Università degli Studi di Napoli Federico II

Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione

Scuola Politecnica e delle Scienze di Base

Corso di Laurea Triennale in Ingegneria Informatica

Implementazione e validazione di controllori supervisivi per veicoli autonomi

Relatore:
Prof.ssa Santini Stefania
Correlatore:
Dott.Ing. Giacomo Basile

Candidato:
Gentile Giovanni
Matr. N46004527

Anno Accademico
2021/2022

Indice

Elenco delle figure

Elenco delle tabelle

Introduzione

Le auto a guida autonoma possono essere definite come veicoli in grado di guidare in molti scenari di traffico sostituendo l'intervento umano. Se è evidente che strade, segnali, semafori e corsie seguano standard riconoscibili dall'uomo, tali scenari sono caratterizzati da una certa variabilità dovuta alle diverse condizioni meteorologiche, al traffico ed al comportamento umano.
Ad oggi, queste sono le principali sfide che i ricercatori sui veicoli autonomi devono affrontare.
Questa tesi raccoglie la sfida descritta e implementa algoritmi di computer vision per l'identificazione di ostacoli situati sulla carreggiata tramite dei sensori presenti sul veicolo.
Quando si parla di auto a guida autonoma, solitamente, si nutrono dubbi sulla possibilità che in un futuro prossimo questi sistemi diventino realtà. Questo accade, in particolare, quando si parla di veicoli completamente autonomi.
Sicuramente la tecnologia sta già trasformando il modo di guida e di viaggio delle persone, ma con i veicoli autonomi cambierà il modo in cui si guarderà ai sistemi di trasporto.
Inoltre, essa apporterà un beneficio sociale dovuto all'assenza di errori umani e all'accessibilità per le persone che, per qualche motivo, non possono guidare.
Ovviamente il passaggio alla guida autonoma comporta alcune problematiche. Queste sono principalmente legate al processo decisionale, cioè alla selezione delle manovre del veicolo in un ambiente dinamico, caratterizzato da un'elevata variabilità.
Per progettare e realizzare un sistema complesso come quello di un'autovettura, è necessario acquisire informazioni significative dall'ambiente con differenti tipologie di sensori, elaborarle e prendere decisioni sulla traiettoria del veicolo. Le telecamere non sono gli unici sensori disponibili. I dati provenienti da GPS, Lidar, Radar e altri vengono raccolti e sfruttati insieme.
Questa importante fase prende il nome di *Sensor Fusion*.

Capitolo 1

Advanced Driver Assistance System

ADAS[1] è la sigla usata per definire tutti quei dispositivi pensati per aumentare il comfort alla guida e i livelli di sicurezza, rendendo l'auto più sicura ed affidabile.
Si tratta, quindi, di dispositivi elettronici installati sulle vetture per ridurre al minimo i rischi di incidente ed agevolare la vita a bordo e nelle vicinanze dell'automobile [4].

1.1 Livello Automatizzazione

I livelli di prestazioni delle auto a guida autonoma sono classificati in sei livelli dall'ente SAE per quanto riguarda la tecnologia e la legislazione nello standard *J3016* [1].
Una panoramica dei livelli di autonomia è rappresentata in *Figura 1.1*, che rappresenta il grafico più aggiornato di tali livelli, come indicato nell'ultima versione di Giugno 2018 dello standard *J3016*. Le auto a guida autonoma sono classificate in sei Livelli in base al loro grado di automatizzazione: dal Livello Zero, ovvero nessuna automazione, fino al Livello SAE Cinque per la piena autonomia del veicolo.

- Il Livello Zero è il livello base, in cui le funzioni di supporto al conducente si limitano a fornire avvisi e assistenza momentanea, come l'avviso di superamento della corsia o l'avviso di angolo cieco.

[1] Advanced Driver Assistance System

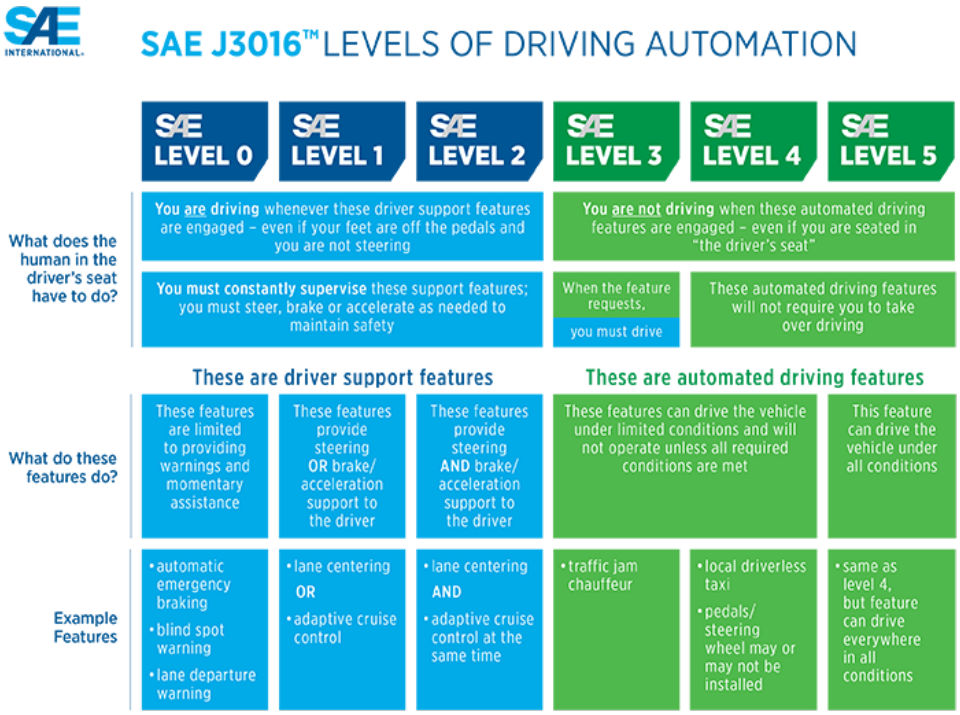


Figura 1.1: SAE Level [9]

In pratica al Livello Zero non c'è nessun tipo di intervento autonomo da parte dell'autovettura.

- Al Livello Uno appartengono tutte le auto sulle quali vengono implementate funzioni dedicate alla guida assistita di base, ovvero il *Cruise Control* che regola la velocità di crociera, l'*ABS*[2] e anche altre tecnologie che sono in grado di rilevare la corsia.

- Il Livello Due fornisce funzioni come il centraggio della corsia e l'*Adaptive Cruise Control*. In generale fanno parte di questo livello le auto in grado di intervenire in accelerazione e in frenata.
  Fino a questo livello il conducente è l'unica persona responsabile del comportamento dell'auto, in quanto le funzioni ADAS[3] sono considerate solo come un'assistente, ma non possono sostituirlo in alcun caso.

- Il Livello Tre è il primo vero livello di automazione dove la responsabilità è condivisa tra il conducente e l'automobile. Il conducente deve

[2]Sistema antibloccaggio
[3]Advanced Driver Assistance Systems

intervenire ogni volta che è presente una richiesta di sistema. I sistemi presenti in questo Livello prendono il controllo solo in determinate situazioni come nella ricerca di un parcheggio o in autostrada.

- Al Livello Quattro viene introdotto un alto livello di automazione. Qui, la guida autonoma è capace di monitorare l'ambiente esterno tramite sensori, radar e telecamere, e, quindi, ricavare le istruzioni di guida. L'autista può riprendere il controllo del veicolo in qualsiasi momento, e, nonostante ciò, il computer di bordo non si disattiverà in modo tale da poter intervenire nel caso in cui il guidatore non risponda in tempo ad una richiesta di intervento.

- Nel Livello Cinque il sistema esegue tutti i compiti menzionati in precedenza in tutte le situazioni incontrate durante l'intero viaggio, in modo tale da non richiedere l'intervento dell'uomo.

Questo standard è uno dei riferimenti più citati per le capacità delle auto autonome. Negli ultimi anni sono state introdotte numerose auto autonome sulle strade, le caratteristiche di ciascun livello sono spiegate più chiaramente nel recente aggiornamento della tabella sopra citata per mostrare come stanno aumentando la sicurezza e la convenienza dei consumatori (*Figura 1.1*).
Questa tesi prenderà in considerazione dal Livello Due al Livello Quattro di autonomia per i casi definiti. Al fine di valutare la sicurezza di un'auto autonoma in scenari di guida critici definiti, verrà implementato un banco di prova composto da alcune funzioni ADAS principali, quali l'*Adaptive Cruise Control*, il *Cruise Control* e l'*AEB*.
La valutazione della sicurezza dei veicoli autonomi è estremamente lunga e complessa a causa del numero potenzialmente illimitato di scenari di traffico e prestazioni del sistema; per questo motivo, sulla base delle ricerche disponibili sulla generazione di scenari complessi, l'attenzione durante questa valutazione della sicurezza può essere focalizzata su alcune situazioni specifiche e quindi il numero di casi di prova necessari può essere ridotto senza mettere a rischio una sufficiente copertura di situazioni.
Come si vedrà nei prossimi Capitoli, le questioni sopra menzionate saranno prese in considerazione per la valutazione del comportamento di un'auto autonoma.

1.2 Obiettivi

L'obiettivo, considerando un AV[4], è quello di progettare un *Supervisor Controller* in grado di prendere decisioni lungo lo scenario del traffico stradale e di adattarne il suo movimento in base all'ambiente circostante.
I principali sistemi utilizzati per implementare gli *ADAS* sono i Radar, i LiDar, telecamere, sensori a ultrasuoni e GPS. Con tali sistemi l'auto ottiene un campo visivo dai 20° fino ai 360°.

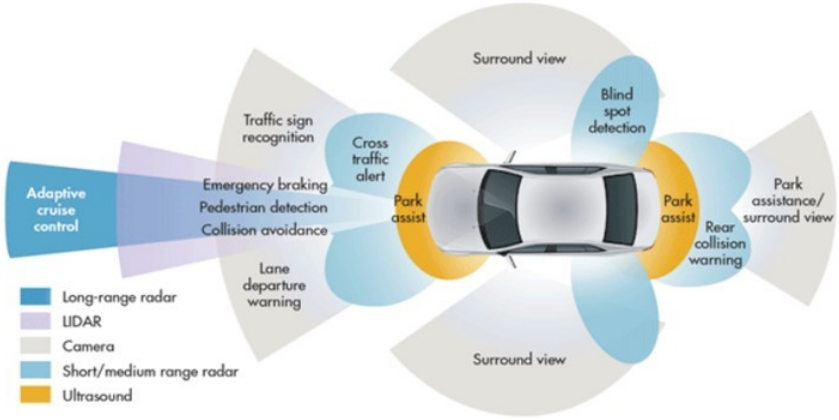


Figura 1.2: Advanced driver assistance system [3]

Nel caso in esame il veicolo è dotato di tre ADAS

1. Il Cruise Control;
2. L'Adaptive Cruise Control;
3. L'Autonomous Emergency Braking.

Il primo dispositivo permette di poter guidare a velocità costante quando non ci sono ostacoli.
Il secondo permette di regolare la velocità durante la marcia considerando le condizioni del traffico e la distanza di sicurezza da mantenere; vengono utilizzati sensori differenti che lavorano in sincronia per garantire un'andatura adeguata alla situazione.
Infine, tra le tecnologie ADAS, c'è il sistema per la frenata automatica di emergenza. La macchina è in grado di rilevare la distanza da pedoni, biciclette ed auto, in modo tale da evitare collisioni con ostacoli fermi sulla

[4]Autonomous Vehicle o Veicolo Autonomo

strada.
Il *Supervisor Controller* sulla base delle informazioni dell'ambiente circostante deve selezionare l'ADAS appropriato al contesto. Esistono vari sistemi di controllo che possono essere utilizzati per la realizzazione delle auto a guida autonoma.

1.3 Supervisor Controller

In questa sezione verrà fornita una breve descrizione di alcuni controllori più utilizzati.

1.3.1 PID Controller

Il Controllore PID presenta una legge di controllo semplice, tanto che esso è applicabile quasi in ogni situazione; tuttavia, esso ha prestazioni inferiori rispetto ad altri approcci.
Presenta tre componenti: la componente Proporzionale (P), la componente Integrale (I) e la componente Derivativa(D).
Per esemplificare l'apporto di ogni termine supponiamo di voler progettare un algoritmo affinché un'auto resti all'interno di una corsia. Ciò si ottiene calcolando l'angolo di sterzata proporzionale al *Cross Track Error*, ovvero la distanza laterale tra l'auto e la traiettoria di riferimento [11].
La componente Proporzionale imposta un angolo di sterzata proporzionale al CTE. Tuttavia, il risultato finale oscilla intorno alla traiettoria di riferimento.
La componente Derivativa utilizza un tasso di variazione per ridurre l'*overshoot*[5] causato dalla componente proporzionale.
La componente Integrale risolve il problema dell'accumulo di errori dovuti ad una distorsione sistematica che potrebbe portare l'auto fuori pista.

