UNIVERSITÀ DEGLI STUDI
DI SALERNO

Dipartimento di

Ingegneria dell'informazione ed Elettrica e Matematica applicata

"AUTONOMOUS VEHICLE DRIVING"

anno 2020/2021

Docenti:

Alessia Saggese

Antonio Greco

Team:

Amorosi Sara	0622701389
Cerino Mario	0622701191
Palma Arcangelo	0622701208
Vitiello Alberto	0622701186

Indice

INTRODUZIONE

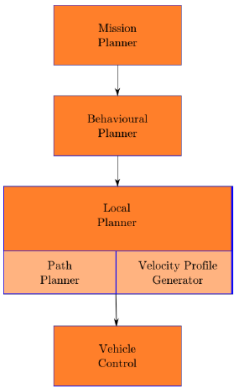
Lo scopo di questo progetto era la realizzazione dei moduli necessari per permettere il funzionamento di un veicolo a guida autonoma, per fare ciò è stato utilizzato un simulatore open-source, CARLA.
Infatti, CARLA è stato sviluppato proprio per supportare lo sviluppo e l'addestramento dei sistemi di guida autonoma. Aspetto molto importante di questo software è che fornisce una rappresentazione della vita reale a tutto tondo, mettendo a disposizione delle mappe rappresentanti città in cui compaiono edifici, veicoli e pedoni.
Grazie a questo sono stati effettuati vari test per provare quante più configurazioni possibili garantendo prima di tutto la *safety* e considerando questo progetto come se dovesse essere utilizzato nei casi di vita reale, infatti l'obiettivo è stato quello di mettere in sicurezza il veicolo stesso, la persona all'interno e le persone e gli oggetti al di fuori dello stesso.
Per implementare questo progetto è stato necessario progettare un Behavioural planner che permettesse al veicolo di navigare in maniera autonoma all'interno della mappa *Town01*.
Nella progettazione un aspetto importante è stata la gestione del comportamento del veicolo in presenza di un semaforo e in presenza di veicoli e pedoni all'interno della scena per evitare collisioni con questi.
L'approccio alla base di questo processo è stata la replicazione, in un certo senso, dei comportamenti che un guidatore esperto assumerebbe in casi di vita reale, come, ad esempio: attraversamento della strada da parte di un pedone; mantenimento della distanza di sicurezza dal veicolo che precede; rallentamento in prossimità di un incrocio per garantire maggiore sicurezza.

Per garantire una generale disciplina per la circolazione stradale del nostro veicolo, è stato imposto un "codice della strada" che riguarda queste regole:

- Se un pedone mostra l'intento di attraversare la carreggiata è necessario che il veicolo si fermi per permettere al pedone di attraversare in totale sicurezza. Il veicolo può ripartire solo quando il pedone ha completato l'attraversamento dell'intera carreggiata, come enunciato anche nell'art. 191 (*Comportamento dei conducenti nei confronti dei pedoni*) del codice stradale.
- Se il colore del semaforo è giallo il veicolo si fermerà comunque in prossimità dello stesso.
- Al veicolo non è consentito sorpassare.

Per il completamento di questo progetto sono stati affrontati problemi di vario tipo come, ad esempio, la gestione delle collisioni con i pedoni che attraversano la strada e si trovano ad una certa distanza per cui ostruiscono il passaggio del veicolo stesso, per questo tipo di circostanze sono state valutate varie possibilità e si è scelto di far fermare sempre il veicolo quando dinanzi vi è un pedone per permettere allo stesso di attraversare in sicurezza.
Altro problema è stato far fermare il veicolo nei punti desiderati rispettando una certa distanza con gli oggetti con cui potrebbe entrare in collisione. Quanto appena detto è stato risolto aggiungendo nuovi waypoints per cercare di evitare frenate brusche e collisioni, tuttavia, in casi particolari come quando il pedone attraversa improvvisamente la strada ad una distanza molto piccola dalla macchina (non risulta possibile far fermare il veicolo al waypoint desiderato), viene effettuata una frenata di emergenza.

PROGETTAZIONE

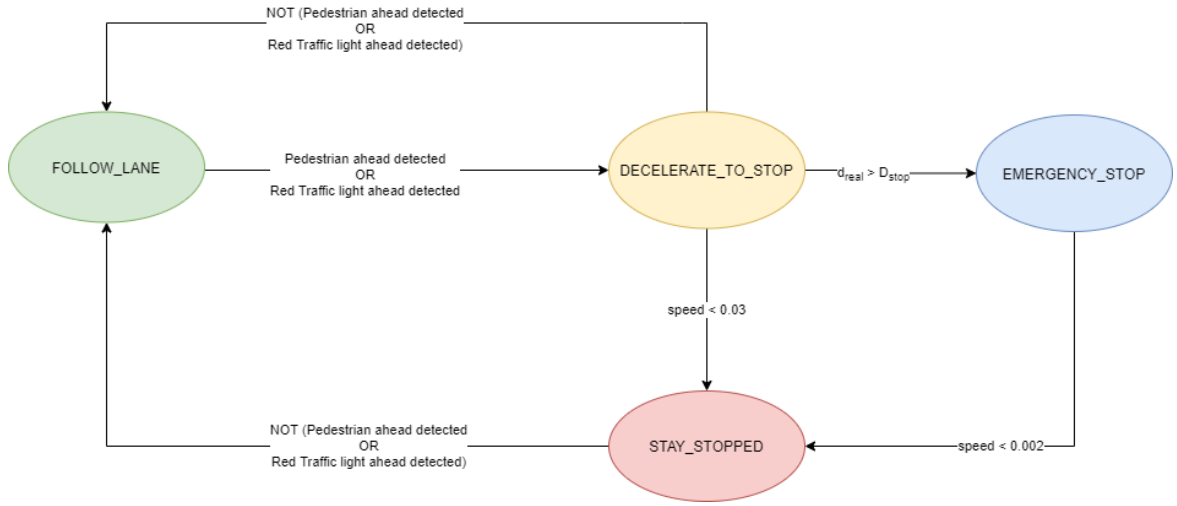


La relazione tra i moduli software utilizzati per questo progetto è di tipo gerarchico.
Il Mission Planner rappresenta il livello di astrazione più elevata di cui non ci siamo occupati durante lo sviluppo di questo progetto.
Il Behavioural Planner è stato fondamentale per l'implementazione dal momento in cui attraverso questo modulo si valuta e si decide quale specifico comportamento assumere sulla base dell'analisi statica e dinamica del mondo, quindi eventuali altri agenti che si muovono con il veicolo all'interno di esso.
Altro modulo utilizzato è stato il Local Planner grazie al quale, sulla base della manovra che il modulo precedente ha impartito, viene calcolata una traiettoria fattibile dal punto di vista fisico.

