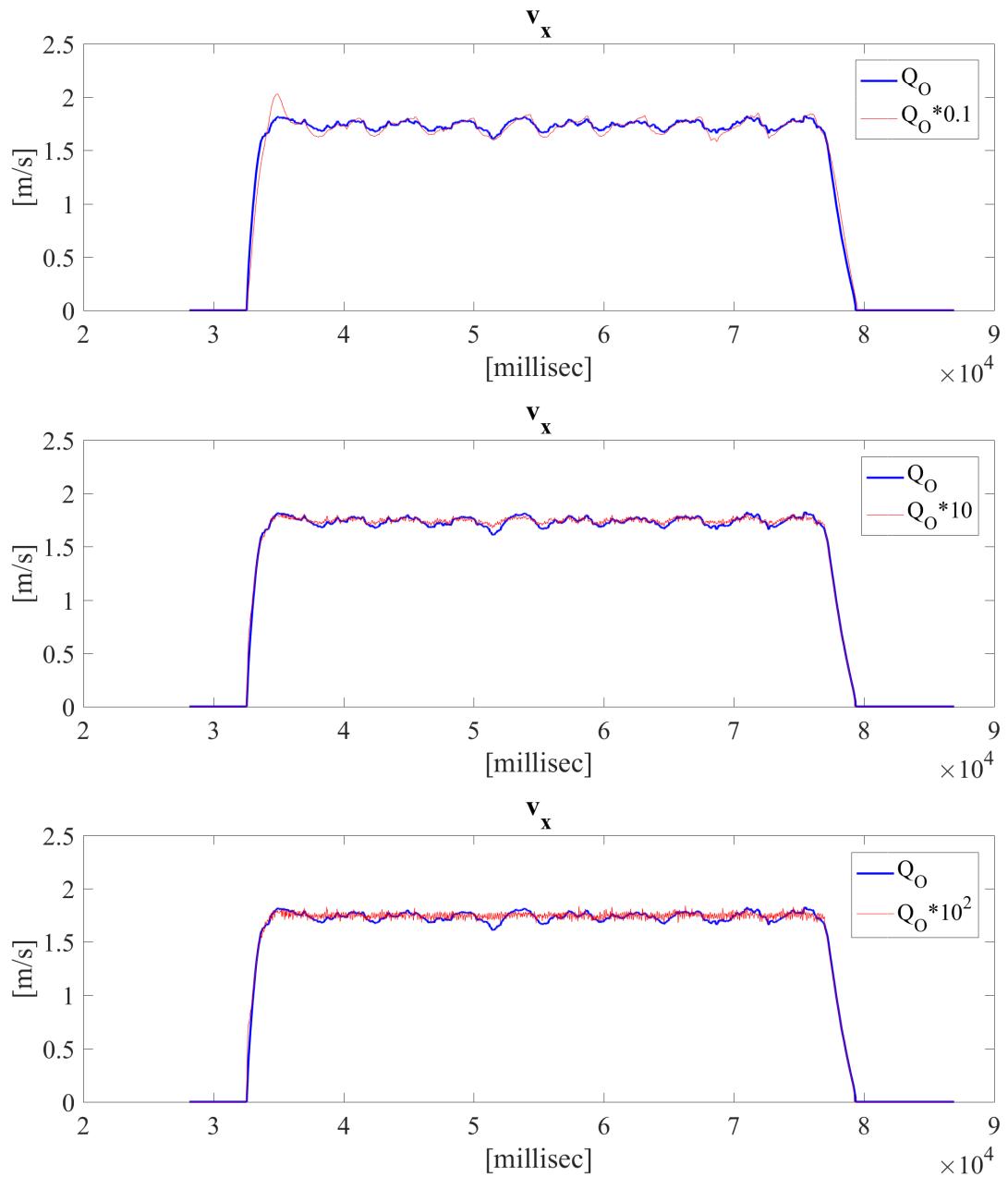


Figura 4.34: Valori di  $v_x$  ottenuti al variare di  $Q$ .