1.3.2 Linear Quadratic Regulator (LQR)

Questo controllore utilizza un modello lineare e una teoria di controllo ottimale per ottenere un controllore in retroazione. Questo approccio richiede le informazioni del modello in anticipo e i segnali di tutti gli stati durante il funzionamento, richiede quindi un osservatore di stato.
Mira a risolvere i problemi di stabilità del controllo del movimento laterale della guida intelligente. Deve solo trovare i parametri corretti in una funzione di costo quadratica per mantenere il sistema in funzione con il minimo costo. L'errore delle variabili di stato può essere eliminato dallo schema, ma le

[5]ampiezza dell'oscillazione intorno alla traiettoria di riferimento

prestazioni del sistema in questo periodo non sono garantite. L'obiettivo principale dell'LQR è quello di trovare i parametri della funzione di costo per far funzionare il sistema con le prestazioni desiderate. Tuttavia, non esistono istruzioni che guidino alla scelta dei parametri dell'LQR [16].

1.3.3 Model Predictive Controller

Il Model Predictive Controller(MPC) riceve informazioni sulla dinamica del veicolo e sulla strada per regolare di conseguenza l'accelerazione e mantenere l'auto nella situazione migliore.
L'MPC è una strategia adottata nell'industria come mezzo efficace per affrontare problemi di controllo. L'idea è quella di scegliere l'azione da eseguire risolvendo ripetutamente un problema di controllo. Ciò mira a creare una sequenza di controllo ottimale, dove il comportamento futuro può essere calcolato in base al modello dell'impianto.
Un importante vantaggio di questo tipo di controllo è la capacità di fare fronte a vincoli stringenti sul controllore e sugli stati [15].
Nel MPC si ottiene una sequenza di azioni di controllo ad anello aperto di cui solo la prima viene effettivamente applicata[6], all'istante di tempo successivo si ripete il procedimento.

1.3.4 Reinforcement Learning

Il *Reinforcement Learning* è un metodo di *Machine Learning* che simula l'apprendimento umano quando si è immersi in un ambiente. Questa tecnica consiste in un'entità, un agente, che deve prendere decisioni, che impara a svolgere un compito attraverso ripetute interazioni, commettendo anche errori. L'agente comunica con l'ambiente attraverso azioni. Oltre l'agente e l'ambiente si possono individuare altri sottoelementi tra i quali un segnale di ricompensa. Se l'azione scelta è buona allora porterà ad un risultato positivo e verrà data una ricompensa positiva. Le tecniche di *Reinforcement Learning* hanno senza dubbio raggiunto buoni livelli di prestazioni, dimostrando la capacità di queste tecniche di apprendere politiche quasi ottimali di gestione efficiente dei diversi sottosistemi del veicolo autonomo. Tuttavia, la maggior parte delle ricerche sono state condotte su vari simulatori o ambienti di prova limitati per una serie di motivi, che vanno dalle normative alla disponibilità di prototipi di veicoli. Di conseguenza, gli attuali modelli di RL non sono in grado di affrontare appieno gli ambienti del mondo reale, che sono pieni di incertezze che ostacolano le garanzie di sicurezza [10].

[6] tecnica *Receding Horizon*

Capitolo 2

Model Design

In questo Capitolo si andrà a studiare l'implementazione del modello adottato.
L'obiettivo principale è evitare ogni tipo di collisione. Dato che le condizioni circostanti non possono essere controllate l'unica possibilità per evitare l'impatto è il corretto funzionamento del controllo della velocità e della frenata. A tal fine sono stati adottati alcuni controllori PID.

2.1 Strategia di Controllo

Si è partiti dalla realizzazione del Controllore, il quale è composto da più blocchi. Ciascun ADAS implementato dispone del proprio.

2.1.1 Cruise Control

La prima implementazione che si va ad analizzare è quella del *Cruise Control*. Esso è un controllo con retroazione in uscita.

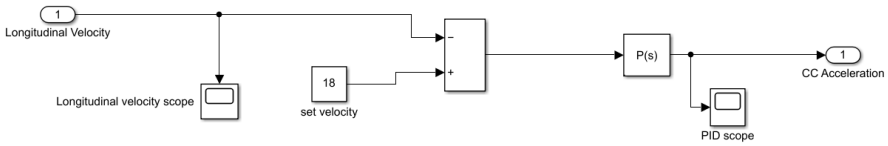


Figura 2.1: *Cruise Control*

In ingresso al blocco si avrà la velocità longitudinale del veicolo ed in uscita

l'accelerazione dello stesso. Come si nota è stato utilizzato un controllore *PID*[1] dove è stata impiegata esclusivamente l'azione Proporzionale, nella quale

$$K_p = 0.5 \tag{2.1}$$

Come si può desumere dall'immagine l'equazione implementata è:

$$u(t) = K_p \cdot (v_{set} - v_x(t)) \tag{2.2}$$

2.1.2 Adaptive Cruise Control

Di seguito il modello in Simulink che rappresenta l'ADAS.

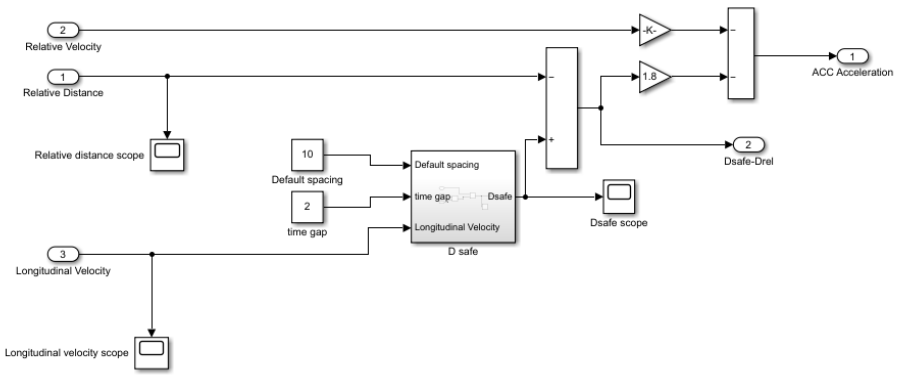


Figura 2.2: *Adaptive Cruise Control*

Esso è progettato come un blocco complementare al *Cruise Control*.
Presenta un controllo in retroazione con *feedforward* dove il guadagno

$$K_v = K = 0.85 \tag{2.3}$$

mentre

$$K_d = 1.8 \tag{2.4}$$

L'equazione di controllo che regola il blocco in esame è:

$$u(t) = K_d \cdot (d_{safe} - d_{rel}) + K_v \cdot v_{rel} \tag{2.5}$$

[1]Controllore che presenta tre azioni: Proporzionale, Integrativo e Derivativo.

In particolare si nota la presenza di un blocco che permette di calcolare il d_{safe} *(Figura 2.3)*.
Nell'equazione il termine d_{rel} esprime la distanza relativa tra veicolo principale e l'oggetto che si trova dinanzi ad esso.

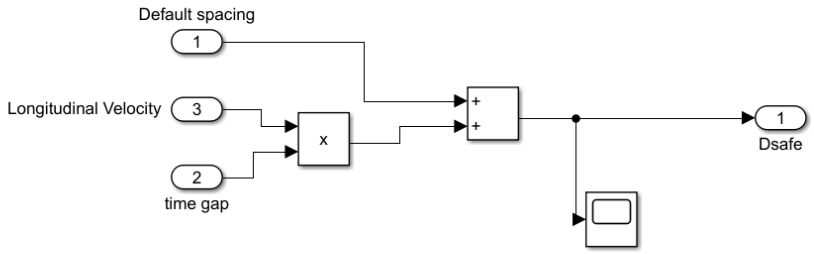


Figura 2.3: d_{safe}

Si è detto che *Cruise Control* e *Adaptive Cruise Control* sono due blocchi complementari proprio perché come possiamo vedere in *Figura 2.4* è presente un blocco *CC/ACC Selector* che permette di selezionare uno dei due ADAS.

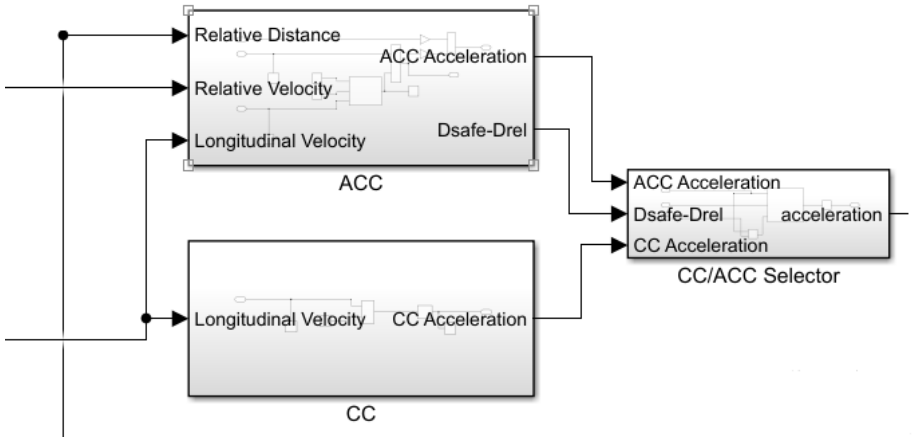


Figura 2.4: CC/ACC Selector

Si può notare la presenza di uno Switch: il segnale passa attraverso l'ingresso 1 se l'ingresso 2 soddisfa il criterio selezionato, ovvero

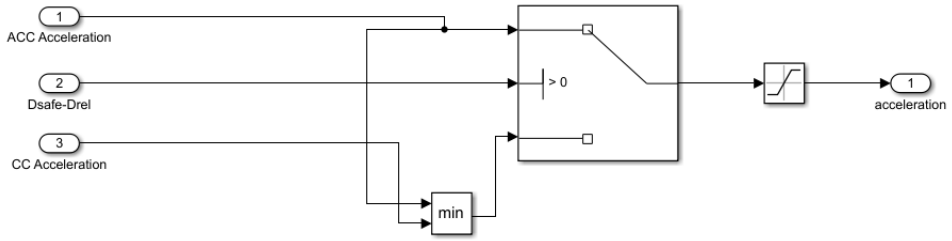


Figura 2.5: CC/ACC Selector Block

$$d_{safe} - d_{rel} > 0 \tag{2.6}$$

alternativamente passa attraverso l'ingresso 3. La prima e la terza sono porte d'ingresso dati, mentre la seconda è una porta di controllo.
Il criterio selezionato in questo caso è quando il *treshold* risulta maggiore di 0:

$$u_2 > 0 \tag{2.7}$$

È presente un blocco *min* che sceglie il valore minimo tra *ACC Acceleration* e *CC Acceleration*.
In uscita è presente un blocco di saturazione.

2.1.3 Autonomous Emergency Braking

Il dispositivo *AEB* è considerato la priorità quando l'ACC non è in grado di evitare la collisione, per cui affinché il blocco riguardante l'*AEB* funzioni il *Cruise Control* deve necessariamente funzionare.
Esso è costituito da vari sotto blocchi:

1. TTC Calculation ovvero il *Time To Collision Calculation* il quale, dato in ingresso la velocità e la distanza relativa tra il veicolo principale e gli altri elementi, dà in uscita il tempo di collisione se la velocità dovesse restare invariata.

2. Nel sottosistema *Stopping Time Calculation* viene calcolato il tempo necessario per raggiungere la velocità 0, date le tre decelerazioni dell'AEB:

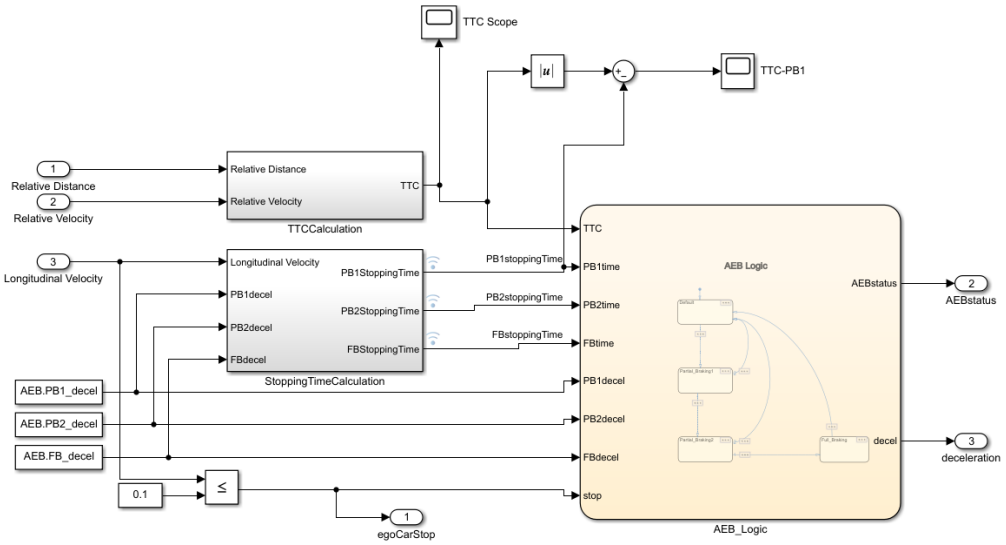


Figura 2.6: AEB Controller

- $AEB\_PB1\_decel = 6\ m/s^2$
- $AEB\_PB2\_decel = 8\ m/s^2$
- $AEB\_PB3\_decel = 9.8\ m/s^2$

3. Infine, nella *AEB_Logic* prendendo gli input dai due blocchi precedenti si determina se impiegare e in quale misura l'*AEB*, aumentando la forza frenante ad ogni fase, fino al completo arresto della vettura.