Behavioural Planner

L'Operational Design Domain (ODD) è un requisito fondamentale perché determina tutte le condizioni operative per cui stiamo progettando il veicolo e per cui questo deve poter lavorare in autonomia, infatti include le condizioni ambientali, il momento della giornata, il tipo di strada e altre caratteristiche per cui il veicolo deve agire in modo affidabile.
Il nostro ODD prevede che il veicolo percorrerà un unico tipo di strada cittadina (*Town01*), il momento della giornata è sempre lo stesso, quindi durante le ore diurne, le condizioni metereologiche non cambieranno e saranno di tipo soleggiato (*Clearnoon*).

Immaginando uno scenario realistico, quindi un ambiente in cui questa applicazione potrebbe essere usata, è stato pensato di gestirla trattando il sistema come un automa, il quale assumerebbe questi stati:



Il sistema di cui sopra si compone di quattro stati e ognuno di questi è stato pensato nell'ambito di circostanze reali in modo da cercare di far assumere al veicolo comportamenti simili a quelli di un guidatore esperto.

Stati

Di seguito vengono descritti nel dettaglio gli stati presenti nella nostra FSM.

Follow Lane

Considerando che la struttura della strada è definita da una linea separatoria che la divide questo stato riporta l'operazione più semplice che consiste nel "mantenimento della strada", cioè riuscire a mantenere sempre la propria carreggiata.

Dunque, come in una comune situazione reale il veicolo si troverà sempre nello stato di *Follow Lane*. Sulla base degli ostacoli presenti sulla strada e sull'azione opportuna da compiere verrà cambiato lo stato.

Ad ogni iterazione viene controllata la presenza di pedoni che attraversano la corsia e semafori di colore rosso, se presenti viene calcolata la distanza desiderata a cui il veicolo dovrà fermarsi e la distanza effettiva a cui il veicolo si fermerà applicando il massimo della decelerazione ($2.5 m/s^2$).

Nel caso in cui vi è un pedone si aggiunge un waypoint alla distanza calcolata, si memorizza il comportamento che si sta assumendo (mi sto fermando per un pedone) e l'indice del pedone in questione. A questo punto viene settato il *goal_index* al waypoint che è stato appena aggiunto e si passa allo stato *DECELERATE_TO_STOP*.

Nel caso in cui viene trovato un semaforo di colore rosso, se la distanza desiderata è maggiore rispetto ad una distanza massima (7m) e la distanza effettiva è minore rispetto alla distanza desiderata, allora il veicolo sta andando troppo lentamente per potersi fermare nel punto desiderato. In caso contrario, viene aggiunto un waypoint alla distanza desiderata e viene settato il *goal_index* al waypoint che è stato appena aggiunto e si passa allo stato *DECELERATE_TO_STOP*.

Decelerate to stop

Aspetto importante durante la guida è saper decelerare, infatti ciascun segnale potrebbe richiedere una specifica manovra, quindi bisogna riuscire a fare la giusta manovra sulla base del segnale che abbiamo dinanzi.

Come detto nel paragrafo precedente alla presenza di un semaforo il veicolo deve riuscire a decelerare, per poi fermarsi in presenza del colore rosso e continuare, riportandosi nello stato di *FOLLOW_LANE*, in presenza del colore verde.

Se il veicolo sta decelerando per la presenza di un semaforo rosso può accedere che, al contempo, sia comparso un altro pedone più vicino al veicolo rispetto al semaforo. In questo caso viene aggiunto un ulteriore waypoint tra il veicolo e il pedone, viene settato l'indice del pedone e non viene cambiato lo stato ma si resta in *DECELERATE_TO_STOP*.

Se, invece, il veicolo si trova molto vicino al semaforo nel momento in cui quest'ultimo cambia colore passando al rosso e la distanza effettiva è maggiore rispetto a quella desiderata si passa allo stato di *EMERGENCY_STOP*.

Altro motivo per cui il veicolo deve decelerare è la presenza di un pedone lungo la propria corsia.

In questo stato si conosce il motivo per cui il veicolo sta decelerando, quindi si può intraprendere un comportamento opportuno.

Se il veicolo sta decelerando per la presenza di un pedone può accadere che, al contempo, sia comparso un altro pedone più vicino al veicolo, oppure che la distanza dal pedone rispetto al veicolo sia diminuita (il pedone si sta dirigendo verso il veicolo), oppure che la distanza rimanga invariata e che quindi il pedone si sta spostando in modo perpendicolare rispetto al veicolo.

Nel primo caso viene aggiunto un ulteriore waypoint tra il veicolo e il nuovo pedone, viene aggiornato l'indice del pedone e non viene cambiato lo stato ma si resta in *DECELERATE_TO_STOP*.

Nei casi rimanenti si confronta la distanza effettiva con quella desiderata e se il veicolo non riesce a fermarsi in tempo si passa allo stato di *EMERGENCY_STOP*.

Nel caso in cui, invece, non fosse più presente alcun pedone si passa allo stato di *FOLLOW_LANE*.

Inoltre, se la velocità del veicolo è minore rispetto a un valore di soglia *stop_threshold* (0.03) si passa allo stato *STAY_STOPPED*.

