RdS
RICERCA DI SISTEMA

ENEA
AGENZIA NAZIONALE PER LE NUOVE TECNOLOGIE, L'ENERGIA E LO SVILUPPO ECONOMICO SOSTENIBILE

MINISTERO DELL'AMBIENTE E DELLA SICUREZZA ENERGETICA

Ricerca di Sistema elettrico

LA2.21 Affinamento del software di pianificazione e navigazione del veicolo

Simone Felicioni, Mirko Leomanni, Gabriele Costante,
Mario Luca Fravolini, Paolo Valigi

unipg
DIPARTIMENTO DI INGEGNERIA

RdS_PTR 22-24_PR 1.7_LA2.21_012

LA2.21 Affinamento del software di pianificazione e navigazione del veicolo

Autori:
Simone Felicioni, Mirko Leomanni, Gabriele Costante, Mario Luca Fravolini, Paolo Valigi

Mese Anno: giugno 2023

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero dell'Ambiente e della Sicurezza Energetica - ENEA
Piano Triennale di Realizzazione 2022-2024
Obiettivo: *Decarbonizzazione/Digitalizzazione ed evoluzione delle reti*
Progetto: Tecnologie per la penetrazione efficiente del vettore elettrico negli usi finali
Linea di attività: 2.21

Responsabile del Progetto: Claudia Meloni, ENEA
Responsabile Linea di Attività: Paolo Valigi, UniPG
Mese inizio previsto: 13
Mese inizio effettivo: 13
Mese fine previsto: 18
Mese fine effettivo: 18

Il presente documento descrive le attività di ricerca svolte all'interno dell'Accordo di collaborazione *"Affinamento del software di pianificazione e navigazione del veicolo, riconoscimento della sede stradale e interfacciamento con un navigator" (primo milestone)*
Responsabile scientifico ENEA: Dr. Sergio Taraglio
Responsabile scientifico Co-beneficiario: Paolo Valigi
Responsabili tecnico-scientifici delle attività: Mario Luca Fravolini e Gabriele Costante

Indice

1 RISULTATI ATTESI 4
2 RISULTATI OTTENUTI 5
3 PRODOTTI ATTESI 6
4 PRODOTTI SVILUPPATI 6
5 ANALISI DEGLI SCOSTAMENTI SU ATTIVITÀ E RISULTATI 6
6 SINTESI DELLE ATTIVITÀ SVOLTE 7
7 DETTAGLIO DELLE ATTIVITÀ SVOLTE 8
8 CONTRIBUTO DELLE EVENTUALI CONSULENZE ALLE ATTIVITÀ SOPRA DESCRITTE 15
9 PUBBLICAZIONI SCIENTIFICHE 16
10 EVENTI DI DISSEMINAZIONE 17

1 Risultati attesi

Risultati/Deliverable della LA2.21: “Affinamento software di pianificazione e navigazione del veicolo”

Per quanto riguarda i risultati attesi per la LA2.21 il primo è rappresentato da uno studio del modello cinematico del veicolo, necessario ad una migliore pianificazione della traiettoria. In particolare, verranno elaborate metodologie per calcolare ed inseguire traiettorie compatibili con le caratteristiche meccaniche e hardware (compresi i sensori disponibili) del veicolo. Gli scenari presi in considerazione saranno di carattere urbano, con navigazione su strada asfaltata.
Il secondo è rappresentato dall’individuazione di un framework o tool open source già disponibile che, a partire da un input fornito dall’utente relativo al punto di arrivo desiderato, elabori dei percorsi codificati attraverso waypoints georeferenziati. Il tool dovrà essere interfacciato con il veicolo per fornire i waypoints calcolati e consentire la successiva elaborazione della traiettoria e navigazione (prevista nella successiva Linea di Attività). Il tool (o navigatore) individuato si baserà su mappe stradali esistenti del C.R: Casaccia per l’elaborazione dei percorsi (es. open street map).
I risultati attesi possono essere quindi riassunti come segue.
Studio ed analisi di un modello cinematico del veicolo ed elaborazione di strategie per il calcolo e l’inseguimento di un percorso. Le prestazioni del sistema di pianificazione e inseguimento saranno valutate in relazione alle caratteristiche hardware del veicolo, ai segnali di controllo e ai sensori disponibili. Risultati preliminari del sistema di interfacciamento tra navigatore open source e pipeline di navigazione del veicolo. L’output è rappresentato da un Rapporto tecnico relativo alle attività svolte nell’ambito della ricerca su “Affinamento del software di pianificazione e navigazione del veicolo”.