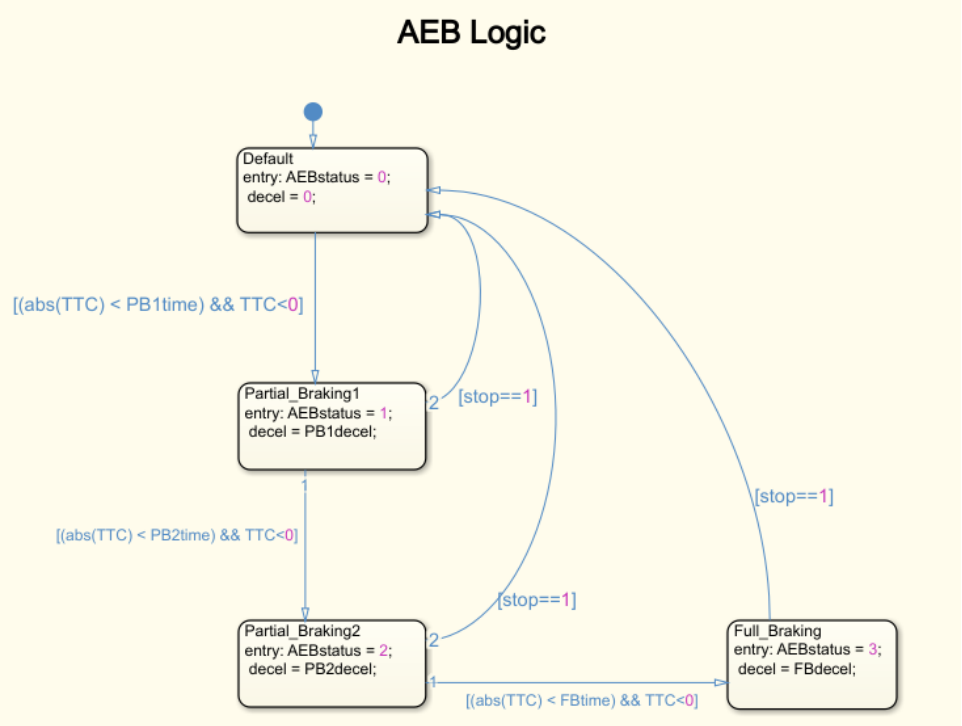


Figura 2.7: *AEB Logic*

Di seguito uno schema sulla logica *AEB*, in relazione alla distanza

Condizione	AEB Status
TTC<FWC_Time && TTC<0	0 = Pronto all'intervento
TTC<PB1_Time && TTC<0	1 = Partial Braking con $6m/s^2$
TTC<PB2_Time && TTC<0	2 = Partial Braking con $8m/s^2$
TTC<PB3_Time && TTC<0	3 = Full Braking con $9.8m/s^2$

Tabella 2.1: AEB Status [8]

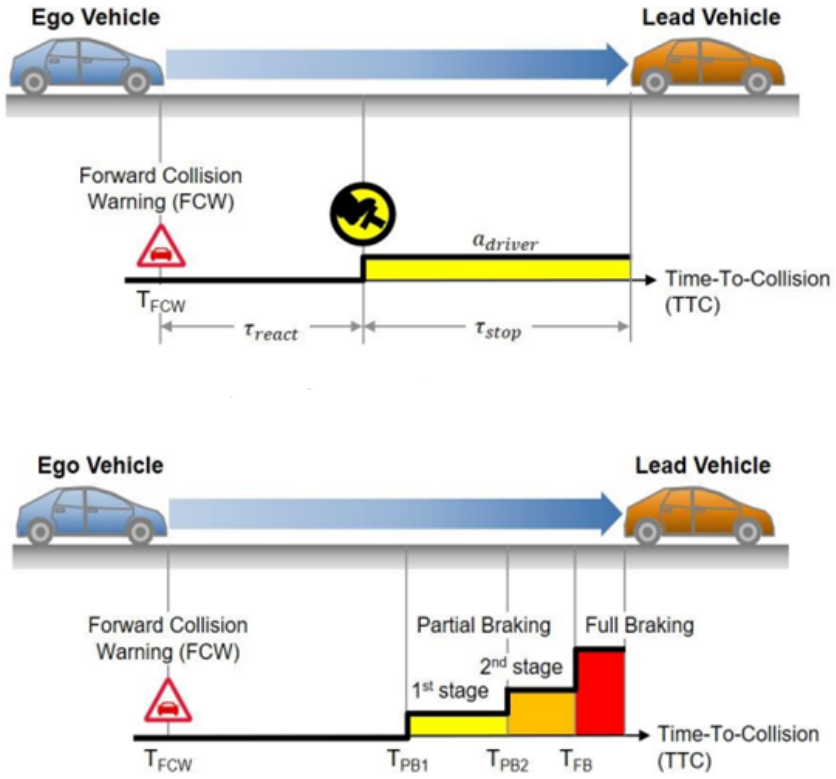


Figura 2.8: AEB Status [2]

Una volta abilitato l'*AEB* le condizioni di frenata vengono calcolate come di seguito:

$$HeadwayTime = \frac{DistanzaRelativa}{VelocitaVeicolo} \tag{2.8}$$

$$TimeToCollision(TTC) = \frac{DistanzaRelativa}{VelocitaRelativa} \tag{2.9}$$

$$ForwardCollisionWarningTime(FCWTime) = \frac{VelocitaVeicolo}{Accelerazione} \quad (2.10)$$

$$PartialBraking1Time(PB1\_Time) = \frac{VelocitaVeicolo}{6} \quad (2.11)$$

$$PartialBraking2Time(PB2\_Time) = \frac{VelocitaVeicolo}{8} \quad (2.12)$$

$$PartialBraking3Time(PB3\_Time) = \frac{VelocitaVeicolo}{9.8} \quad (2.13)$$

Infine, in uscita è presente un *Accelerator/Decelerator Selector* che rilascia l'acceleratore non appena l'AEB viene attivato.

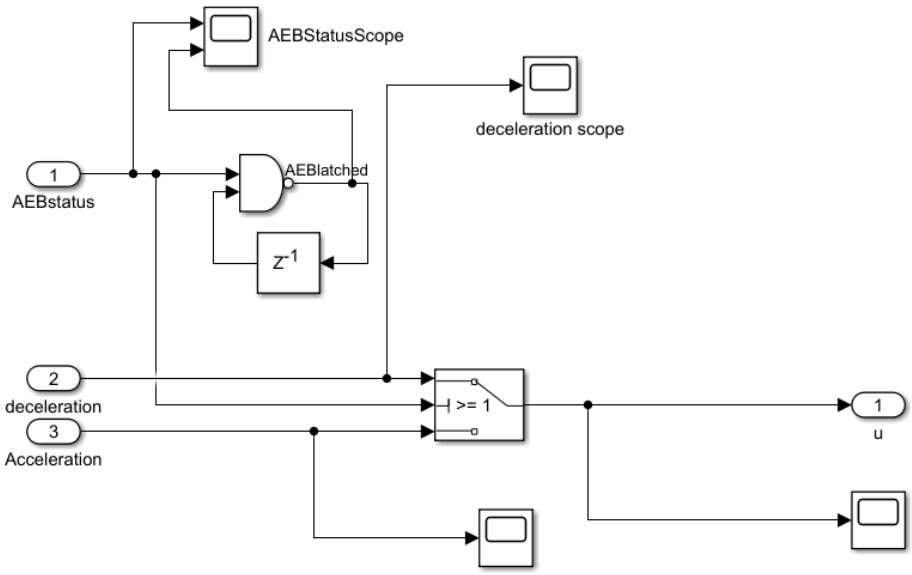


Figura 2.9: Accelerator/Decelerator Selector

Dalla connessione dei blocchi precedenti ricaviamo il modello finale del controllore che consente di implementare i 3 ADAS.

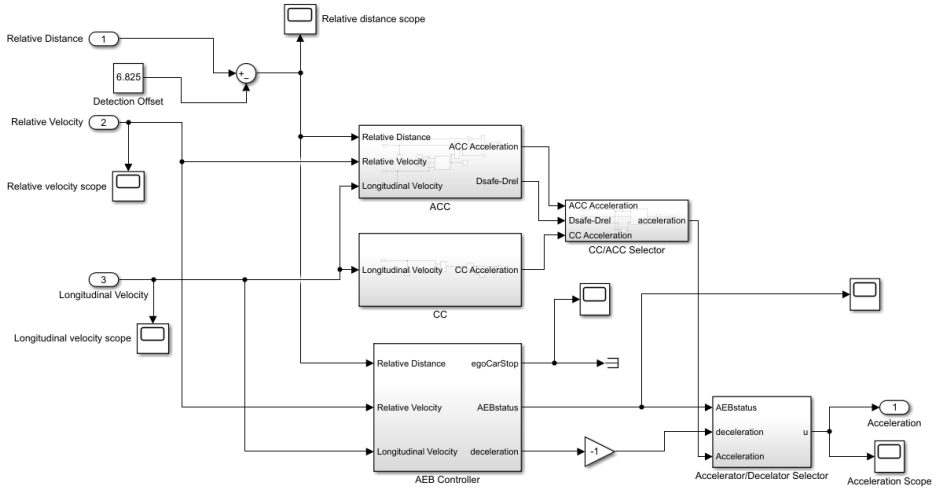


Figura 2.10: Controllore

2.2 Tracking and Sensor Fusion

Poiché nessun sistema singolarmente è in grado di fornire un funzionamento corretto al 100% in tutti gli scenari a causa delle numerose condizioni variabili che si verificano durante la guida di un'auto, fondendo insieme diversi sistemi di rilevamento è possibile costruire un sistema coerente che agisca in modo sicuro in diverse situazioni. Il *Sensor Fusion* offre la possibilità di unire dati provenienti dai vari sensori e dare priorità al segnale più accurato da utilizzare. Ad esempio, l'*AEB* può utilizzare il *Radar* per distanze a corto e lungo raggio, mentre il *Lidar* per distanze a medio raggio, per la rilevazione di oggetti e velocità. Inoltre, le telecamere possono essere utilizzate per ottenere informazioni dettagliate sulla posizione dei segnali stradali e dei pedoni. Per dare un'idea delle capacità dei sensori, i radar riescono a rilevare quasi tutti gli oggetti; tuttavia, non sono in grado di riconoscere il tipo di oggetto. Al contrario i sistemi di visione sono in grado di leggere segnali stradali e linee di demarcazione delle corsie, ma le loro prestazioni non sono ottimali in situazioni di luce diretta solare o quando la lente della telecamera potrebbe risultare sporca. Tuttavia, la presenza del *Lidar* colma queste carenze.
La simulazione di rilevamenti radar e visivi offre la possibilità di creare eventi di interesse e testare algoritmi.
Su di un veicolo autonomo sono presenti diversi sistemi sensoriali che lavorano simultaneamente per effettuare rilevazioni. Tali segnali vengono inviati al computer centrale dell'auto e tradotti tramite metodi del *Sensor Fusion*,

ed infine inviati agli attuatori[2]. Dunque gli attuatori eseguono il compito richiesto ed attivano o disattivano le funzioni ADAS a seconda dei segnali ricevuti.
Tema importante è il posizionamento dei sensori e delle telecamere in modo da ricoprire più campo visivo possibile; di conseguenza, quando i dati provenienti dai sensori sono sufficienti, il controllore è in grado di attivare/disattivare le funzioni degli ADAS per mantenere l'auto nelle migliori condizioni durante scenari di guida critici.

