UNIVERSITÀ DEGLI STUDI DI SALERNO

DIPARTIMENTO DI INFORMATICA

TESI DI LAUREA

IN

INFORMATICA

Intelligenza Artificiale per il rilevamento dei pedoni in ambiente simulato

Relatore:

Ch.mo. Prof. Andrea F. Abate

Correlatore:

Dott. Ignazio Passero

Candidato:

Ferrara Carmine

Matr.05121/05255

ANNO ACCADEMICO 2019/2020

RINGRAZIAMENTI

Giungere alla fine di un cammino complesso quale il compimento di uno ciclo di studi è sempre stato per me un'esperienza molto difficile da immaginare e da affrontare. Adesso mi ritrovo qui, tra le pagine di un documento che segna le battute finali di tre anni che presso l'Università degli Studi di Salerno mi hanno visto crescere sia sul piano personale che culturale.

Innanzi tutto, se si parla di ringraziamenti non posso far altro che rivolgere un pensiero al carissimo professore Gerardo Scarano, mio docente all'ultimo anno di scuola, il quale forse per primo mi ha dato la giusta carica per intraprendere il cammino universitario, se non fosse stato per lui probabilmente ad oggi non avrei nemmeno avuto questa possibilità.

Dopo di lui tanti sono stati i docenti che nell'arco dei tre anni, a loro modo anche indirettamente mi hanno sempre spronato a dare sempre il massimo, Come non pensare:

- Alla grande umanità del professore Deufemia, con cui ho forse percepito per la prima volta un modo nuovo di vedere l'informatica, a metà strada tra i primi formalismi e la passione di un programmatore;
- Alla grande tenacia del professore De Lucia e delle professoresse Rescigno e Anselmo, che ad ogni lezione hanno cercato in tutti i modi di aiutare noi studenti dei primi anni a formalizzare i nostri concetti in modo rigoroso, senza lasciare nulla al caso;
- Al grande impegno della professoressa Ferrucci dalla quale ho appreso l'importanza avere un approccio analitico e schematico dall'inizio alla fine di qualsiasi esperienza, sia essa lavorativa o meno;
- Alla grande solarità e dedizione del professore Carrabs, che durante l'ultimo semestre, se pur a distanza, mi ha fatto capire una volta per tutte, quanto l'applicazione di un buon formalismo matematico sia davvero utile per noi

giovani informatici, che troppo spesso vogliamo cercare la soluzione al problema senza sapere nemmeno da dove partire.

I docenti citati sono in realtà solo alcuni dei tanti a cui vorrei dedicare un pensiero in questo momento, ma data la limitatezza di queste pagine, mi limito a porgere un sentito ringraziamento a tutti i docenti che nel corso dei tre anni, hanno dedicato il loro tempo all'arricchimento delle mie conoscenze e a quelle di tanti altri studenti.

Un particolare ringraziamento va al professore Andrea F. Abate e al dottor Ignazio Passero, che mi hanno aperto le porte al mondo della Computer Graphics e delle innumerevoli tecnologie e applicazioni ad essa correlate, tra cui l'enorme esperienza formativa che ha caratterizzato il mio lavoro di tirocinio. Grazie a loro e alla grande disponibilità dell'azienda Kineton (che purtroppo, date le circostanze, ho potuto conoscere solo a distanza) ho cominciato a conoscere, se pur nel mio piccolo, l'intelligenza artificiale e il machine learning, campi molto avvincenti che da sempre mi hanno incuriosito. Oltretutto ci tengo a ribadire che la loro disponibilità nel seguirmi anche a distanza con numerosi meeting, ha fatto la differenza in questa nuova avventura.

Come non ringraziare poi le innumerevoli persone che da sempre supportano le mie scelte, a cominciare dai membri della mia famiglia, ed in particolar modo i miei genitori e i nonni sempre pronti ad aiutarmi qual ora ne avessi avuto bisogno. Un ringraziamento altrettanto importante senz'altro va dedicato agli amici di sempre Andrea, Francesco, Agostino, Fabio, Assunta, Mariapia e Nicola, che da anni ormai mi sono vicini e sono determinanti in tutte le mie scelte se pur non di carattere prettamente universitario.

Per finire mi sembra anche doveroso ringraziare anche Ferdinando e Nunzia, grandi consiglieri nei momenti di incertezza più totale, Michelantonio ed Emmanuel di cui sono stato il fedele braccio destro in questi tre anni. Benedetta, che nell'ultimo anno si è rivelata una persona su cui si può sempre contare, Simone, Simona, Vincenzo, Luigi, Rosario, Sanny, Marco, Felice e Gianluca amici con cui si può chiacchierare e scherzare davvero su mille cose, il "Collega" Francesco, amico dalla grande simpatia, ma al

contempo molto dedito e caparbio in ogni suo impegno, con cui ho avuto il piacere di affrontare quest'enorme esperienza conclusiva e tanti altri amici e amiche con cui che nel corso dei tre anni a modo loro mi sono stati vicino.

ABSTRACT

Al giorno d'oggi la viabilità stradale è un contesto che sta diventando sempre più sensibile alla necessità di automazione correlata da una buona dose di sicurezza, infatti quotidianamente un qualsiasi conducente d'auto fa i conti con una serie di problematiche che a colpo d'occhio possono sembrare banali, ma richiedono un'elevata dose di attenzione. Una valutazione corretta alla guida, magari supportata dagli opportuni suggerimenti, può infatti fare la differenza tra una manovra sicura e un tamponamento, tra la scelta della corretta velocità stradale in relazione alle condizioni atmosferiche e un brutto incidente in cui si rischia troppo spesso la vita, tra il rispetto della segnaletica e l'incolumità dei passanti in prossimità di un attraversamento e la possibilità di ferire seriamente qualcuno.

L'intelligenza Artificiale e la sensoristica di bordo hanno fatto notevolmente progredire il settore dei sistemi di assistenza alla guida. Da informatico, mi sono sempre chiesto in che modo un sistema software o un sistema automatizzato potessero intervenire al fine di garantire un corretto comportamento alla guida, e la mia esperienza di tirocinio nell'azienda Kineton è stata davvero utile a dare risposta a questa domanda.

La mia esperienza si è incentrata su un primo approccio al mondo del Machine Learning, una branca dell'informatica che al giorno d'oggi trova applicazione in svariati settori nei quali la tecnologia sta diventando sempre di più indispensabile. Per quanto riguarda il mondo dell'automotive, il Machine Learning è stato incorporato in molte componentistiche software e hardware che interagiscono con le componenti elettriche dell'autovettura e forniscono informazioni al conducente al fine di garantire un'esperienza di guida sicura. Di particolare rilievo sono i sensori ADAS (Advanced Driving Assistance System), i quali sono addestrati a rilevare e intervenire in modo congruo e preventivo di fronte a potenziali situazioni di pericolo su strada.

Sfruttando le potenzialità dell'engine 3D Unity e della libreria Python MLAgents, il mio progetto è stato incentrato nel simulare una scena di attraversamento pedonale in

contesto urbano, in un tratto lineare di strada, nella quale un'auto munita di un sistema di sensoristica per la tenuta di strada e il rilevamento pedonale (atto a simulare in toto il comportamento ottimale di un sensore ADAS) è stata addestrata, tramite l'approccio basato su premiazione, usato generalmente per le applicazioni di Reinforcement Learning, a tenere una corretta traiettoria di strada in sensi di marcia differenti, dall'inizio alla fine dell'ambiente di scena e a regolare la velocità e l'utilizzo dei freni in presenza o assenza di pedoni, quando si trova in prossimità delle strisce di attraversamento.

Dopo svariati tentativi, l'auto ha imparato correttamente a sfruttare le sue potenzialità, sviluppando un comportamento ottimale di fronte al pedone, gestendo in maniera completamente automatizzata l'utilizzo dei freni e la corretta ripresa di marcia.

INDICE

INTRODUZIONE

LA PROBLEMATICA

Da circa un secolo e mezzo oramai, l'essere umano ha a che fare con l'automobile, uno strumento che dalla sua messa in circolazione si è ben presto rivelato un'arma a doppio taglio: da un lato ha portato con se una serie di innumerevoli vantaggi che hanno semplificato di molto la realtà quotidiana dell'essere umano, ma dall'altro con essa sono nate una serie di pericoli che hanno influenzato, non di poco, la concezione quotidiana di sicurezza urbana ed extraurbana.

Al giorno d'oggi infatti, i pericoli connessi al mondo della strada sono una forte tematica di studio che coinvolge numerose realtà accademiche e aziendali (dagli studi del diritto stradale alla ricerca di nuove tecnologie) che, se pur in modo differente, si cimentano quotidianamente nella continua ricerca di soluzioni per garantire stabilità e sicurezza comune a tutte le personalità che ogni giorno hanno a che fare con l'automobile.
Ogni studioso che si approccia per la prima volta allo studio della sicurezza urbana, si pone di fronte ad una serie di domande che non hanno una semplice risposta, ad esempio:

- "Quali sono le principali problematiche stradali che necessitano di maggiore attenzione?"
- "Come il mio settore può intervenire per migliorare questa realtà?"
- "Quali strumenti posso mettere in gioco al fine di trovare una soluzione ottimale ad uno di questi problemi?"

Ovviamente le risposte a queste domande e le soluzioni proposte sono svariate, ma è difficile dire se alcune di esse sono ottimali o se lo saranno in futuro, l'ambito della circolazione stradale è un qualcosa di molto dinamico e complesso, ogni esperienza di guida deve essere ben analizzata e monitorata, in quanto anche il più piccolo cavillo può fare la differenza tra un corretto modo di agire o un brutto incidente su strada.

INTELLIGENZA ARTIFICIALE E AUTOMAZIONE DEL VEICOLO

Da studioso d'informatica, da sempre scienza affine all'automazione e alla ricerca di soluzioni per svariate gamme di problemi, mi sono approcciato per la prima volta alla problematica di circolazione stradale osservando che lo sviluppo tecnologico legato al mondo Automotive si incentra, da più di una decina d'anni, sulla creazione di sistemi automatizzati che rendono il mezzo di trasporto in grado di assistere il guidatore nelle scelte corrette da avere in qualsiasi situazione di pericolo su strada.

In particolare, l'Intelligenza Artificiale e il Machine Learning hanno reso possibile la progettazione e la messa in esercizio dei sensori ADAS (Advanced Driver Assistance Systems), componentistiche che vengono programmate e addestrate da zero al riconoscimento di una potenziale situazione di pericolo e a garantire in maniera autonoma una corretta reazione del veicolo, migliorando non di poco l'esperienza di guida del conducente. Un sensore ADAS può avere innumerevoli applicazioni, ad esempio: la corretta esecuzione di un parcheggio, una corretta valutazione in corrispondenza di un semaforo, il rispetto della segnaletica stradale, la regolazione di velocità del mezzo in relazione alle condizioni atmosferiche, oppure il riconoscimento di pedoni in corrispondenza di un attraversamento, e conseguente frenata automatica per consentirne il passaggio in sicurezza.

Proprio su quest'ultima problematica si è incentrato il mio lavoro di tirocinio tramite l'azienda Kineton, con la quale è stato progettato e implementato, in ambiente 3D simulato, un esempio di veicolo dotato di sensoristica ADAS, atta al riconoscere pedoni su strada, tramite l'ausilio di Machine Learning.

In particolare, è stato usato l'approccio puro di addestramento del Reinforcement Learning, tramite il quale il modello di auto, inserito senza alcuna conoscenza pregressa in una scena di circolazione stradale lineare, con attraversamenti pedonali, ha progressivamente imparato come riconoscere i pedoni e mantenere un corretto

comportamento sia in termini di viabilità, sia per garantire l'incolumità del pedone in corrispondenza dell'attraversamento.

SISTEMA PROPOSTO

Come già accennato nel paragrafo precedente, il sistema proposto si articola nella realizzazione di una simulazione 3D, realizzata con l'engine Unity per lo sviluppo di giochi e simulazioni bidimensionali e tridimensionali, di una quotidiana scena di attraversamento stradale in cui un' modello d'auto è stato dotato di un sistema di sensoristica per l'analisi dell'ambiente circostante (di cui successivamente verranno forniti maggiori dettagli) atto a simulare il comportamento di una serie di sensori ADAS per il corretto movimento su strada e il riconoscimento dei pedoni quando essi sono situati in prossimità dell'auto. Per sviluppare corretto comportamento da avere in prossimità dei pedoni e per la tenuta di strada è stato scelto l'utilizzo della libreria Python ML Agents (libreria sviluppata l'apprendimento automatico di agenti in una generica scena Unity), tramite la quale l'agente di guida (l'auto della simulazione) ha come obbiettivo quello di imparare, nel modo più autonomo possibile, il corretto comportamento da avere all'interno della simulazione.

STRUTTURA DEL DOCUMENTO

Il documento di tesi sarà articolato nel seguente modo:

- Capitolo 1 – Paradigmi e Metodologia utilizzati – Nozioni fondamentali alla base del progetto implementate, e interconnessioni tra le stesse;

- Capitolo 2 – Tecnologie Implementative e strumenti di sviluppo utilizzati;

- Capitolo 3 – Progettazione e Sviluppo del progetto:
  Il terzo capitolo illustrerà come la scena di attraversamento pedonale è stata progettata e successivamente realizzata, ponendo l'accento sia sul Set-Up

dell'ambiente di simulazione, sia sulle caratteristiche principali delle entità coinvolte nella simulazione, e di principi di implementazione e strategie per il machine learning adottate per la predisposizione alle successive sessioni di Training;

- Capitolo 4 – Difficoltà e risultati del training:
  Nel quarto capitolo del documento verrà spiegato come l'autovettura è stata addestrata a mantenere una corretta tenuta di strada e al corretto comportamento da avere in caso di rilevazione di pedoni in attraversamento, ponendo l'attenzione sull'organizzazione generale della fase di training, dei principi di addestramento adottati ad ogni macro-fase di training, dei risultati ottenuti ad ognuna di esse e delle difficoltà incrementali introdotte con il progredire dei progressi;

- Capitolo 5 – Prototipo e sviluppi futuri:
  Nel quinto capitolo, vengono illustrate le caratteristiche fondamentali del prototipo dimostrativo del progetto di tirocinio completo, e alcuni piccoli cenni ai possibili sviluppi futuri;

- Bibliografia.

CAPITOLO 1 – PARADIGMI E METODOLOGIA

1.1 SISTEMI ADAS

Al giorno d'oggi ogni auto di nuova fabbricazione è dotata di dispositivi elettronici atti a ridurre al minimo i rischi di incidente ed agevolare la vita a bordo dell'automobile, questi sistemi stanno avendo un successo sempre più ampio nell'ultimo decennio, tanto da spingere la Commissione Europea a renderli obbligatori su auto di nuova omologazione a partire dal 2022, secondo la quale i sistemi ADAS potranno prevenire circa 25.000 morti e 140.000 feriti sulle strade europee tra il 2022 e il 2038 [1].

FIGURA 1 - RILEVAZIONE AUTOMATICA DI UN PEDONE SU STRADA

Come suggerisce il significato più profondo del termine, i sistemi ADAS (Advanved Driver Assistence System per l'appunto) sono sistemi elettronici che pilotano l'autovettura nelle scelte da avere nelle più svariate situazioni, tra cui: sensori di parcheggio, l'avviso di cambio corsia, il riconoscimento automatico di segnali, sensori per l'andamento di velocità e per il rilevamento atmosferico. Di rilevante importanza per il progetto sviluppato, sono state le metodologie di approccio dei sensori:

- di mantenimento della carreggiata [1]: telecamere ottiche o radar piazzati sull'autovettura che riconoscono le linee di demarcazione di una carreggiata o una corsia e identificano quando le stesse vengono oltrepassate;
- di avviso di collisione frontale [1]: per il riconoscimento di ostacoli o persone su strada durante la circolazione, in generale questi sensori avvertono il guidatore tramite segnale acustico, ma possono anche essere utilizzati al fine di migliorare i comportamenti automatici che un'auto può o deve avere in situazioni ottimali;

- per la frenata di emergenza automatica [2]: sistemi di controllo per i freni, al fine di fermare il veicolo di fronte ad una potenziale situazione di pericolo, quali ad esempio un'altra autovettura troppo vicina o un pedone troppo vicino all'auto in velocità.

FIGURA 2 - TIPICO ESEMPIO DI FRENATA AUTOMATICA TRAMITE AUSILIO ADAS

Tanti sono i progetti di applicazione futura per i sensori ADAS, in particolare sono in studio sistemi che permetteranno all'auto di eseguire valutazioni e azioni con livelli di autonomia molto elevati, da affiancare alla presenza del guidatore, che mantiene il compito di sorveglianza e supporto al veicolo in caso di necessità. Dal riconoscimento e analisi di ostacoli tridimensionali o di altri veicoli su strada, fino alla pura valutazione autonoma di una situazione di pericolo, l'industria dell'automazione considera questi concetti sempre meno utopistici, tanto da aver introdotto dei formalismi e classificazioni molto rigorose come ad esempio il grado di autonomia di cui un veicolo dispone [2] (strutturato in 5 livelli nella classificazione definita dalla SAE – Society of Automobile Engineers).

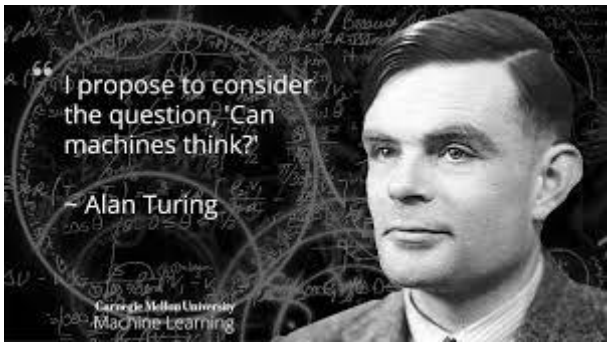
- Livello 0: nessuna automazione.
- Livello 1: guida assistita o presenza di supporti per la guida assistita.

- Livello 2: automazione parziale.
- Livello 3: automazione condizionata.
- Livello 4: alta automazione.
- Livello 5: guida interamente autonoma.

1.2 MACHINE LEARNING E METODOLOGIE DI APPROCCIO UTILIZZATE

Il Machine Learning [3] è una particolare branca dell'informatica molto affine ai concetti di intelligenza artificiale, atta allo sviluppo di automi intelligenti in grado di migliorare le proprie capacità e prestazioni nel tempo. Il campo di applicazione del Machine Learning è molto complesso in quanto esso stesso prevede differenti modalità, strumenti e tecniche per l'apprendimento e lo sviluppo di algoritmi, dando vita ad altrettante possibilità di utilizzo che allargano il campo dell'apprendimento automatico rendendone difficile una definizione specifica del concetto.

In altri termini, è possibile affermare che il compito di un generico algoritmo di Machine Learning, è quello di definire una serie di notazioni primitive, le quali saranno poi elaborate da un agente, in una specifica fase di apprendimento definita training, durante la quale vengono osservate le caratteristiche dell'ambiente circostante e si interagisce con esso sulla base delle osservazioni effettuate.



Storicamente la nascita del Machine Learning è da collocare agli inizi degli anni Cinquanta del Novecento, quando per la prima volta studiosi del calibro di Alan Turing, ipotizzarono la necessità di realizzare macchine in grado di apprendere. In quegli stessi anni, anche gli studi sull'intelligenza artificiale, in particolar modo sulle reti neurali (circuiti neurali artificiali atti a simulare il funzionamento delle più svariate reti neurali umane), portarono a numerosi investimenti del settore. Per trovare "linfa nuova"

l'apprendimento automatico dovrà aspettare però la fine degli anni Novanta, quando una serie di innovative tecniche legate ad elementi statistici e probabilistici, hanno portato il Machine Learning ad essere un ramo della ricerca riconosciuto e altamente richiesto.

Come già accennato l'apprendimento automatico è una branca molto complessa dell'informatica e classificarne le principali metodologie e approcci non risulta affatto semplice, tuttavia i diversi paradigmi fondanti del machine learning, possono essere classificati in due grandi aree:

- L'apprendimento supervisionato che consiste nel fornire al sistema informatico di una macchina una serie di notazioni specifiche e codificate, ossia esempi che permettono di costruire un vero e proprio data base di informazioni ed esperienze su problemi specifici e caratterizzati, dalle quali attingere e formulare ipotesi per generare la miglior risposta a problemi di tipo più generale.

- L'apprendimento senza supervisione che invece non prevede informazioni codificate inserite all'interno della macchina, ossia l'automa stesso ha la possibilità di attingere a determinate informazioni fornite o apprese in fase di addestramento senza avere alcun esempio del loro utilizzo. Dovrà essere onere dello stesso quello di catalogare tutte le informazioni in proprio possesso, organizzarle ed imparare il loro significato, il loro utilizzo e, soprattutto, il risultato a cui esse portano. L'apprendimento senza supervisione offre maggiore libertà di scelta alla macchina che dovrà organizzare le informazioni in maniera intelligente e imparare quali sono i risultati migliori per le differenti situazioni che si presentano.

Al di là della strategia di Learning utilizzata, caratteristica comune ad ogni macchina che deve essere sottoposta ad un processo di apprendimento automatico, è la fase di Addestramento o Training, fase in cui l'automa (in modo pilotato o semi pilotato via software) acquisisce informazioni inerenti all'ambiente circostante o rielabora quelle già immagazzinate al fine di produrne di più complesse.

1.3 Reti Neurali e Machine Learning

[4] Uno dei fondamenti base a cui il machine learning si appoggia è l'utilizzo di Reti Neurali Artificiali, una rete neurale può essere vista come una rete di tante piccole informazioni matematiche computabili e traducibili in dati di input per un agente al fine di essere elaborati per produrre l'output desiderato in un'altra forma. Il concetto di Rete Neurale Artificiale è inspirato alla biologia umana e al modo in cui i neuroni in un cervello umano elaborano gli input dai 5 sensi.