Stay stopped

In determinate circostanze il veicolo deve essere in grado di fermarsi per una certa quantità di tempo o fin quando non si verifica una particolare situazione.
Questo stato rappresenta una situazione di stallo del veicolo.
A seconda della condizione per cui si è entrati in questo stato, se quest'ultima non è più verificata si passa allo stato di *FOLLOW_LANE*.

Emergency stop

Fin quando il veicolo si trova in questo stato viene settato *brake* al valore massimo (1). Si resta in questo stato finché la velocità del veicolo non scende oltre una certa soglia, *emergency_stop_threshold* (0.002). Questa soglia è inferiore a *stop_threshold* (0.03) perché quando il veicolo si trova in *EMERGENCY_STOP* si ha la necessità che sia completamente fermo prima di rilasciare il freno.
Superato questo valore di soglia, *emergency_stop_threshold*, si passa allo stato di *STAY_STOPPED*.

Scelte progettuali

Camera

Una delle scelte progettuali che sono state affrontate durante questo progetto è stata la valutazione del numero delle camere da posizionare sull'auto per avere una visione della strada. In particolare, è stato scelto di utilizzare due camere: Camera centrale e Camera Right. Nella foto sottostante viene riportato quanto appena detto:

Figura 1: Progettazione della macchina a guida autonoma

Nelle immagini che seguiranno, invece, verranno mostrati degli esempi dell'utilizzo su strada delle due camere, quindi si può notare che la Camera Right è molto importante principalmente quando bisogna vedere il semaforo ad una certa distanza mentre la Camera centrale è utilizzata per vedere il semaforo quando il veicolo si trova molto vicino a esso.
Tra queste due camere vige il principio della mutua esclusione, quindi se una delle due camere individua il semaforo non viene considerata anche l'altra camera.

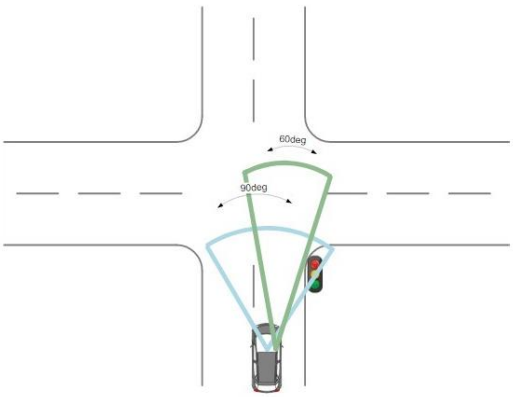


Figura 2: Utilizzo camera centrale

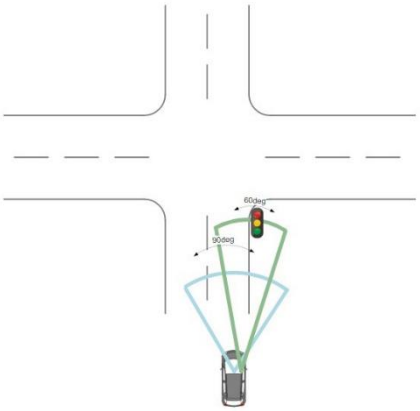


Figura 3: Utilizzo camera right

Curve

Per migliorare il comportamento del veicolo in curva è stato pensato di diminuire il *lookahead* ponendolo pari a 16. Diversamente da una strada dritta, in curva è necessario che il veicolo prenda come goal_index i waypoints più vicini per evitare di allargarsi troppo in curva.
Per quanto riguarda i pedoni, durante una curva, il *lookahead* viene settato a 8 per permettere al veicolo di vedere solo i pedoni presenti lungo la traiettoria che sta percorrendo, non considerando, quindi, i pedoni presenti dinanzi ad esso ma lungo una traiettoria diversa. Nell'immagine sottostante viene mostrato un esempio di quanto appena detto.

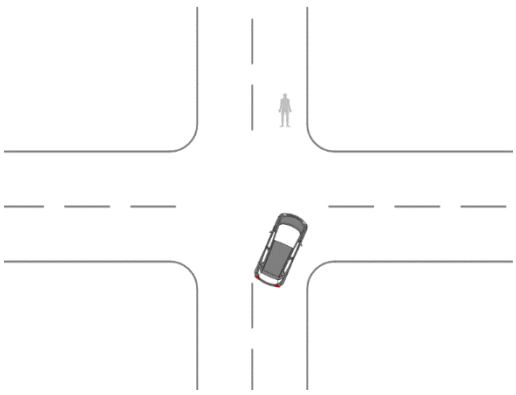


Figura 4: Lookahead for pedestrian

Per gli stessi motivi che hanno portato alla scelta di un *lookahead* minore rispetto al normale per i pedoni, per i veicoli viene diminuito ancora e settato a 6.

Strada

Per la gestione dei pedoni si è deciso di calcolare la larghezza della carreggiata che in Carla risulta essere circa 8m ma per una maggiore sicurezza è stato aggiunto un padding e messa a 10m.

Implementazione

In questo capitolo viene descritta nel dettaglio l'implementazione del codice e viene spiegato il perché di alcune scelte. Come prima cosa vengono descritte le scelte effettuate per l'implementazione del detector e tutte le assunzioni che sono state fatte per migliorare le performance dello stesso. In seguito, vengono descritte nel dettaglio ciò che è stato implementato.

Detector

Per l'individuazione dei semafori è stato utilizzato il modulo Traffic Light Detector fornitoci a lezione.

A intervalli regolari di due frame, viene applicato il detector sull'immagine fornita dalla camera di Carla e viene individuato un punto che rappresenta la posizione sulla mappa del semaforo. Sulla base di questo primo punto individuato viene costruito un cluster, il quale viene rappresentato come un cerchio con raggio di 5 metri. Ogni individuazione che avverrà successivamente sarà inserita all'interno del cluster più vicino se si trova nell'intorno di 5 metri da esso, altrimenti verrà costruito un nuovo cluster.