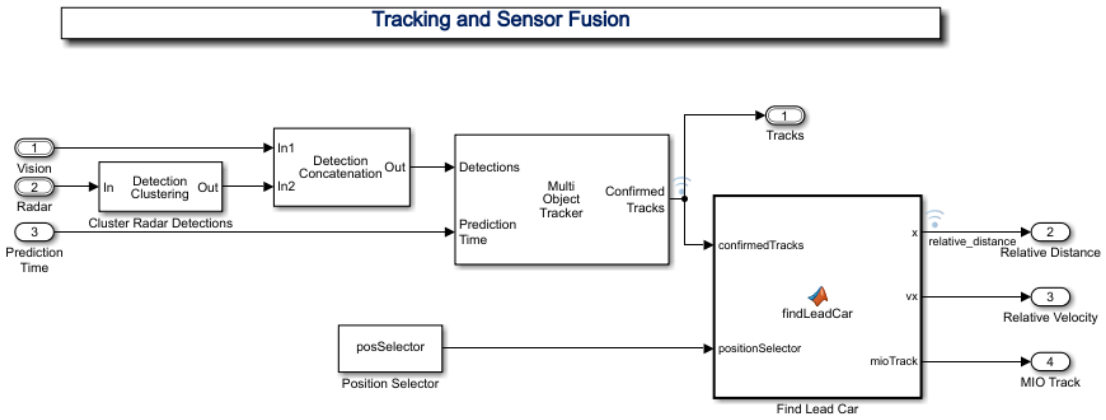


Figura 2.11: Tracking and Sensor Fusion

Vengono raggruppati tutti i rilevamenti generati da un sensore, se i rilevamenti sono entro una certa distanza, chiamata *Cluster Size*, l'uno dall'altro. Successivamente vengono convogliati i bus di rilevamento di più sensori su un unico bus di uscita. È utile quando i rilevamenti da più blocchi sensore vengono passati in un *Multi-Object Tracker Block*. Tale blocco crea e gestisce le tracce degli oggetti in movimento. Gli input del tracker sono apporti di rilevamento generati dai blocchi *Radar Detection Generator* e *Vision Detection Generator*.
Infine, la funzione *findLeadCar* è implementata in modo da poter riconoscere l'auto che precede l'*Ego Vehicle*[3].

[2]Un attuatore è definito come un qualsiasi dispositivo che converte energia da una forma ad un'altra, in modo che questa agisca nell'ambiente fisico al posto dell'uomo[12]
[3]Ego Car o Ego Veichle è il veicolo principale dotato dei sensori

2.3 Vehicle and Environment

Ultimo blocco presente nel modello è quello che riguarda il veicolo e l'ambiente circostante ad esso. Si andrà a considerare prima tutto ciò che concerne la dinamica del veicolo implementata tramite il blocco *Vehicle Dynamics* e successivamente gli attori e l'ambiente tramite il blocco *Actor and Sensor Simulation*.

2.3.1 Vehicle Dynamics

Per progettare efficacemente un sistema di controllo per veicoli autonomi è necessario un modello dinamico del sistema in esame valido e semplice da gestire. Verranno affrontati, di seguito, gli aspetti cinematici e dinamici per dare una visione d'insieme al lettore. Il modello di veicolo sfruttato per la progettazione del controllore è derivato dal noto modello di bicicletta, la cui dinamica è implementata nel $VehicleDynamicsBlockset$ di Matlab.
Il movimento del veicolo è descritto in termini di velocità longitudinale, velocità laterale, velocità verticale, rollio, beccheggio e imbardata nel sistema di coordinate fisse del veicolo, riferite ad un sistema di riferimento inerziale. Nel caso esaminato, dato il modello scelto, il moto risulta planare ovvero parallelo alla superficie stradale e non è presente alcun moto verticale, di beccheggio o di rollio[4]. Quindi il veicolo, in questo caso, è controllato da comandi di sterzata e di accelerazione. Non sono necessari blocchi di saturazione per queste grandezze poiché i vincoli sono già incorporati nel controllore sotto forma di vincoli matematici [7].
Come si può evincere dalla *Figura 2.12* il segnale in ingresso è applicato per metà sull'asse anteriore e per metà sull'asse posteriore.

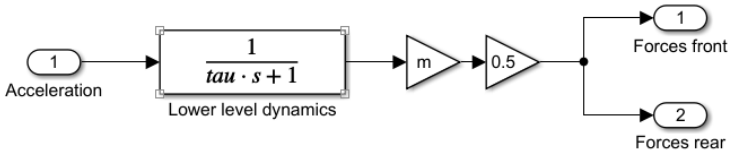


Figura 2.12: Lower Level Dynamics

[4]Gli angoli di slittamento laterale dello pneumatico destro e sinistro sono uguali.

Applicando l'equazione di Newton lungo la direzione longitudinale otterremo:

$$\dot{x}(t) = v(t) \tag{2.14}$$

$$\dot{x}(t) = \frac{\eta}{R \cdot m} \cdot u(t) - g \cdot \sin\theta - g \cdot f_r \cdot \cos\theta - \frac{0.5}{m} \cdot \rho \cdot C_d \cdot C_h \cdot A_f \cdot v_i^2(t) \tag{2.15}$$

L'*equazione 2.15* rappresenta la dinamica del veicolo descritta dal seguente sistema dinamico longitudinale non lineare.
Dove:

- v è la velocità del veicolo espressa in m/s^2;
- p è la posizione del veicolo espressa in m;
- u è l'input di controllo espresso in m/s^2;
- m è la massa del veicolo espressa in kg, posta pari a $1521kg$;
- f_r è il coefficiente di resistenza al rotolamento, posto pari a 0.015;
- ρ è la densità dell'aria espressa in kg/m^3, posta pari a $1,2kg/m^3$;
- C_d è il coefficiente di resistenza aerodinamica, posto pari a 0.28;
- η è l'efficienza, posta pari a 0.93;
- R è il raggio della ruota, posto pari a 0.5;
- C_h è il fattore di altitudine, pari a $1 - 0.085H$[5];
- A_f è l'area frontale del veicolo espresso in m^2, posta pari a $2.33m^2$;
- θ è la pendenza della carreggiata, espressa in rad

Bisogna tenere conto che sui parametri dei veicoli sopra presentati è presente un'incertezza del 10%.

[5]H (km) è l'altitudine della strada

2.3.2 Actor and Sensor Simulation

Per quanto concerne l'*Actor and Sensor Simulation Block* avremo in ingresso *quattro* parametri relativi all'$EgoVehicle$:

- Posizione;
- Velocità;
- Angolo di imbardata;
- Velocità di imbardata;

I vari parametri passeranno attraverso degli ZOH[6], che ricampionano ad un'unica frequenza di campionamento tutte le uscite.
Successivamente le grandezze fisiche dell'*Ego Vehicle* convertite vengono incapsulate in una struttura Matlab attraverso la funzione *packEgo* che viene sfruttata dal blocco *ScenarioReader*. Tale blocco, il quale utilizzo sarà esemplificato nel capitolo successivo, legge le informazioni dello scenario e aggiorna i comportamenti dell'*Ego Vehicle* provenienti dal blocco sulla dinamica del veicolo. La struttura dati citata si ottiene dall'applicazione *DrivingScenarioDesigner*, toolbox che si approfondirà anch'esso nel capitolo successivo.
Elementi costitutivi di questo modello sono, inoltre, il *VisionDetectionGenerator* ed il *RadarDetectionGenerator*. Essi vengono utilizzati per generar rilevamenti visivi a partire da una telecamera e da un radar montati sul veicolo. Le rilevazioni effettuate sono riferite al sistema di coordinate del veicolo. È stato sfruttato, inoltre, un blocco Matlab per convertire alcune grandezze angolari da gradi a radianti.

[6]Zero Order Holder

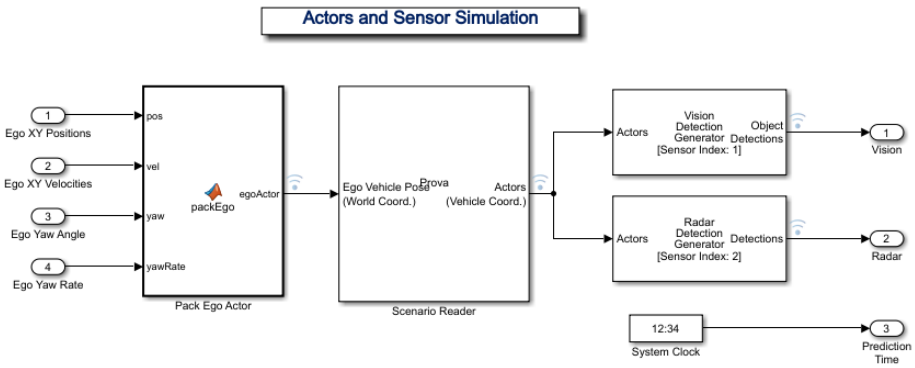


Figura 2.13: Actor and Sensor Simulation

Capitolo 3

Implementazione Scenario

3.1 Introduzione all'Automated Driving Tool

In questo capitolo si tratterà il tema della creazione di uno scenario in *Matlab Simultink* tramite l'*Automated Driving Toolbox*.
Tale toolbox fornisce algoritmi e strumenti per la progettazione, simulazione e test di sistemi ADAS e di guida autonoma. È possibile progettare e verificare sistemi di percezione e di visione, così come algoritmi di *Sensor Fusion*, pianificazione percorsi o controllori dei veicoli.
Il toolbox supporta, inoltre, la generazione di codici C/C++ per prototipazione rapida e test HIL[1], con assistenza per algoritmi di *Sensor Fusion*, tracciamento, pianificazione dei percorsi e controllori per veicoli.
Le caratteristiche principali di questo toolbox sono l'applicazione *drivingScenarioDesigner*, il *Bird's-Eye Plot View*, il *Vision Detector Generator* per la generazione di rilevamenti sintetici e lo *Scenario Reader* [5].
Come riportato in *Automated Driving Toolbox™ User's guide* [6] il toolbox usa il seguente sistema di coordinate:

- Mondo: un sistema di coordinate universali fisso (X, Y, Z) in cui sono collocati i veicoli ed i sensori. È una proprietà del *Driving Scenario* e definiscono in modo univoco la posizione di ciascun attore nello scenario.

- Veicolo: è il sistema di coordinate associate al veicolo (x, y, z) a cui è collegato. x punta in avanti rispetto al veicolo, y punta a sinistra se il veicolo è visto con una prospettiva frontale e z punta in alto.

[1]Con *Hardware in the Loop* si indicano le tecniche di verifica (testing) di unità di controllo elettronico, come le centraline delle automobili [13]

- Sensore: è il sistema di coordinate (X_c, Y_c, Z_c) relative al sensore che puntano nella stessa direzione di quelle del Veicolo.
- Spaziale: sistema di coordinate (x, y) che definisce la griglia discreta di pixel.

3.2 Creazione Scenario

Questa applicazione Matlab fa parte dell'*Automated Driving Toolbox* ed è stata utilizzata in questo lavoro per progettare scenari di guida per algoritmi di controllo.
Analizziamo la costruzione di uno scenario.
Per aprire l'applicazione inserire, nel prompt di comandi di Matlab, la dicitura "*drivingScenarioDesigner*".

3.2.1 Strade

Una volta aperto il canvas è possibile modellare a piacere lo scenario con, ad esempio, la creazione di una strada.
Nella barra degli strumenti dell'app selezionare "Add Road".

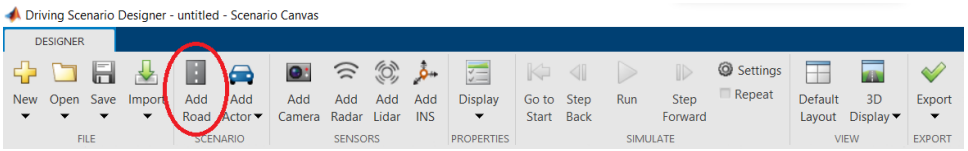


Figura 3.1: Barra Strumenti Scenario

Quindi creare la strada con il cursore direttamente dal canvas.
Per rendere la strada curva bisogna aggiungere un centro strada intorno al quale curvarla. Basta fare click con il pulsante destro del mouse e selezionare "Add Center Road". Trascinare quindi il centro stradale per generare la curva.
Il procedimento si può reiterare per costruire strade più complesse.
Di default la strada presenta un'unica corsia di larghezza 3.6 metri, standard per una tipica larghezza autostradale.
Per rendere lo scenario più realistico si può aumentare il numero delle corsie: nel pannello di sinistra entrare nella scheda "Roads" ed espandere la sezione "Lanes".