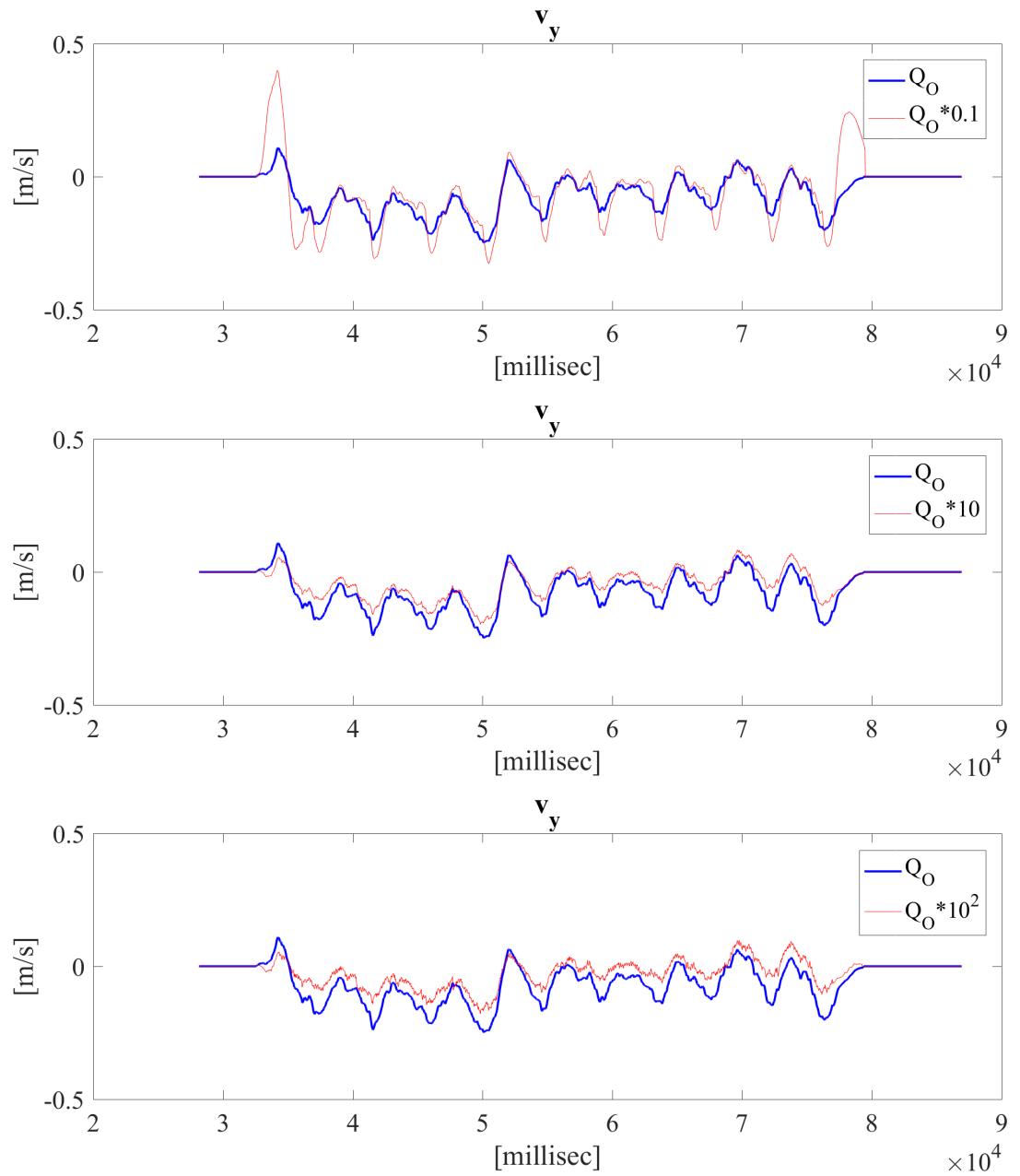


Figura 4.35: Valori di  $v_y$  ottenuti al variare di  $Q$ .