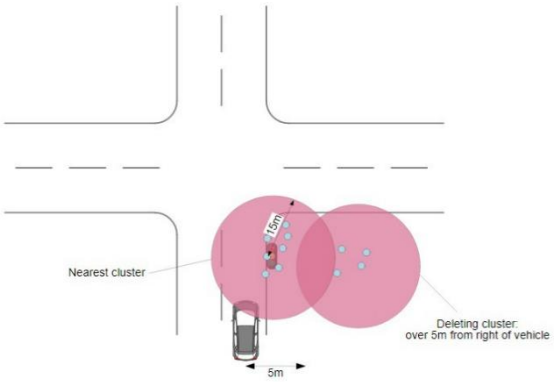


Figura 5: Cluster measurement

Dopo aver identificato il cluster con maggiori punti e quindi, dopo aver individuato la posizione del semaforo sulla mappa è stato necessario capire se il semaforo in questione si trovasse nel senso di

percorrenza del veicolo per poter poi decidere, sulla base del colore, il comportamento che bisogna assumere. Dunque, al momento in cui un semaforo viene analizzato e quindi il veicolo ha svolto il proprio compito questo cluster dovrà essere eliminato per poterne poi costruire uno nuovo alla prossima intersezione (se presente il semaforo).

Dal momento in cui i semafori presenti possono essere più di uno e soprattutto possono trovarsi ad entrambi i lati della strada, quindi sia a destra che a sinistra, sono state filtrate queste misurazioni seguendo alcune euristiche. Per inserire il punto nel cluster questo prima di tutto deve trovarsi nel nostro senso di percorrenza, deve trovarsi prima della prossima intersezione e in un intorno di 20 metri rispetto a quest'ultima.

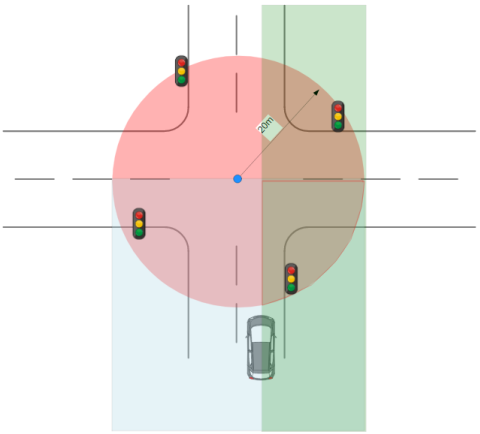


Figura 6: Filter measurement

Durante l'implementazione, nonostante l'utilizzo delle immagini *Epic* con le quali è stato addestrato il detector, sono stati ricontrati vari problemi. Il primo problema affrontato è stato l'imprecisione nella "cattura" del semaforo quando questo si trova a più di una certa distanza dal veicolo, in questo caso, dal momento in cui spesso i bounding box rappresentativi del semaforo potevano non contenerlo si è deciso di estenderli in entrambe le direzioni di 50px, ma, per poter avere una posizione corretta del semaforo era necessario individuare almeno un punto su questo. Questo tipo di problema è stato affrontato confrontando il bounding box con la semantic segmentation e prendendo la maschera del semaforo, dopodiché è stato calcolato il centro di massa della forma catturata dal bounding box (il motivo di questo è che il semaforo sarebbe potuto anche non essere ripreso interamente nel box) per poi essere trasformato in coordinate nel mondo ed essere localizzato all'interno della mappa. Questo tipo di soluzione ha consentito di aumentare notevolmente la precisione anche perché i bounding box che non contenevano il semaforo sono stati scartati, evitando misurazioni errate nella trasposizione del punto nel mondo.

Di seguito viene mostrata un'immagine in cui si può notare la differenza tra la prima e la seconda foto in termini di larghezza del bounding box, mentre nella terza foto si mostra come vengono applicati alla semantic segmentation i bounding box allargati.

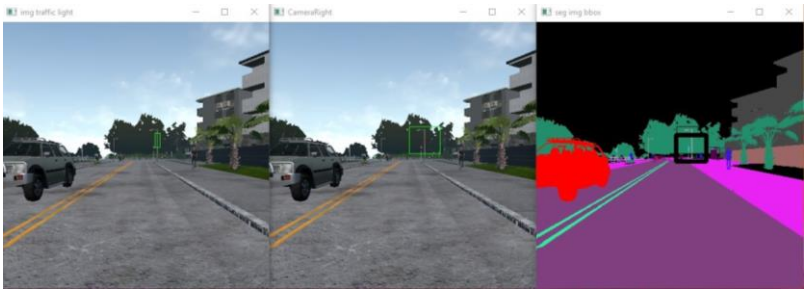


Figura 7: Estensione del bounding box

Quando il semaforo è verde sono stati riscontrati errori di misdetection se la distanza del veicolo da esso risulta essere maggiore di 7 metri, pertanto si è deciso, nel caso di semaforo rosso, di far avvicinare quanto più possibile la macchina al semaforo prima di fermarsi, in modo tale che il detector possa continuare a rilevare il semaforo anche quando vi è un cambiamento di colore dal rosso al verde, tuttavia è importante che il veicolo non si avvicini a più di tre metri dal semaforo altrimenti quest'ultimo uscirebbe dalla vista e non saremo più in grado di capirne il colore. Per stabilire la posizione in cui il veicolo deve fermarsi prima del semaforo è stato introdotto un nuovo *waypoint* in modo che il semaforo si trovi tra i 4 e i 6 metri di distanza dal veicolo.

Il colore del semaforo è stato verificato tramite un metodo di computer vision.

Si è deciso di utilizzare la tecnica di detection sullo spazio di colore HSV dal momento in cui il nostro ODD prevede condizioni di illuminazione e metereologiche stabili.

Questo approccio ci ha permesso di migliorare i risultati ottenuti dal detector e avere prestazioni migliori, consentendoci di non avere detection sbagliate per quanto riguarda il colore del semaforo soprattutto in caso di colore giallo.