2 Risultati ottenuti

2.1 Studio del modello cinematico

Considerati gli elementi meccanici (auto con coppia di ruote anteriori sterzanti e coppia di ruote posteriori motrici) ed hardware (compresi i sensori disponibili), la dinamica del veicolo è stata descritta tramite un modello senza retromarcia. In particolare, è stato adattato un modello di Ackermann che tiene conto dell'angolo di sterzo delle ruote anteriori, la lunghezza del passo del veicolo, e l'assenza della retromarcia.

2.2 Pianificazione del percorso stradale

È stato effettuato uno studio per la scelta del framework software open source che, a partire da un punto di partenza noto, in seguito da un input fornito dall'utente relativo al punto di arrivo desiderato, elabora dei percorsi codificati da waypoints georeferenziati.
È stato quindi sviluppato un software che permette l'immissione da parte di un utente del punto di arrivo tramite un'interfaccia grafica, calcola la sequenza di waypoints intermedi per raggiungerlo e invia quest'ultimi al sistema di navigazione. In particolare, l'interfaccia grafica mostra l'immagine satellitare della porzione di mappa in cui il veicolo opera e la sua posizione corrente, e permette di selezionare graficamente la posizione desiderata da raggiungere. Il sistema di pianificazione scelto è così in grado di calcolare un percorso stradale tra il punto di partenza e il punto di arrivo selezionato.

2.3 Pianificazione locale e inseguimento del percorso

È stata effettuata un'analisi preliminare sulle diverse componenti necessarie per lo sviluppo e l'implementazione di uno stack di navigazione autonoma a partire da waypoints georeferenziati. In particolare, il sistema di navigazione individuato è suddiviso in due moduli fondamentali: 1) modulo di pianificazione locale reattivo per evitare collisioni con oggetti rilevati durante la navigazione, e 2) modulo per l'inseguimento del percorso calcolato. Coerentemente con gli obiettivi di questa linea di azione, è stato effettuato il design architetturale del sistema di navigazione ed è stato preliminarmente implementato il modulo di inseguimento di un percorso definito da un insieme di set-points di posa. L'implementazione e la validazione dell'intero sistema di navigazione saranno oggetto della successiva linea di azione.

3 Prodotti attesi

Per la LA 2.21 era prevista la realizzazione di un report delle attività svolte, delle metodologie sviluppate e degli esperimenti effettuati.

4 Prodotti sviluppati

È stato realizzato il presente report delle attività in cui sono dettagliate le metodologie sviluppate e gli esperimenti svolti per la validazione.

5 Analisi degli scostamenti su attività e risultati

Non si riscontrano scostamenti rilevanti tra le attività pianificate e quelle effettivamente svolte.

6 Sintesi delle attività svolte

Le attività svolte in questa LA hanno riguardato lo sviluppo di un sistema di navigazione autonoma che consenta al veicolo di pianificare e seguire un percorso stradale evitando gli ostacoli lungo il cammino.
In questa fase è stato effettuato uno studio preliminare sul modello cinematico che descrive il veicolo utilizzato e sulla scelta del framework per la generazione di percorsi stradali a partire da coordinate georeferenziate. È stata inoltre svolta un'analisi sulle diverse componenti necessarie per implementare uno stack di navigazione autonoma.
Gli esperimenti preliminari sono stati condotti in ambiente simulato utilizzando un motore grafico che offre numerosi vantaggi, come la possibilità di generare ambienti fotorealistici e testare il sistema di navigazione autonoma in modo ripetibile su scenari controllati, senza il rischio di provocare danni al veicolo o alle persone.

7 Dettaglio delle attività svolte

Lo scopo del progetto è lo sviluppo di un sistema di navigazione autonoma che consenta al veicolo di pianificare e poi di seguire un percorso stradale, evitando gli ostacoli lungo il cammino, a partire da coordinate georeferenziate. In particolare, è stato studiato un approccio modulare che garantisce un'elevata precisione nella stima della posizione, una pianificazione di percorsi efficiente e una rilevazione affidabile degli ostacoli.