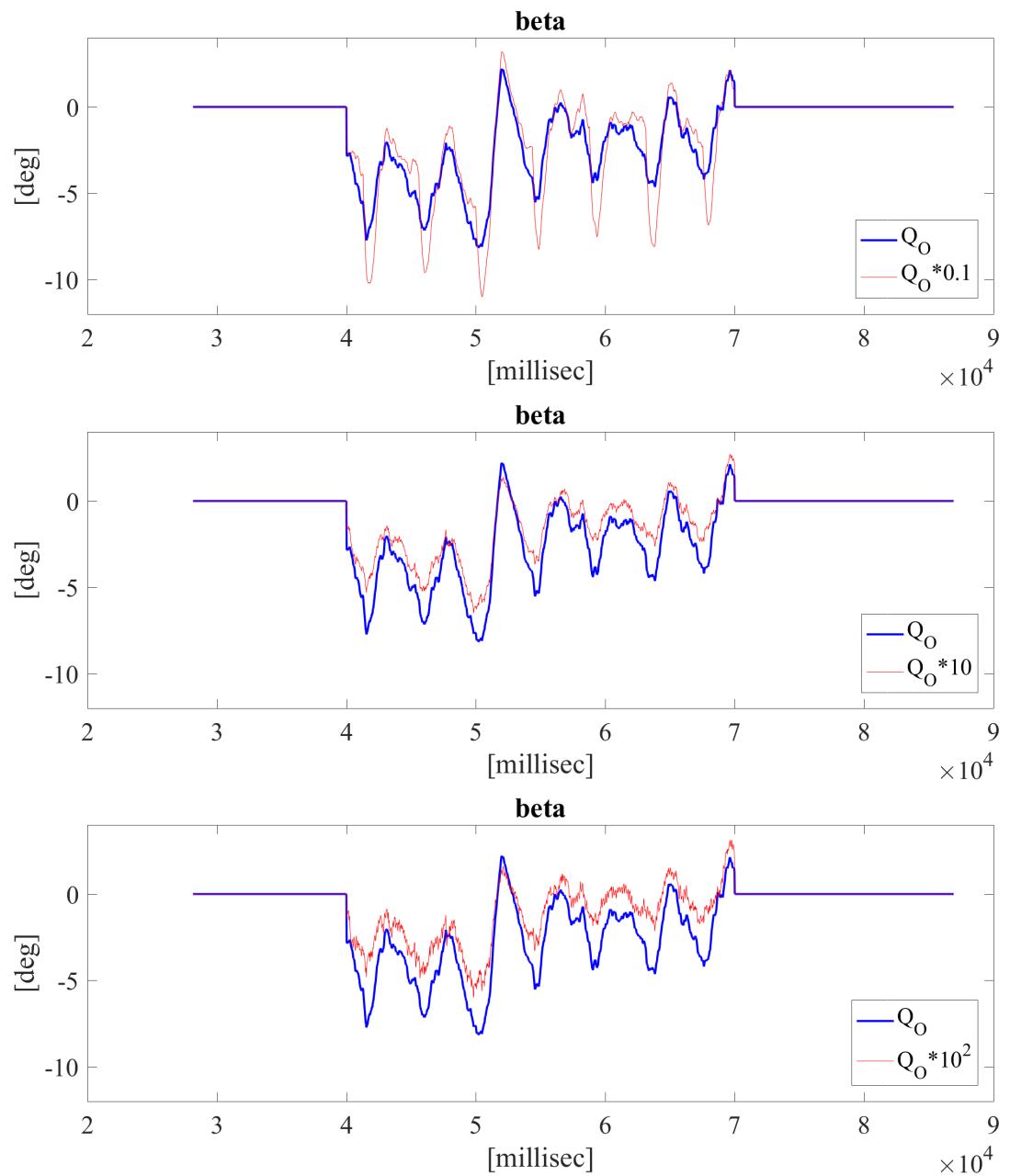


Figura 4.36: Valori di  $\beta$  ottenuti al variare di  $Q$ .

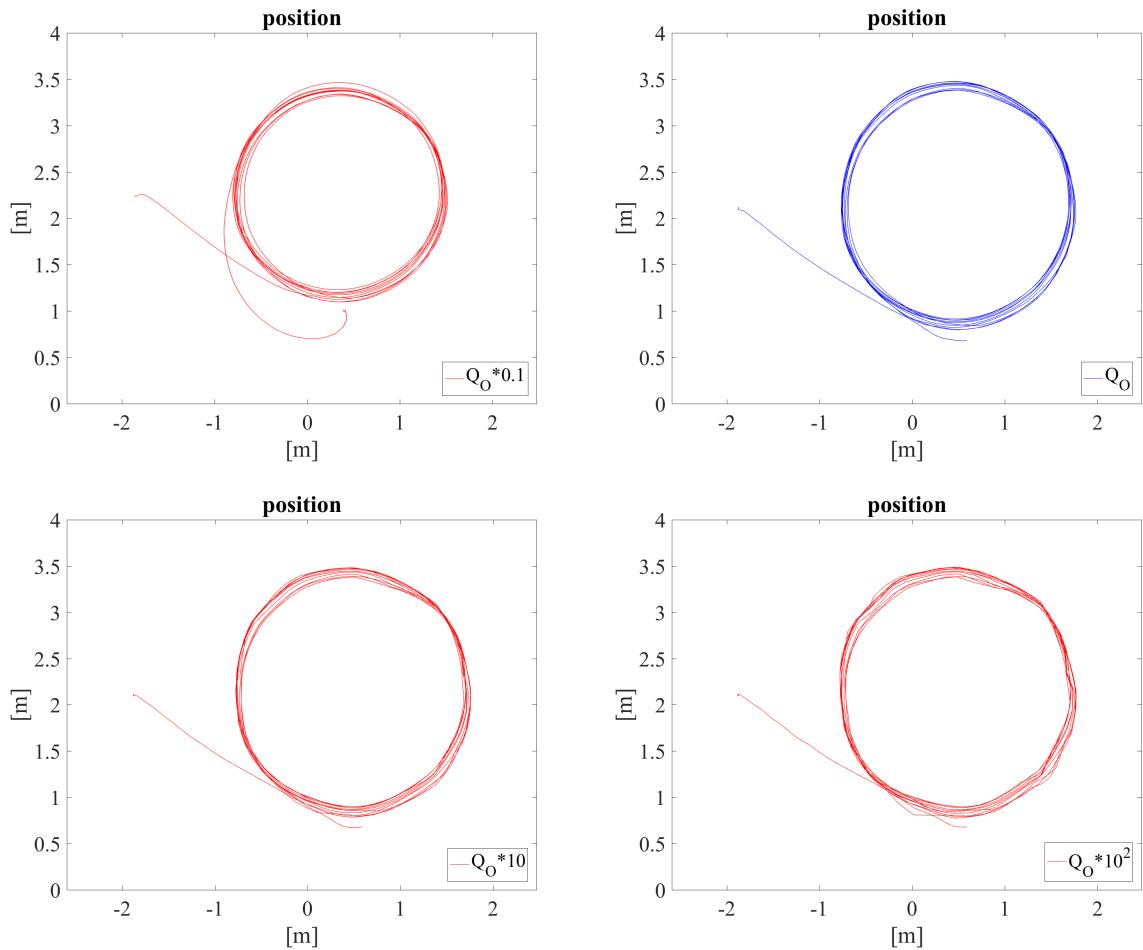


Figura 4.37: Traiettorie ottenute al variare di  $Q$ .