Per migliorare ulteriormente la detection del semaforo e far in modo che il veicolo riesca sempre o almeno nella maggior parte dei casi a fermarsi sono state utilizzate due camere: Camera Right e Camera RGB. È stata fatta questa scelta perché la Camera Right, la quale è utile soprattutto per individuare il semaforo ad una certa distanza, avendo una focale più stretta non riesce a carpire il semaforo quando il veicolo si trova troppo vicino a questo. In questi casi è utile la Camera RGB che, avendo una focale maggiore, riesce ad individuare il semaforo anche in circostanze un po' più "estreme". Dal momento in cui i semafori da considerare si trovano alla destra del veicolo, si è scelto di spostare la Camera Right di 50cm a destra rispetto alla Camera RGB.

Altro problema riscontrato è che nonostante il filtraggio mediante la semantic-segmentation, non venivano distinti i segnali stradali dai semafori dal momento in cui il detector resta sempre attivo durante tutto il percorso, quindi grazie alle euristiche di cui sopra è stato possibile filtrare gli altri

segnali basandoci anche sulla loro distanza dalle intersezioni e permettendoci di ottenere la reale condizione del semaforo.

Dato quanto appena detto viene mostrato il flusso di tutte le operazioni che vengono effettuate per consentire il corretto funzionamento del semaforo.

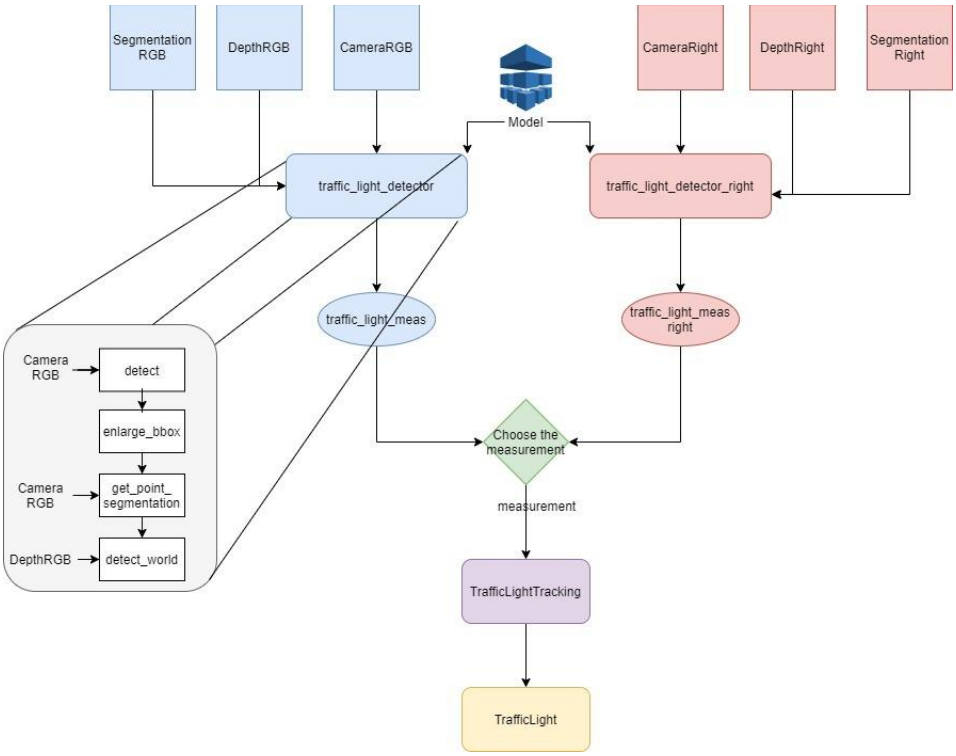


Figura 8: Flusso di localizzazione del semaforo

Di seguito viene riportato l'UML delle classi per quanto riguarda la parte dedicata al semaforo:

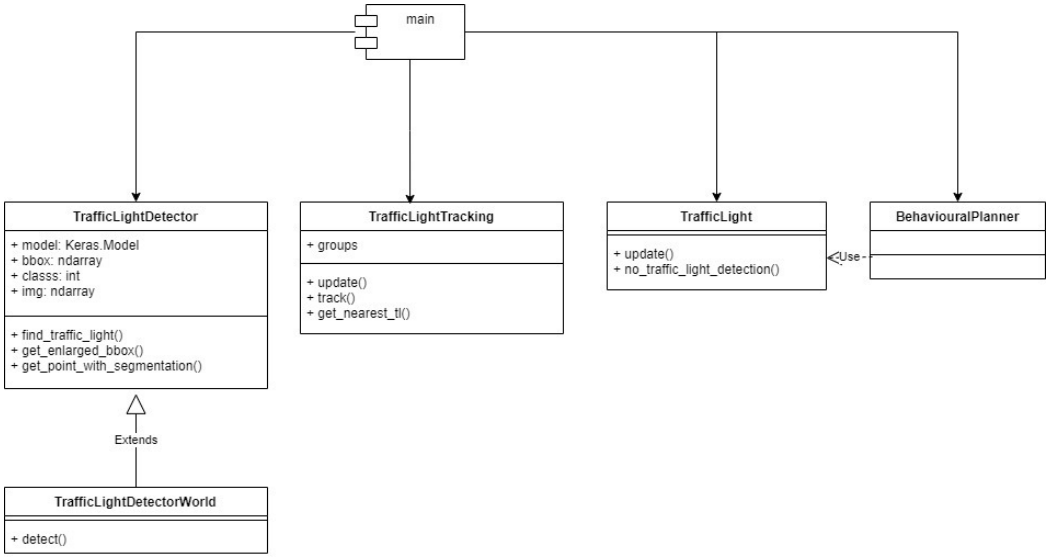


Figura 9: UML

Funzionalità aggiuntive

Di seguito vengono descritte nel dettaglio tutte le funzionalità aggiuntive a quelle della baseline che sono state implementate. Nella realizzazione del sistema sono stati affrontati i problemi riguardanti le collisioni con veicoli e pedoni presenti sulla strada.