A seguito dell'analisi condotta, il sistema di navigazione è stato suddiviso nei seguenti moduli funzionali:

- **Localizzazione**: la localizzazione del veicolo è una componente chiave per la navigazione autonoma. Questa può essere fornita da un sistema GPS (Global Positioning System), oppure stimata utilizzando diverse fonti di informazioni, quali ad esempio misure GPS e dati odometrici. La stima odometrica, in particolare, fornisce la posizione del robot basandosi su algoritmi specifici che sfruttano misure provenienti da strumenti come telecamere o sensori laser.

- **Pianificazione globale**: il modulo di pianificazione globale si occupa di definire un percorso tra il punto di partenza e il punto di arrivo desiderato. Il pianificatore globale impiegato, denominato GraphHopper, permette di calcolare un percorso descritto da una sequenza di waypoints georeferenziati, a partire dalla posizione iniziale del veicolo e da punto di arrivo desiderato. A tal riguardo, è stata sviluppata un'interfaccia grafica che permette all'utente di selezionare il punto di arrivo.

- **Pianificazione locale**: il pianificatore locale utilizza la sequenza di waypoints forniti dal pianificatore globale insieme ad informazioni sulla presenza di eventuali ostacoli lungo il percorso, al fine di calcolare un percorso che permetta al veicolo di raggiungere la destinazione, evitando al contempo le collisioni. Il percorso restituito, definito da un insieme di set-points di posa, deve risultare il più vicino possibile al percorso indicato dal pianificatore globale, tenendo conto della dinamica del veicolo e della presenza di ostacoli.

- **Generazione della mappa**: per l'individuazione di percorsi privi di ostacoli è necessario generare una rappresentazione dettagliata dello spazio di manovra, che indichi chiaramente le regioni dello spazio libere e quelle occupate da ostacoli. Questa rappresentazione prende il nome di *occupancy grid*, ed in questo studio è generata a partire dalle misure di un sensore LiDAR (Light Detection and Ranging). Il LiDAR è un dispositivo di rilevamento remoto che emette impulsi laser e misura il tempo impiegato per il ritorno del segnale, consentendo di ottenere informazioni dettagliate sulla distanza e sulla forma degli oggetti circostanti.

- **Controllore**: per l'inseguimento del percorso è necessario implementare un sistema di controllo in retroazione che valuti in tempo reale la differenza tra la posa attuale del veicolo e la posa desiderata lungo il percorso pianificato. Questa deviazione, nota come errore di inseguimento, deve essere azzerata dal controllore in retroazione al fine di portare il veicolo sul percorso desiderato e di attenuare i disturbi che agiscono sulla dinamica dello stesso.

Lo schema a blocchi del sistema di navigazione progettato è rappresentato in Figura 1.

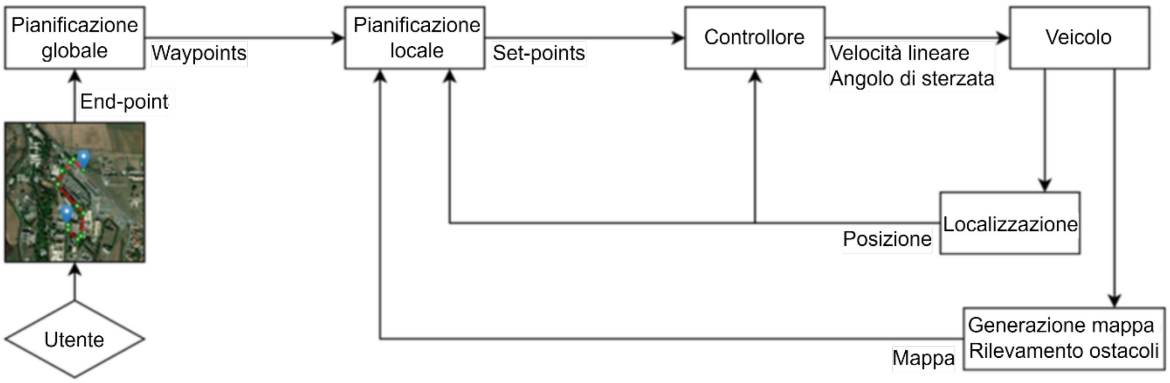


Figura 1: sistema di navigazione: il punto di arrivo viene selezionato tramite interfaccia grafica e, successivamente, il pianificatore globale GraphHopper genera una sequenza di waypoints fra la posizione attuale del veicolo ed il punto di arrivo. Il modulo di pianificazione locale calcola un percorso che congiunge gli waypoints restituiti dal pianificatore globale ed evita al contempo gli ostacoli. Il controllore in retroazione garantisce l'inseguimento del percorso fornito dal pianificatore locale.