# Conclusioni e sviluppi futuri

Questo lavoro di tesi si è focalizzato principalmente su due aspetti:

- la progettazione della architettura hardware e software del veicolo eRumby;
- lo sviluppo di un algoritmo di *sensor fusion* per la rielaborazione dei dati raccolti dai sensori montati sul veicolo.

Per quanto riguarda il primo punto, il veicolo è ora in grado di ricevere informazioni da un GPS indoor, da una IMU e da quattro Encoder, salvarle e inviarle al PC utilizzando una scheda Arduino Mega e una BeagleBone. Inizialmente era stata utilizzata una IMU “9 Degrees of Freedom - Sensor Stick”, prodotta dalla azienda Sparkfun, ma si è rivelata poco utile a causa dell’eccessivo rumore nei dati in uscita e per questa è stata sostituita con la IMU “Adafruit BNO055”. Il GPS Indoor si è dimostrato uno strumento molto utile, con una ottima precisione e una frequenza di aggiornamento più che accettabile. La possibilità di eseguire la raccolta dati su una scheda e la rielaborazione di questi su una seconda unità distinta ha permesso la realizzazione di un sistema stabile, robusto e sufficientemente performante. La Arduino Mega si occupa della lettura dei sensori e della gestione dei motori, mentre la BeagleBone si occupa della rielaborazione dei dati e della determinazione degli input necessari per il controllo del veicolo. Con il veicolo è stato possibile svolgere diverse tipologie di prove sperimentali e i dati raccolti dai vari sensori, nonostante la presenza di rumore, si sono dimostrati coerenti e accettabili e validato sperimentalmente.

Il secondo punto affrontato è stato lo sviluppo di un algoritmo per la rielaborazione dei dati raccolti. Per ridurre il rumore e per fondere assieme tra loro le informazioni provenienti dal GPS, dalla IMU e dagli Encoder è stato sviluppato un Filtro di Kalman Esteso, basato sul modello cinematico del veicolo. I parametri relativi agli errori di stima e di misura sono

stati ricavati mediante osservazioni sperimentalni. Per l'ottimizzazione del filtro è stato utilizzato un approccio simulativo, facendo uso di dati off-line e andando ad ottimizzare le matrici legate agli errori di stima e di misura. Una volta ottenute delle simulazioni soddisfacenti, l'algoritmo di *sensor fusion* è stato implementato sulla scheda BeagleBone presente sul veicolo.

Allo stato attuale il veicolo può essere utilizzato per provare e sperimentare nuovi algoritmi di *sensor fusion* e di controllo. La struttura modulare alla base del progetto consente un cambio semplice sia dei moduli software che di moduli hardware -i.e. nel caso si volesse sostituire un sensore o un algoritmo di lettura. Una modifica importante da valutare è l'aggiunta di un secondo controllore, che potrebbe essere una Arduino Mega, o di una scheda Microduino per eseguire la lettura e il *processing* degli encoder. Lo spostamento della lettura degli encoder dalla attuale Arduino Mega utilizzata porterebbe ad una aumento della frequenza di lettura dei dati sia della IMU che degli encoder stessi. Potrebbe essere interessante cambiare il protocollo di comunicazione tra la Arduino Mega e la BeagleBone, passando dalla attuale comunicazione seriale a una comunicazione tramite protocollo I2C, migliorando in questo modo la velocità e la sincronizzazione della trasmissione dei dati. Potrebbe essere inoltre utile sostituire la BeagleBone Black con il modello BeagleBone Green, comparabile alla prima, ma dotata di WiFi, al fine di sostituire il modulo XBee e di gestire il veicolo direttamente dal suo cervello centrale.

Un possibile *upgrade* del controllo del veicolo potrebbe essere l'introduzione di un *Traction Control System* (TCS), detto anche *Anti-Slip Regulation* (ASR), un sistema elettronico per impedire lo slip delle ruote motrici del veicolo in fase di accelerazione. Tramite i sensori posti sulle ruote è possibile individuare lo slittamento delle ruote e intervenire sulla alimentazione del motore riducendo la potenza erogata e sottrarre alle ruote la coppia che determinava il loro slittamento. L'uso di un sistema TCS permetterebbe di configurare una trazione integrale sul veicolo e permetterebbe una migliore lettura dai sensori a bordo del veicolo e un sistema più robusto. Per poter introdurre il TCS è però necessario passare ad un controllo in coppia del motore. Per facilitare questo passaggio, uno dei progetti già attivi sul progetto eRumby è volto alla sostituzione del motore *brushless* con un motore *brushed*. Questo permetterebbe di controllare in modo più facile il motore e gestire il veicolo anche a velocità molto basse. Al TCS potrebbe essere affiancato anche un *Vehicle Dynamic Control* (VDC), un sistema di controllo della stabilità che regola la potenza del

motore in fase di sbandata del veicolo. Arrivati a questo punto le implementazioni sul veicolo sono molteplici. Si potrebbe pensare allo sviluppo di algoritmi per evitare le collisioni o per evitare ostacoli in movimento oppure algoritmi per eseguire particolari manovre che possono andare dalle manovre di parcheggio alle manovre di *lane change*.