Pedoni

Dal momento in cui durante la simulazione i pedoni attraversano la strada improvvisamente, il veicolo deve essere pronto a reagire.

Per ovviare a ciò, vengono considerate le posizioni di tutti i pedoni in un determinato range rettangolare intorno al veicolo. Viene fissato un punto che si trova alla base del bounding box del pedone, il quale viene poi trasformato in coordinate immagine in riferimento alla camera presente sul veicolo. Tale punto in coordinate immagine viene confrontato con i bordi della strada trovati tramite la *Lane Detection*; per tali linee sono stati calcolati i coefficienti angolari e le intercette con l'asse y, infatti, utilizzando tali parametri è stato possibile verificare se il pedone fosse o meno sulla carreggiata. Sono stati effettuati anche altri controlli per verificare se la direzione del pedone fosse di ostacolo al proseguimento del veicolo lungo la traiettoria.

Nei frame successivi, all'entrata del pedone in carreggiata, non vengono più confrontati i risultati della *Lane Detection* con il punto del pedone che è stato precedentemente individuato, bensì si controlla se questo punto risiede in un intorno destro o sinistro del veicolo (distanza dal marciapiede sinistro e marciapiede destro).

Queste due distanze vengono calcolate scartando le lane sulla base del coefficiente angolare e prendendo solo quelle che potrebbero effettivamente essere del marciapiede destro o sinistro (dal momento in cui queste si trovano sempre in un range fissato).
Dal momento in cui solitamente si ha la lane destra (dato che la vista della camera non è ostruita da altri veicoli presenti sulla carreggiata) per calcolare quella sinistra viene usata un'euristica sulla larghezza della carreggiata, quindi considerando che si è posta la larghezza della carreggiata pari a 8m la prima lane trovata (quindi quella che abbiamo) viene spostata di 1m e l'altra viene posta a 10m da essa, in modo da averle a 10m di distanza l'una dall'altra e avere una dimensione della carreggiata pari a 10m. Lo stesso ragionamento viene fatto quando si ha la lane sinistra e bisogna calcolare la destra.
Nel caso in cui dovessero mancare entrambe, viene usato un range rettangolare. Quindi, calcolata la posizione del pedone con cui potrebbe verificarsi la collisione, e data la velocità con cui si sta muovendo il veicolo si fa in modo che il veicolo riesca a non collidere con il pedone (frenando o decelerando semplicemente). In tal caso viene aggiunto un nuovo waypoint e modificato il *goal_state* in modo da potersi fermare a questo. Nel caso in cui una semplice decelerazione non basti per non generare una collisione, viene applicata una frenata brusca per far in modo che il veicolo si fermi.

Veicoli

Come prima cosa si è deciso seguire l'andamento del veicolo che precede il nostro senza superarlo.
Per fare ciò, è stato identificato il veicolo più vicino al nostro ma che ci precede considerando un range rettangolare (che può variare a seconda della presenza o meno di un incrocio).
Sono stati considerati solo i veicoli presenti in questo rettangolo e che abbiano una direzione contenuta in un range di 45° (destro e sinistro) rispetto a quella del nostro veicolo. Infine, per considerare tale veicolo come un *lead_vehicle*, ovvero un veicolo da seguire, si va a verificare se quest'ultimo si trovi all'interno del *lookahead* settato. Tutti gli altri veicoli, invece, vengono considerati come possibili veicoli con cui si può verificare una collisione, per questo motivo vengono scartati quelli più lontani di una certa distanza e vengono verificati i percorsi del local planner con i bounding boxes dei restanti veicoli.

Analisi sperimentale

Di seguito vengono riportate le analisi quantitative e qualitative dei test effettuati e dei risultati ottenuti.

Analisi quantitativa

L'applicativo è stato testato ripetutamente durante lo sviluppo in modo da poter avere una visione separata delle parti scritte e capire se quanto implementato facesse esattamente quello che era stato pensato e se fosse necessario considerare ulteriori varianti.
Dopo aver completato i test delle parti singolarmente sono stati effettuati i primi test più lunghi per capire se l'unione delle singole parti creasse ulteriori problemi. Questi ultimi sono stati effettuati in scenari complicati per poter testare i comportamenti del veicolo in tutte le possibili situazioni. Da questi test sono emersi piccoli problemi che sono stati risolti nell'immediato per poi poter passare ai test finali in condizioni meno "eccessive".
I problemi riscontrati durante questi test sono stati:

- Individuazione di *lead_car*.
- Collisioni con pedoni che stanno per attraversare la strada e non venivano individuati come oggetti con cui poteva avvenire una collisione
- Collisioni con altri veicoli in curva
- Individuazione sbagliata del semaforo quando si partiva da un *player_start_index* troppo vicino
- Arrivo in ritardo nello stato *decelerate_to_stop* con conseguente tamponamento del veicolo
- Cambi di direzione di un pedone fanno in modo che il veicolo resti fermo per diverso tempo (non risolto perché non è stato modificato il comportamento dei pedoni all'interno della mappa)

Per poter effettuare al meglio i test per quanto riguarda i semafori sono stati individuati i punti di start ed end sulla mappa che ci permettevano di attraversare almeno due o tre semafori e almeno un incrocio per cui bisognava svoltare a destra. Per poter individuare e poi risolvere il problema sopra descritto sono stati effettuati dei test in cui il veicolo si trovava abbastanza vicino al primo semaforo al momento della partenza.
Per poter effettuare test sui pedoni, invece, si è scelto di inserire all'interno della mappa da 100 a 300 pedoni in modo da poter considerare sia i casi più semplici che quelli più complessi. In questo modo

siamo riusciti a capire in quali condizioni il pedone veniva investito e in quali condizioni, invece, non veniva considerato come oggetto con cui sarebbe potuta avvenire una collisione.
Ulteriori test sono avvenuti per individuare le possibili collisioni con i veicoli, come descritto anche sopra. Anche in questo caso è stato tutto risolto.
Altri problemi non risolti riguardano pedoni che camminano incontro alla macchina ma al di fuori delle due camere, ad esempio camminando al centro del veicolo o totalmente dietro.
Di seguito viene riportato un grafico in cui sono stati inseriti i vari test effettuati prima di passare ai test completi, come si può notare il numero di test conseguiti in modo corretto è maggiore rispetto a quelli conseguiti con qualche errore, questo per garantire maggiore sicurezza prima di effettuare i test finali completi. L'unico test che non differisce da quanto detto è quello che riguarda il cammino della macchina in curva poiché non è stato modificato rispetto alla baseline che ci è stata fornita.