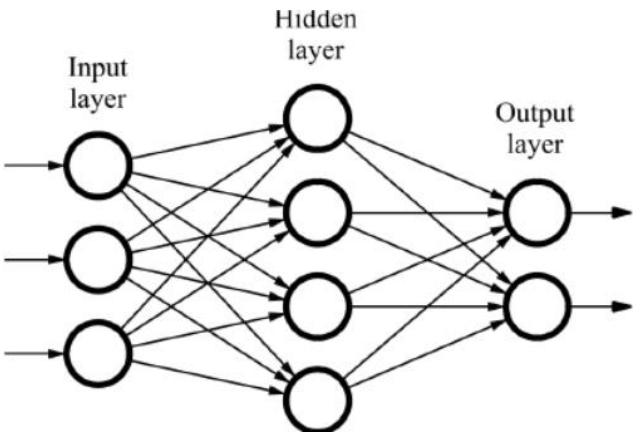


Figura 3 - Esempio di Rete Neurale

Le reti neurali possono essere viste come un insieme di strumenti e approcci usati in un algoritmo di machine learning, le quali vengono perfezionate solo tramite l'acquisizione di nuove esperienze tramite la fase apprendimento, che permette all'automa di migliorare le informazioni in suo possesso ed elaborarne di nuove costruendo così reti neurali sempre più complesse.

L'apprendimento automatico, in particolar modo nella sua variante non supervisionata, sfrutta le reti neurali al fine di immagazzinare informazioni sempre più complesse per migliorare le proprie esperienze di percezione e interazione con l'ambiente circostante.

1.4 Reinforcement Learning e Reti Neurali

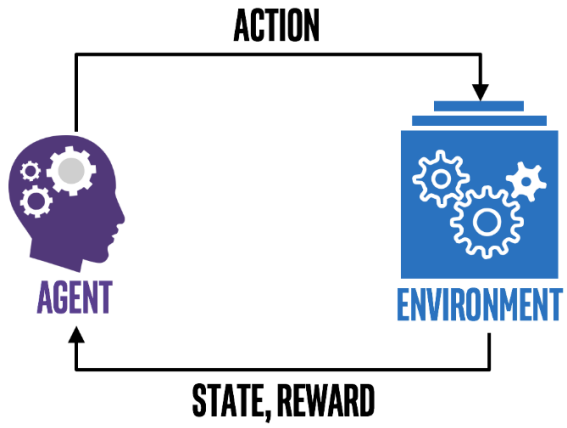


FIGURA 4 - SCHEMA BASE DELL'APPRENDIMENTO PER

La metodologia di apprendimento automatico tramite rinforzo [3] rappresenta forse il sistema di apprendimento più complesso, prevedendo che l'agente sia dotato di sistemi e strutture in grado di migliorare il proprio apprendimento, in stretta relazione alle caratteristiche dell'ambiente circostante. In particolare, un algoritmo di Deep Reinforcement Learning [5], **l'agente*** in addestramento ha l'obbiettivo di costruisce una rete neurale sulla base di informazioni derivanti dal progressivo apprendimento di informazioni derivanti dall'ambiente circostante.

- **Agente** – Sinonimo per indicare una macchina o un modello tridimensionale sottoposto ad un addestramento di Machine Learning. -

In fase di addestramento, Per un agente (a cui è stato predisposto un sistema di sensoristica adeguato ad apprendere informazioni dall'ambiente circostante) che ha come base un algoritmo di apprendimento basato su rinforzo, viene definito un sistema di premiazione, che fa discriminare quali informazioni salvare nella nuova rete neurale e quali discriminare. In particolar modo un agente viene premiato positivamente, quando effettua le azioni attese in output, e negativamente nel caso opposto.

1.5 Reinforcement Learning e Sistemi ADAS

L'apprendimento per rinforzo è tipico nelle simulazioni di guida per l'Automotive, infatti un modello di autovettura, reale o virtuale, dotato un complesso sistema di sensori di supporto è in grado di percorrere strade cittadine e non, riconoscendo eventuali ostacoli, seguendo le indicazioni stradali e molto altro [3].

Esempio di notevole importanza può essere la simulazione 3D di un'autovettura su una strada urbana nei pressi di un attraversamento. La quale dotata di un sistema di tipo Ray Tracing atto a simulare sensori ADAS per il rilevamento di corsia, rilevamento di collisioni e frenata assistita in presenza di pedoni, può essere addestrata a mantenere un comportamento corretto tramite un algoritmo di Deep Reinforcement Learning.

FIGURA 5 - AUTOMOTIVE E MACHINE LEARNING

Tale esempio, come già anticipato, è stato potato avanti durante l'esperienza di tirocinio da me sostenuta presso l'azienda Kineton, partendo appunto da un agente (autovettura dotata di raggi), del tutto inconscia del corretto comportamento da avere al fine di circolare correttamente su strada ed evitare di collidere con un passante di un attraversamento pedonale posto davanti alla traiettoria del mezzo. Scopo del training di Reinforcement Learning applicato, è stato appunto quello di far sviluppare all'agente le corrette nozioni da avere al fine di migliorare sempre di più il comportamento con la scena circostante, sviluppando man mano reti neurali sempre più precise, che hanno messo l'autovettura in condizione tale di migliorare in maniera costante e minuziosa la sua interazione con l'ambiente circostante.

- Dettagli di progettazione e implementativi verranno trattati nei successivi capitoli del documento in seguito alle specifiche inerenti alle tecnologie utilizzate e agli strumenti ausiliari utilizzati per il training basato su apprendimento per rinforzo.

1.6 Game Engine e Sviluppo di Simulazioni

Prima di introdurre e analizzare gli strumenti e le tecnologie utilizzate per il lavoro progettuale è doveroso fare una breve digressione su come è stato possibile creare un agente virtuale e un ambiente 3D da utilizzare come Environment per la simulazione di Machine Learning.

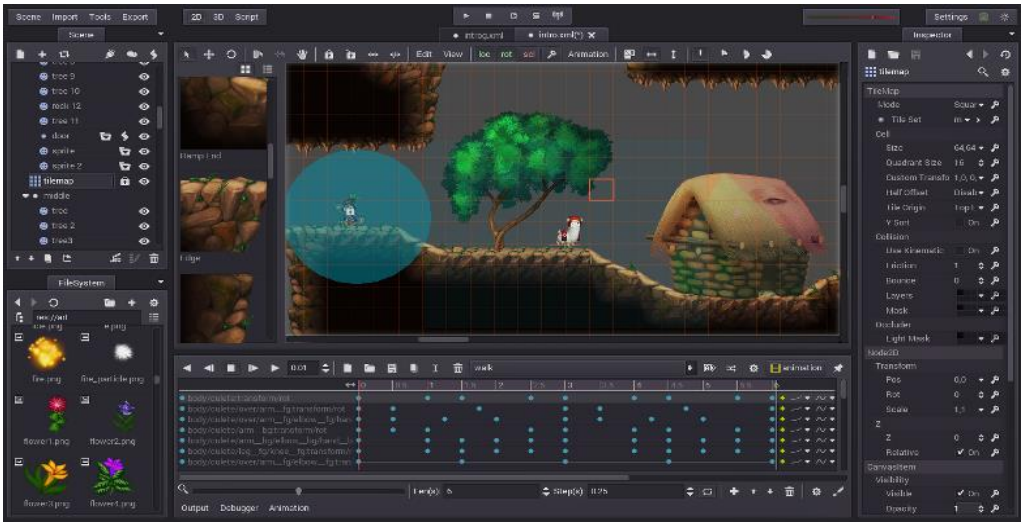


Figura 6 - Making di una scena tramite motore grafico

Un motore grafico [6] (o game engine) è un software che fornisce un set di funzionalità necessarie per lo sviluppo di giochi e simulazioni virtuali in due o tre dimensioni in maniera semplice e veloce. È possibile intendere lo sviluppo di applicazioni tramite un Game Engine come un framework del game development che supporta e collega tra di loro in modo quasi del tutto trasparente funzionalità molto differenti tra di loro.

Tra le principali aree "core" di un Game Engine, sono senz'altro di rilievo:

- La gestione della grafica: i moderni motori grafici (Unreal Engine, Unity, …) forniscono API molto potenti per l'organizzazione e l'interconnessione di contenuti grafici (modelli 3D ad esempio) in una scena a due o tre dimensioni, la cui resa grafica non pecca di basse prestazioni, data le complesse capacità di rendering visuale di cui i moderni Engine dispongono. La realizzazione di una scena di gioco è molto facilitata date le possibilità di includere asset già modellati anche di produzione differente;

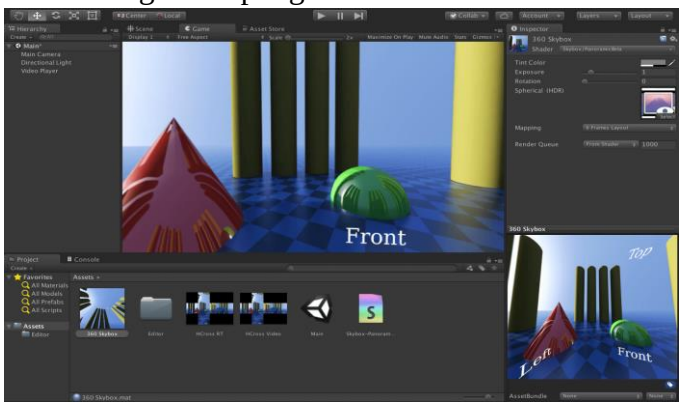


FIGURA 7 - UNITY RENDERING EXAMPLE

- Gestione dei file multimediali e dell'interfaccia grafica: Oltre le funzionalità di modellazione per scena 3D, un Engine mette a disposizione la possibilità di integrare in maniera nativa anche effetti sonori, registrazioni vocali, immagini, componentistiche per la realizzazione di interfacce grafiche (quali testi, bottoni, slider, top down menù), al fine di garantire un'esperienza di gioco piena e completa all'utente finale;

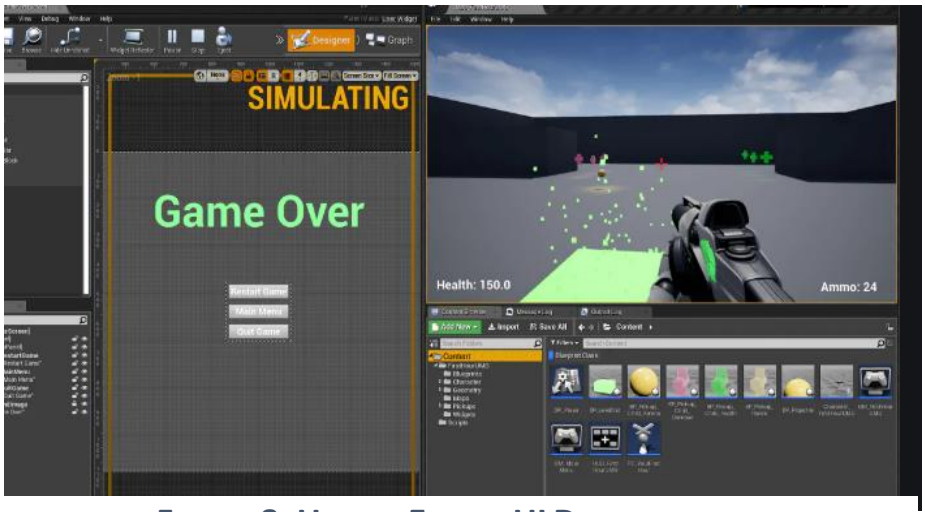


FIGURA 8 -UNREAL ENGINE UI DEFINITION

- Un aspetto cruciale di un Game Engine è rappresentato dagli script che definiscono la logica comportamentale delle componenti di gioco tramite l'ausilio di elementi di programmazione. Tramite lo scripting di un Game Engine, è facilmente possibile intercettare qualsiasi parametro o componente di un oggetto di scena e definirne gli aspetti comportamentali, esempi lampanti sono i controlli: di un player di gioco, di un NPC, di altre componenti (quali ad esempio una camera). È inoltre possibile definire tramite script interazioni e animazioni statiche e dinamiche tra differenti componenti di gioco.

- Di rilevante importanza sono i sistemi fisici che in un Game Engine forniscono una simulazione fisica molto realistica in maniera quasi automatica. Ogni componente di una scena, a seconda del grado di complessità dell'Engine, può essere personalizzato in termini di masse, forze e attriti di vario tipo, in modo da generare differenti reazioni in relazione con la gravità, l'evironment di scena e soprattutto con le collisioni con altri oggetti. Scripting e Gestione della fisica possono essere tra di loro interconnessi, infatti i motori grafici permettono di definire e manovrare caratteristiche fisici con poche linee di codice.

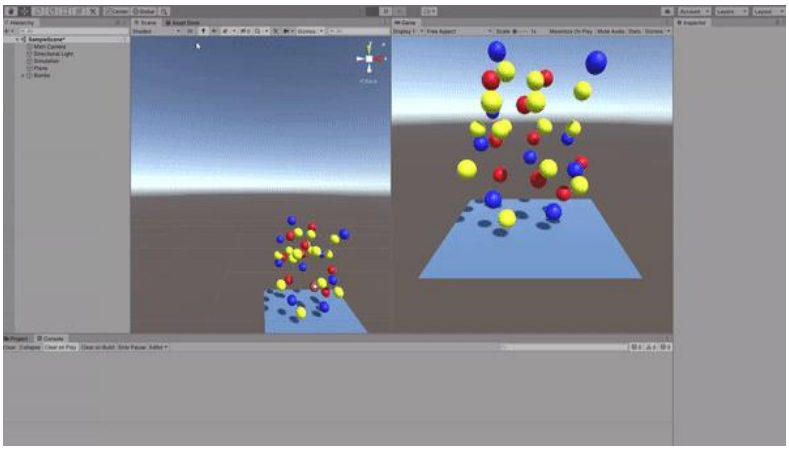


FIGURA 9 - SIMULAZIONE FISICA IN AMBIENTE UNITY

- Molti Game Engine, ma non tutti, permettono di sviluppare applicazioni multiplayer, tramite l'utilizzo di High Level API molto complesse e strutturate, tramite un'architettura di rete sottostante (Client – Server o Peer to Peer), con protocolli di trasporto e rete quasi del tutto trasparenti allo sviluppatore.

CAPITOLO 2 – TECNOLOGIE E STRUMENTI UTILIZZATI

In questo capitolo verranno introdotte tutte le tecnologie utilizzate per l'implementazione del progetto, partendo dall'ambiente di modellazione della scena 3D, passando per le tecnologie di machine learning utilizzate, linguaggi di programmazione utilizzati e supporti ausiliari.

2.1 UNITY 3D

Unity 3D è un efficiente Game Engine per lo sviluppo videoludico, il quale offre molteplici funzionalità gratuitamente.

Unity permette la definizione di videogiochi e simulazioni tridimensionali (ma anche bidimensionali) tramite l'ausilio di modelli predefiniti detti GameObjects. In Unity ogni GameObject è caratterizzato da una serie di proprietà fondamentali come la sua disposizione nello spazio virtuale, in termini di posizione su coordinate e la sua rotazione o lo scaling rispetto alla dimensione originale; possono essere definite e personalizzate altre caratteristiche di un singolo GameObject tramite l'utilizzo di textures e materiali, scripts e componenti inerenti alla fisica [8].

FIGURA 10 - UNITY LOGO

Unity, tramite la sua semplice interfaccia, permette agli sviluppatori di gestire ogni singola sfaccettatura del proprio progetto di Game Making. Ogni sottosezione dell'interfaccia è dedita ad una particolare caratteristica dell'iter di sviluppo:

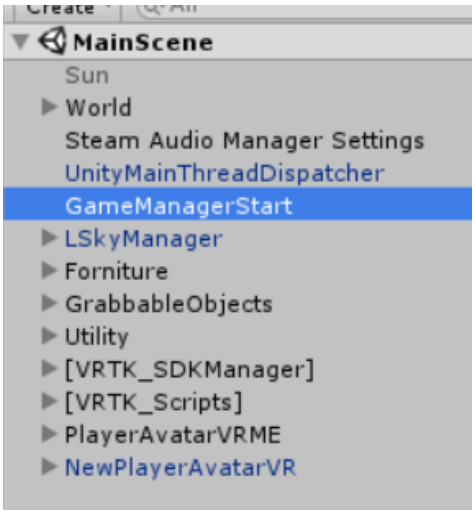


FIGURA 11 - GERARCHIA

- Gerarchia: tiene traccia di tutti gli oggetti presenti in scena in formato gerarchico e permette la loro gestione, tramite un'interfaccia a livelli. Gli oggetti di scena, in questo modo,

catalogabili in strutture padre-figlio, in modo da personalizzare alcune proprietà di un oggetto in funzione di un altro posto al livello superiore;

- Finestra Progetto: permette l'organizzazione e la gestione di tutti i contenuti importati e creati durante lo sviluppo del progetto videoludico (di default tali contenuti sono posizionati nella cartella Assets). Inoltre, la project view, tiene traccia di tutte le librerie importate nel progetto;

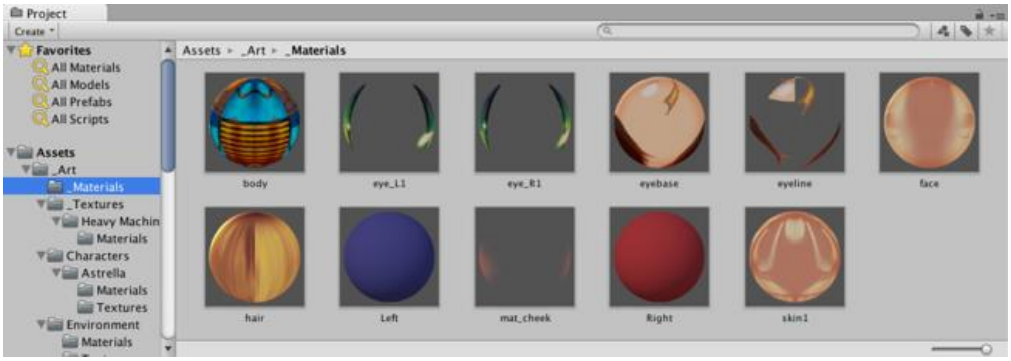


FIGURA 12 - PROJECT WINDOW

- Scene view: finestra dedita alla realizzazione e modellazione di un ambiente simulato, detto anche ambiente di scena. Nella finestra di scena è quindi possibile integrare nuovi GameObjects, modellare caratteristiche e dimensioni in maniera manuale tramite appositi strumenti messi a disposizione dall'editor;
- Game view: visuale dedita al rendering in tempo reale ciò che accade nella scena durante l'esecuzione di un'anteprima di gioco, dal punto di vista della camera di gioco attualmente attiva. Dando una visione dettagliata e realistica delle future interazioni tra i GameObjects nella build stand alone del videogioco.
- Inspector: permette la visualizzazione e la gestione di tutte le proprietà inerenti ad un singolo GameObject, quindi permette l'immissione di scripts (e personalizzazione dei singoli parametri annessi), dei colliders, proprietà fisiche e di posizionamento, ed altro;

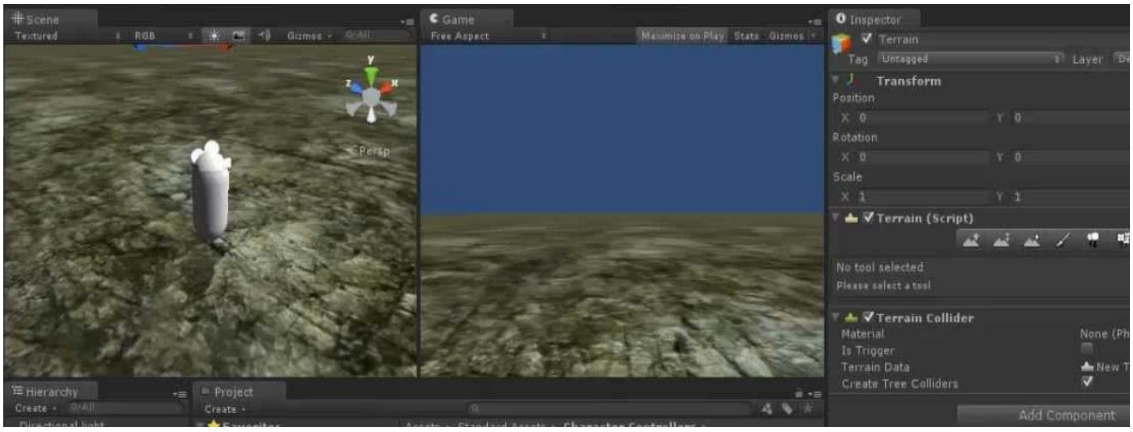


FIGURA 13 - SCENE VIEW, GAME VIEW E INSPECTOR

- Asset Store: Store integrato in Unity per l'acquisto o il download gratuito di risorse messe a disposizione da altri sviluppatori videoludici [7].

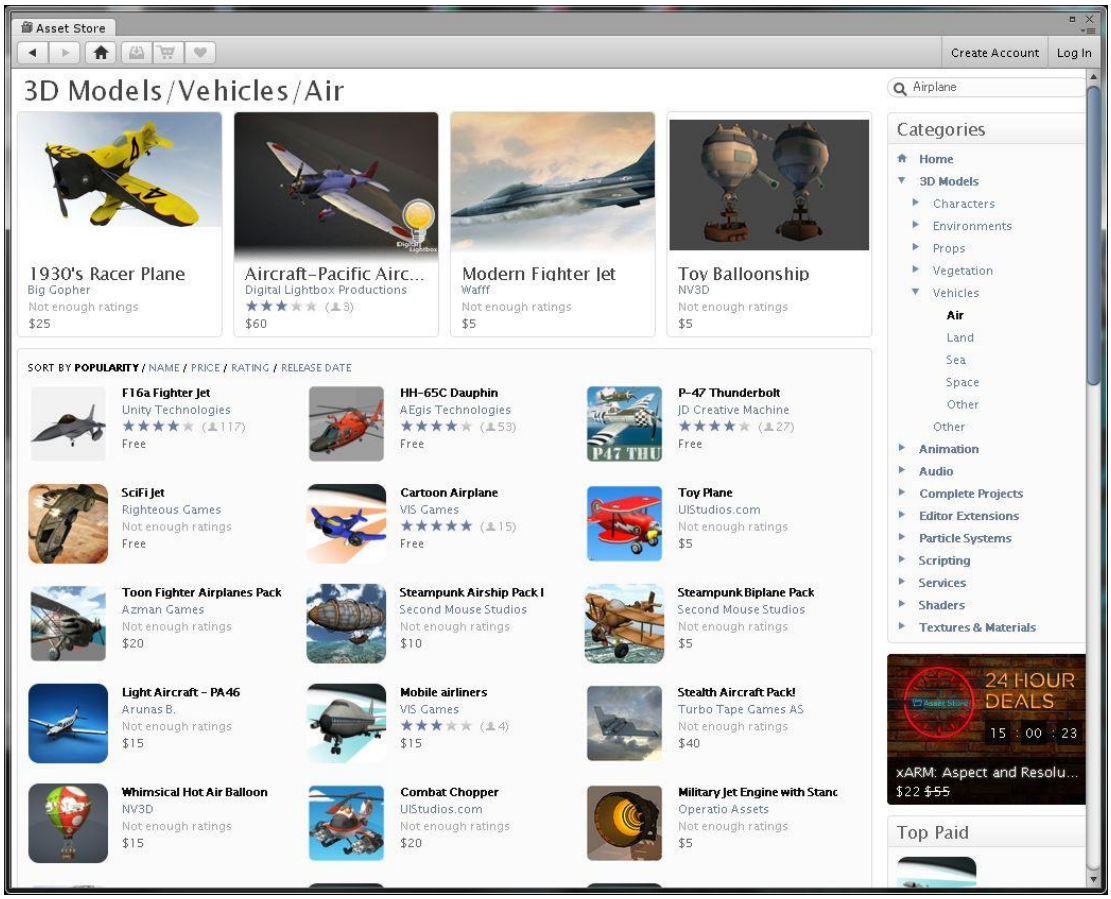


FIGURA 14 - UNITY ASSET STORE

2.1.1 SCRIPTING E LINGUAGGIO C#

Durante lo sviluppo di un progetto in Unity, è possibile definire e personalizzare il comportamento di uno specifico GameObject in scena tramite l'operazione di scripting. Molte sono le differenze tra lo scripting e la normale programmazione, infatti in uno script si delineano comportamenti dei singoli oggetti e non l'infrastruttura di un applicativo da eseguire, in quanto è l'Engine ad occuparsi di compilazione e di esecuzione integrata dei codici.