# Appendice

## Assemblaggio del veicolo

Si riporta lo schema delle connessioni elettriche tra schede e sensori impiegati.

ENCODER			
Indice	Funzione	Colore	Collegamento
1	GND	Nero	Polo negativo batteria
2	5V	Rosso	Polo positivo batteria
3	Canale A	Bianco	Pin 50/51/52/53 Arduino Mega
4	Canale B	Marrone	
5	Canale I	Verde	

Tabella 4.1: *Cavi in uscita dall'Encoder.*

GPS			
Indice	Funzione	Colore	Collegamento
1	GND	Viola	GND Arduino Mega
2	TX	Verde	Pin 19 Arduino Mega

Tabella 4.2: *Cavi in uscita dal GPS.*

IMU			
Indice	Funzione	Colore	Collegamento
1	Vin	Rosso	Pin 5V Arduino Mega
2	GND	Marrone	Pin GND Arduino Mega
3	SDA	Blu	Pin 20 Arduino Mega
4	SCL	Bianco	Pin 21 Arduino Mega

Tabella 4.3: *Cavi in uscita dalla IMU.*

## Appendice

---

XBEE			
Indice	Funzione	Colore	Collegamento
1	GND	Nero	Pin P9_2 BeagleBone
2	3V	Rosso	Pin P9_4 BeagleBone
3	RX	Blu	Pin P9_22 BeagleBone
4	TX	Giallo	Pin P9_21 BeagleBone

Tabella 4.4: *Cavi in uscita dalla XBee.*

RICEVENTE RADIO			
Indice	Funzione	Colore	Collegamento
1	input modalità	Bianco - Viola	Pin 2 Arduino Mega
2	GND	Nero - Grigio	GND Arduino Mega
3	GND	Nero - Grigio	GND Arduino Mega
4	input sterzo	Blu - Verde	Pin 8 Arduino Mega
5	output sterzo	Blu - Rosa	Pin A12 Arduino Mega
6	input trazione	Giallo - Arancione	Pin 9 Arduino Mega
7	output trazione	Giallo - Azzurro	Pin A11 Arduino Mega