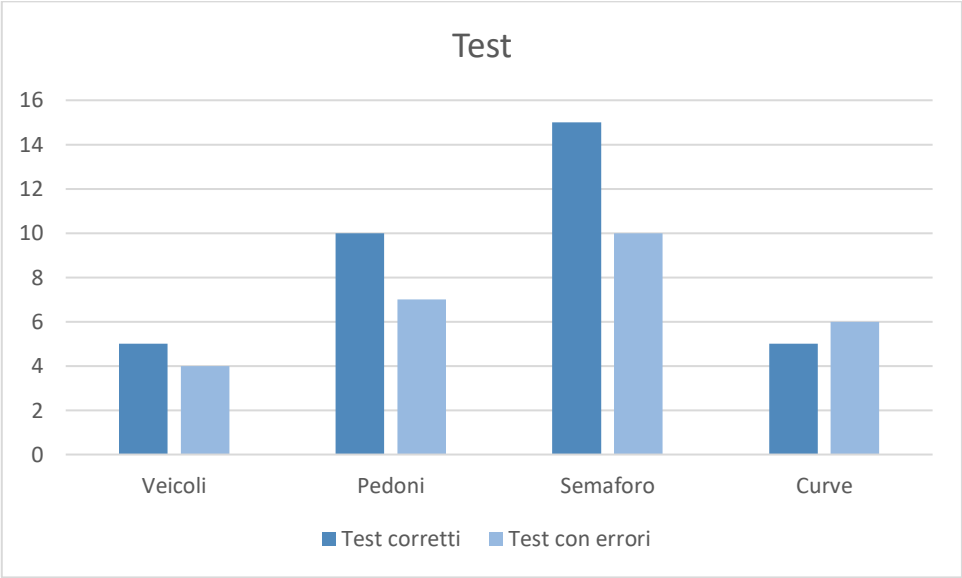


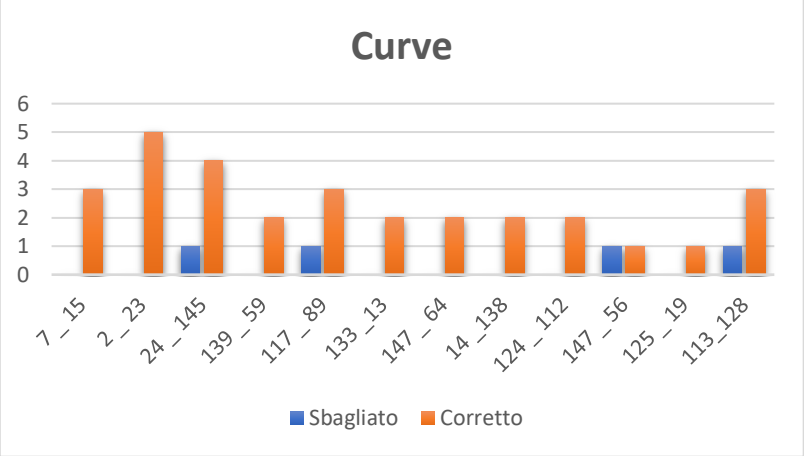
Figura 10: Test preliminari

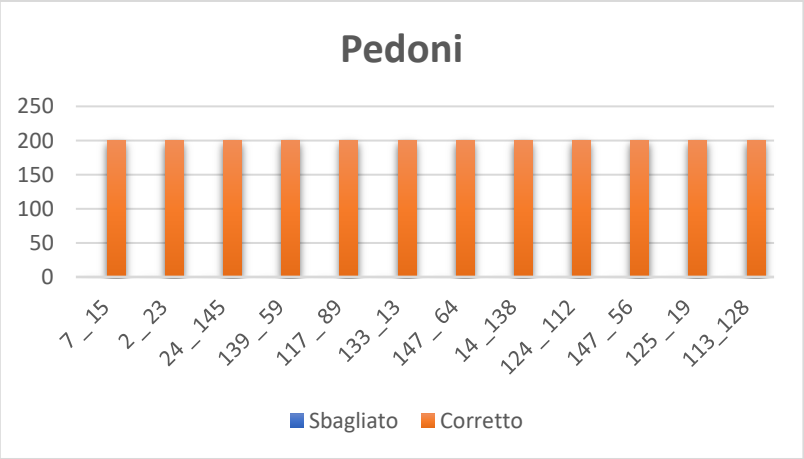
Per quanto concerne i test finali, invece, dopo aver risolto i vari problemi testando le singole parti in condizioni più "estreme", sono stati effettuati test completi sui percorsi scelti.