Gli script sono scritti in C#, linguaggio object-oriented, oramai standard ed integrato in modo nativo nell'intero iter di sviluppo di un progetto Unity [8].

Ogni script è una classe che, in genere, eredita MonoBehaviour, classe di libreria in cui sono specificate funzioni utili alla manipolazione del GameObject e delle sue componenti. Funzioni standard e molto utilizzate, in uno script che eredita MonoBehaviour, sono:

- La funzione `start()`, che viene alla creazione del GameObject;
- La funzione `update()`, che invece viene eseguita in modo costante ad ogni frame.

  MonoBehaviour non è l'unica classe ereditabile in uno script Unity: è possibile estendere altre classi che forniscono comportamenti diversi basati su situazioni diverse. Un esempio è la classe Agent che offre funzionalità riguardanti il Machine Learning, oppure la classe NetworkBehaviour che offre funzionalità riguardanti il multiplayer.

2.2 ML-AGENTS

Lo Unity Machine Learning Agents Toolkit, conosciuto come ML-Agents, è un progetto open source che permette di istruire agenti intelligenti tramite svariati meccanismi di Machine Learning, Deep Reinforcement Learning nel caso del progetto sviluppato, con il supporto di una API scritta in Python.

L'addestramento di agenti tramite tecnologie di Machine Learning può essere utile a tante problematiche come: controllo degli NPC (personaggi non giocabili), testing autonomi nelle build di gioco, o per lo sviluppo di simulazioni della realtà in ambienti simulati.

ML-Agents è utile sia in ambito di sviluppo che di ricerca e didattico, siccome permette di addestrare agenti in simulazioni sviluppate in ambiente Unity, tramite una semplice interfaccia shell di comando [9].

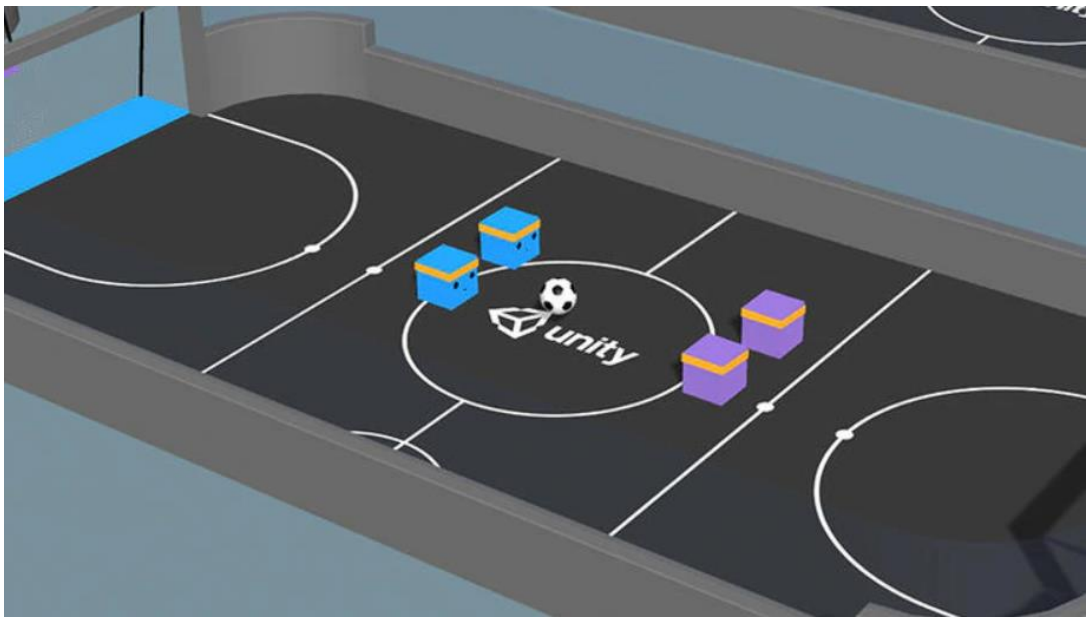


FIGURA 15 - ML-AGENTS TOOLKIT

2.2.1 AGENTE

Un'agente è un'oggetto di scena che viene addestrato a prende decisioni in maniera autonoma, utilizzando il paradigma del Machine Learning tramite le informazioni ottenute interagendo con l'ambiente durante le sessioni di training.

L'agente si basa su tre entità per modellare il suo comportamento e costruirne una rete neurale:

- Le osservazioni, cioè ciò che l'agente percepisce dall'ambiente circostante, esse possono essere di tipo numerico o visuale. L'ambiente esterno può essere visualizzato e analizzato tramite l'ausilio di appositi sensori, come specifiche camere

integrate per le osservazioni visuali e sistemi basati su collisioni di raggi (RayPerception Sensors in Unity) con ostacoli per osservazioni di tipo numerico;

- Le azioni sono le decisioni che l'agente prende in seguito ad ogni specifica osservazione. Possono essere continue (eseguite ad ogni update) o discrete (eseguite in maniera sporadica). La tipologia di azioni utilizzate dipende dalla complessità dell'ambiente circostante;
- Le ricompense, scalari matematici utilizzati per far comprendere all'agente se ciò che sta eseguendo sia corretto o meno, in base al valore positivo o negativo, rispecchiando a pieno il paradigma del Reinforcement Learning;

Inoltre, il GameObject agente è caratterizzato dalla componente Behaviour Parameters, la quale permette di definire la tipologia e il numero di azioni che l'agente dovrà eseguire in fase di training e di incapsulare una Neural Network prodotta dal training già concluso.

Il comportamento dell'agente viene definito da script, il quale estende la classe Agent nella quale sono definiti una serie di metodi utili all'addestramento.

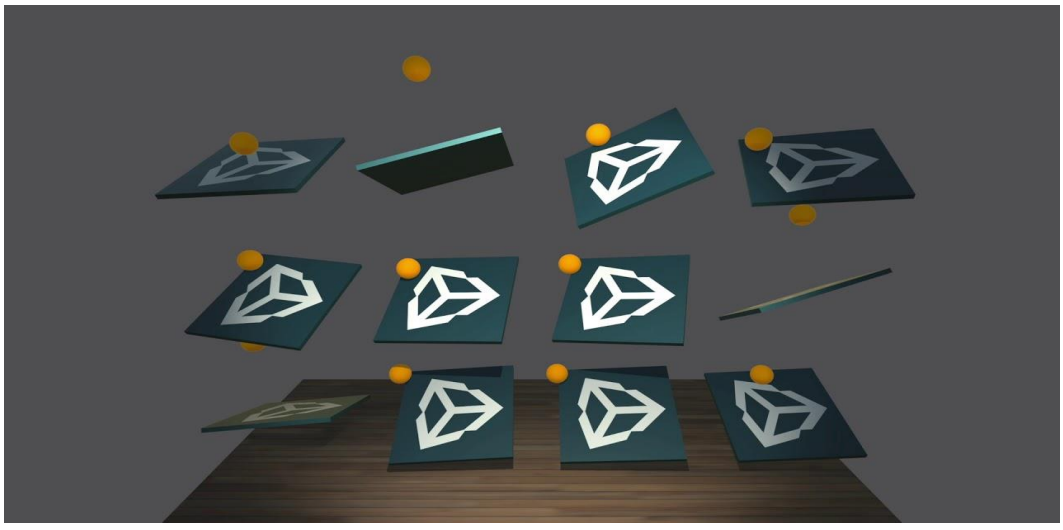


FIGURA 16 - ADDESTRAMENTO DI MOLTEPLICI AGENTI NEL MANTENERE IN EQUILIBRIO UNA PALLINA

Alcuni metodi più rilevanti di questa classe sono i seguenti:

- `Initialize()`: contiene il codice utile all'inizializzazione dell'agente, quindi permette l'assegnamento di valori alla creazione di quest'ultimo;
- `CollectObservations(VectorSensor sensor)`: permette di definire i parametri da osservare, inerenti all'ambiente circostante, di cui si terrà conto durante l'addestramento, al fine di migliorare il comportamento dell'agente;
- `Heuristic(float[] actionsOut)`: permette di definire un profilo dell'agente manovrabile dal giocatore tramite comandi manuali. Questo metodo è molto utile durante la fase di debug e per altre tipologie di learning, come l'Imitation Learning;
- `OnActionReceived(float[] vectorAction)`: tramite questa funzione, l'agente utilizza un vettore di azioni, composto da valori numerici, che influenzerà le proprie scelte e decisioni. Che siano giusti o sbagliati, dipende dai reward assegnati. Le azioni presenti in questo vettore possono assumere determinati valori in base alla tipologia di azione a cui si fa riferimento: se l'azione è continua, il valore numerico avrà valore k tale che $-1 \leq k \leq 1$; se l'azione è discreta, il valore numerico è univoco per una determinata scelta che l'agente dovrà intraprendere;
- `OnEpisodeBegin()`: in questa funzione viene definito ciò che accade ad ogni inizio di un test, il quale viene anche chiamato episodio;
- `AddReward(float increment)`: funzione utile per l'assegnamento di ricompense positive o negative, in base al valore di incremento passato come argomento. È preferibile che tale funzione venga richiamato al succedere di un certo evento, onde evitare che un determinato punteggio venga assegnato in ogni frame;
- `EndEpisode()`:Determina la conclusione dell'episodio e si occupa del reset dell'agente, richiamando a sua volta il metodo `OnEpisodeBegin()`.

L'agente è caratterizzato da un Brain, il quale incapsula il processo che permette di prendere decisioni sulla base di osservazioni, azioni e algoritmi di learning.

Esistono quattro tipi di Brain:

- Player: non è stata affrontata alcuna sessione di training e il giocatore può muovere liberamente l'agente in modo da testare tutte le sue funzionalità;

- External: per addestrare un agente, il suo Brain dovrà essere impostato su External siccome permette di ottenere ricompense tramite appositi algoritmi di learning. Tale tipologia di Brain permetterà all'agente di ottenere informazioni dalle osservazioni, le quali permetteranno di prendere decisioni in maniera autonoma;
- Internal: una volta conclusa la sessione di training, il Brain potrà essere impostato su Internal per interpretare i risultati del training e per far effettuare decisioni all'agente tenendo conto di ciò che ha imparato;
- Heuristic: è una tipologia di Brain sperimentale che permette al programmatore di intervenire sulle scelte dell'agente [9].

2.2.2 LINGUAGGIO PYTHON

Python è un linguaggio di programmazione alto livello orientato agli oggetti, molto versatile nello sviluppo di software, proprio in Python vengono realizzate numerose API per gli scopi più svariati, come ad esempio API per interconnessioni di rete ad alto livello, API per il Web Development, oppure le più svariate tecnologie di supporto per la computer Graphics.

Famoso ed affermato nel mondo della programmazione moderna Python, è molto apprezzato anche per la sua particolare sintassi, tanto da essere contrassegnato dalla community con i simpatici concetti di:

- Beautiful is better than ugly;
- Simple is better than complex;
- Complex is better than complicated;
- Sparse is better than dense.

Svariate sono, anche, le librerie Python dedicate alle applicazioni di Machine Learning. D'altronde, il toolkit ML-Agents si basa proprio su Python.

2.3 TensorFlow e TensorBoard

TensorFlow è una libreria open source utilizzata nel mondo del Machine Learning, inizialmente creata dall'organizzazione AI di Google. ML-Agents usa questa libreria per computare i risultati dei training e per analizzare i parametri dell'applicazione [11]. TensorBoard è un tool grafico che interpreta e mostra i dati inerenti all'evoluzione dell'agente generati dal tool TensorFlow, tramite l'ausilio di grafici.

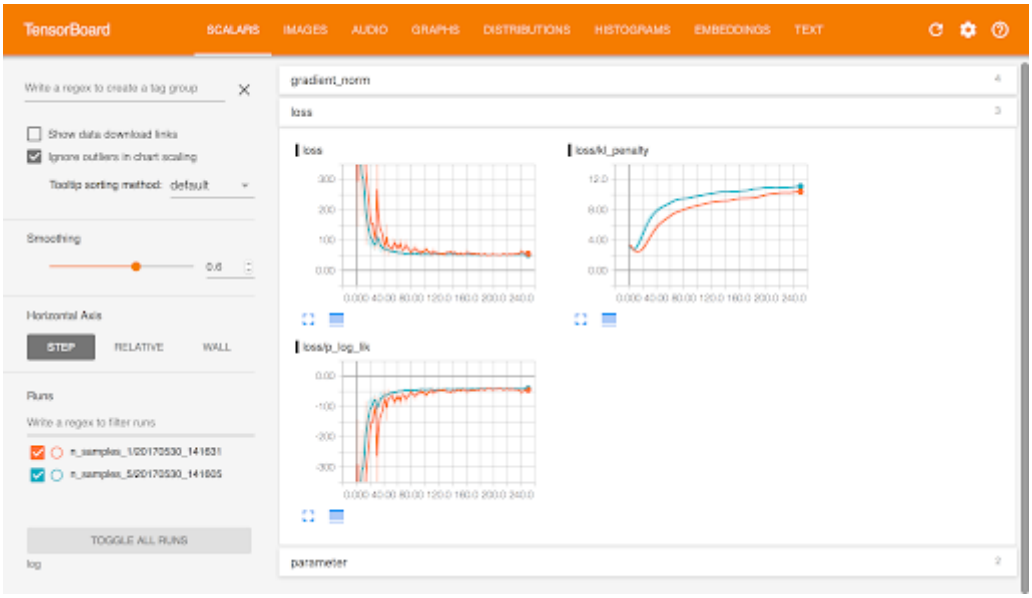


FIGURA 17 - ALCUNE SESSIONI DI TEST SU TENSORBOARD

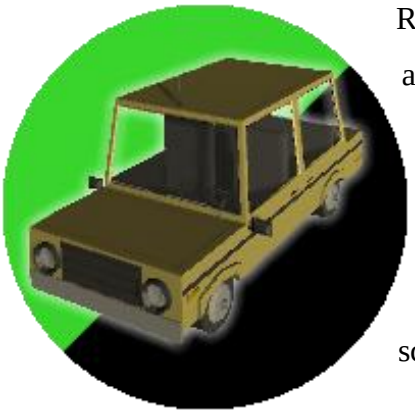
CAPITOLO 3 – PROGETTAZIONE E SVILUPPO DEL PROGETTO

Il terzo capitolo del documento si incentra sulle caratteristiche salienti di progettazione e implementazione del progetto di tirocinio AI Car Kineton che simula, tramite l'utilizzo della libreria ML Agents, l'automazione di un'autovettura dotata di sensori ADAS in corrispondenza di un attraversamento pedonale e in una scena di parcheggio, in particolare in questo capitolo si spiegheranno nel dettaglio i meccanismi di progettazione e realizzazione della scena di attraversamento pedonale (La scena di parcheggio, viene illustrata e trattata nel dettaglio nel documento di tesi "Simulazione di assistenza al parcheggio tramite Deep Learning" a cura di Francesco Abate). Al fine di garantire una panoramica completa e minuziosa sull'intero iter di sviluppo, questo

capitolo sarà diviso in differenti sottosezioni, ognuna delle quali tratterà di uno specifico aspetto del progetto sviluppato, entrando nel dettaglio di quesiti come:

- Perché questo particolare aspetto è utile ai fini della simulazione?
- Quali scelte sono state prese per realizzarlo?
- Quali sono stati i meccanismi implementativi utilizzati?

3.1 Progettazione e Realizzazione dell'environment



Realizzare un progetto di apprendimento automatico in ambiente 3D necessita di scelte molto chiare e precise, per garantire la buona riuscita dell'addestramento e il raggiungimento degli obbiettivi prefissati è stato necessario fin da subito cominciare a porsi qualche domanda di incipit per progettare l'environment di scena, ad esempio:

- In che modo l'autovettura dovrà muoversi all'interno della scena?
- Qual è l'obbiettivo principale da raggiungere?
- Con quali e quanti ostacoli, l'agente dovrà interfacciarsi?
- Quale sarà il grado di autonomia dell'autovettura?
- Quali, invece, le restrizioni da apportare al fine di garantire la buona riuscita dell'addestramento?

Anche se tali quesiti possono sembrare banali, al fine di realizzare i movimenti e le reazioni di una semplice autovettura che dovrà muoversi in traiettoria lineare e frenare non appena avvista un pedone, c'è comunque da considerare che far apprendere anche il più piccolo movimento tramite apprendimento per rinforzo, non è un'operazione banale, perché come già anticipato precedentemente l'agente, non appena viene catapultato all'interno della simulazione, non ha nessuna conoscenza dell'ambiente circostante, nessuna informazione pregressa gli è stata fornita, e di conseguenza anche

la più scelta può essere di rilevante importanza al fine di determinare la buona riuscita di un addestramento.

Per rispondere a questi quesiti le chiavi di lettura sono stati essenzialmente i concetti di semplicità nell'ambiente circostante all'agente, e di predisposizione alle dinamiche di addestramento per difficoltà incrementali. Sotto tale ottica sono stati prima di tutti determinati, in modo accurato e schematico quali elementi includere in scena, e quali interazioni essi dovessero avere tra di loro:

- Strada urbana con attraversamento pedonale, su cui far muovere auto e pedoni;
- Auto: (Agente da addestrare), che in maniera autonoma percorre la strada fino ad un punto di fine, e in corrispondenza dell'attraversamento evita l'incrocio con i pedoni;
- Pedone: componente programmata che attraversa in modo continuo l'attraversamento pedonale da un lato all'altro ciclicamente, e si pone al passaggio dell'autovettura, davanti a quest'ultima per stimolarla alla frenata;
- End Game Point: Punto di arrivo dell'autovettura per terminare correttamente la simulazione
- Walk Limit Wall: muri di delimitazione di corsia per garantire i corretti movimenti dell'auto.

Di seguito viene riportato schematicamente come gli oggetti saranno poi disposti tra di loro e un accenno di una corretta interazione tra gli stessi:

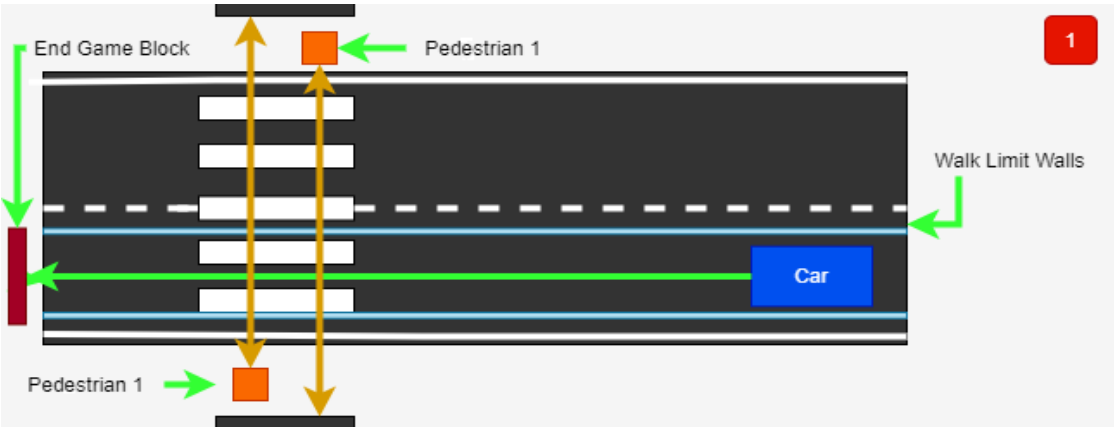


FIGURA 18 - AUTO IN MARCIA VERSO L'END GAME ALL'INTERNO DEI WALK LIMIT WALLS

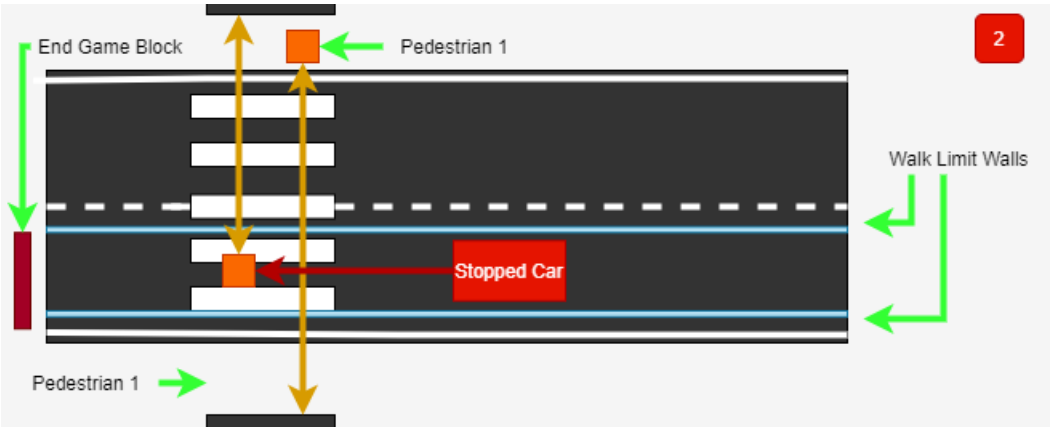


FIGURA 19 - AUTO FERMA IN CORRISPONDENZA DELL'ATTRAVERSAMENTO, CHE ATTENDE IL PASSAGGIO DI UN PEDONE

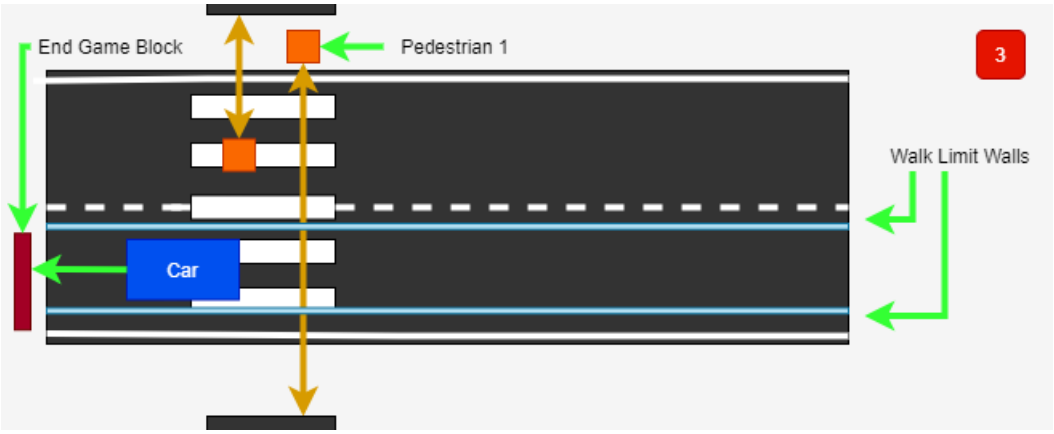


FIGURA 20 - AUTO CHE RIPRENDE CORRETTAMENTE LA MARCIA DOPO IL PASSAGGIO DEL PEDONE

Dopo la progettazione degli elementi di scena ed un minimo accenno alle loro interazioni, si è passato alla realizzazione della scena in Unity.