7.1 Gestione della comunicazione

La gestione della comunicazione fra i diversi moduli di navigazione è affidata a sistema operativo ROS (Robot Operating System). ROS è un software open-source flessibile che mette a disposizione un insieme di strumenti, librerie e convenzioni atte a semplificare lo sviluppo di applicazioni robotiche. In particolare, è disponibile una vasta gamma di funzionalità, tra cui la gestione delle comunicazioni tra i componenti di un sistema robotico distribuito, la gestione dei sensori e degli attuatori, l'elaborazione e la visualizzazione dei dati. Utilizzando questo software, è stato realizzato un sistema di messaggistica distribuita che consente lo scambio efficiente di dati tra i diversi moduli del sistema.

Una delle caratteristiche distintive di ROS è la sua architettura basata su nodi che interfacciano le diverse componenti di un sistema robotico (sensori, attuatori, algoritmi di controllo, etc.). I nodi possono comunicare tra loro scambiando messaggi su dei cosiddetti *topic*. Questo approccio modulare favorisce la scalabilità e la flessibilità del sistema.

7.2 Pianificazione di percorsi su strada a partire da coordinate georeferenziate

7.2.1 Generazione del percorso stradale

La pianificazione dei percorsi su strada si basa su GraphHopper, un sistema open-source per il routing e la navigazione basato su grafi, il cui scopo principale è quello di fornire algoritmi e strumenti finalizzati a calcolare il percorso ottimale tra due o più punti all'interno di una rete stradale. In particolare, questo sistema modella la mappa in nodi, connessioni, e informazioni semantiche (*meta-tag*) come segnali stradali.

GraphHopper fornisce delle API HTTP, come ad esempio Route e Maps. La prima viene utilizzata per il calcolo del percorso, mentre la seconda per la visualizzazione della mappa. Il sistema può essere configurato per usare diversi algoritmi di pianificazione (come Dijkstra[1] o A*[2]), e per specificare il tipo di veicolo.

Ai fini della pianificazione, è stata sviluppata un'interfaccia software basata su un servizio web in Python e su una pagina Javascript CSS HTML che permette di inviare richieste http per l'invio delle coordinate dei nodi, ovvero degli waypoints. La parte grafica dell'interfaccia mostra l'immagine satellitare della porzione di mappa in cui il veicolo opera, la posizione corrente del robot e la posizione desiderata da raggiungere. La posizione corrente viene indicata come *startpoint*, mentre la posizione di arrivo come *endpoint*. Queste informazioni di posizione sono scambiate tra il servizio web e l'interfaccia grafica: la pagina HTML fa richiesta al server per ottenere le coordinate GPS del robot, mentre la posizione di arrivo viene selezionata dall'utente direttamente sull'interfaccia grafica, e le coordinate GPS corrispondenti sono pubblicate dal servizio web. Lo *startpoint* e l'*endpoint* vengono inviati all'API Route di GraphHopper per la generazione del percorso stradale. Il punto di partenza corrisponde alle coordinate GPS della posizione corrente del veicolo. In particolare, la posizione del veicolo viene aggiornata sulla mappa ogni tre secondi utilizzando le coordinate GPS ricevute dal veicolo stesso. Il punto di arrivo invece viene selezionato dall'utente direttamente sulla mappa mostrata all'interno dell'interfaccia grafica.