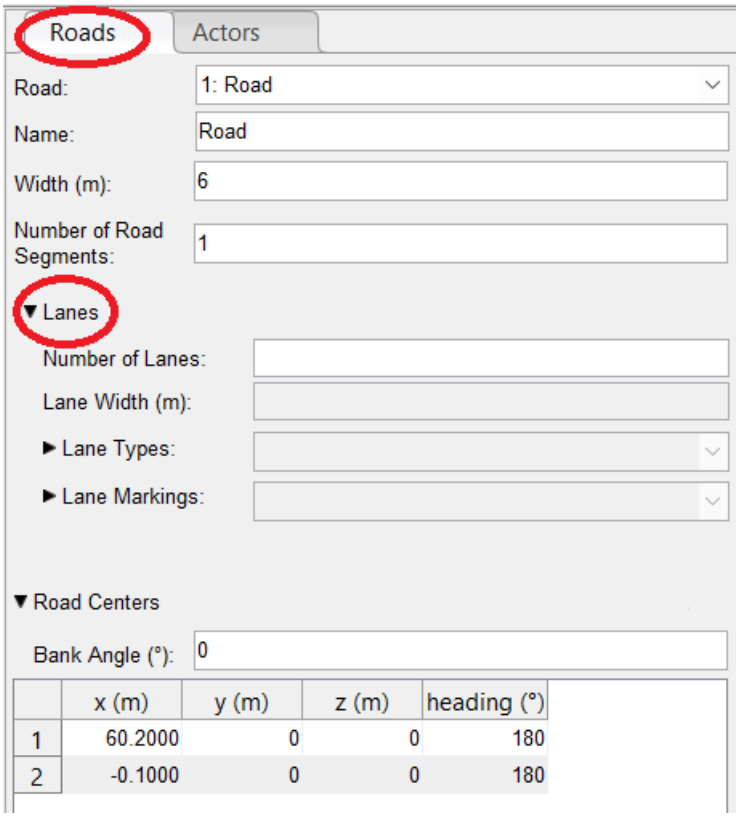


Figura 3.2: Road Panel

Qui è possibile scegliere il numero di strade: per aumentare le corsie dello stesso senso di marcia basta modificare nella sezione "Lanes" il "Number of Lanes".
Al contrario, se si vogliono aggiungere più corsie per senso di marcia bisogna impostare il "Number of Lanes" attraverso la dicitura $[nn]$.
La striscia gialla al centro indica che la corsia è a doppio senso di marcia.
Per ispezionare o modificare le corsie create, dalla sezione a sinistra "Lane Markings" e "Lane Types" selezionarne una e modificarne i parametri.
Inoltre, è possibile anche aggiungere ai lati della strada due tipologie di barriere:

- Guardrail;
- Jersey Barrier[2].

[2]Barriera modulare in plastica o in cemento impiegata per separare le corsie di traffico e limitare al minimo i danni al veicolo in caso di contatto[14]

Per aggiungere le barriere ai bordi della strada si può utilizzare il menu relativo alla strada scelta direttamente dal canvas cliccando con il tasto destro sulla strada:

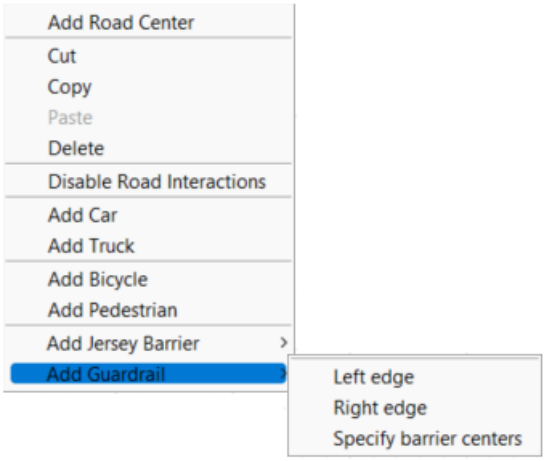


Figura 3.3: Aggiunta Guardrail/Jersey Barrier

Bisognerà scegliere anche il lato stradale sul quale aggiungere la barriera. È possibile, inoltre, aggiungere uno spazio tra i segmenti della barriera andando a modificare il valore della proprietà "Segment Gap" nella scheda "Barriers."

3.2.2 Attori

Di default la prima macchina che si può aggiungere è l'*Ego Vehicle*, ovvero la macchina principale all'interno dello scenario. Esso è l'unico veicolo sul quale vengono montati i sensori che rilevano linee di demarcazione delle corsie, pedoni o altre auto all'interno dello scenario.
L'aggiunta dell' *Ego Vehicle* può essere effettuata direttamente dalla barra strumenti tramite la sezione "Add Actor". Alternativamente, tasto destro del mouse sul Canvas e "Add Car", in modo da creare l'auto direttamente in quel punto. Si può anche specificare la traiettoria che l'auto deve seguire; per farlo basta cliccare con il tasto destro sull'auto e selezionare "Add Forward Waypoints" o "Add Reverse Waypoints". È possibile aggiungere waypoints in sequenza per creare una traiettoria più precisa, andando a regolarne le coordinate nell'apposita sezione.
Per regolare la velocità dell'auto, nel riquadro a sinistra, modificare la voce relativa ad essa per ogni singolo waypoint.

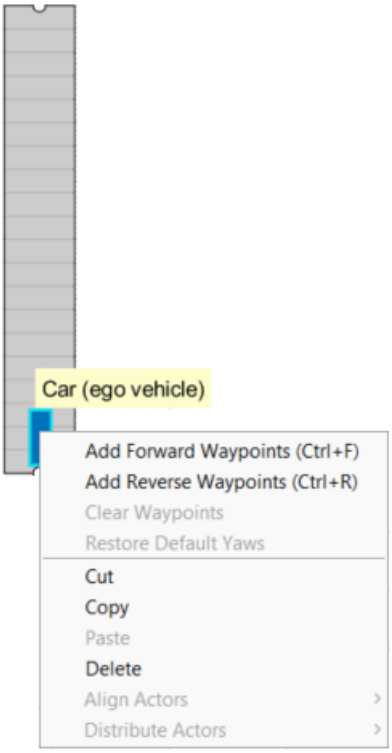


Figura 3.4: Aggiunta Waypoints

Waypoints, Speeds, Wait Times, and Yaw

	x (m)	y (m)	z (m)	v (m/s)	wait (s)	yaw (°)
1	0	0	0	30	0	0
2	10	0	0	30	0	0
3	20	0	0	30	0	0

Figura 3.5: Interfaccia Waypoints

Dopo aver creato una traiettoria per l'*Ego Vehicle* è possibile aggiungere all'interno dello scenario altri veicoli da far rilevare ai sensori posizionati su quest'ultima. Sempre nella barra degli strumenti scegliere "Add Actor" e selezionare "Car". Analogamente a quanto visto in precedenza è possibile aggiungere, per la seconda auto, i waypoints e creare una traiettoria.
Si possono, poi, aggiungere i pedoni alla stregua di quanto visto per le auto, il cui comportamento viene sempre caratterizzato da waypoints.
Generalmente la simulazione termina quando il primo attore completa la sua traiettoria. In alternativa, si può scegliere di far terminare la simulazione solo dopo che tutti gli attori hanno completato la traiettoria andando a modificare le impostazioni riguardo "Simulation Setting" e in particolare la condizione

"Stop Condition".

3.2.3 Sensori

Il toolbox mette a disposizione varie tipologie di sensori da poter implementare sull'*Ego Vehicle*.
Nella barra degli strumenti scegliere "Add Camera".
Verrà aperto il pannello "Sensor Canvas" dal quale è possibile scegliere la posizione del sensore tra quelle disponibili. Scegliere la posizione anteriore per aggiungere una camera sul paraurti del veicolo. Generalmente la telecamera rileva solo gli oggetti e non le corsie; per cambiare questa impostazione, nella scheda "Sensori", espandere la sezione "Sensor Parameters" e modificare il tipo di oggetti che si vuole andare a rilevare.
Una volta applicata la telecamera è possibile aggiungere altri sensori quali il Radar. Per impostazione predefinita questa tipologia di sensori è a corto raggio. Si può, inoltre, scegliere l'area di copertura andando a modificare l'area di visione.
Tale sistema utilizza onde elettromagnetiche per il rilevamento e la determinazione (in un dato sistema di riferimento) della posizione ed eventualmente della velocità di oggetti (bersagli, target) sia fissi che mobili. A seconda del sistema, le onde radio vengono trasmesse in modo pulsato o continuo dal trasmettitore, le quali si riflettono sull'oggetto e tornano al ricevitore, fornendo informazioni sulla posizione dell'oggetto.
Altra tipologia di sensore è il Lidar, ovvero un *Laser Imaging Detection and Ranging*, che consiste in una tecnica di telerilevamento che permette di determinare la distanza di un oggetto o di una superficie usando un impulso laser. Il Lidar fa rimbalzare la luce sugli oggetti per rilevarne la posizione, alla stregua di come il Radar usa le onde radio. L'ampio campo visivo del Lidar, rispetto a quello più ristretto dei radar, è un vantaggio in quanto il Lidar può creare dati 3D rilevando la forma degli oggetti. L'area grigia che compare intorno al veicolo simboleggia l'area di copertura del sensore.

3.2.4 Birdy's Eye

Su Simulink, una volta dopo aver eseguito lo script Matlab per impostare il controllore e i vari parametri è possibile aprire una serie di tool, tra cui il *Bird's Eye Scope*, per analizzare e ispezionare i dati dello scenario. Durante l'esecuzione il *Bird's-Eye Plot* mostra rilevamenti tramite una visuale isometrica, mentre l'*Ego-Centric View* mostra lo scenario dalla prospettiva dell'*Ego Vehicle*. Questo strumento offre una vista dall'alto dell'auto e dello

scenario di guida, e permette di capire visivamente ciò che succede durante la simulazione. Ad ogni passo il blocco "Scenario Reader" riceve le informazioni e sposta gli attori di conseguenza.
Una volta avviata la simulazione l'*Ego Vehicle* resterà al centro dello scenario e gli altri attori si muoveranno di conseguenza per simulare le azioni. Parametro importante è il tempo di simulazione che non può essere superiore rispetto al tempo in cui l'auto raggiunge la posizione finale. Questa posizione viene calcolata automaticamente a priori a partire dalle informazioni sui waypoints e sulla velocità dell' auto. In caso contrario si verificherà un errore.

3.2.5 Scenario Reader

Il blocco "Scenario Reader" legge le intenzioni degli attori e i dati della strada da uno scenario di guida.
La descrizione che segue è stata riportata come presentata sulla guida di Matlab: il blocco produce il comportamento degli attori nel sistema di coordinate dell'*Ego Vehicle* o nelle coordinate del Mondo dello scenario. Il blocco consente anche l'output dello stato dell'*Ego Vehicle*, che include le misure di accelerazione.
Una volta creato lo scenario, esso deve essere caricato all'interno del modello attraverso l'interfaccia:

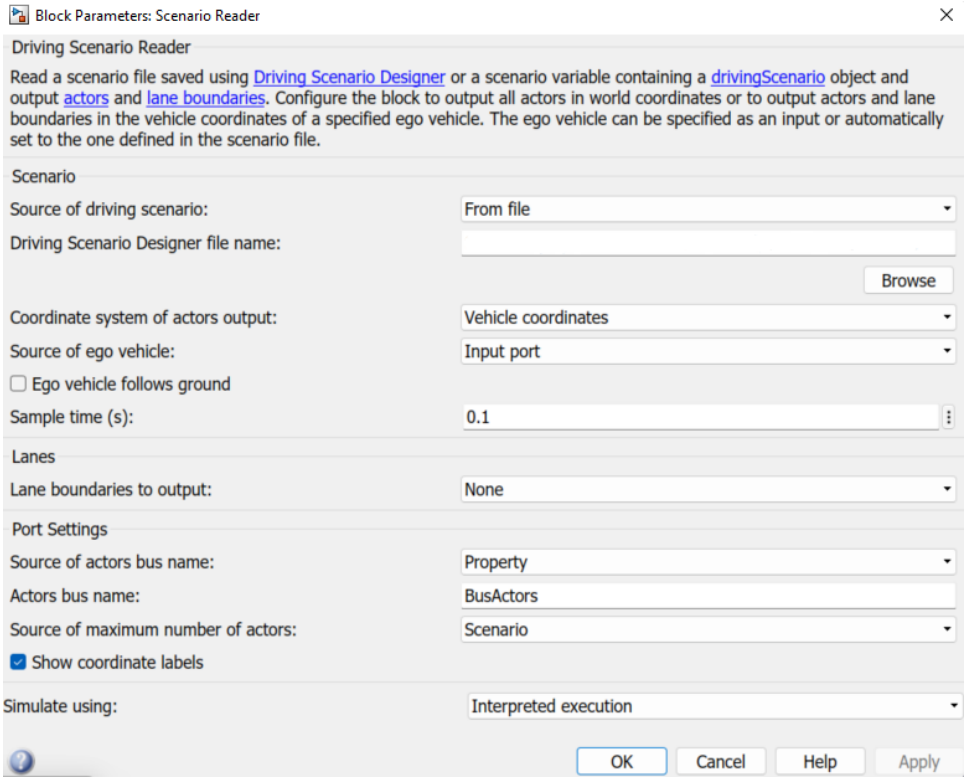


Figura 3.6: Scenario Reader

Capitolo 4

Simulazione Scenario

Partendo dagli elementi base messi a disposizione dal toolbox Automated Drived di Matlab si è andati a costruire uno scenario più complesso in cui, attraverso il modello in Simulink, si modella il comportamento dell'*Ego Vehicle* in base alle situazioni che possono presentarsi. Andiamo a inserire lo scenario all'interno del modello visto in precedenza e attraverso il *Bird's-Eye Scope* vediamo cosa succede.