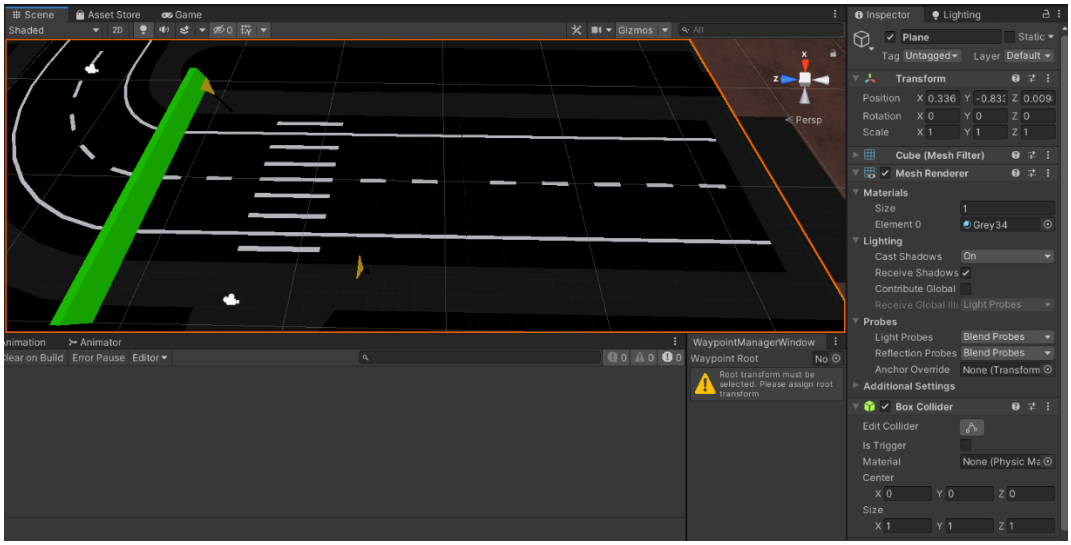


FIGURA 21 - AICAR KINETON - SCENA DI ATTRAVERSAMENTO 1 – PRIMA COSTRUZIONE

Come primo step, è stato realizzato il primo tratto stradale su cui successivamente sarebbe avvenuto l'addestramento dell'autovettura, a tale scopo è stato usato un pacchetto di asset free per la realizzazione di un semplice ambiente per la circolazione stradale in grafica low poly scaricabile dall'asset store. Nella sua semplicità, il primo environment permette di addestrare il veicolo in maniera agevole e progressiva, garantendo la possibilità di imparare la corretta traiettoria dall'inizio della strada, passando per l'attraversamento pedonale, fino al raggiungimento del blocco di demarcazione di fine strada in verde (End Game Object), con la possibilità di incontrare durante il tragitto, un numero molteplice di pedoni.

Altra caratteristica di rilievo dell'ambiente di scena è la scalabilità dato che è stata lasciata la possibilità di introdurre complicazioni di vario tipo in simulazione, dall'introduzione di un singolo pedone in movimento vicino alle strisce ad un numero di pedoni superiore, dalla viabilità in un singolo senso di marcia alla circolazione più realistica in entrambi i sensi di percorrenza della strada, fino alla presenza di attraversamenti pedonali multipli.

Inseguito sono stati aggiunti alcuni edifici decorativi al fine di rendere l'ambiente di scena più realistico e simile ad un vero e proprio centro cittadino e sono state introdotte camere multiple in corrispondenza dell'attraversamento pedonale o in punti diversi della

strada, per successivamente osservare al meglio i comportamenti dell'agente in scena da differenti angolazioni.

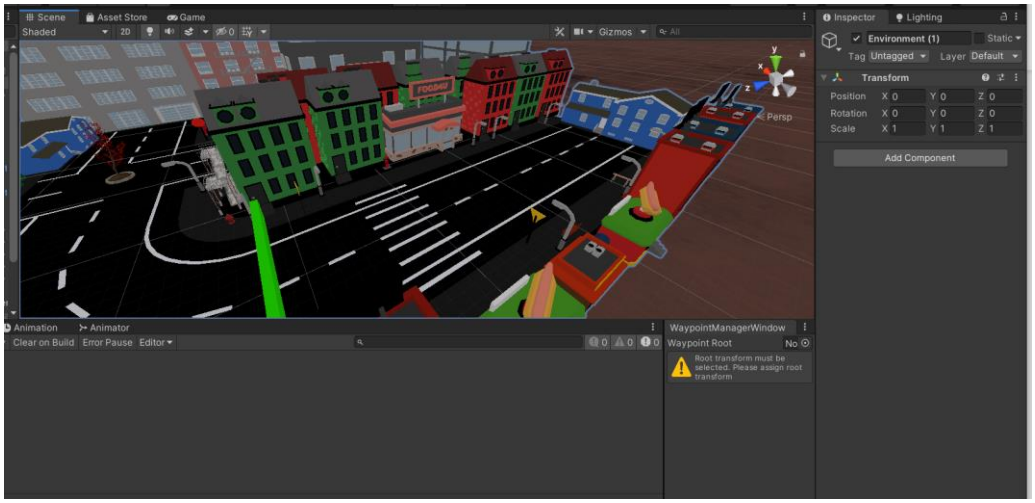


FIGURA 22 - AICAR KINETON - SCENA DI ATTRAVERSAMENTO 1 – AGGIUNTA DEGLI EDIFICI

3.2 SISTEMA DI TRAFFICO PEDONALE

FIGURA 23 - CHARACTER ELF IN IDLE

Il primo vero problema da affrontare, dopo la predisposizione dell'environment, è stato quello di creare e gestire in maniera ottimale il traffico pedonale. Al fine di simulare il comportamento del tipico pedone su strada, si è scelto di utilizzare il modello tridimensionale di umanoide "Character ELF", scaricabile gratuitamente dall'Asset Store di Unity e molto versatile per l'applicazione di varie componenti Unity per l'animazione della Mesh. Il modello, infatti, possiede un Animator Controller già integrato, e risponde bene a differenti animazioni della componente Transform di un generico Game Object in Unity, simulando movimenti molto realistici, tra cui corsa, camminata, salti o anche semplice stato di IDLE. Tale Mesh inoltre è un ottimo compromesso tra semplicità e funzionalità, ad esempio è molto versatile anche per

l'applicazione della componente RagDoll, per la gestione realistica di collisioni con altri game object, tramite la generazione automatica di collider, Rigidbody e giunti.

Al fine di animare e far seguire una traiettoria fissa al prototipo di pedone, sono state applicate due animazioni per i personaggi in terza persona presenti nel pacchetto "Standard Asset" di Unity:

- Idle Animation: che ferma la mash del personaggio in un semplice stato di attesa;
- Walk Animation: la quale anima il personaggio, facendolo camminare in traiettoria lineare rispetto ai suoi parametri di posizione e rotazione.

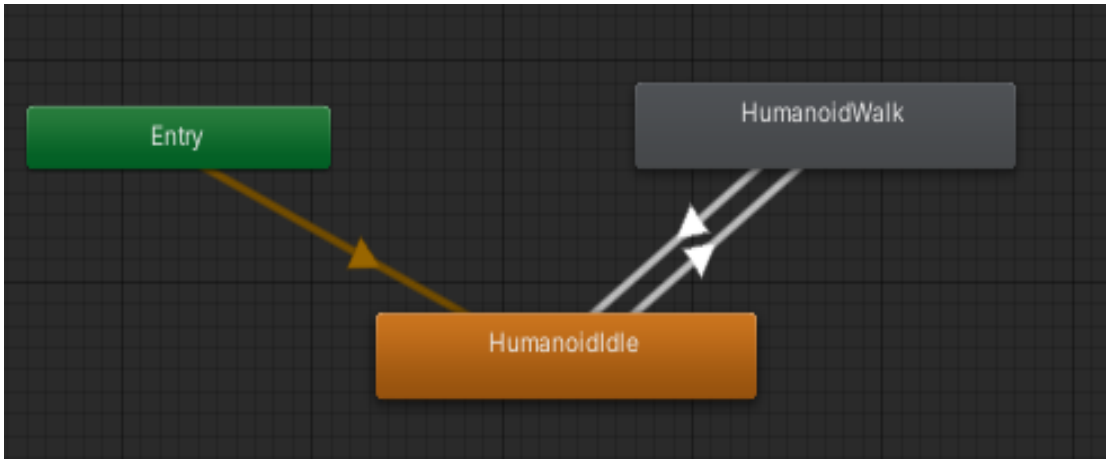


FIGURA 24 - ANIMATORE PER I MOVIMENTI PEDONALI

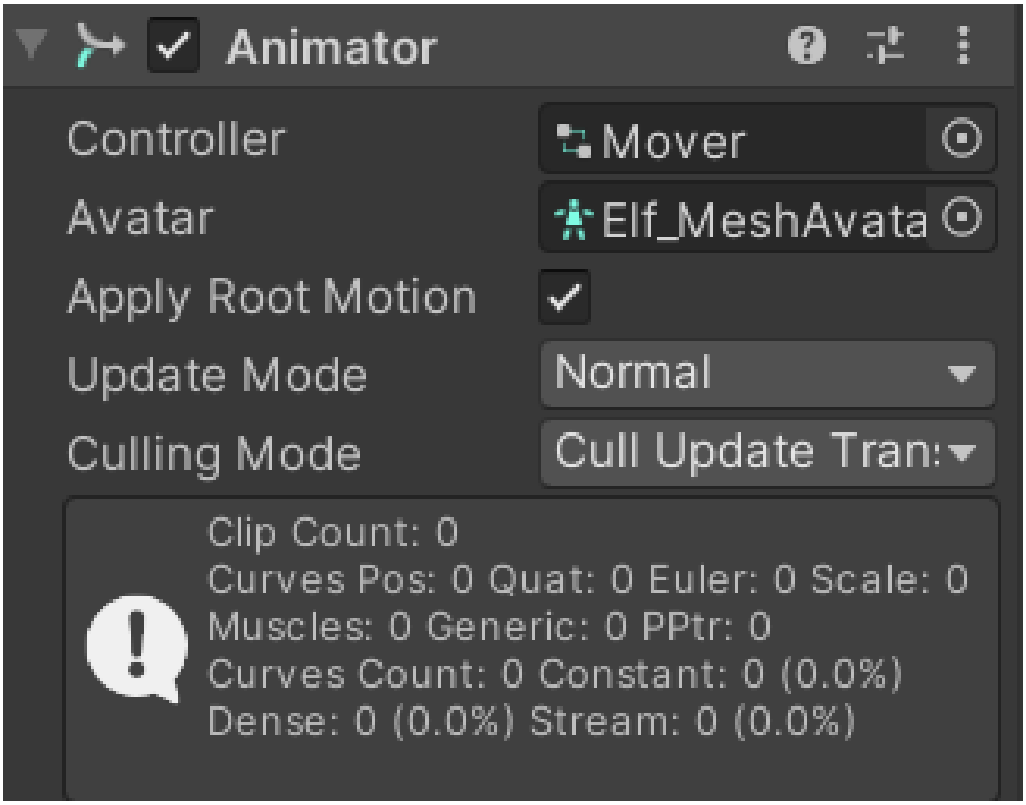


FIGURA 25 - ANIMATOR CONTROLLER DEL SINGOLO PEDONE

Le due animazioni sono state connesse tra di loro tramite il meccanismo dell'Animator di Unity, secondo il paradigma di una generica macchina a stati (dove ogni singola animazione rappresenta uno stato) interconnessi da transizioni, il passaggio di transizione dallo stato di Idle a quello di Walk o viceversa, viene effettuato semplicemente tramite l'utilizzo di un fleg booleano che determina lo stato di Walk come stato attivo (true), e lo stato di Idle, come stato passivo (false).

Come determinabile dagli schemi di progettazione, si capisce facilmente che il pedone nel muoversi ha la necessità di seguire una precisa traiettoria da percorrere per tutta la

durata della simulazione di gioco, a tale scopo è stato implementato un sistema di waypoints a lista concatenata per ogni pedone.

Via script sono stati definiti i Gizmos per la modellazione dei singoli waypoint e per i collegamenti tra gli stessi, in modo da avere una visione grafica e precisa della traiettoria che il pedone dovrà seguire durante l'intera durata della simulazione.

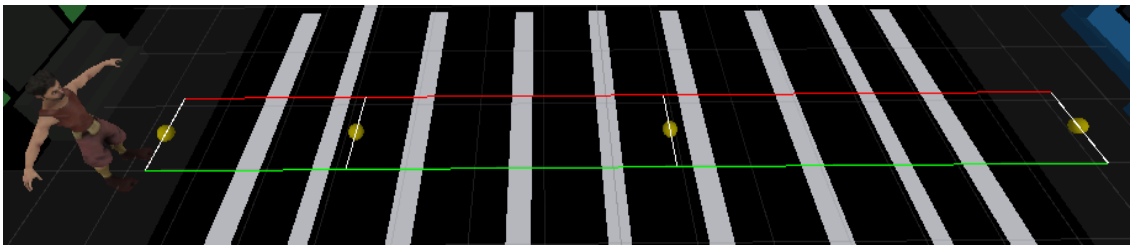


FIGURA 26 - PEDONE IN CORRISPONDENZA DI PEDONALE CON LISTA DI WAYPOINTS ASSOCIATA

Il funzionamento del meccanismo di traiettoria in Waypoints per il movimento del pedone è stato implementato via Script e può essere così riassunto:

- La testa della lista di way points associati al pedone viene agganciata ad esso via script;
- Primo passaggio, fatto non appena viene avviata la simulazione, dallo script di navigazione è far passare il pedone, posto inizialmente nello stato di Idle, allo stato di Walk,

- Ad ogni frame lo script controlla la position e la rotation del che vengono conseguenzialmente modificate in modo tale da raggiungere il waypoint attualmente puntato;

- Quando la distanza tra pedone e waypoint puntato è sufficientemente approssimabile allo zero, lo script di navigazione dei prevede a sostituire il waypoint puntato con il successivo, in modo tale procedere avanti nel cammino;

- Al termine della lista, quando il pedone ha raggiunto l'ultimo waypoint, lo script comincia a scorrere la lista in modo opposto, in modo tale da riportare il pedone in corrispondenza del primo waypoint per così continuare fino all'arresto della simulazione.

Tale sistema di movimento per i pedoni risulta molto flessibile, in quanto la singola traiettoria, può essere progettata nei modi più differenti possibili, con la possibilità di adattarsi a pieno all'ambiente di scena sia se si vuole predisporre uno o più waypoints in un tratto lineare o ricurvo. Inoltre, la lista di way points introdotta può essere trattata come un normalissimo Game Object, duplicata senza complicazioni ed ogni copia può essere predisposta al passaggio di pedoni differenti in maniera indipendente dalle altre.

3.3 MODELLO D'AUTO E MOVIMENTI

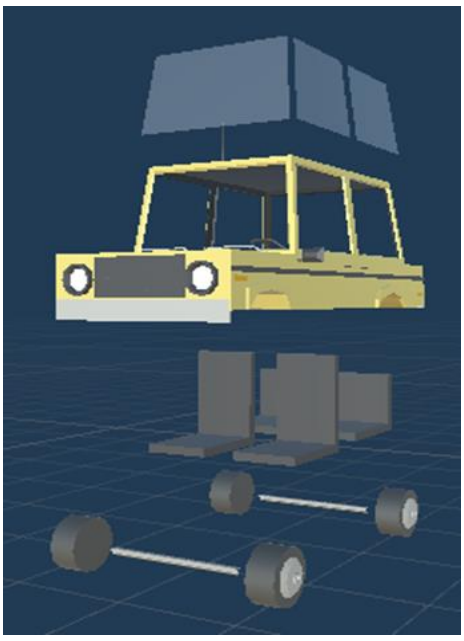


FIGURA 27 - STRUTTURA AUTO

Al fine di realizzare la simulazione il corretto comportamento di un'autovettura all'interno della scena di attraversamento pedonale con pedoni predisposta, si è reso necessario predisporre un modello di auto in grado di muoversi in modo autonomo, simulando i normali movimenti di un veicolo su strada.
A tale scopo è stato utilizzato un modello molto semplice di Autovettura, scaricabile come altri asset utilizzati in modo gratuito dall'Asset Store di Unity. Come osservabile in figura le

componentistiche del modello sono molto semplici.

Il modello è dotato di una struttura esterna, di sediolini, di finestrini e quattro ruote poi animate per l'implementazione dei movimenti dell'agente. Una piccola decorativa aggiunta apportata al modello sono state le luci posteriori per segnalare il corretto utilizzo. Per la visione del veicolo in movimento alle camere di scena, è stato poi aggiunto un ulteriore set di camere per permettere all'utente di visualizzare la scena di attraversamento, da diverse prospettive, tra cui la visione in prima o terza persona, la visuale laterale rispetto all'auto o la visuale dall'alto.

Le ruote dell'auto sono state dotate di Wheel Colliders, componenti Unity capaci di simulare le reazioni dell'auto al sistema fisico di Unity, evitando quindi un onere molto complesso allo sviluppatore. Come movimenti dell'auto sono stati previsti i movimenti di marcia in avanti, retromarcia, frenata e di sterzata verso destra o sinistra. Per l'implementazione dei movimenti e quindi del comportamento generale del veicolo, è stato dapprima previsto uno script di tipo MonoBehaviour per definire ed ottimizzare i mezzi del veicolo manualmente, convertito poi in script di tipo Agent per l'addestramento automatico del veicolo.
I movimenti nello script per il controllo manuale del veicolo sono stati definiti tramite i seguenti comandi:

- Movimento Verticale in avanti tramite la pressione del tasto "W";
- Movimento Verticale in retromarcia tramite la pressione del tasto "S";
- Movimenti di sterzata del mezzo tramite la pressione dei tasti "A" e "D";

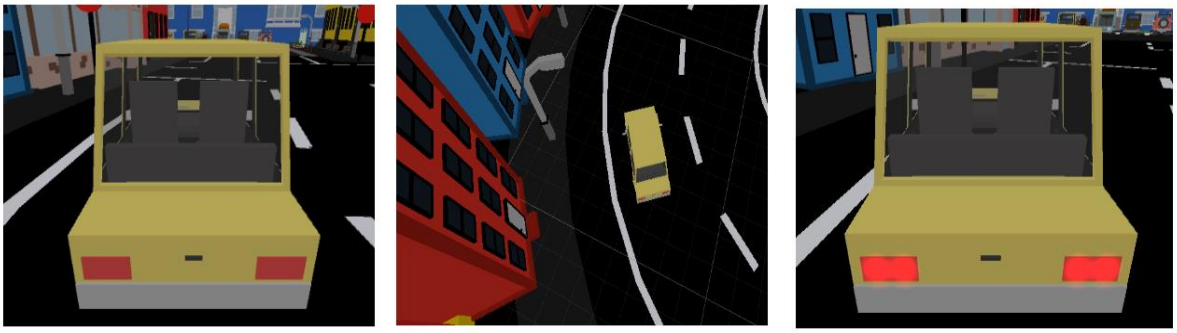


FIGURA 28 - MOVIMENTI GENERALI DELL'AGENTE (IN AVANTI, IN CURVA E IN FRENATA)

- Frenata del mezzo tramite la pressione del tasto “Q”.