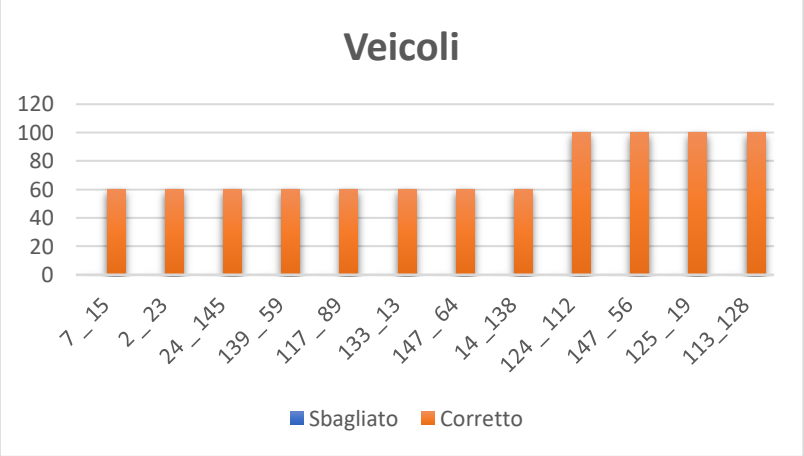
I test vengono considerati sbagliati o corretti in queste condizioni:

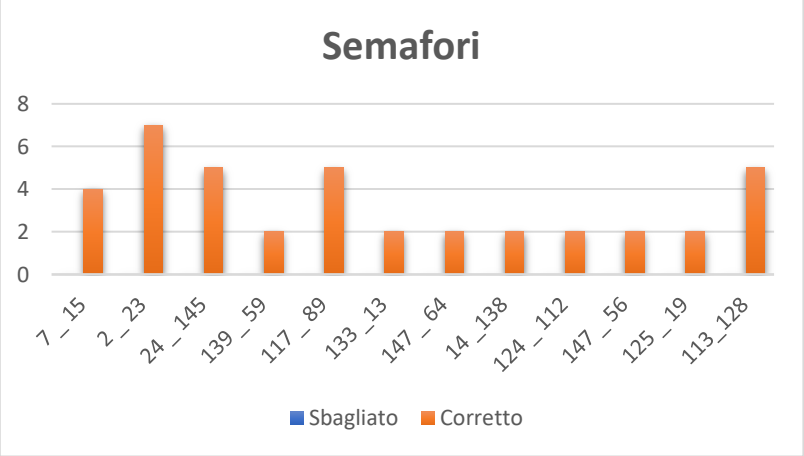
Test	Corretti	Sbagliati
Curve	Non ci sono stati errori	Il veicolo sbaglia il percorso ed entra in contatto con il guard rail.
Pedoni	Non ci sono state collisioni	Il veicolo investe un pedone
Veicoli	Non ci sono state collisioni	Il veicolo entra in collisione con un altro veicolo
Semafori	La posizione del semaforo e il colore dello stesso viene riconosciuto correttamente	Non viene riconosciuto il giusto colore del semaforo e pertanto il veicolo esegue un'azione sbagliata oppure non viene individuato il semaforo

Di seguito vengono riportati dei grafici, ognuno per ogni problema, che presentano sulla base dei test effettuati quanti errori si sono verificati.









Come si nota da questi grafici, gli unici problemi rimasti riguardano soltanto le curve poiché non sono state affrontate in modo approfondito durante lo sviluppo di questo progetto.

Analisi qualitativa

I problemi avuti con i pedoni sono stati due, uno riguarda l'aggiunta dei waypoints alla giusta distanza in modo tale da permettere alla macchina di fermarsi in tempo evitando una collisione e l'altro riguarda l'orientamento del pedone, quindi capire dove è rivolto e se si trova in una posizione tale da andare incontro al veicolo, inoltre sempre per questo problema è stata scelta una distanza di "azione" che ci permettesse di capire se il pedone stesse per passare dinanzi al veicolo.

Per quanto riguarda il semaforo i problemi avuti durante i test finali riguardavano principalmente la posizione in cui si fermava il veicolo, poiché se ad una distanza troppo piccola rispetto al semaforo nessuna delle due camere riusciva a carpire il semaforo.

Per i problemi avuti con i veicoli vale il ragionamento simile a quello affrontato con i pedoni, se il veicolo stava percorrendo la strada in curva con una velocità abbastanza elevata, ad esempio quando il semaforo durante l'intersezione era di colore verde, non riusciva a decelerare in tempo quando il nuovo veicolo entrava nel suo raggio di visione e quindi si verificava una collisione. Questa cosa è stata risolta mantenendo in curva una velocità sempre bassa.

Di seguito vengono riportati alcuni screenshot effettuati duranti i test intermedi che mostrano che i casi di errore affrontati negli scenari critici riportati nel capitolo precedente sono stati risolti.

Pedone

Il veicolo si ferma per la presenza di un pedone che attraversa la carreggiata:

Semaforo

Viene rilevato il colore verde del semaforo e il veicolo non si ferma continuando la propria traiettoria:

Viene rilevato il colore rosso del semaforo e il veicolo si ferma nel punto desiderato attendendo il passaggio del semaforo da colore rosso a verde:

In questa cartella di Google Drive, in cui è presente anche il progetto, sono stati inseriti anche tre video in cui in uno è presente un pedone per cui il veicolo è costretto a fermarsi, in un altro è presente il semaforo e viene mostrato come il veicolo si ferma al rosso e l'ultimo una situazione in cui sono presenti veicoli sia dinanzi al nostro sia nell'altra corsia.
https://drive.google.com/drive/folders/1F4qKnbsDxoPK4S3--Sj40T2DTcLdjQAG?usp=sharing

CONCLUSIONI

L’aspetto più importante per questo progetto è stato la *safety*, infatti si è cercato di considerare tutte le situazioni più critiche che potrebbero verificarsi e sono state sviluppate le funzioni adatte per poter mettere in sicurezza prima di tutto gli oggetti dinamici al di fuori del veicolo e anche il veicolo stesso. Inoltre, si è cercato di rendere la guida molto fluida evitando di far fermare il veicolo per ogni oggetto dinamico e facendogli assumere comportamenti diversi quando dinanzi sono presenti pedoni o veicoli. A valle di quanto detto il veicolo si fermerà soltanto quando necessario e quando si troverà in situazioni che possono essere definite critiche, fermandosi in modo tale da non creare danni.
In seguito, sarebbe opportuno procedere con il miglioramento di queste funzioni e con l’ampiamento delle stesse prendendo in considerazione altri casi che per motivi di tempistiche non sono stati considerati. Quindi si potrebbe pensare di rendere la guida all’interno di CARLA quanto più simile possibile a quella di un guidatore esperto sotto tutti i punti di vista.