Tabella 4.5: *Cablaggio della ricevente radio.*

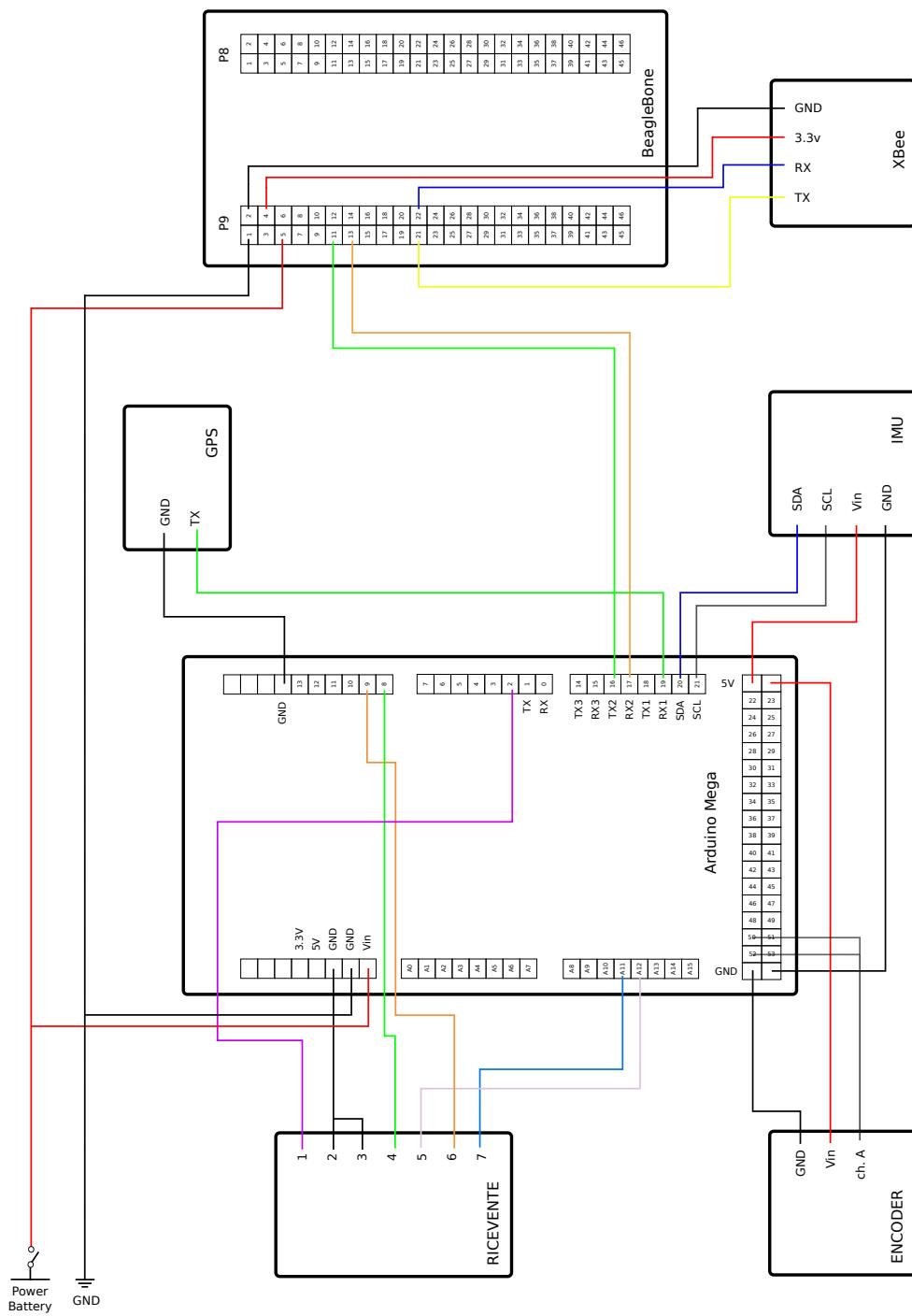
ARDUINO MEGA			
Indice	Funzione	Colore	Collegamento
	GND	Nero	Polo negativo batteria
	Vin	Rosso	Polo positivo batteria
	5V	Rosso	Pin 1 IMU
	GND	Marrone	Pin 2 IMU
	GND	Nero - Grigio	Pin 2&3 Ricevente Radio
	GND	Viola	Pin 1 GPS
2	input modalità	Bianco - Viola	Pin 1 Ricevente Radio
8	input sterzo	Blu - Verde	Pin 4 Ricevente Radio
9	input trazione	Giallo - Arancione	Pin 6 Ricevente Radio
A11	output trazione	Giallo - Azzurro	Pin 7 Ricevente Radio
A12	output sterzo	Blu - Rosa	Pin 5 Ricevente Radio
16	TX2	Verde	Pin P9_11 BeagleBone
17	RX2	Arancio	Pin P9_13 BeagleBone
19	RX1	Verde	Pin 2 GPS
20	SDA	Blu	Pin 3 IMU
21	SCL	Bianco	Pin 4 IMU
50	Encoder RR	Bianco	Pin 3 encoder rear right
51	Encoder FR	Bianco	Pin 3 encoder front right
52	Encoder RL	Bianco	Pin 3 encoder rear left
53	Encoder FL	Bianco	Pin 3 encoder front left

Tabella 4.6: *Cablaggio della scheda Arduino Mega.*

BEAGLEBONE			
Indice	Funzione	Colore	Collegamento
P9_1	GND	Nero	Polo negativo batteria
P9_2	GND	Nero	Pin 1 XBee
P9_4	3.3V	Rosso	Pin 2 XBee
P9_5	SYS_5V	Rosso	Polo positivo batteria
P9_11	UART4_RXD	Verde	Pin 16 Arduino Mega
P9_13	UART4_TXD	Arancio	Pin 17 Arduino Mega
P9_21	UART2_RXD	Giallo	Pin 4 XBee
P9_22	UART2_TXD	Blu	Pin 3 XBee

Tabella 4.7: *Cablaggio della scheda BeagleBone*

## Appendice





## Appendice

---

# Elenco delle figure

1	<i>Carrello di Leonardo da Vinci.</i>	1
2	<i>Infografica dell'evoluzione dei veicoli a guida remota e autonoma.</i>	4
3	<i>Prototipo RUMBy.</i>	8
4	<i>Prototipo eRumby.</i>	9
1.1	<i>Kyosho Inferno VE Race.</i>	12
1.2	<i>Componenti del veicolo eRumby e relative connessioni.</i>	13
1.3	<i>Motore brushless.</i>	15
1.4	<i>Motore Vortex 8 Evo e ESC Vortex R8 di Team Orion.</i>	16
1.5	<i>Radiocomando e ricevente Syncro KT.</i>	16
1.6	<i>Modalità di funzionamento del veicolo.</i>	17
1.7	<i>Encoder E5 US-Digital.</i>	18
1.8	<i>Schema di lettura dei canali A e B degli encoder.</i>	19
1.9	<i>Adafruit BNO055 Absolute Orientation Sensor.</i>	21
1.10	<i>GPS indoor Marvelmind Robotics.</i>	23
1.11	<i>Schema del sistema IPS.</i>	24
1.12	<i>Arduino Mega.</i>	26
1.13	<i>Digi XBee.</i>	27
1.14	<i>BeagleBone Black.</i>	28