3.3.1 SOTTOSISTEMI DI SEMPLIFICAZIONE DI GUIDA

Al fine di garantire un’esperienza di guida più realistica e agevole, sia per la guida manuale del veicolo che per quella automatica, l’auto è stata dotata via script di due sottosistemi, trasparenti all’utente che influiscono non di poco sulla traiettoria di strada. Il primo sottosistema per il controllo di velocità in curva (che ha assunto il nome formale di Steering Control) obbliga il mezzo ad attivare in automatico i freni, se l’auto si attesta ad entrare in curva a velocità molto elevate, in modo tale da evitare la fuoriuscita dalla corsia, per l’apprendimento della tenuta di strada in un tratto lineare, in particolar modo nella scena di attraversamento. Questo sottosistema è stato di vitale importanza dato che

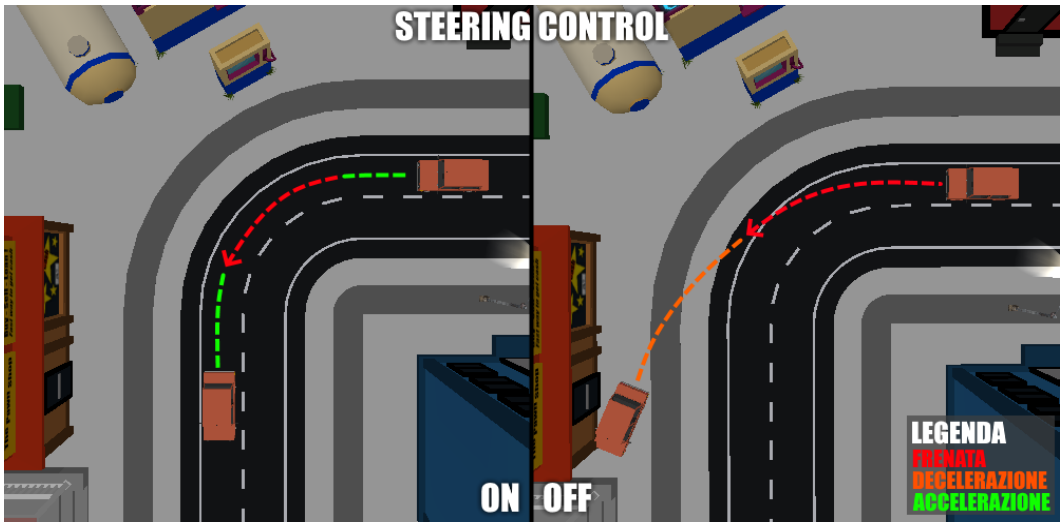


FIGURA 29 - STEERING CONTROL IN CURVA

esso limita di molto la possibilità di sbandamento ed uscita dalla corsia. La velocità a cui il sistema di Steering Control è impostabile manualmente tramite Inspector di Unity, in modo tale da poter garantire la sua messa in esercizio nelle più svariate situazioni, ad esempio tenendo conto delle velocità diverse che un veicolo raggiunge a seconda se si trovi su strada urbana o extraurbana.

Dato che l’utilizzo dei Wheel collider non garantiscono un cambiamento del senso di marcia molto agevole quando l’auto non è ferma, allora è stato progettato un secondo sottosistema di assistenza che provvede in tale situazione ad abbassare la velocità

dell'autovettura in corsa, fino a farla raggiungere rapidamente zero e ad invertire il senso di marcia delle ruote non appena ferme, rendendo l'esperienza di guida ancora più confortevole.

3.4 AUTOVETTURA COME AGENTE

Dopo aver terminato l'implementazione dei movimenti dell'autovettura, la stessa è stata convertita in agente da addestrare nelle successive fasi progettuali. Predisporre l'auto all'addestramento tramite rinforzo, è stata forse la procedura più complessa durante la restante parte dell'esperienza di tirocinio, infatti trovare il giusto equilibrio tra Reward, sensoristiche ReyCast e movimenti corretti del mezzo, è stata una procedura con una forte variabilità, molto complessa da monitorare, ma necessaria al fine di arrivare ad un compromesso accettabile per avere esiti di training ottimali.

3.4.1 SENSORI ADAS SIMULATI CON TECNOLOGIA RAY CAST

Come già introdotto nel primo capitolo, al fine di automatizzare in modo corretto il comportamento dell'auto in una scena di attraversamento pedonale, è stato necessario riprodurre nell'ambiente di scena l'utilizzo di moderni sensori ADAS per il rilevamento dei limiti di corsia e per il rilevamento dei pedoni. A tale scopo è stato molto utile l'utilizzo della componente **RayPerseptionSensor 3D,** introdotta in Unity tramite l'inculusione della libreria di supporto MLAgents, la quale, tramite un sistema RayCast, aumenta in modo considerevole il numero di osservazioni che un agente può rilevare, in sinergia con l'utilizzo della componente VectorObservation, in modo da produrre un'analisi molto precisa e accurata dell'environment circostante in fase di addestramento.

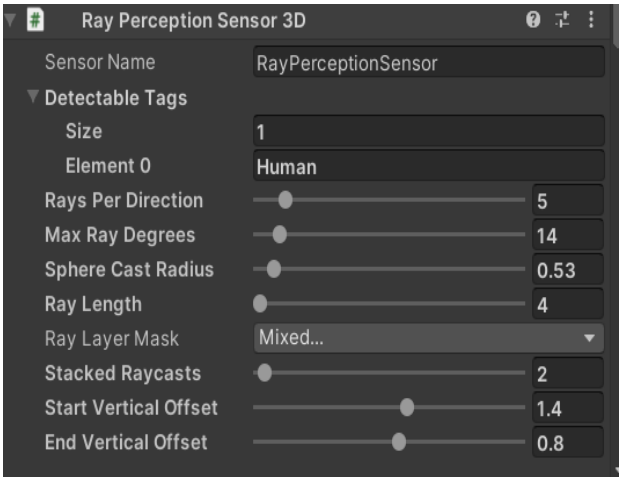


FIGURA 30 - RAY PERCEPTION SENSOR 3D INSPECTOR VIEW

Come visibile dalla Figura 30, il RayPerseptionSensor3D permette una personalizzazione molto elevata delle sue caratteristiche, in particolare parametri che sono stati ampliamente modificati durante l'intero iter progettuale sono stati:

- Detactable Tags: i quali indicano quali GameObject (o gruppi di Game Object) marcati con uno dei tag presenti nella lista, devono essere osservati dal fascio di raggi in fase di addestramento;
- Rays Per Direction: direttamente correlati al parametro superiore, indicano il numero di raggi ottimale al fine di garantire un'analisi ottimale di tutti i game object specificati. Questo parametro necessita di specifici accorgimenti e deve essere posto sempre in giusta proporzione già dall'inizio dell'addestramento, in quanto anche una piccolissima modifica comporta la necessità di ricominciare l'addestramento da zero, in quanto il numero di raggi dei singoli RayPerseptionSensor3D e le dimensioni delle componenti osservate nel VectorObservation, sono parametri di base fondanti per le reti neurali prodotte in fase di training;
- MaxRayDegree: indica l'ampiezza del fascio di raggi (da 0 a 360 gradi), la quale incide molto sulla qualità del sensore simulato, infatti con ampiezze differenti il comportamento dell'agente, nelle successive fasi di training, ha assunto notevoli variazioni, nonostante la presenza di reti neurali derivanti da precedenti training;

- Ad esempio, per il rilevamento di pedoni, un'ampiezza ottimale è stata riscontrata, quando il parametro è stato posto a 14, creando un ventaglio di visibilità amplio quanto tutto lo spettro della corsia, in modo tale da avere una ampiezza tale da rendere possibile l'attraversamento ottimale del pedone in corrispondenza di tutto il perimetro frontale dell'autovettura, mentre un caso totalmente opposto, con ampiezza di 180 (intero perimetro dell'auto), è stato quello del sensore di parcheggio, al fine di garantire l'intera visibilità dell'area di parcheggio.

- Ray Length: la lunghezza del singolo raggio, è un altro parametro che ha inciso molto in fase di addestramento: avere dei raggi sufficientemente lunghi, significa determinare la vicinanza di rilevazione e quindi la rapidità con cui le singole osservazioni vengono elaborate;
- I restanti parametri della componente RayPerceptionSensor che sono stati modificati più volte in corso di addestramento sono stati: SphereCastRadious (indice di sensibilità del singolo raggio, ad ampiezze maggiori corrispondono sensibilità maggiori dei singoli raggi), StartVerticalOffset e EndVerticalOffset, che indicano le angolazioni dei punti iniziale e finali del fascio di raggi.

Ai fini della simulazione, i sensori ADAS per il rilevamento dei pedoni e per la tenuta di strada sono stati modellati ciascuno con una singola componente RayPerceptionSensor 3D (test con più detactable tags, in un singolo fascio di raggi, per la rilevazione congiunta hanno portato a risultati poco soddisfacenti, data la necessità di effettuare rilevazioni diverse e mirate per le singole esigenze).

In particolare, lo schema ottimale del sensore per la tenuta di strada consiste in un ventaglio di raggi posto in modo tale da non superare l'altezza dei Walk Limit Walls, con una lunghezza e ampiezza di sfere tale da permettere preventivamente all'auto di analizzare la posizione dei muri rispetto alle ruote, e ruotare le stesse in modo tale da uscire dalla traiettoria di collisone con il muro, e mantenere un andamento lineare verso l'end game.

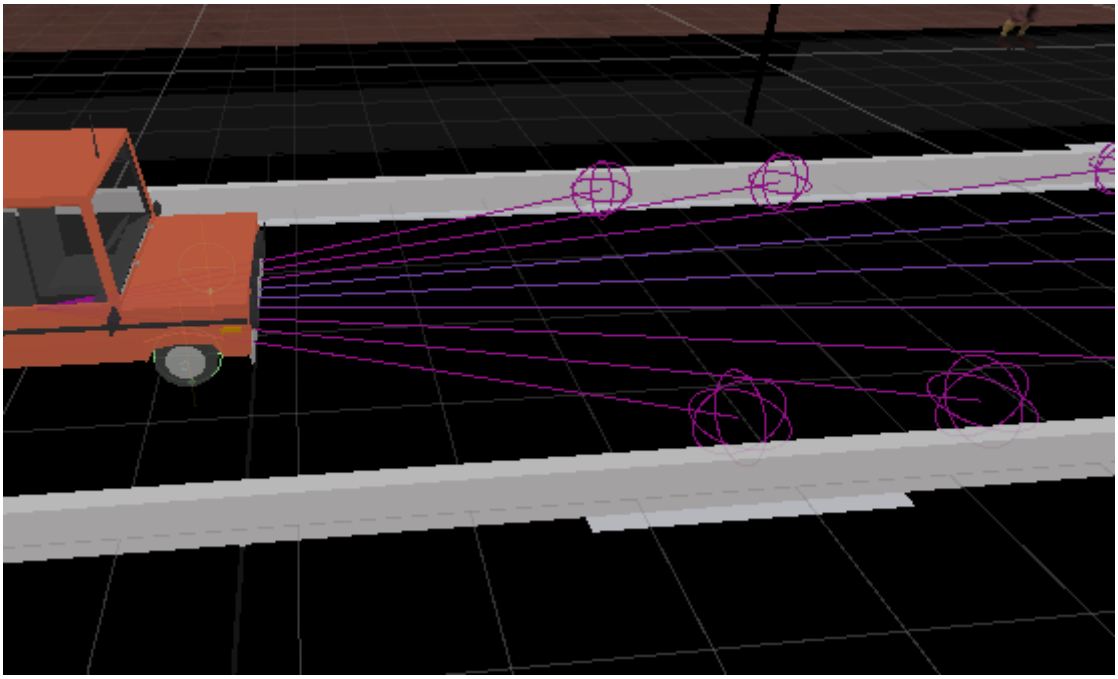


FIGURA 31 - FASCIO DI RAGGI PER RILEVAZIONE DEI LIMITI DI CORSIA E ANALISI DI TRAIETTORIA

Invece il sensore per la rilevazione dei pedoni ha assunto un comportamento ottimale venendo progettato come un fascio di raggi posto dall'alto verso il basso, in modo da simulare l'angolazione visiva del conducente con tali caratteristiche:

- Un'ampiezza di copertura poco più grande dello spazio compreso tra i limiti di corsia;
- Un'ampiezza delle sfere ottimale è stata posta su un valore medio alto, in modo tale da avere visione dell'intero spostamento del pedone davanti alla macchina, evitando la possibilità di far percepire all'auto la possibilità di percepire spazi vuoti data la presenza di troppi raggi liberi, e consequenzialmente innescare movimenti erronei.
- Lunghezza dei raggi di media grandezza, realizzata sfruttando le potenzialità dell'environment di scena.
    - In particolare, l'asset di attraversamento pedonale utilizzato, prevede la presenza di un cartello di pericolo poco distante dalle strisce pedonali (distanza approssimabile a 1.5 metri nella realtà), data tale agevolazione, il fascio di raggi è stato reso abbastanza lungo da poter avere una corretta

visione dell'intero tratto di attraversamento (intera ampiezza delle strisce) partendo appunto dalla posizione del segnale.

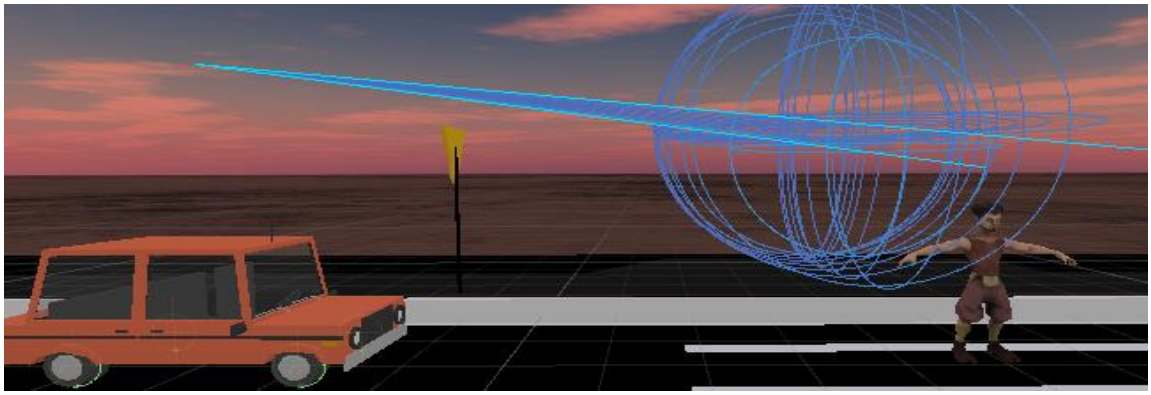


FIGURA 32 - RILEVAZIONE E FRENATA OTTIMALE SU VISUALE LATERALE

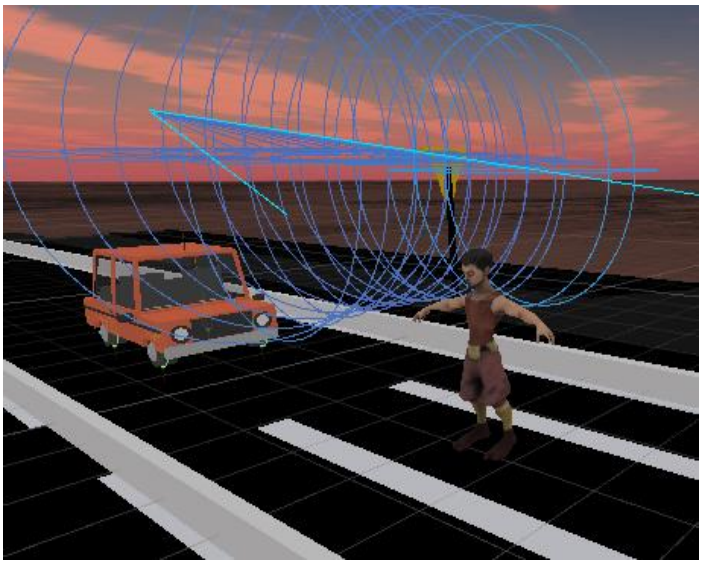


FIGURA 33 - RILEVAZIONE E FRENATA OTTIMALE SU VISUALE FRONTALE

A scopo migliorativo, è stato aggiunto nei test conclusivi dell'iter progettuale anche un sensore per la percezione dell'End Game point, nella sezione dedita ne verranno riportati i dettagli.

3.4.2 PRINCIPI BASILARI PER L'ASSEGNAZIONE DI REWARD

Gestire i Rewards per un addestramento di Reinforcement Learning, in una scena ad alta variabilità con componenti differenti, tra cui alcune di esse addirittura in costante movimento, può diventare un qualcosa di molto complesso se non si progettano meccaniche di analisi che sappiano intercettare costantemente le singole interazioni tra

gli agenti e gli oggetti di scena, ed assegnano ricompense e penalità in modo opportuno discriminando correttamente un'interazione positiva da una negativa, tenendo anche conto del fattore di variabilità che la scena possiede data la presenza di altri oggetti dinamici oltre l'agente.

In particolar modo tale problematica nella scena di attraversamento pedonale, se pur non con elevata difficoltà di gestione, la si può riscontrare pensando a come l'agente debba comportarsi rispetto ai pedoni in movimento sulle strisce pedonali. In condizioni di addestramento normali, al passaggio dell'auto sulle strisce, possono verificarsi due circostanze molto differenti che incidono su come l'agente debba prendere decisioni:

- Una prima circostanza, più agevole, avviene quando le strisce pedonali sono libere e l'agente può continuare la sua marcia indisturbato;
- Un'ulteriore eventualità si verifica quando uno o più pedoni possono trovarsi in traiettoria di collisione con l'auto al passaggio di quest'ultima vicino alle strisce. In tal caso al fine di permettere un corretto svolgimento della simulazione, l'agente deve fermarsi non appena il sensore apposito avvista uno o più pedoni e consentirne il passaggio restando fermo.

Determinare i Reward da assegnare all'auto in presenza o in assenza di pedoni sulla sua traiettoria, non è stato un ragionamento veloce, infatti è stato necessario "capire" via script, quali comportamenti dell'auto premiare e quali penalizzare tenendo conto del dinamismo appena descritto. Sicuramente, in primo luogo, sono state penalizzate di parecchio le collisioni tra pedone e agente, in modo tale da addestrare quest'ultimo sotto l'ottica che il Game Object "Pedestrian" non andava in nessun modo toccato, però a tal proposito oltre alla frenata preventiva, una valutazione scorretta che resta possibile all'agente è quella di sterzare in modo erroneo e pericoloso in vicinanza del pedone. Se si pensa a quest'ultima eventualità, in un contesto reale, quindi un sensore ADAS per il rilevamento dei pedoni in modo automatico a cui si lascia la possibilità di sterzare in modo troppo ravvicinato ad un passante, ci si rende subito conto di quanto ciò possa essere pericolosa sia per il conducente, ma soprattutto per l'incolumità dei passanti.

FIGURA 34 - AGENTE CHE DOPO UNA CORRETTA VALUTAZIONE LASCIA PASSARE IL PEDONE FRENANDO

FIGURA 35 - AGENTE CHE DOPO UN ANALISI SCORRETTA TENTA DI SORPASSARE IL PEDONE DAVANTI AD ESSO, RISCHIANDO INEVITABILMENTE LA COLLISIONE

Quindi come fare per "obbligare" l'auto a fermarsi a distanza di sicurezza da un pedone in movimento sulle strisce?

A tal proposito, è stato nuovamente di grande aiuto l'asset di attraversamento utilizzato per l'attraversamento pedonale, sfruttando le caratteristiche della mesh, è stato possibile calcolare un perimetro di sicurezza (considerando come punti di riferimento la posizione

del segnale stradale e la retta perpendicolare alla fine delle strisce), entro il quale l'agente, approssimativamente, inizia a percepire la presenza di pedoni sulle strisce (se presenti) e senza controlli aggiuntivi valuta, a seconda della posizione del pedone rilevato, il comportamento da avere in una delle due modalità soprariportate .

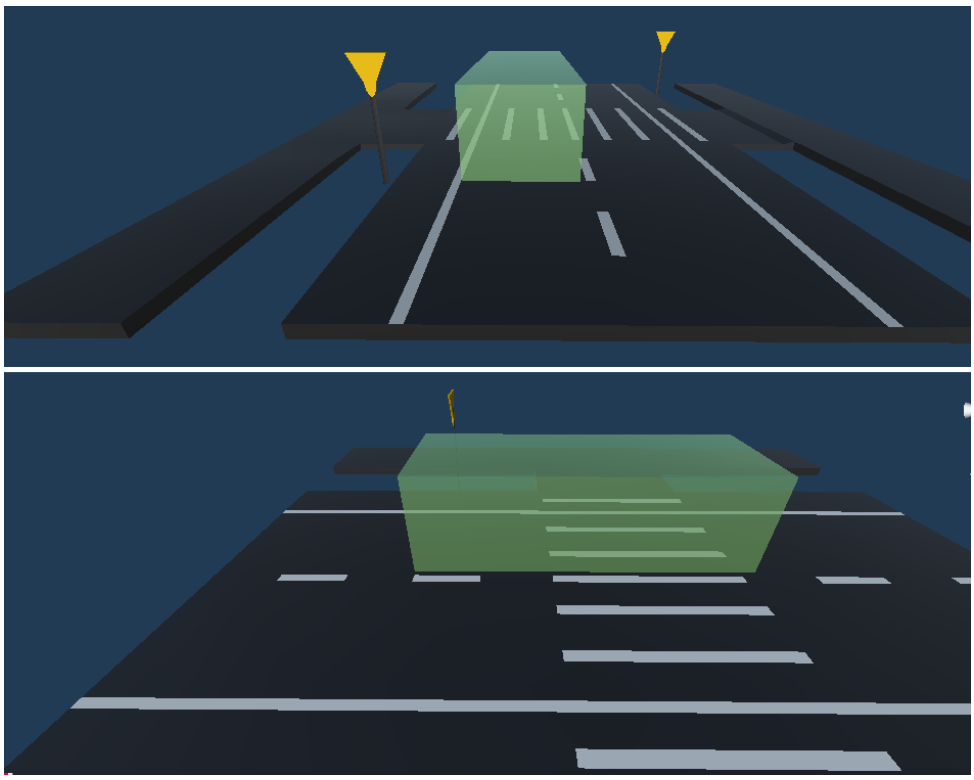


FIGURA 36 – RAPPRESENTAZIONE GRAFICA DEL PERIMETRO DELLA AREA DI "RISCHIO" VISTO RISPETTO AGLI ASSI X E Z

Il perimetro individuato ha un'ampiezza rettangolare che può essere facilmente calcolata in termini di coordinare X e Z rispetto al centro dell'auto, quindi vedendola in termini numerici, quest'area di pericolo astratta, può essere calcolata in termini di distanze tra i singoli pedoni e l'agente, quindi "generata frame per frame" ad ogni spostamento dell'auto, senza creare alcuna dipendenza con le strisce pedonali.
Secondo tale ragionamento nello script per l'agente, sono stati presi dinamicamente i riferimenti di tutti i pedoni presenti in scena, e ad ogni frame, sono stati calcolati due Range di coordinate (per la X e la Z), tramite il quale poter effettivamente svalutazioni differenti e dinamiche sul comportamento dell'auto in presenza o meno di pedoni che intralciano il suo corretto cammino verso l'End Game Object.