L'output di GraphHopper è una sequenza di waypoints verso la destinazione (generati in modo ottimale, in base alla presenza di svincoli o incroci stradali). Le coordinate GPS degli waypoints sono convertite in coordinate cartesiane utilizzando il modello matematico della Terra WGS84 (World Geodetic System 1984), che è basato su un ellissoide di riferimento. Le seguenti formule permettono di convertire latitudine (lat) e longitudine (lon) fornite dal GPS nel sistema cartesiano ($x,\ y,\ z$):

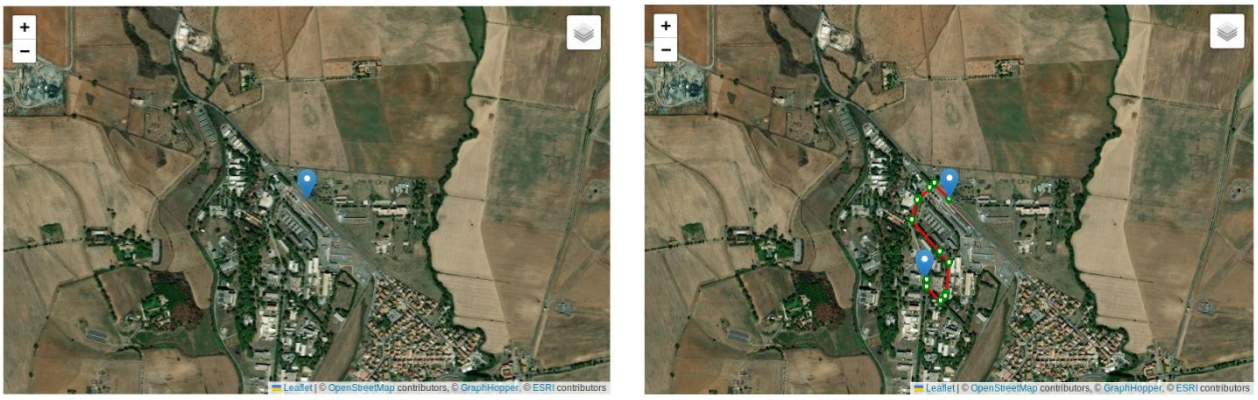
$$x = R * \cos(lat) * \cos(lon)$$
$$y = R * \cos(lat) * \sin(lon)$$
$$z = R * \sin(lat)$$

laddove R indica il raggio approssimato della Terra (6371 km). La trasformazione inversa risulta essere definita come segue:

$$lat = a\sin(z/R)$$
$$lon = a\tan 2\,(y, x)$$

Le coordinate cartesiane degli waypoints sono pubblicate su uno specifico topic ROS per renderle accessibili agli altri moduli del sistema di navigazione.

[1] Dijkstra, E. W. (1959). A note on two problems in connection with graphs. Numerische Mathematik, 1(1), 269–271.
[2] Hart, P., et al. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. IEEE Transactions on Systems Science and Cybernetics, 4(2), 100–107.



(a) (b)

Figura 2: Interfaccia utente del servizio web: (a) coordinate GPS della posizione di partenza del robot; (b) posizione di arrivo scelta dall'utente (in *blu*) insieme ai punti intermedi calcolati da GraphHopper (in *verde*), e il relativo percorso stradale (in *rosso*).

7.2.2 Modello cinematico del veicolo

A seguito dell'interazione con ENEA, sulla base dei dati sperimentali disponibile (utilizzati per il calcolo della relazione tra angolo di sterzo al volante e raggio di curvatura) e delle caratteristiche meccaniche (auto con coppia di ruote anteriori sterzanti e coppia di ruote posteriori motrici) ed hardware (sensori ed attuatori disponibili), il modello di Ackermann è risultato essere sufficientemente rappresentativo per descrivere la cinematica del veicolo.

Il modello di Ackermann è un'estensione del modello cinematico per robot denominato *car-like*, che può essere descritto, in prima approssimazione, dalla velocità traslazionale e dall'angolo di sterzata, nonché dalla velocità di sterzata del veicolo. Il modello impiegato assume che le ruote anteriori possano sterzare, al contrario di quelle posteriori.

Il modello di Ackermann descrive le relazioni tra le velocità delle ruote ed i movimenti di traslazione e rotazione del veicolo. In particolare, il modello tiene conto dell'angolo di sterzata delle ruote anteriori (δ) e della distanza tra gli assi delle ruote anteriori e posteriori (passo o wheelbase, w) e della velocità lineare (v) e angolare (ω) del veicolo.