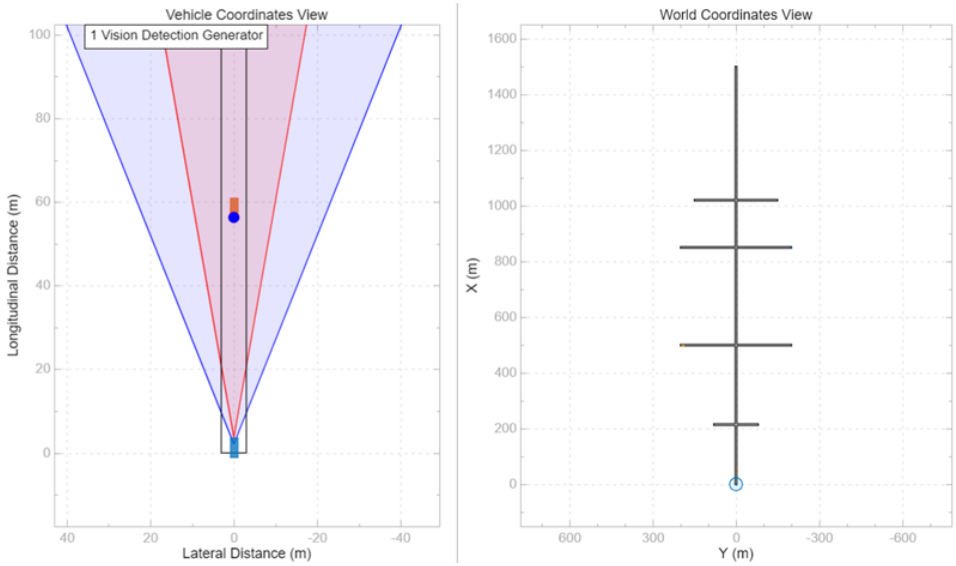


Figura 4.1: Panoramica Scenario

Lo scenario presenta una strada dritta con quattro incroci.
Il primo incontro sarà con un'auto che si muove a velocità minore rispetto alla principale, essa dovrà, allora, adattare la velocità in funzione di quella. Al cambio di corsia dell'ostacolo, l'*Ego Vehicle* riprenderà a muoversi con marcia sostenuta.
Al secondo incrocio si presenterà un ostacolo a centro corsia, l'auto dovrà frenare fino a fermarsi, per poi ripartire una volta che l'ostruzione non ingombra più.
Gli ultimi due incroci presentano un'immissione e l'uscita per una bicicletta. L'*Ego Vehicle* adatterà la propria velocità a quella della biciletta, per poi accelerare una volta che essa avrà svoltato.

La simulazione inizia con l'*Ego Vehicle* che percorre la strada rettilinea con una velocità iniziale impostata pari a $15m/s$.
Nel corso della simulazione si presenta un'automobile che si muove con velocità inferiore a $15m/s$.
In questo caso, il sensore noterà che è presente un oggetto, un veicolo, davanti all'auto e andando a misurare la distanza di sicurezza, appena essa scenderà sotto una soglia impostata, pari a $d_{safe} - d_{rel}$ lo switch modificherà l'andamento e si passerà da un *Cruise Control* ad un *Adaptive Cruise Control*.

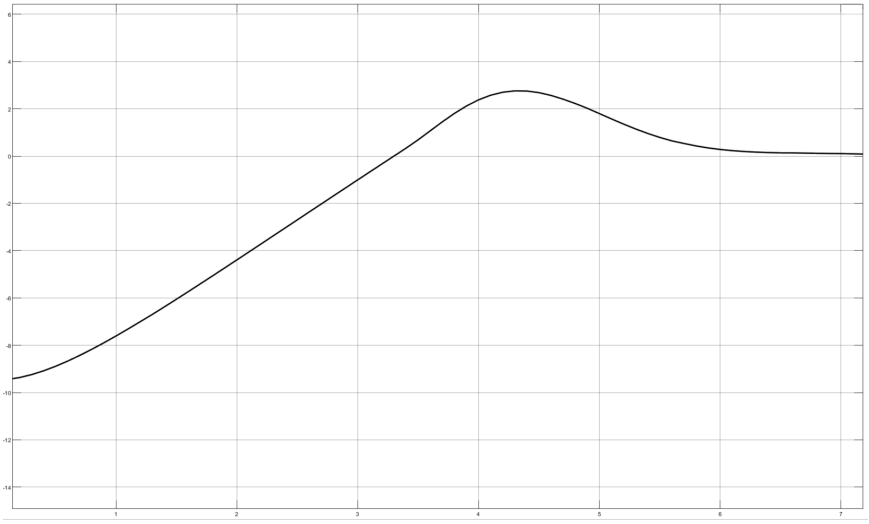
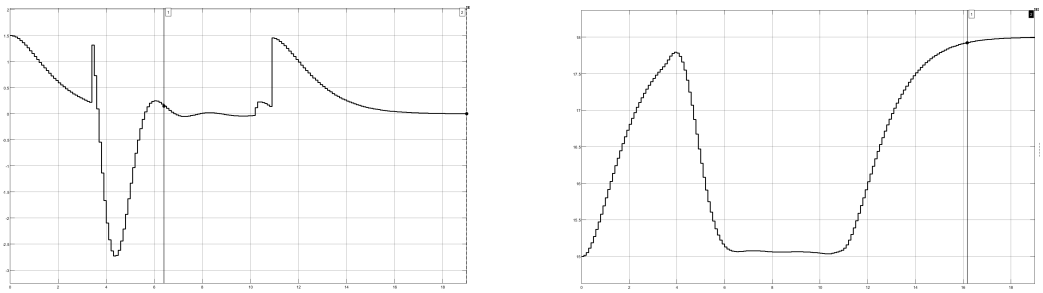


Figura 4.2: $d_{safe} - d_{rel}$

Evidenziamo nel grafico due punti di interesse

- Il primo è quando la quantità $d_{safe} - d_{rel} > 0$, dato che per come è stato implementato l'ACC/CC *Selector* si passerà da un CC ad un ACC.

- Analogamente nel momento in cui a circa 6s la distanza torna ad essere 0 poiché l'auto ha svoltato riprende il *Cruise Control*.



(a) (b)

Figura 4.3: Dinamica del veicolo. Accelerazione(a) e Velocità(b)

Nel momento in cui l'auto secondaria svolta a destra il sensore dell' *Ego Vehicle* rileverà che non è presente alcun vincolo sulla carreggiata e quindi accelererà, come si evince anche dai grafici.

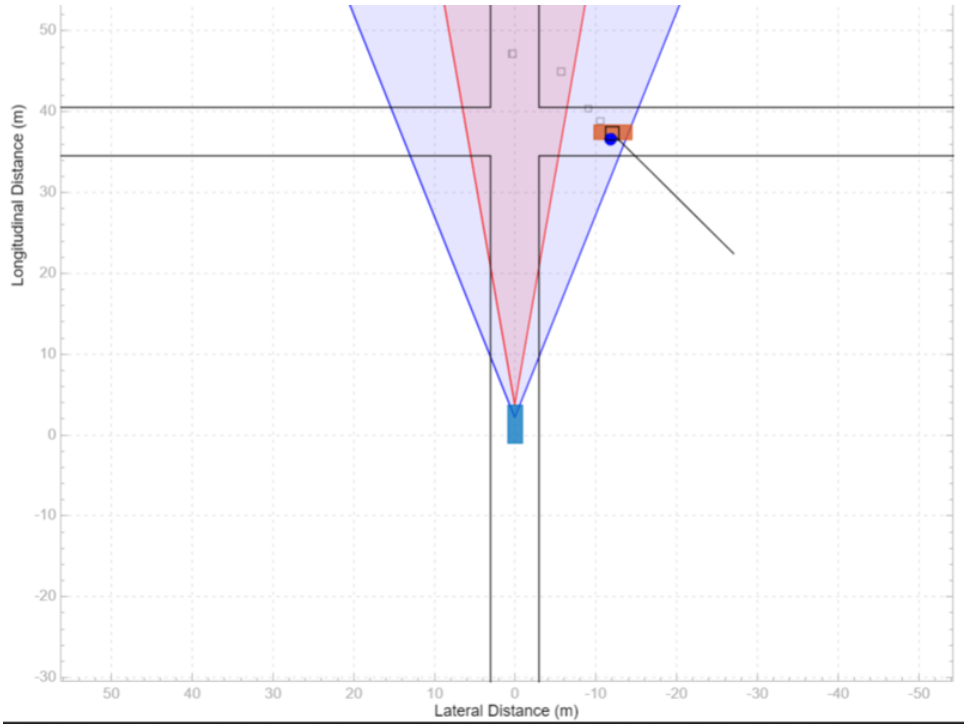


Figura 4.4: Svolta Auto Secondaria

Successivamente, avremo un veicolo che, attraversando l'incrocio, si pone al centro della corsia (*Figura 4.5, 4.6*). Il veicolo parte con *Cruise Control* e viene attivato l'AEB prima attraverso un *Partial Braking* e successivamente, dato che l'ostacolo viene percepito dal sistema ad una distanza molto ravvicinata, attraverso un *Full Braking*.