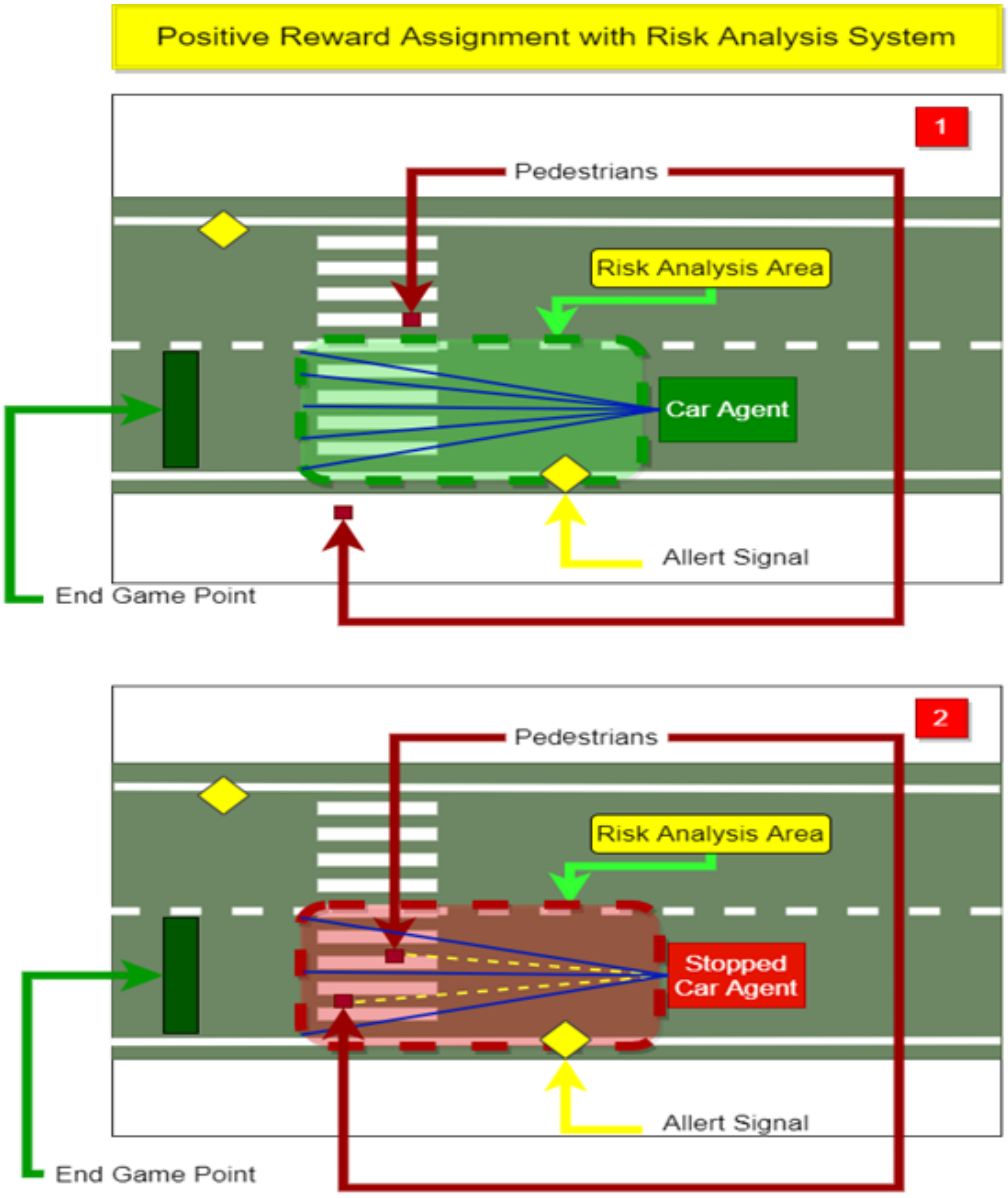


FIGURA 37 - SCHEMA PROGETTUALE PER L'ASSEGNAZIONE CORRETTA DI REWARD IN PRESENZA O IN ASSENZA DI PEDONI

In linea di principio, il funzionamento del sistema di assegnamento di ricompense e penalità dinamiche in termini dell'analisi di potenziali collisioni tra Agente e pedone, è così riassumibile (per semplicità si consideri un singolo pedone in scena, ma il principio il discorso è ampliabile anche ad un numero di pedoni superiore): Considerando al tempo Ti, la posizione dell'auto alla coordinata (xi, 0.1, zi) e la coordinata di posizionamento del pedone (xpi, 0,1, zpi),

- Se $(0 \leq |zi - zpi| \leq 10\ e - 5 \leq |xi - xpi| \leq 5)$ allora
  - la variazione di coordinate tra pedone e auto è troppo ristretta e quindi c'è rischio di collisone se l'auto continua effettivamente a muoversi –

    L'auto acquisisce una o più ricompense nel caso in cui:
    - mantiene una velocità approssimabile allo zero;
    - ha inoltrato correttamente e preventivamente il comando per la frenata;
    - La posizione di arresto è compresa nei limiti della zona di rischio, in particolar modo, quest'ultimo Reward è tanto più grande, quanto più il differenziale $|zi - zpi|$ è vicino al 10.
  - l'auto acquisisce una o più penalità se invece:
    - La sua velocità è superiore a 0 (in particolare maggiore di 0.3), quindi continua a muoversi nonostante la presenza di pedoni;
    - Ruota i Wheel Colliders erroneamente mentre il pedone sta transitando davanti ad esso.

- Altrimenti (nel caso in cui il pedone non sia davanti alla traiettoria dell'auto oppure talmente lontano da non avere il rischio di collisione),
  - L'auto acquisisce ricompense se:
    - Mantiene un andamento costante in direzione dell'End Game Object;
    - Mantiene una traiettoria più simile possibile ad una linea retta tra il centro dell'auto e il centro dell'End Game Object;
  - L'auto acquisisce penalità negative nel momento in cui:
    - Tende a fermarsi erroneamente su strada quando ha campo libero;
    - Mantiene una traiettoria scorretta, rivolta verso i Walk Limit Wals e non rispetto all'End Game Object;
    - Nel caso peggiore collide con un muro di corsia.

Altri piccoli controlli sui movimenti dell'auto sono stati poi aggiunti per migliorarne i movimenti o accelerare le valutazioni in presenza o in assenza di pedoni, per esempio quasi sempre è stata penalizzata la possibilità di andare in retromarcia per questa scena, in quanto in una situazione reale di circolazione urbana, tale movimento non è ammesso, a meno di particolari esigenze.

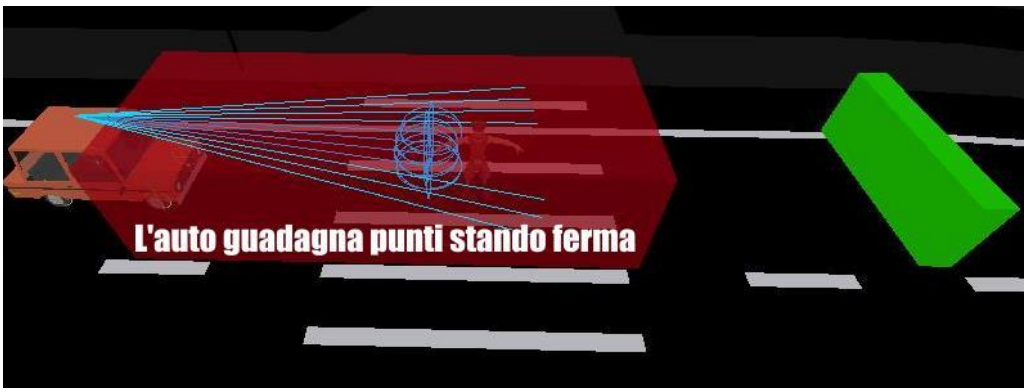


FIGURA 38 - CORRETTO COMPORTAMENTO IN PRESENZA DI PEDONI SULLA TRAIETTORIA DI MARCIA

FIGURA 39 - CORRETTO COMPORTAMENTO IN ASSENZA DI PEDONI SULLA TRAIETTORIA DI MARCIA

La scelta di progettare un sistema di assegnamento per i Rewards così granulare è senz'altro motivata dalle meccaniche generali da un addestramento di Reinforcement Learning, in generale è buona prassi mantenere, durante l'addestramento, il valore medio di guadagno, tra la simulazione in corso e la successiva presso che costante, perché, se l'agente viene eccessivamente penalizzato, tende a dimenticare le precedenti decisioni effettuate, e di conseguenza anche i progressi, se tale valore è troppo alto, allora l'agente tende a lasciare poco spazio a nuove valutazioni, prediligendo azioni che evidentemente avevano comportato troppo guadagno durante l'**episodio** precedente.

***Episodio**: Nel gergo utilizzato dai tool per il Machine Learning (tra cui MLAgents), un episodio di training, consiste nel periodo che intercorre tra l'inizio della simulazione, fino al Reset dell'agente per la simulazione successiva, durante il quale l'agente è libero di osservare l'environment circostante, e testare azioni a seconda dei parametri rilevati.

Considerando tale principio nella scena di attraversamento pedonale, soprattutto rispetto al grande tasso di variazione derivante dal comportamento dei pedoni, gestire i reward con meno granularità sarebbe stato sicuramente di più dispendioso e più difficile da equilibrare. La ricerca di equilibrio tra premi e penalità è stata un delle problematiche principali che ha reso la fase di addestramento una delle più dispendiose e con più alto indice di variabilità per l'intera esperienza progettuale. Infatti, coordinare movimenti corretti in traiettoria rispetto all'End Game Object, evitando di collidere con i Walk Limit Object, e mantenere un comportamento corretto in presenza di pedoni, non è stato un compito molto semplice. Per far assorbire, gradualmente, tali abilità all'agente e migliorare di volta in volta lo script correlato, si è deciso di procedere per step incrementali, ponendo all'inizio alcune semplificazioni nei movimenti o nel numero di pedoni, per poi aumentare man mano che l'agente rispondeva positivamente alle difficoltà trovate nei training precedenti.

CAPITOLO 4 – DIFFICOLTÀ E RISULTATI DEL TRAINING

Avendo ben definito sotto quali basi l'agente avesse dovuto imparare a conoscere ed interagire con l'ambiente di scena, si è passati poi alla definizione delle modalità, e soprattutto del grado di complessità da dedicare ad ogni singolo training di addestramento. Come già anticipato, in questa fase progettuale ha influito molto l'alta variabilità che ha caratterizzato l'analisi dell'agente, dovuta soprattutto alle forti influenze dall'environment e dal comportamento dinamico dei pedoni in scena, infatti

con il susseguirsi dei training in funzione di tale variabilità, la forte alternanza di Rewards positivi e negativi è sempre stata un fattore molto evidente.

Nonostante ciò, con il progredire delle conoscenze a disposizione dell'agente, e soprattutto con il perfezionamento dei valori di premio o penalizzazione assegnati via script, i miglioramenti dell'agente sono stati sempre più evidenti, infatti sarà illustrato anche come, se pur con lievi picchi, l'andamento generale dei training ha comunque riportato un forte andamento positivo.

Ai fini di addestrare correttamente l'agente, l'obbiettivo della simulazione è stato riformulato in logica incrementale, definendo, quali obbiettivi l'agente avesse dovuto raggiungere con il progredire delle fasi di addestramento. Innanzitutto, sono state formulate le azioni necessarie all'agente per sviluppare i comportamenti basilari di un'auto reale, quali:

- Il corretto movimento in avanti su strada urbana lineare;
- Il corretto utilizzo del meccanismo di sterzata per mantenere una tenuta di strada lineare corretta;
- Riconoscimento dei pedoni davanti alla traiettoria di marcia;
- Corretto utilizzo dei freni rispetto ai pedoni.

Successivamente sarebbe stata poi prerogativa dell'addestramento, quella di coordinare correttamente le nuove conoscenze acquisite, al fine di arrivare ad avere un comportamento ottimale con un singolo pedone su strada. In fine, una volta acquisiti i corretti comportamenti in questa situazione semplificata, si è pensato di aggiungere alcune piccole complicazioni al fine di aggiungere maggior grado di realismo alla simulazione, come: la possibilità di movimento in entrambe le corsie in sensi di marcia opposti, la presenza di più pedoni su un singolo attraversamento e la presenza di un doppio attraversamento pedonale per dinamizzare ancora di più le interazioni tra agente e pedoni.

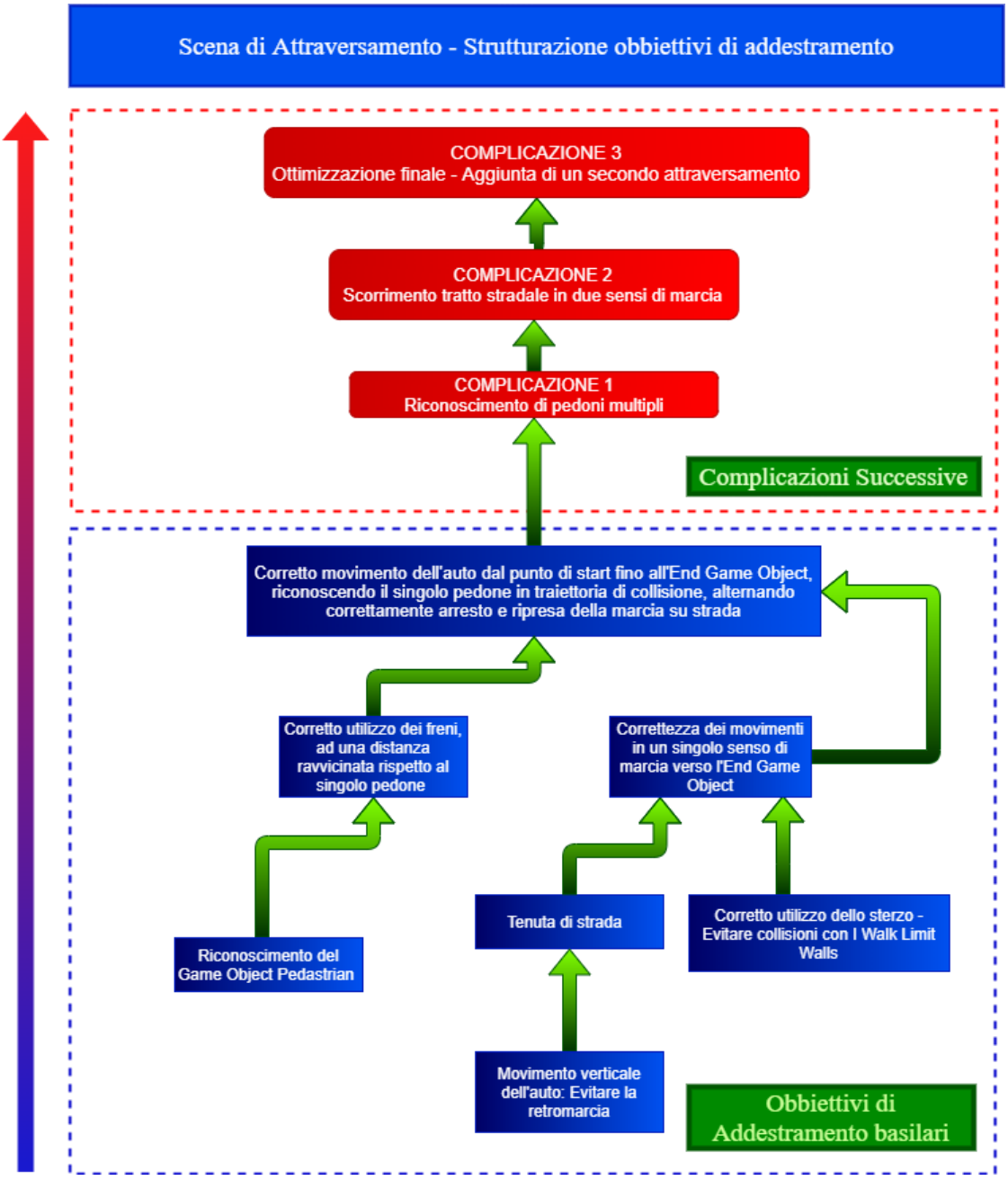


FIGURA 40 – STRUTTURAZIONE PER GLI OBBIETTIVI DI ADDESTRAMENTO

Una volta definiti in modo più granulare gli obbiettivi di addestramento, la sfida più difficile è stata la comprensione di quante complicanze riportare all'interno della scena in ogni fase di addestramento, tenendo conto di quanto l'agente avesse già imparato nelle precedenti sessioni di training, oppure quanto lo stesso fosse in grado di apprendere ripartendo da zero con i test. Sotto quest'ottica il training dell'agente è stato

strutturato in 9 sessioni, ognuna delle quali ha riportato differenti risultati (positivi e negativi). Per le prime sessioni (coincidenti con i delineamenti finali della struttura implementativa del sistema, di cui si è discusso nel capitolo precedente), l'agente è stato addestrato in maniera quasi svincolata, per poi passare ad un'ottica più precisa e mirata al raggiungimento (per incrementi successivi) degli obbiettivi formulati (secondo l'ordine riportato dalla figura 40).

4.1 Sessioni 1 e 2 – Test iniziali

Prima sessione di addestramento

I primi test di training, come già anticipato, sono stati effettuati in maniera quasi contemporanea alla definizione basilare della struttura implementativa, sia in termini di tenuta di strada che per il riconoscimento del pedone. In particolar modo, per la prima sessione, non erano presenti né i Walk Limit Walls né il sistema per il calcolo delle distanze tra agente e pedone (Sistema Risk Zone Analysis, dettagliato nel precedente capitolo), ma erano solo previsti reward di premio per collisione con l'End Game Object e di penalizzazione per collisioni con palazzi e con i pedoni. In tale sessione si lasciava libera l'auto di muoversi su strada e di interagire anche con oggetti in scena di non primario interesse, quali ad esempio gli edifici decorativi.

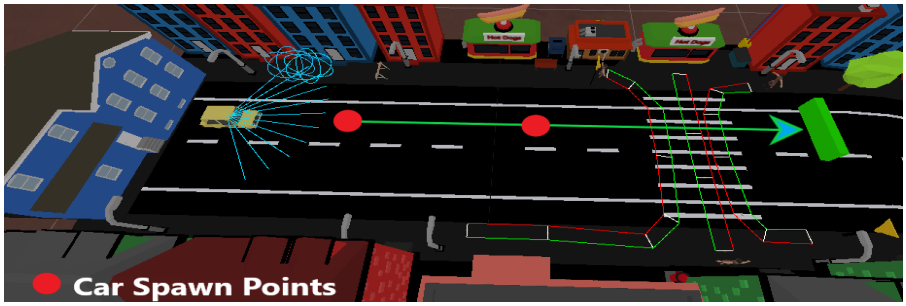


Figura 41 - Elementi in scena durante la prima sessione di addestramento

Comportamenti dell'agente durante la prima sessione di training e prime problematiche

Come visibile dalla figura, in questa prima sessione (così come anche per la successiva), l'auto era dotata di un singolo fascio di raggi, per il riconoscimento generico di un qualsiasi oggetto presente in scena. Non avendo alcuna base da cui partire, ed essendo

quindi l'agente totalmente allo scuro delle fattezze dell'environment circostante, sono stati previsti per la vettura due punti di Spawn randomici, da cui avviare gli episodi di training: il primo ad inizio strada per monitorare come l'auto reagisse in termini di tenuta di strada, il secondo più vicino all'attraversamento, per permettere alla vettura di iniziare a prendere conoscenza dei pedoni.

Tenendo conto che nessuna base preventiva era stata fornita all'agente e che il sistema di Reward ancora troppo impreciso, la prima sessione di Training non è stata delle più soddisfacenti, infatti in ognuno dei tre training avviati l'agente dall'inizio alla fine si muoveva in maniera totalmente casuale, dapprima frenando di continuo apparendo quasi fermo, per poi procedere con l'evitare totalmente l'utilizzo dei freni ed iniziare a muoversi in modo caotico da una corsia all'altra, se non arrivando addirittura fuori strada in collisione con gli edifici. Da notare inoltre che nei primi test l'auto non aveva nemmeno coscienza dei corretti movimenti da fare per il raggiungimento dell'End Game Object, infatti le collisioni con lo stesso sono state molto limitate.

A peggiorare ancora di più la prima sessione, è stata l'interazione con i pedoni, non riuscendo a coordinare i movimenti in modo corretto verso l'End Game Object, l'agente quando veniva a contatto con i pedoni (quasi sempre quando l'episodio di simulazione iniziava dal secondo punto di Spawn), l'auto collideva con gli stessi davvero molto spesso, se ciò non accadeva, il tutto era giustificabile, dal fatto che l'auto, appena resettata, avesse forti incertezze sull'utilizzo dei freni, utilizzandoli in modo improprio e magari inoltrando anche contemporaneamente il comando di frenata e quello per il movimento verticale.

A confermare che l'auto non avesse avuto nessun indice di miglioramento e oltre le cause soprariportate, è soprattutto necessario considerare che l'environment di scena

fosse troppo complesso da far analizzare ad un agente non ancora addestrato, sono stati i report grafici del tool Tensor Board:

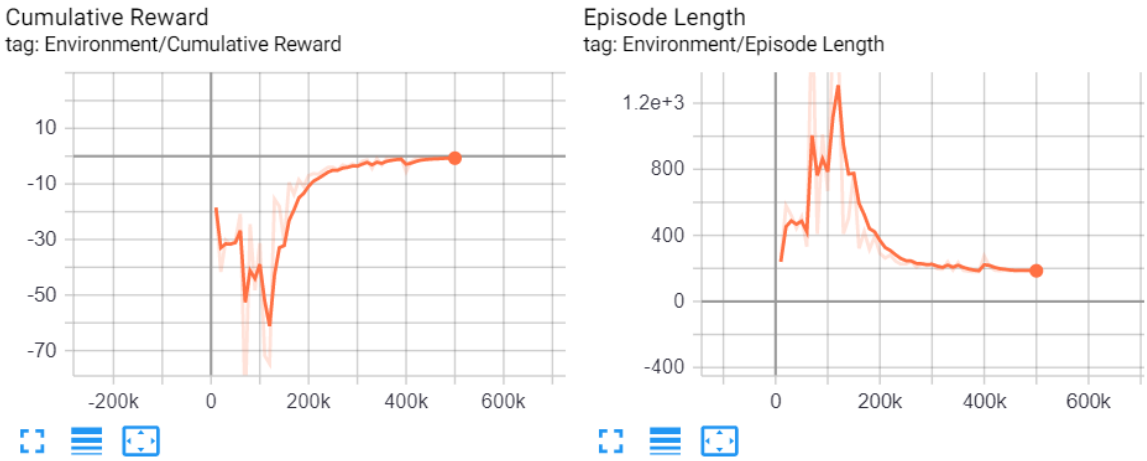


FIGURA 42 - TENSOR BOARD REPORT TRAINING 1_1

per il primo training, come visibile, l'addestramento ha portato ad assegnazioni di Rewards molto caotiche, ed è possibile notare come l'agente venisse molto frequentemente penalizzato dato un utilizzo scorretto dei comandi. Verso la fine del primo addestramento l'auto ha attestato attorno allo 0 i suoi progressi, comportamento che poi ha mantenuto anche per i due training successivi, in effetti l'agente dopo aver esplorato la zona circostante ha continuato la sua analisi cercando di perdere punti il meno possibile, evitando i pedoni o gli edifici, senza effettuare nessun tipo di valutazione ulteriore.