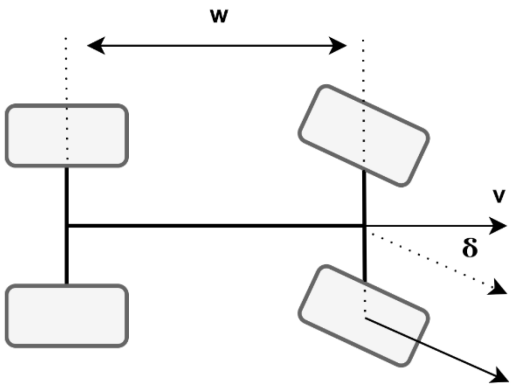


Figura 3: rappresentazione del modello cinematico di Ackermann.

Il modello appena descritto è discretizzato utilizzando il metodo di Eulero come in Paden et al. (2016)[3]. Definendo la posizione dell'asse anteriore del veicolo come $\left(x_f(t),\ y_f(t)\right)$ e il suo orientamento come $\theta(t)$, ed indicando lo stato del veicolo all'istante di campionamento attuale come $\left(x_f(t),\ y_f(t),\ \theta(t)\right)$, l'angolo di sterzata come $\delta(t)$, e lo stato del veicolo all'istante di campionamento successivo a quello attuale come $\left(x_f(t+1),\ y_f(t+1),\ \theta(t+1)\right)$, si ottengono le seguenti equazioni alle differenze:

$$x_f(t+1) = x_f(t) + v_f \cos(\theta(t) + \delta(t))\,\Delta T$$
$$y_f(t+1) = y_f(t) + v_f \sin(\theta(t) + \delta(t))\,\Delta T$$
$$\theta(t+1) = \theta(t) + \frac{v_f}{w}\sin(\delta)\,\Delta T$$

dove ΔT denota il passo di campionamento.

7.2.3 Inseguimento del percorso

Il sistema di controllo impiegato per l'inseguimento di un percorso, definito da una sequenza di set-points di posa, si basa sul metodo di Stanley[4]. In questo algoritmo, si considerano sia l'errore di posizione che l'errore di orientamento del veicolo. L'errore di posizione (e) è pari alla differenza tra la posizione attuale del veicolo ed il set-point più vicino sul percorso desiderato, mentre l'errore di orientamento (ψ) è definito come l'angolo tra la direzione attuale del veicolo e quella desiderata. Per compensare entrambi gli errori, il controllore di Stanley agisce sull'angolo di sterzata della vettura $\delta(t)$ come segue:

$$\delta(t) = \psi(t) + \tan^{-1}\left(\frac{ke(t)}{k_s + v(t)}\right), \qquad \delta(t) \in [\delta_{min}, \delta_{max}]$$

laddove $[\delta_{min}, \delta_{max}]$ è l'intervallo che contiene i valori ammissibili dell'angolo di sterzata. In particolare, è necessario innanzitutto individuare il set-point di posa più vicino al centro geometrico dell'asse anteriore del veicolo. Il segnale di controllo è costituito due componenti che correggono gli errori di orientamento e di posizione sulla base delle seguenti considerazioni:

1. Una prima componente $\delta_1(t) = \psi(t)$ si occupa di eliminare l'errore di orientamento, imponendo che l'angolo di sterzata sia uguale all'angolo tra la retta definita dalla direzione attuale del veicolo e l'orientamento del set-point più vicino.

2. Per compensare l'errore di posizione, è invece necessario calcolare la distanza tra il veicolo e il set-point più vicino, definita come $e(t)$. Il contributo risultante in termini di angolo di sterzata può essere descritto come segue:

   $$\delta_2(t) = \tan^{-1}\left(\frac{ke(t)}{v(t)}\right)$$

   Questa componente fa sì che l'angolo di sterzata sia proporzionale alla distanza tra il veicolo e il percorso desiderato. Al fine di evitare eventuali singolarità, un termine di offset costante k_s viene infine sommato al denominatore del segnale di controllo appena definito, ottenendo:

[3] Paden, B., et al. (2016). "A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles". IEEE Transactions on Intelligent Vehicles, 1, 33-55.

[4] Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., et al. (2007). "Stanley: The Robot That Won the DARPA Grand Challenge". The 2005 DARPA Grand Challenge. Springer Tracts in Advanced Robotics, 36, 1.

$$\delta_2(t) = \tan^{-1}\left(\frac{ke(t)}{k_s + v(t)}\right)$$

Il modulo di inseguimento (controllo in retroazione) legge da un topic ROS un percorso definito come sequenza di set-points di posa (il calcolo del percorso locale a partire dagli waypoints restituito dal pianificatore globale sarà oggetto della successiva linea di azione). Dopo aver calcolato l'errore di posa tra la posa corrente del veicolo e il set-point di posa più vicino, il controllore restituisce un segnale di controllo dato da velocità lineare e angolo di sterzata.