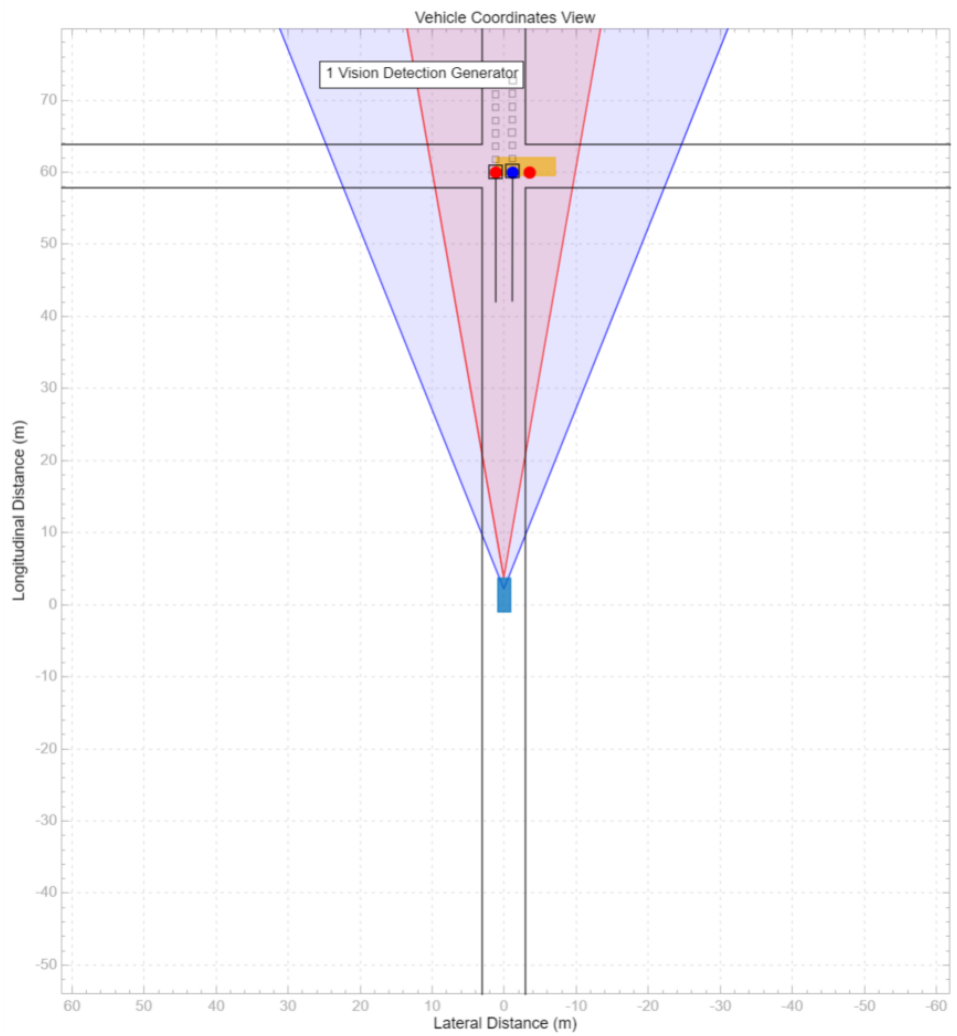


Figura 4.5: Veicolo centro corsia

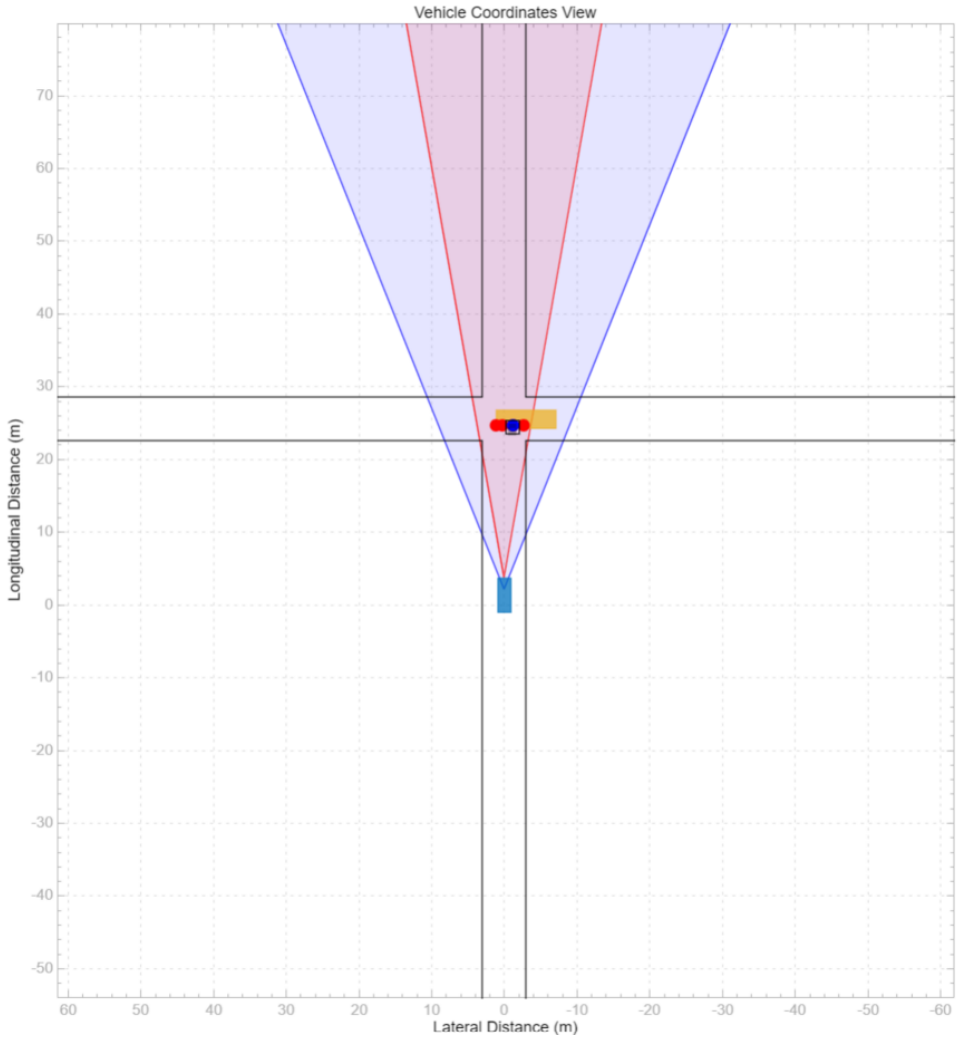
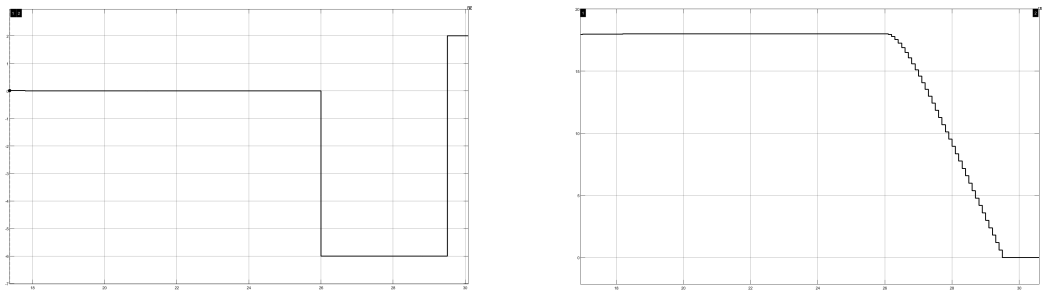


Figura 4.6: Veicolo fermo



(a) (b)

Figura 4.7: Dinamica del veicolo. Accelerazione(a) e Velocità(b)

Il veicolo resta fermo fin quando rileva l'ostruzione presente sulla corsia. Nel momento in cui il sensore non rileva più l'ostacolo la vettura riparte con *Cruise Control*. (*Figura 4.8*)

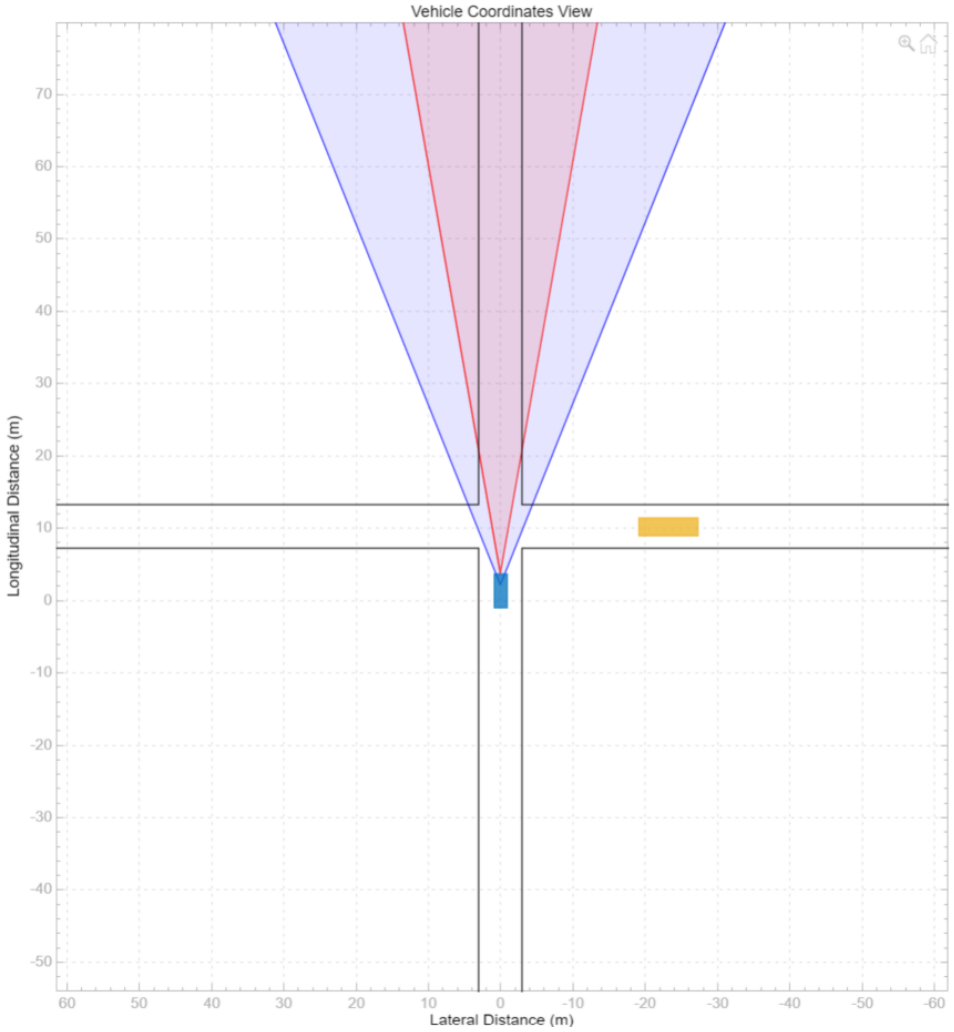


Figura 4.8: Strada sgombera

Successivamente si è implementato un attraversamento pedonale in cui più pedoni attraversano la strada.(*Figura 4.9*)

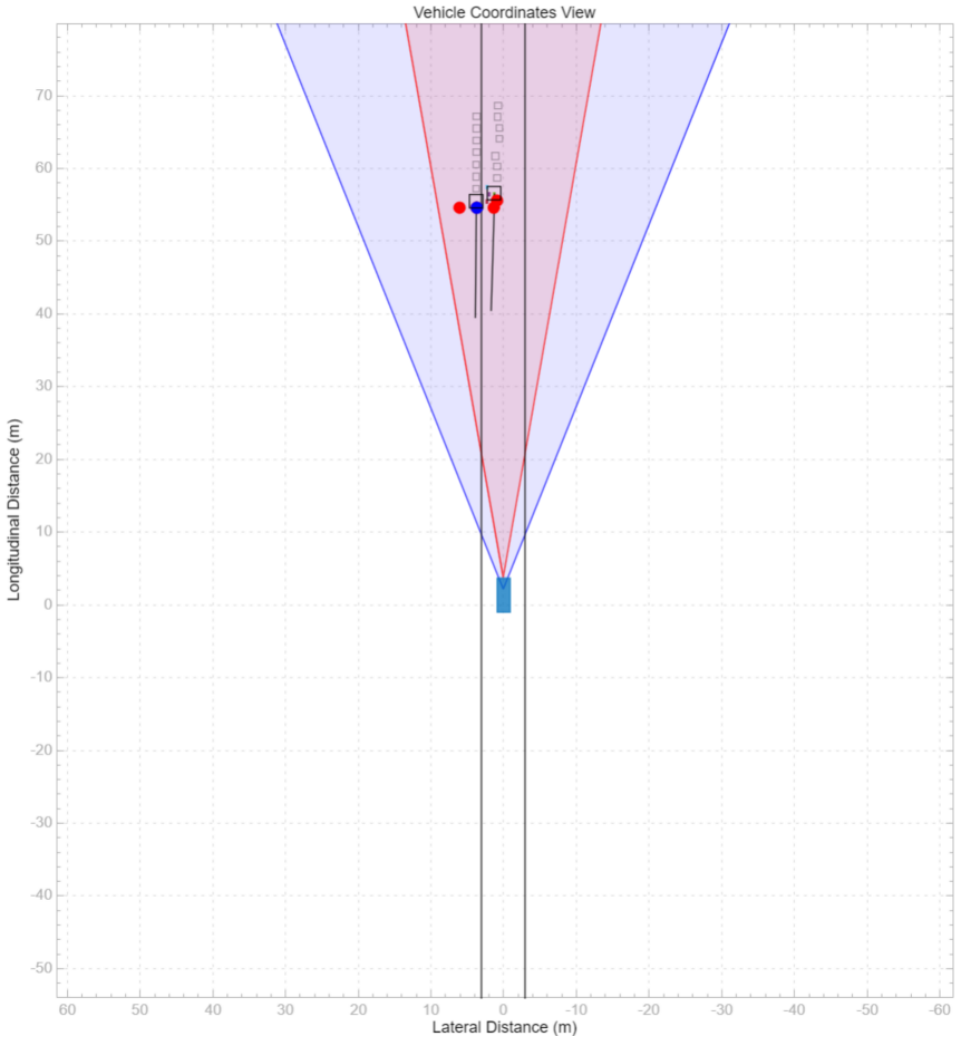


Figura 4.9: Attraversamento Pedonale

L'auto, come nel caso visto precedentemente, attiverà un *Full Braking* per evitare la collisione con essi. (*Figura 4.10, 4.11*)

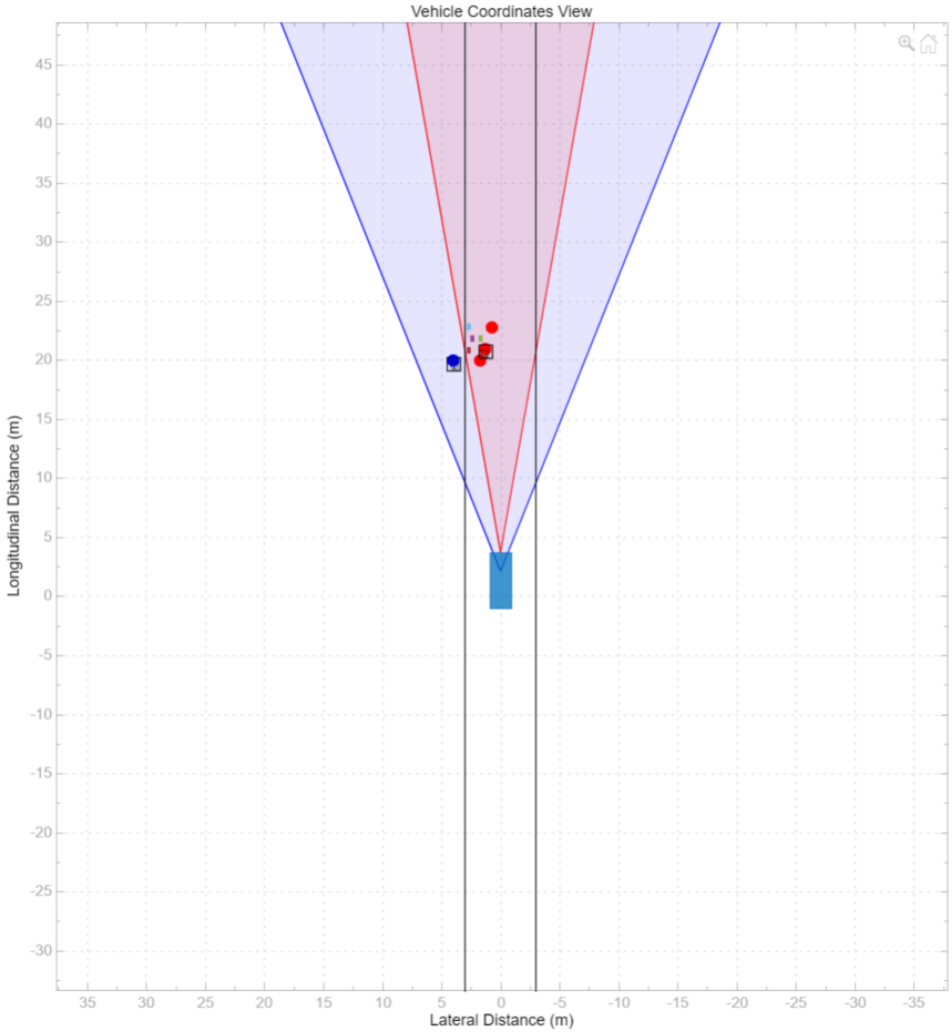
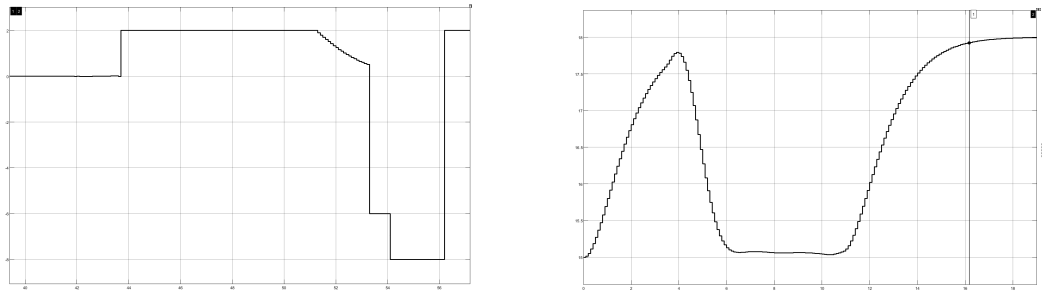