CONSIDERAZIONI ALLA FINE DELLA PRIMA SESSIONE DI ADDESTRAMENTO

Addestrare un agente tramite rinforzo per la prima volta, in particolar modo in una scena non molto semplificata, non può essere fatto partendo da una realtà complessa come quella illustrata, quindi onde evitare nuovamente dei test così caotici e confusionali, è stato necessario introdurre alcuni piccoli accorgimenti che già dalla successiva fase di training hanno portato subito dopo a dei lievi miglioramenti.

Un'ulteriore problematica, già evidenziata dalla prima sessione, che poi ha impattato anche lo svolgersi delle successive alla seconda, è stata la difficoltà dell'agente nel capire come "cambiare comportamento" in vicinanza delle strisce di attraversamento.

SECONDA SESSIONE DI ADDESTRAMENTO

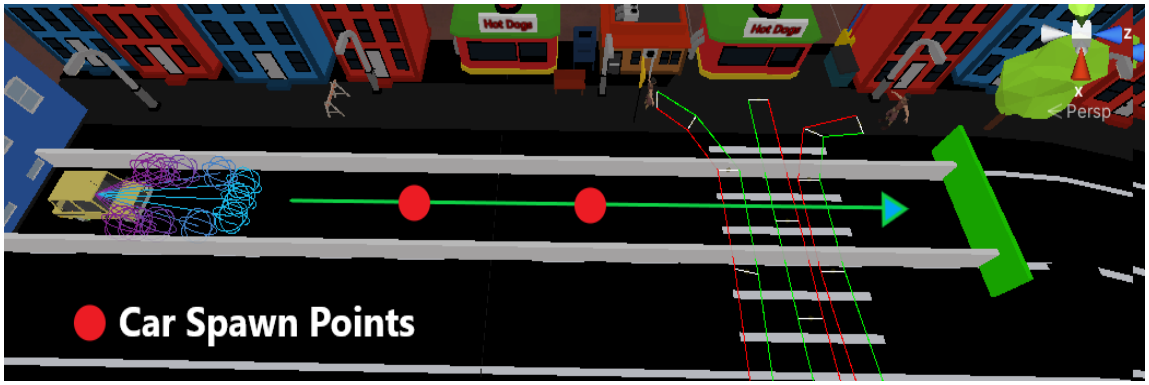


FIGURA 43 - ENVIRONMENT DI ADDESTRAMENTO PER LA SECONDA SESSIONE DI TRAINING

Dopo aver preso atto delle difficoltà riscontrate durante la prima sessione di addestramento, è stato necessario ricominciare il tutto da zero con una nuova chiave di lettura. Seguendo effettivamente, per la prima volta, la logica di semplificazione e decomposizione progettata (Schema incrementale – figura 40), la seconda fase di addestramento è cominciata con l'obbiettivo di addestrare l'agente in modo tale da fargli acquisire innanzi tutto gli elementi basilari al fine della buona riuscita della simulazione, cioè:

- Utilizzare correttamente i movimenti di marcia e di sterzata, al fine di raggiungere l'End Game Object partendo dall'inizio del tratto stradale;
- Imparare a determinare correttamente come frenare e ripartire in modo corretto in seguito alla rilevazione di un pedone sulle strisce pedonali.

Come già anticipato però, il corretto utilizzo di questi due "macro-funzionalità" in maniera combinata, senza basi di addestramento è da considerarsi come uno step successivo, quindi durante la prima metà dell'iter di addestramento si è deciso di considerarli in maniera autonoma.

Al fine di correggere la traiettoria dell'auto, sono stati introdotti per la prima volta nella scena di addestramento i Walk Limit Wall, i quali sono stati resi attraversabile (tramite l'opzione trigger del BoxCollider associato ai Game Object), per permettere comunque ai pedoni di poter attraversare la strada senza intralci. Inoltre, sono state aggiunte, allo script, la penalità di collisione con i muri di corsia, ovviamente per stimolare l'agente a

non sfiorarli e quindi mantenere una traiettoria quasi lineare, tramite Rewards è stato penalizzato l'utilizzo di retromarcia, in quanto tale movimento non è giustificabile in contesti di circolazione reale.

COMPORTAMENTI DELL'AGENTE E CONSIDERAZIONE DELLA SECONDA SESSIONE DI ADDESTRAMENTO

I primi training di questa sessione, nonostante la presenza del Walk Limit Wall, non hanno portato grandi miglioramenti visivi nel comportamento generale dell'auto, però quasi da subito, venendo penalizzata per l'utilizzo della retromarcia, l'auto ha iniziato a muoversi in maniera costante in avanti. Verso metà sessione poi (verso la fine del 3° training, su 6 di cui è composta la seconda sessione), l'agente ha assunto anche la tendenza a non collidere più con i Walk Limit Walls, e ha iniziato a procedere in maniera quasi costante in direzione dell'End Game Object (se pur in maniera molto approssimativa e lontana dal sembrare una traiettoria lineare).

In termini di bilanciamento premi/penalità, la nuova fase di addestramento non ha variato di molto il comportamento riscontrato, nella precedente sessione, infatti

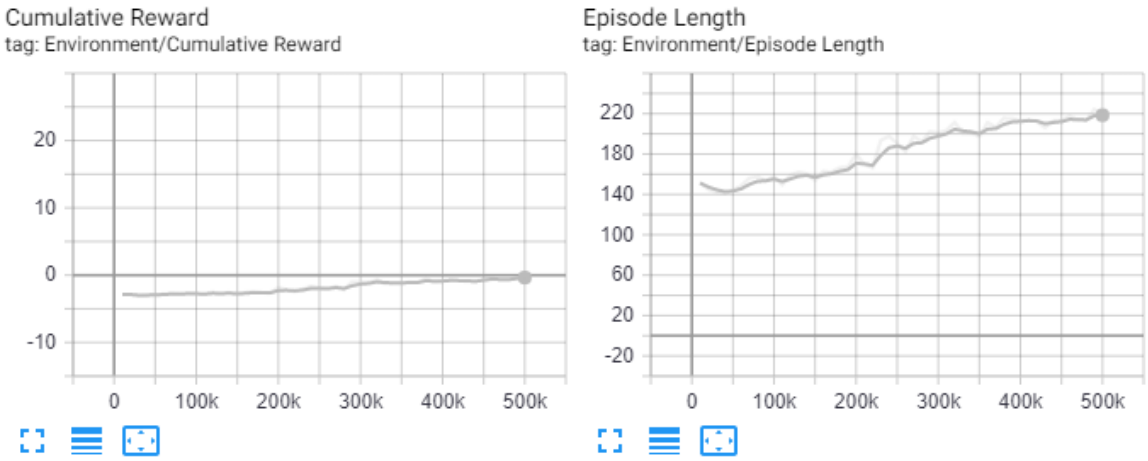


FIGURA 44 - TRAINING 2.4 TENSOR BOARD REPORT

non sono stati riportati grandi cambiamenti in termini di Reward, e come si nota dalla figura, visualmente l'auto (dopo aver imparato a muoversi correttamente in avanti) ha mantenuto un comportamento abbastanza lineare, muovendosi verso il punto di fine simulazione.

Di cruciale differenza rispetto alla prima sessione, sono stati però le risposte avute dall'agente rispetto all'ambiente di scena, nonostante l'auto non avesse ancora alcuna coscienza dei pedoni, e di come interagire con essi, la possibilità di concentrarsi sull'analisi di due singoli GameObject (I Walk Limit Walls), ha permesso all'auto di migliorare nei movimenti in maniera progressiva.

Restava a questo punto in sospeso la problematica di riconoscimento dei pedoni e su come comportarsi correttamente al fine di consentire il corretto attraversamento degli stessi all'arrivo dell'auto in corrispondenza delle strisce pedonali.

Oltretutto, se per un attimo si tralascia il fattore cruciale del corretto comportamento da avere in vicinanza ai pedoni, l'auto aveva ancora da migliorare il suo movimento verso l'End Game Object, infatti nonostante un primo accenno di miglioramento nei movimenti, la traiettoria era ancora molto imprecisa e poco realistica. In altri termini, l'auto se pur riusciva a raggiungere molto più spesso l'End Game, per fare ciò non manteneva ancora non manteneva ancora un movimento equidistante rispetto ai delimitatori, ma tendeva a sterzare in modo erroneo non appena la distanza era troppo ristretta rispetto ad uno di essi.

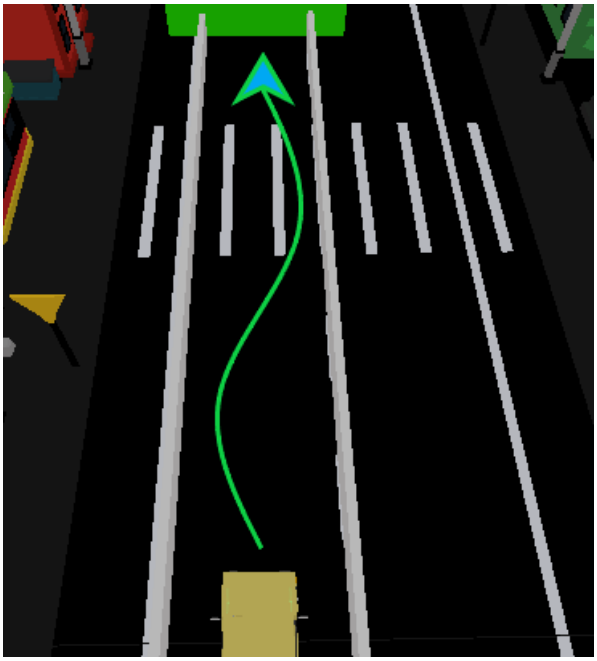


FIGURA 45 - COPORTAMENTO ASSUNTO DALL'AUTO IN TERMINI DI TENUTA DI STRADA ALLA FINE DELLA SECONDA SESSIONE DI ADDESTRAMENTO

Nonostante le forti imprecisioni, la seconda sessione di addestramento, ha fatto evincere una metodologia corretta per il raggiungimento dell'End Game Object, che sarà poi ripresa e raffinata per successive fasi di addestramento, con l'introduzione di un sensore ADAS per la tenuta di strada, come già illustrato nel precedente capitolo.

4.2 SESSIONI 3 E 4 – ADDESTRAMENTO SEMPLIFICATO

Riconoscere i pedoni rispetto agli altri oggetti di scena e innescare il corretto comportamento da avere rispetto ad essi, è stata la prerogativa principale che ha distinto questa nuova fase dell'addestramento dell'agente. Nonostante alla fine della seconda sessione di training, fosse stata introdotto nello script una prima base del sistema di riconoscimento di analisi per l'assegnamento dei Rewards dinamico (Risk Analysis System), questa sessione è stata terminata preventivamente con i soli risultati sui movimenti, visto che si è ritenuto necessario apportare semplificazioni ulteriori al fine di migliorare il comportamento generale dell'auto in corrispondenza di un pedone.

Al fine di acquisire i giusti comportamenti, sono stati temporaneamente sospesi i comandi di rotazione delle ruote, in modo tale da "forzare" l'auto a seguire una traiettoria lineare e non avere altre alternative se non l'utilizzo dei freni per evitare collisioni coni i pedoni, questa semplificazione, l'aggiunta del sistema di assegnazione dei Reward basato su distanze rispetto ai pedoni (in questa fase ancora specifico per il singolo senso di marcia), e le riduzione ad un singolo pedone nell'ambiente di addestramento sono stati i parametri su cui si sono basate la sessione di training 3 e 4.

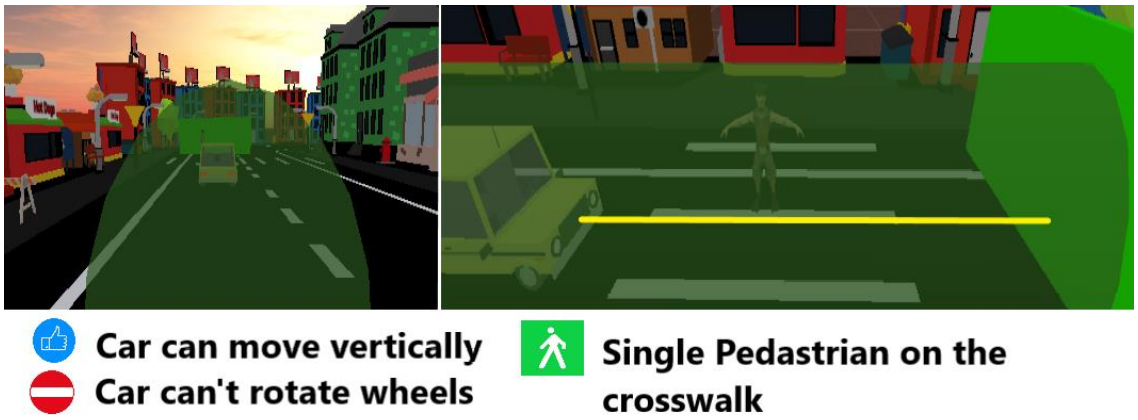


FIGURA 46 - SEMPLIFICAZIONI APPORTATE ALL'AUTO PRIMA DELLA TERZA SESSIONE DI ADDESTRAMENTO

SESSIONI DI ADDESTRAMENTO 3 E 4

Questa nuova fase di addestramento, la prima in cui è stato introdotto il sistema di analisi dinamico per l'assegnazione dei Reward, è stata indice di grande variabilità nel comportamento dell'agente. Non potendo ricorrere ai movimenti di sterzata, lo stesso agente ha facilmente appreso come muoversi in traiettoria lineare verso l'End Game Object (complice anche la penalità assegnata per l'utilizzo della retromarcia, unico residuo rimasto attivo derivante dalla precedente fase di training), ma molto di lunga durata è stato l'apprendimento per l'utilizzo dei freni in vicinanza ai pedoni sulle strisce.

Come già accennato in precedenza, sono stati i serrati controlli sulle distanze tra pedone e agente a creare forte instabilità nei comportamenti acquisibili in fase di addestramento: nella maggior parte dei casi l'agente (data la maggior velocità acquisita dopo aver appreso correttamente come muoversi verso l'End Game Object) veniva premiato per i

suoi movimenti, ma in altri casi quando l'agente, intento a proseguire per la sua strada, si trovava in traiettoria di collisione con il pedone intento ad attraversare, non aveva altra possibilità se non collidere con lo stesso, dato che fino ad ora l'utilizzo dei freni per permettere il corretto attraversamento del pedone, non era mai stato considerato come ipotesi rilevante.

Di conseguenza per circa metà della terza sessione di addestramento, l'agente e il pedone si sono tra di loro scontrati molto frequentemente, e le forti penalità, sia per le collisioni, che per le distanze troppo ravvicinate, hanno creato forti oscillazioni nei report grafici dell'addestramento.

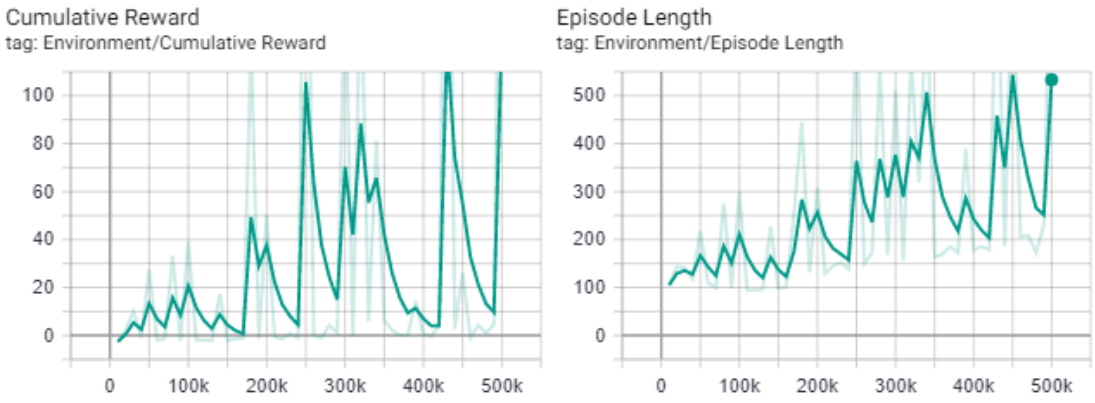


FIGURA 47- TRAINING 3.4 TENSOR BOARD REPORT

Con il concludersi della sessione 3, però, l'agente in modo quasi spontaneo (complice anche l'ingrandimento del raggio sferico del sensore Ray Cast) ha iniziato ad accennare all'utilizzo del sistema frenante, non appena il pedone veniva percepito come potenziale pericolo, ma ancora ad una distanza molto ravvicinata dallo stesso.

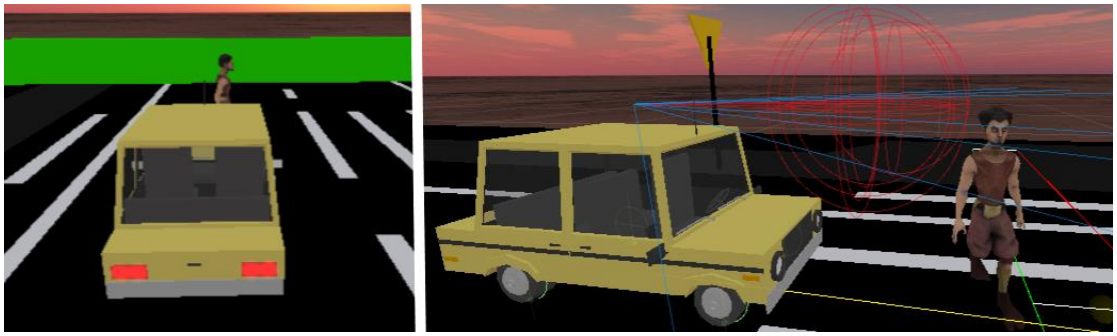


FIGURA 48 - COMPORTAMENTO ASSUNTO DALL'AUTO ALLA FINE DELLA TERZA SESSIONE DI ADDESTRAMENTO

Questa tendenza comportamentale, date le condizioni semplificate in cui è avvenuta questa fase di addestramento, è stata maggiormente accentuata e migliorata con la successiva quarta sessione di addestramento. Durante la quarta sessione di training, al fine di far concepire alla vettura come agire correttamente in corrispondenza di un pedone si è deciso di apportare pochissime variazioni alla scena e al codice, in previsione di miglioramenti con la singola continuazione della fase di addestramento , infatti con il progredire della sessione la vettura ha iniziato a frenare molto più spesso, mantenendo anche un costane miglioramento nella distanza di sicurezza di sicurezza (sull'asse Z), tra il segnale stradale e l'attraversamento. In particolare, il punto di arresto dell'auto è stato sempre più prossimo al raggiungimento del limite massimo del bound dinamico tra agente e pedone in scena calcolato via script (dettagli implementativi riportati nel paragrafo 3.4.2 del documento). Un ulteriore ottimo risultato è stato riscontrato anche nel valutare modi e tempi di ripartenza ottimali, non appena il pedone fosse fuoriuscito dall'area di pericolo.

VALUTAZIONE DEI RISULTATI OTTENUTI E CONSIDERAZIONI ALLA FINE DELLA QUARTA SESSIONE

Al termine della quarta sessione di addestramento, in termini di interazione con il Game Object pedestrian, l'agente ha acquisito, se pur in condizioni ancora semplificate, un ottimo comportamento, che senz'altro ha dato la prova di come il sistema di analisi per l'assegnazione di penalità fosse un meccanismo ottimale per una corretta analisi comportamentale di un pedone su strada.

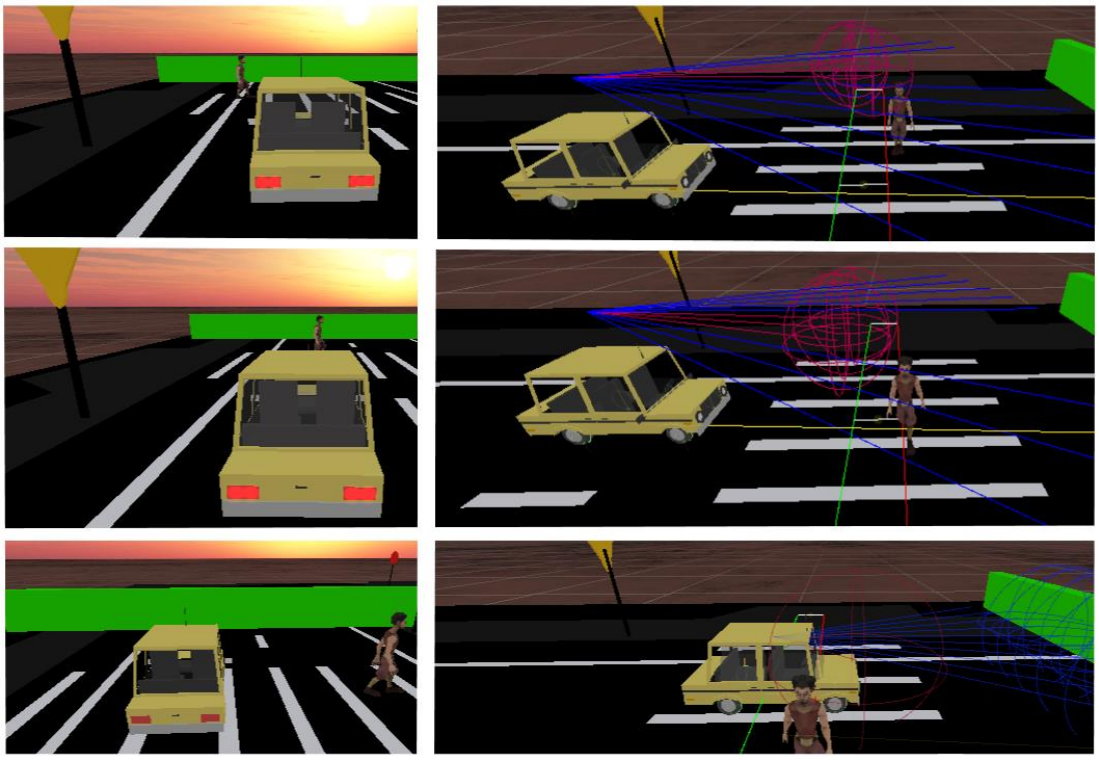


FIGURA 49 - COMPORTAMENTO DELL'AGENTE AL TERMINE DELLA QUARTA SESSIONE DI TRAINING

TEST COMPORTAMENTALE CON RETROMARCIA AMMESSA

In via del tutto **sperimentale**, per testare in che modo l'auto si relazionasse con le distanze imposte dal sistema di Reward dinamico, l'agente è stato posto in addestramento senza la penalità negativa per l'utilizzo della retromarcia quando la distanza (rispetto all'asse Z) tra auto e pedone era ad un valore tale da rientrare nel bound di rischio. Durante il training, era possibile osservare come l'auto continuasse ad andare in avanti verso l'End Game Object quando la corsia era libera dal passaggio pedonale, ma non appena il pedone veniva rilevato, non solo l'auto tendeva a fermarsi quanto prima, ma anche a correggere la distanza indietreggiando, fino ad arrivare a posizionarsi esattamente ad un punto tale da avere la distanza massima dal pedone, in modo tale da garantire un assegnazione di premio più alta (si ricorda che, per come è stata progettata la zona di rischio, distanza tra agente in attesa e pedone in attraversamento, e reward assegnati per questo specifico punto sono direttamente proporzionali tra di loro).