La velocità lineare (v) è attualmente impostata ad un valore costante ($5\ m/s$), mentre assume valore nullo ($0\ m/s$) solo nel caso in cui ci sia un ostacolo non evitabile e quindi non esista un percorso percorribile in modo sicuro dal veicolo. L'angolo di sterzata (δ) è invece calcolato come dettagliato precedentemente per annullare l'errore di posizione e di orientamento del veicolo. Velocità lineare e angolo di sterzata sono infine pubblicati su uno specifico topic ROS per comunicare con il veicolo.

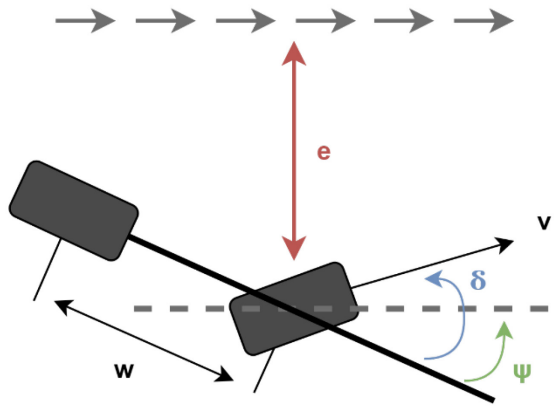


Figura 4: Grandezze impiegate nel controllore di Stanley, che considera congiuntamente l'errore di rotta (ψ) e l'errore di posizione (e) per l'inseguimento del percorso.

7.3 Esperimenti

Gli esperimenti sono stati svolti in ambiente fotorealistico simulato attraverso il motore grafico Unreal Engine 5 (UE5). È stato innanzitutto creato un ambiente urbano (Figura 5), per poi implementare la cinematica desiderata sul robot simulato, su cui sono presenti un sensore GPS e un sensore LiDAR, quest'ultimo con le stesse specifiche del Velodyne LiDAR Puck VLP-16, tra cui:

- Angolo di visione orizzontale: 360° (da -180° a 180°)
- Angolo di visione verticale: 30° (da -15° a 15°)
- Risoluzione angolare: 0.4°

(a) (b)

Figura 5: Simulazione di ambiente urbano in Unreal Engine 5 (UE5), visto dall'alto (a) e dalla camera frontale presente sul sistema robotico (b).

Una delle componenti principali per la navigazione autonoma in un ambiente è l'inseguimento di un percorso predefinito. In particolare, questo percorso è definito da una sequenza di set-points di posa e dovrà essere generato a partire dalla sequenza di waypoints georeferenziati che definiscono il percorso stradale restituito da GraphHopper. Come descritto nella Sezione 7.1.3, per l'inseguimento del percorso è stato implementato il controllore di Stanley, che, a partire da una sequenza di set-points di posa, calcola l'angolo di sterzata per la compensazione dell'errore di inseguimento. In Figura 6 vengono riportati i risultati qualitativi degli esperimenti preliminari effettuati.

Figura 6: considerato un percorso predefinito, costituito da una sequenza di set-points di posa, il modulo di inseguimento implementato calcola l'angolo di sterzata utile per compensare gli errori di orientamento e posizione.

8 Contributo delle eventuali consulenze alle attività sopra descritte

Nello sviluppo delle varie attività previste per questa LA non è stato fatto ricorso a consulenze esterne. Tutte le attività qui descritte sono state sviluppate dal Team di ricercatori UniPG elencati all'inizio di questo report.

9 Pubblicazioni scientifiche

Le attività di questa prima LA hanno quasi esclusivamente riguardato il test di tecnologie software allo stato dell'arte e la loro integrazione al fine di implementare un sistema di navigazione completo.
Poiché attualmente le attività del progetto sono a livello di "work in progress" si rimanda la possibile pubblicazione dei risultati ad una fase successiva del progetto quando saranno disponibili i risultati sperimentali dei test.

10 Eventi di disseminazione

Dalla presente Linea di Attività non sono scaturiti eventi di disseminazione.