## ELENCO DELLE FIGURE

---

2.1	<i>Architettura software del veicolo</i> . . . . .	32
2.2	<i>Sistema di riferimento della IMU solidale al veicolo</i> . . . . .	36
2.3	<i>Problema del Preview Control</i> [21]. . . . .	44
2.4	<i>Geometria alla base del Pure Pursuit</i> [22]. . . . .	44
2.5	<i>Geometria alla base del Stanley Method</i> [12]. . . . .	45
3.1	<i>Struttura centralizzata</i> . . . . .	49
3.2	<i>Struttura distribuita</i> . . . . .	49
3.3	<i>Logica alla base delle reti neurali</i> . . . . .	50
3.4	<i>Logica Fuzzy</i> . . . . .	51
3.5	<i>Schema generale del filtro di Kalman</i> . . . . .	53
3.6	<i>Schema generale dell'algoritmo di EKF</i> . . . . .	56
3.7	<i>Modellazione del veicolo eRumby</i> . . . . .	59
3.8	<i>Sideslip angle</i> . . . . .	61
4.1	<i>Accelerazioni in condizioni statiche</i> . . . . .	64
4.2	<i>Velocità angolari in condizioni statiche</i> . . . . .	65
4.3	<i>Orientazione in condizioni statiche</i> . . . . .	66
4.4	<i>Valori in ingresso ai motori di sterzo e trazione durante la traiettoria circolare</i> . .	67
4.5	<i>Velocità di rotazione media delle ruote durante la traiettoria circolare</i> . . . . .	67
4.6	<i>Velocità di rotazione delle ruote durante la traiettoria circolare</i> . . . . .	68
4.7	<i>Traiettoria percorsa dal veicolo durante la traiettoria circolare</i> . . . . .	69
4.8	<i>Accelerazioni durante la traiettoria circolare</i> . . . . .	69
4.9	<i>Angoli in condizioni durante la traiettoria circolare</i> . . . . .	70
4.10	<i>Velocità angolari durante la traiettoria circolare</i> . . . . .	71
4.11	<i>Sideslip angle durante la traiettoria circolare</i> . . . . .	71

4.12 <i>Stato del veicolo durante la traiettoria circolare.</i> . . . . .	72
4.13 <i>Traiettoria ricavata con EKF confrontata con quella fornita dal GPS durante la prova su traiettoria circolare.</i> . . . . .	73
4.14 <i>Valori in ingresso ai motori di sterzo e trazione durante la traiettoria rettilinea.</i> .	74
4.15 <i>Accelerazioni durante la traiettoria rettilinea.</i> . . . . .	74
4.16 <i>Angolo di yaw e yaw rate durante la traiettoria rettilinea.</i> . . . . .	75
4.17 <i>Traiettoria ricavata con EKF confrontata con quella fornita dal GPS durante la prova su traiettoria circolare.</i> . . . . .	76
4.18 <i>Sideslip angle durante la traiettoria rettilinea.</i> . . . . .	76
4.19 <i>Stato del veicolo durante la traiettoria rettilinea.</i> . . . . .	77
4.20 <i>Manovra di lane change.</i> . . . . .	78
4.21 <i>Valori in ingresso ai motori di sterzo e trazione durante il lane change.</i> . . . . .	79
4.22 <i>Accelerazioni durante il lane change.</i> . . . . .	79
4.23 <i>Valori in ingresso ai motori di sterzo e trazione durante il lane change.</i> . . . . .	80
4.24 <i>Traiettoria ricostruita con EKF confrontata con quella fornita dal GPS durante il lane change.</i> . . . . .	81
4.25 <i>Sideslip angle durante il lane change.</i> . . . . .	81
4.26 <i>Stato del veicolo durante il lane change.</i> . . . . .	82
4.27 <i>Circuito nel quale è stata eseguita la prova.</i> . . . . .	83
4.28 <i>Traiettoria ricostruita con EKF confrontata con quella fornita dal GPS durante la prova nel circuito.</i> . . . . .	84
4.29 <i>Traiettoria percorsa dal veicolo durante la prova nel circuito.</i> . . . . .	84
4.30 <i>Stato del veicolo durante la prova nel circuito.</i> . . . . .	85
4.31 <i>Sideslip angle durante la traiettoria rettilinea.</i> . . . . .	86
4.32 <i>Valori di <math>a_x</math> ottenuti al variare di <math>Q</math>.</i> . . . . .	88
4.33 <i>Valori di <math>a_y</math> ottenuti al variare di <math>Q</math>.</i> . . . . .	89

## ELENCO DELLE FIGURE

---

4.34 <i>Valori di <math>v_x</math> ottenuti al variare di <math>Q</math>.</i> . . . . .	90
4.35 <i>Valori di <math>v_y</math> ottenuti al variare di <math>Q</math>.</i> . . . . .	91
4.36 <i>Valori di <math>\beta</math> ottenuti al variare di <math>Q</math>.</i> . . . . .	92
4.37 <i>Traiettorie ottenute al variare di <math>Q</math>.</i> . . . . .	93
4.38 <i>Cablaggio del veicolo.</i> . . . . .	102