Figura 4.10: Dettaglio Attraversamento Pedonale



(a) (b)

Figura 4.11: Dinamica del veicolo. Accelerazione(a) e Velocità(b)

Soltanto dopo che l'ultimo pedone ha attraversato la strada ed è uscito fuori dal raggio di azione del radar, l'auto riprenderà il suo cammino.
Ultimo caso è l'immissione da una strada laterale di una bicicletta. (*Figura 4.12*)

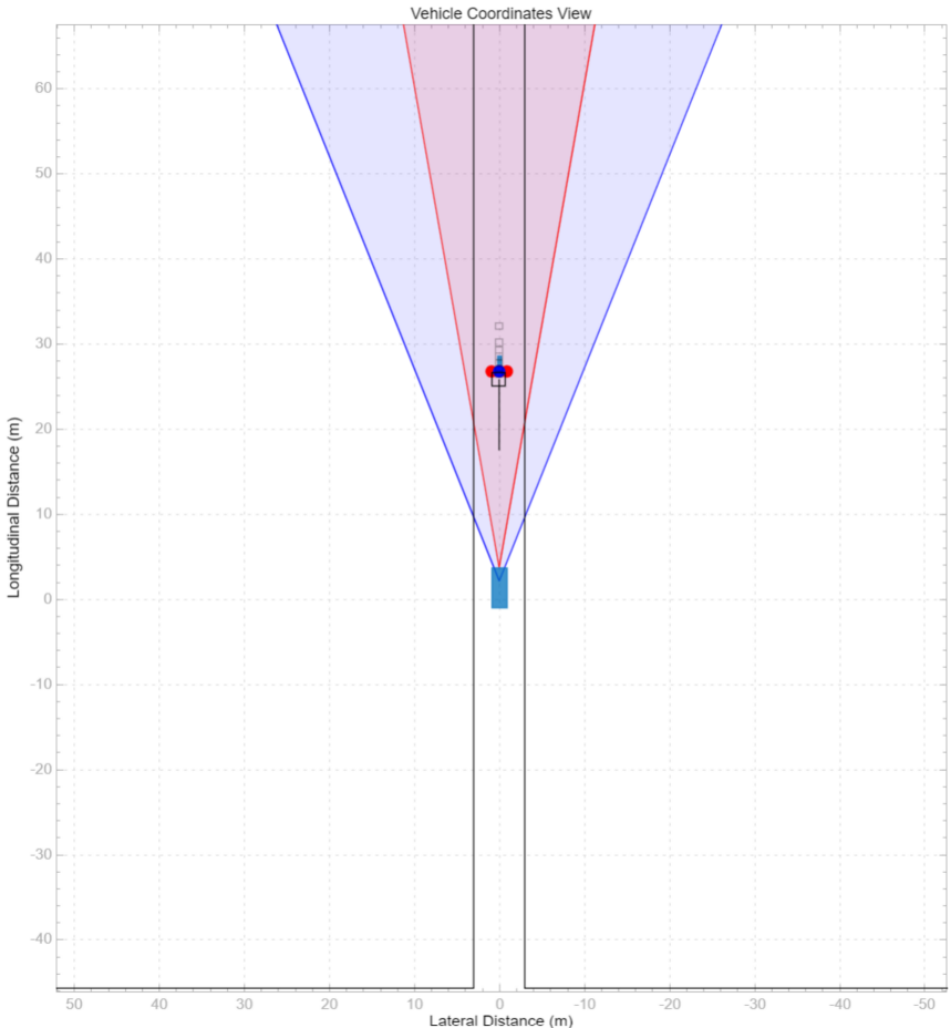
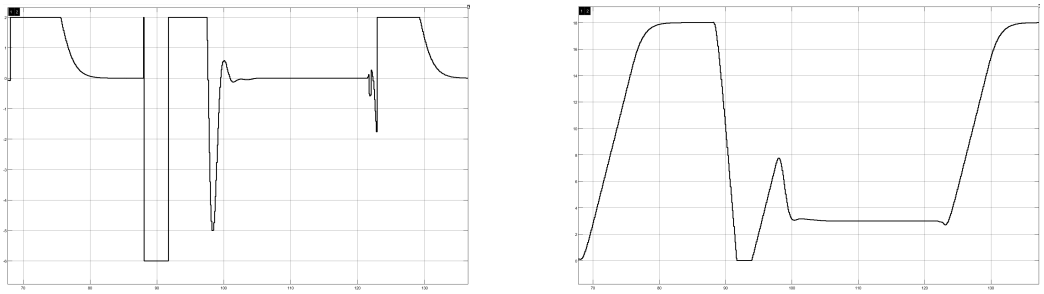


Figura 4.12: Bicicletta centro corsia

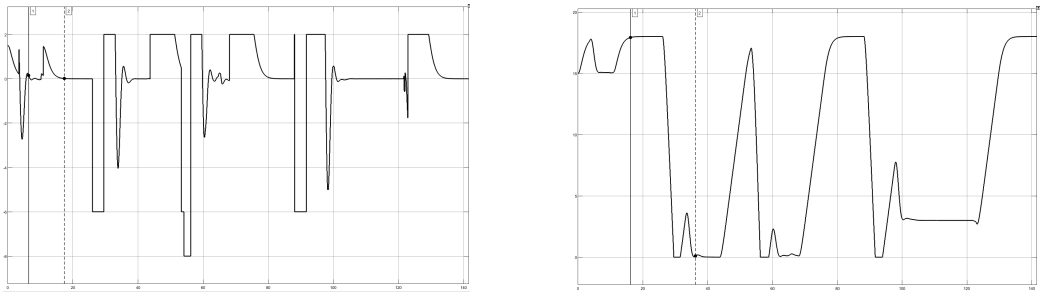
L'auto, che stava mantenendo una velocità di crociera, rilevando la bicicletta come oggetto presente sulla strada attiva l'*AEB*, ma la velocità di quest'ultima è talmente bassa (5 m/s) che l'auto deve attivare un *Full Braking*. Una volta evitato lo scontro l'auto riprenderà a muoversi attraverso *Adaptive Cruise Control* avendo come riferimento la bicicletta e, successivamente, quando quest'ultima cambia strada con *Cruise Control*.(*Figura 4.13*)



(a) (b)

Figura 4.13: Dinamica del veicolo. Accelerazione(a) e Velocità(b)

Lo scenario sopra descritto implementa tutti gli ADAS visti in precedenza: si passa dal *Cruise Control* all'*Adaptive Cruise Control* ed infine all'*Autonomous Emergency Braking* ed ancora.
È possibile vedere nel grafico l'andamento della velocità e dell'accelerazione lungo tutto il percorso. (*Figura 4.14*)



(a) (b)

Figura 4.14: Dinamica del veicolo. Accelerazione(a) e Velocità(b)

Conclusioni

Si è iniziato implementando il *Cruise Control* ADAS: il controllore scelto è stato un controllore a retroazione in uscita con solo azione proporzionale, che riesce a soddisfare le specifiche. Successivamente si è andati a implementare l'*Adaptive Cruise Control* ADAS dove è stato impiegato un controllore ad azione feedback e feedforward per soddisfare le richieste. Infine, l'ultimo ADAS implementato è stato l'*Autonomous Emergency Braking*, testato sia in *Partial Braking* che in *Full Braking*.
L'unione degli ADAS sopra descritti crea il *Supervisor Controller*.
L'obiettivo di questa tesi è stato lo sviluppo e la simulazione di algoritmi di controllo per veicoli autonomi.
Il modello dinamico utilizzato per la progettazione è noto ed è derivato dal modello della bicicletta. Sebbene questa strategia di controllo abbia portato ad un'efficace implementazione degli ADAS, manca l'obiettivo del comfort dei passeggeri. Inoltre, il fatto che il modello utilizzato trascuri parte della dinamica del veicolo suggerisce che, nei sistemi reali, questa tecnologia può trovare applicazione su strade principalmente lineari e con auto che si muovono a velocità pressappoco costante.
Dal punto di vista pratico degli ambienti di simulazione, è stato esplorato e testato l'*Automated Driving Toolbox* fornito da Matlab.
Il Capitolo 4 è stato dedicato a questo strumento che permette di costruire scenari di guida, e che, tramite un'integrazione in Simulink consente di testare i sistemi di guida autonoma.
Il pregio del toolbox è la possibilità di accedere a informazioni di alto livello senza dover implementare algoritmi di visione e quindi concentrarsi sulla progettazione del controllore senza guardare troppo ad altri aspetti di un algoritmo completo per la guida autonoma. La principale debolezza, d'altro canto, è la mancanza di un ambiente urbano fotorealistico e pronto all'uso, che limita la possibilità di algoritmi.
In questa tesi si considerano esclusivamente scenari rettilinei; tuttavia, gli scenari testati possono essere estesi anche a strade non rettilinee per verificare la coerenza delle prestazioni del sistema. Inoltre, apportando esperimenti

su scenari più estremi si potranno avere numerosi progressi nell'ambito della guida autonoma, come ad esempio, testando scenari con soglia di *time gap* più piccola, per quanto riguarda l'attivazione dell'*AEB*, si renderà possibile analizzare le prestazioni del sistema in condizioni più critiche.

In definitiva, il *Supervisor Controller* creato è in grado di prendere decisioni lungo lo scenario di traffico stradale e di adattare il movimento dell'auto in base all'ambiente circostante, come si evince dai grafici esposti. Il sistema di controllo si può, dunque, ritenere affidabile in questo caso di studio.

Bibliografia

[1] SAE Standards News: J3016 automated-driving graphic update. https://www.sae.org/news/2019/01/sae-updates-j3016-automated-driving-graphic.

[2] Iain Knight Alix Edwards Matthew Avery Hulshof, Wesley and Colin Grover. Autonomous emergency braking test results. *Paper Number 13-0168*, 2013.

[3] Yeong D.J. Sensor Technology in Autonomous Vehicles. Encyclopedia. https://encyclopedia.pub/entry/8356.

[4] Marcar. https://gruppomarcar.it/adas-tutto-quello-che-ce-da-sapere/.

[5] MATLAB. Automated driving toolbox™ getting started guide, 2019.

[6] MATLAB. Automated driving toolbox™ user's guide, 2019.

[7] MATLAB. Model predictive control™ user guide, 2018, 2021.

[8] MATLAB. *https://it.mathworks.com/help/driving/ug/autonomous-emergency-braking-with-sensor-fusion.html?searchHighlight=aeb&s_tid=srchtitle_aeb_2*. The MathWorks Inc., Natick, Massachusetts, 2021.

[9] Levels of Driving Automation. https://www.sae.org/news/2019/01/sae-updates-j3016-automated-driving-graphic.

[10] S. Shammah S. Shalev-Shwartz and A. Shashua. Safe multi-agent reinforcement learning for autonomous driving. *Paper Number 13-0168*, 2013.

[11] Raj Uppala. Pid controller for autonomous vehicles, 2017.

[12] Wikipedia. Attuatore, 2011.

[13] Wikipedia. Hardware-in-the-loop, 2011.

[14] Wikipedia. Jersey barrier, 2011.

[15] Wikipedia. Model predictive control, 2011.

[16] Sun Yao. A study of an lqr drifting controller for rc cars, 2018.