# Elenco delle tabelle

1.1	<i>Caratteristiche del veicolo eRumby.</i>	14
1.2	<i>Caratteristiche generali degli encoder “E5” della US-Digital.</i>	20
1.3	<i>Caratteristiche della IMU Adafruit BNO055.</i>	22
1.4	<i>Caratteristiche generali del sistema GPS Indor Marvelmind Robotics.</i>	25
1.5	<i>Caratteristiche della scheda Arduino Mega 2560.</i>	26
1.6	<i>Caratteristiche della scheda BeagleBone Black.</i>	29
2.1	<i>Indirizzi I2C IMU.</i>	35
2.2	<i>Convenzioni sulla rotazione degli angoli.</i>	36
2.3	<i>Valori per la ricostruzione dei segnali</i>	38
2.4	<i>Struttura del file di log</i>	40
2.5	<i>Struttura del file di log</i>	41
3.1	<i>Comparazione tra le tecniche di sensor fusion prese in esame.</i>	52
4.1	<i>Cavi in uscita dall’Encoder.</i>	99
4.2	<i>Cavi in uscita dal GPS.</i>	99
4.3	<i>Cavi in uscita dalla IMU.</i>	99
4.4	<i>Cavi in uscita dalla XBee.</i>	100
4.5	<i>Cablaggio della ricevente radio.</i>	100

## ELENCO DELLE TABELLE

---

4.6	<i>Cablaggio della scheda Arduino Mega</i>	101
4.7	<i>Cablaggio della scheda BeagleBone</i>	101

# Bibliografia

- [1] Anderson J., Kalra N., *Autonomous Vehicle Technology. A guide for policymakers*, RAND Corporation, 2016.
- [2] Åström K. J., Murray R. M., *Feedback systems. An introduction for scientists and engineers*, Princeton University Press, 2008.
- [3] Bernardo G., *Easy Bee. Guida alla scelta e alla comprenione dei moduli XBee*, RobotItaly, 2011.
- [4] Bertozzi M., Broggi A., Conte G., Fascioli A., *Automatic vehicle guidance. The experience of the ARGO autonomous vehicle*, World Scientific Publishing Co. Pte. Ltd., 1999.
- [5] Canestrini G., *Leonardo costruttore di macchine e di veicoli*, Tumminelli & C., 1939.
- [6] Cox I. J., Wilfong G. T., *Autonomous robot vehicles*, Springer, 1990.
- [7] Dickmanns E. D., *Dynamic vision for perception and control of motion*, Springer, 2007.
- [8] Faragher R., *Understanding the basis of the Kalman filter via a simple and intuitive derivation*, IEEE Signal Processing Magazine, Settembre 2012.
- [9] Farrelly J., Wellstead P., *Estimation of vehicle lateral velocity*, in Proc. IEEE Int. CCA, Dearborn, MI, 1996.
- [10] Fukada Y., *Slip-angle estimation for vehicle stability control*, Vehicle System Dynamics, 32, 1999.

## BIBLIOGRAFIA

---

- [11] Guiggiani M., *Dinamica del veicolo*, CittàStudi edizioni, Nuova edizione 2007.
- [12] Hoffman G. M., Tomlin C. J., Montemerlo M., Thrun S., *Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing*, Stanford University, 2005.
- [13] Horvath D. E., Horvath T., *Indiana Cars. A history of the automobile in Indiana*, CreateSpace Independent Publishing Platform, 2013.
- [14] Korosec K., *Elon Musk Says Tesla Vehicles Will Drive Themselves in Two Years*, Fortune, 21 Dicembre 2015.
- [15] Lipson H., Kurman M., *Driverless. Intelligent cars and the road ahead*, The MIT Press, 2016.
- [16] Malan A., *Google lancia Waymo per la guida autonoma*, Il sole 24 ore, 13 Dicembre 2016.
- [17] Meyer M. T., *One Man's Vision. The life of automotive pioneer Ralph R. Teetor*, CreateSpace Independent Publishing Platform, 2011.
- [18] Panzani G., Corno M., Tanelli M., Zappavigna A., Savaresi S. M., Fortina A., Campo S., *Designing On-Demand Four-Wheel-Drive Vehicles via Active Control of the Central Transfer Case*, IEEE transactions on intelligent transportation systems, vol. 11, no. 4, Dicembre 2010.
- [19] Shoup D., *Cruising for parking*, Transport Policy 13, 2006.
- [20] Stein S. K., *From torpedoes to aviation. Washington Irving Chambers and technological innovation in the New Navy 1876-1913*, The University of Alabama Press, 2007.
- [21] Tomizuka M., *Optimal continuous finite preview problem*, IEEE Trans. Automat. Contr., vol. AC-20, no. 3, 1975.
- [22] Wallace R., Stentz A., Thorpe C. E., Maravec H., Whittaker W., Kanade T., *First Results in Robot Road-Following*, in IJCAI, 1985.

- [23] Zimmermann H.J., *Fuzzy set theory and its applications*, Kluwer Academic Publishers, IV edizione.