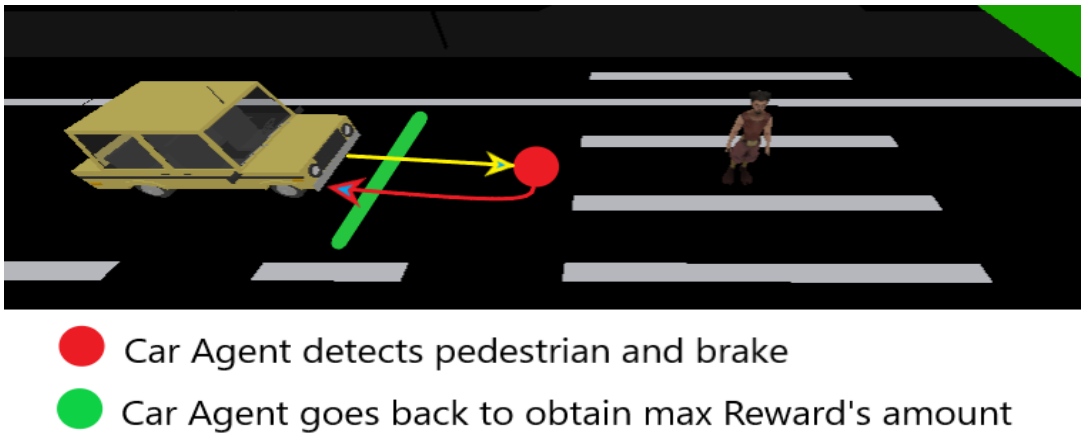


FIGURA 50 - TENDENZA DELL'AUTO AD INDIETREGGIARE PER OTTENERE UN PREMIO PIÙ ALTO

4.3 SESSIONI 5, 6 E 7 – CASI REALISTICI

Concepire come l'auto si relazionasse all'apprendimento semplificato, in fasi distinte, dei movimenti per la corretta tenuta di strada e della corretta interazione con i pedoni, è da considerarsi senz'altro un'ottima base per l'aggiunta di complicanze nell'ambiente di scena.

La fase di addestramento in cui vengono raggruppate le sessioni 5, 6 e 7, infatti ha avuto come obbiettivo quello di far progredire le conoscenze dell'agente sia sul piano delle difficoltà (quali ad esempio numero di pedoni superiore ad 1), ma soprattutto nel pieno coordinamento di movimenti (marcia e rotazione) e comportamento corretto in prossimità di un attraversamento.

Procedendo sempre con lo stesso approccio utilizzato per le sessioni precedenti, basato sull'incremento graduale del grado di difficoltà della simulazione (meglio esplicitata all'inizio di questo capitolo), in questa fase sono stati individuati 3 macro-obbiettivi principali:

- Il riconoscimento di pedoni molteplici in prossimità di un attraversamento stradale;

- Il ripristino dei movimenti di sterzata, con integrazione al riconoscimento dei pedoni;
- L'addestramento dell'agente in due sensi di marcia, al fine di migliorare l'aspetto dinamico del suo comportamento.

SESSIONE 5 – COME REAGISCE L'AUTO FIN ORA ADDESTRATA CON PRESENZA DI PEDONI MULTIPLI SULLE STRISCE?

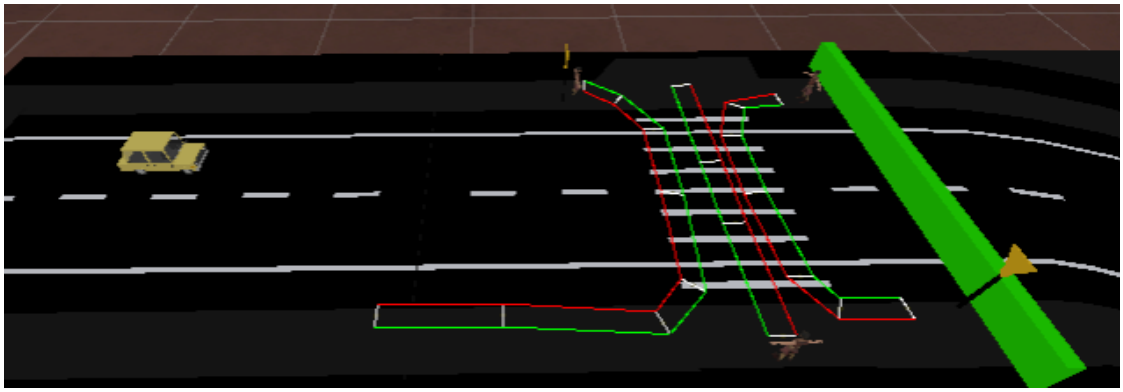


FIGURA 51 - TIPICA CONFIGURAZIONE DELL'ENVIRONMENT DURANTE LA QUINTA SESSIONE DI TRAINING

Partendo dai progressi fatti durante la quarta sessione di training, ci si è posto l'obbiettivo di addestrare l'agente in una condizione leggermente più complessa, l'aggiunta di pedoni multipli sull'attraversamento.

Aumentare il numero di pedoni, non ha comportato molte variazioni in termini di sensoristica di rilevamento, mentre le penalità per movimenti pericolosi in vicinanza all'attraversamento occupato, e per le collisioni auto/pedone, sono state lievemente ritoccate, al fine di poter agevolare ancora di più l'agente nella rapidità delle sue scelte. Inoltre, gli addestramenti sono stati di molto allungati, passando da training da 500.000 steps (uno step di addestramento è l'intervallo che intercorre da un'azione dell'agente alla successiva), ad addestramenti tra il milione e i 3 milioni di step ciascuno, al fine di consentire all'agente di poter metabolizzare meglio la nuova difficoltà introdotta.

Con un numero di pedoni superiore ad uno, si è notato che l'agente tendesse a mantenere, in diretta continuazione con quanto appreso precedentemente, un comportamento constante. L'autovettura in effetti riusciva perfettamente a riconoscere

anche più di un pedone sulla sua traiettoria ed essa ed aspettarne il passaggio, ma in alcuni episodi, proprio l'attendere il passaggio di un secondo pedone, ha causato collisioni laterali con il primo rilevato, intento a tornare indietro dopo aver completato il suo attraversamento in un senso (si ricorda che il movimento pedonale è stato progettato in modo tale da essere percorso da un estremo all'altro per poi riprendere in senso opposto).

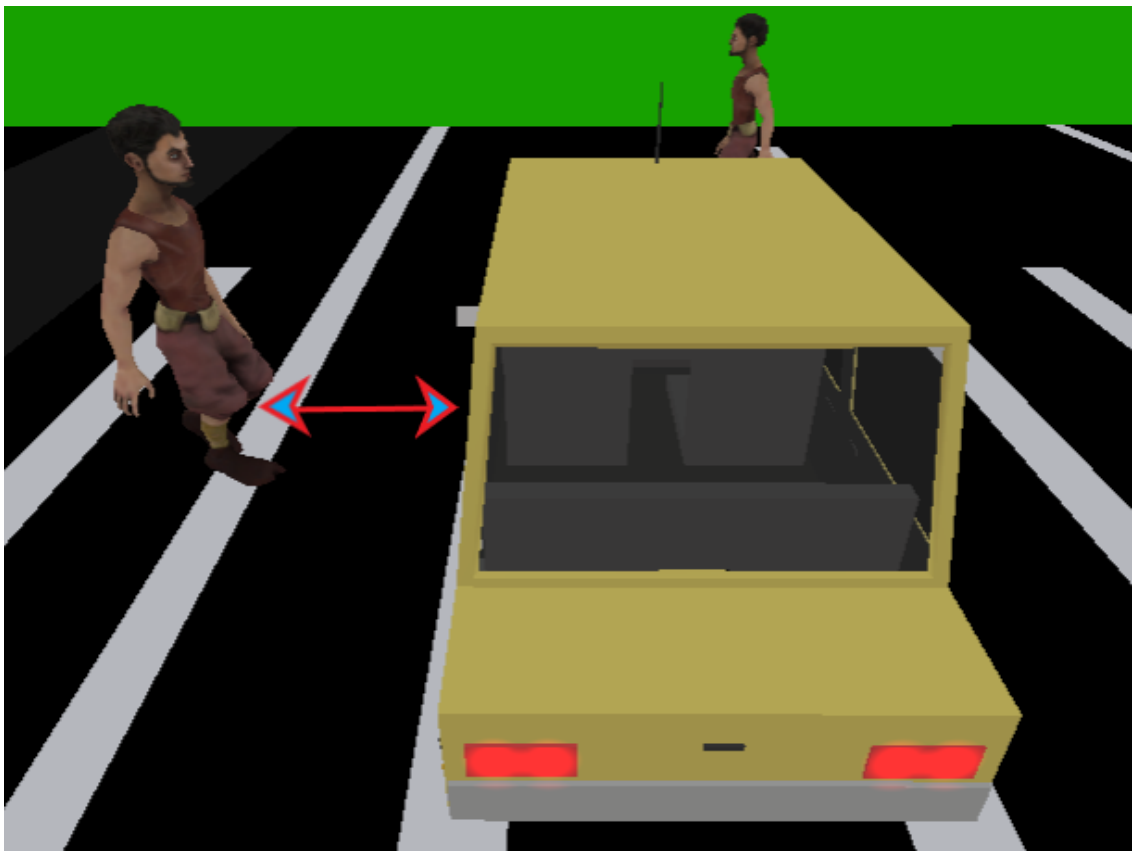


FIGURA 52 - PUNTO CRITICO DURANTE LA SESSIONE 5

Come illustrato dalla figura 52, qui riportata, è possibile notare come l'agente nell'attesa di attraversamento del secondo pedone (ancora rilevato dal sistema di raggi) non percepisce l'arrivo del primo lateralmente, che inevitabilmente entrerà in collisione con esso, causando una perdita di punti.

Tali perdite, causate da collisioni laterali, sono anche riscontrabile anche dai diagrammi di report prodotti da Tensor Board. Dove si può facilmente notare come il

comportamento dell'agente, pur avendo ottimi margini di miglioramento con assegnazioni di premi e penalità (su larga scala) abbastanza bilanciati, è soggetto a dei picchi negativi dovuti soprattutto all'aumento di collisioni con i pedoni al di fuori del margine di osservabilità dell'agente, molto spesso posti lateralmente rispetto all'auto.

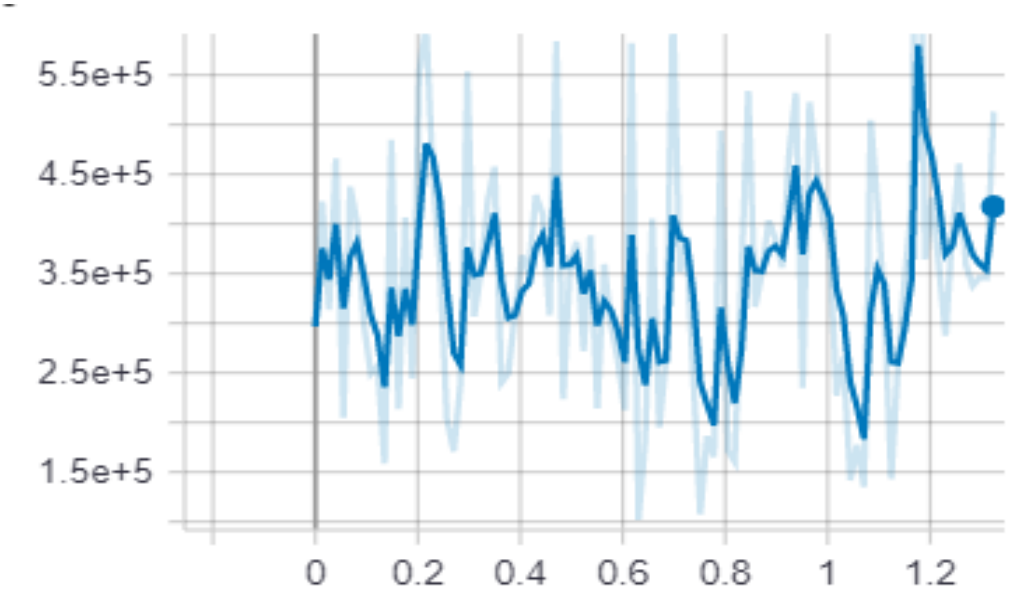


FIGURA 53 - TRAINING 5_9 - TENSOR BOARD REPORT

Al fine di affievolire questa problematica, il fascio di raggi è stato leggermente allargato in modo tale da far percepire all'agente il repentino cambio di marcia da un pedone, e quindi attendere nuovamente il passaggio dello stesso, verso l'altro capo della strada. Nonostante ciò il problema non è stato risolto completamente, ma in alcuni casi estremi, dove il primo pedone usciva totalmente dal campo visivo dell'auto, le collisioni laterali non sono mancate.

VALUTAZIONE DEI RISULTATI OTTENUTI E CONSIDERAZIONI ALLA FINE DELLA QUINTA SESSIONE DI TRAINING

Al termine della quinta sessione di addestramento, l'agente ha imparato a relazionarsi in modo ottimale con 3 pedoni su strada, fermandosi non appena essi venivano rilevati frontalmente dal sistema di raggi associato al veicolo, tale riscontro ha permesso, nelle

fasi di training 6 e 7 per l'integrazione dei movimenti di sterzata e dinamismi aggiunti per incrementare le conoscenze del veicolo, di effettuare test differenziati sia con un singolo pedone che con pedoni multipli.

Per quanto riguarda la criticità riscontrata nella quinta sessione, c'è da considerare che il tutto, è stato anche un po' estremizzato dal tratto di percorrenza fisso che il singolo pedone è vincolato a seguire un percorso stabilito durante l'intero iter di simulazione, in un contesto reale infatti, è molto poco probabile che un pedone si trovi lateralmente ad un'auto ferma ad attendere il passaggio di altri passanti, e se pure ciò accadesse, il contatto tra pedone e auto non comporterebbe un rischio paragonabile come un travolgimento frontale dovuto all'arrivo di un auto in corsa.

Quindi, sulla base di tali considerazioni, e di un non trascurabile miglioramento del comportamento dell'agente, dovuto all'ampliamento del campo di visibilità, è possibile affermare che la quinta sessione di addestramento ha comunque portato ad ottimi vantaggi, e soprattutto ha messo in luce gli estremi entro quali è possibile addestrare l'agente, al fine di ottenere risultati attendibili.

COME GARANTIRE MAGGIOR REALISMO? – MODIFICHE E AGGIUNTE ALL'AGENTE DURANTE LA NUOVA FASE DI APPRENDIMENTO

A questo punto del progetto, avendo dato ampio spazio al corretto comportamento da avere in vicinanza ad un attraversamento stradale, si è ritenuto necessario dare maggior dinamismo all'auto, al fine di ottenere una simulazione quanto più simile alla realtà quotidiana di un circuito urbano.

Innanzitutto, si è ritenuto impensabile pensare che un'autovettura non possa girare le ruote durante la circolazione, quindi si è proceduto a riabilitare tali movimenti, e a reintrodurre i Walk Limit Wall all'interno dell'ambiente di scena.

Problema che si è riscontrato fin da subito, è stato nuovamente il sovraccarico di informazioni a cui il singolo fascio di raggi (atto a simulare un unico sensore ADAS per la tenuta di strada che per il riconoscimento dei pedoni) era sottoposto non portava ad

avere risultati ottimali. Quindi, data la possibilità di integrare più di una singola componente Ray Perseption Sensor 3D in un singolo agente, si è ipotizzato di addestrare l'agente con due sensori separati, il primo atto alla tenuta di strada, a cui sarebbe spettato il compito di simulare e riapprendere il comportamento del singolo sensore durante la seconda fase di addestramento (analisi delle distanze rispetto ai limiti di corsia) e il secondo dedicato al riconoscimento dei pedoni su strada.

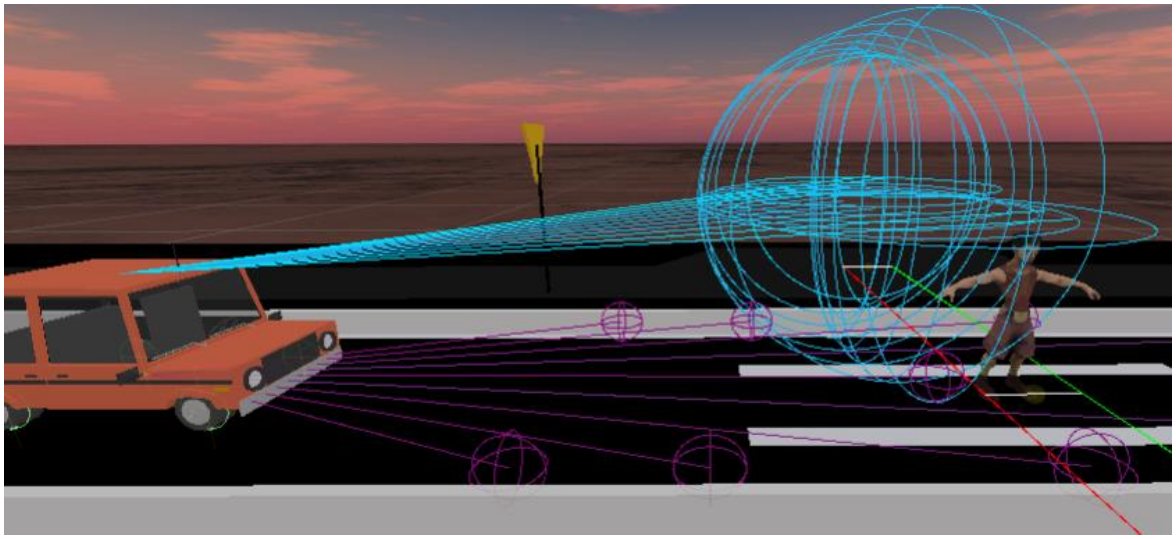


FIGURA 54 - SENSORISTICA IN DOTAZIONE ALL'AGENTE PRIMA DELLE SESSIONI 6 E 7 DI ADDESTRAMENTO

Oltretutto, la seconda sessione di addestramento, pur avendo portato a dei movimenti corretti verso l'End Game Object, lasciava i controlli sulle rotazioni del veicolo ancora molto svincolati (come spiegato infatti la traiettoria del veicolo risultava, alla fine della seconda sessione, molto ondulatoria. Al fine di migliorare tale comportamento, e far seguire una traiettoria più lineare all'auto, è stato introdotto un sistema di assegnazione di premi/penalità sull'angolo di rotazione delle ruote, rispetto alla posizione dell'auto e all'angolo compresa tra la stessa e l'End Game Object da raggiungere.

- Tale funzione è stata implementata ed utilizzata sia per la scena di attraversamento che per quella di parcheggio del progetto, date le necessità comuni -

```
//Checking by direction
if (actualDirection == Direction.Acceleration)
{
    /*
     * OK CASES
     */
    //straight check
    if ((carAngle < 3 && carAngle >= 0 && wc_fl.steerAngle <= 15f && wc_fl.steerAngle >= 0) ||
        (carAngle > -3 && carAngle < 0 && wc_fl.steerAngle >= -15f && wc_fl.steerAngle < 0))
        assignment = reward * 3;
```

FIGURA 50 - STRALCIO DI CODICE DELLA FUNZIONE DI CONTROLLO DELL'ANGOLO DELLE RUOTE RISPETTO ALL'END GAME OBJECT

L'idea di funzionamento di questa nuova accortezza introdotta non è molto complessa, considerando il verso di avanzamento dell'auto (marcia in avanti o retromarcia), vengono effettuati controlli sull'ampiezza dell'angolo tra il centro dell'auto e il centro dell'End Game Object, nel caso in cui esso rientri in un certo bound positivo e l'auto effettui una rotazione molto tenue al fine di proseguire verso l'End Game Object (monitorata grazie al parametro Steer Angle della componente Unity Wheel Collider) allora la stessa viene premiata, in caso opposto la stessa viene penalizzata. Ovviamente i controlli fatti da tale funzione sono stati molteplici, tenendo conto di volta in volta di del movimento, necessario all'auto, al fine di correggere la sua traiettoria verso l'End Game Object, in termini di verso di avanzamento, posizione attuale e soprattutto posizionamento dell'auto rispetto al centro dell'angolo monitorato (posizionamento a destra o sinistra).

Oltre queste modifiche apportate all'agente per migliorare osservazioni e assegnazioni di premi in fase di addestramento, al fine di aumentare maggiormente il grado di realismo della scena, si è pensato di predisporre la possibilità di aggiungere un secondo agente su strada, posto in senso di marcia opposto a quello utilizzato fino a questo momento. Questo nuovo upgrade, però ha necessitato di una lieve modifica al codice, in quanto il sistema di assegnazione di analisi per le distanze, pensato e predisposto utilizzando come punto di riferimento il singolo senso di marcia originale, non calcolava correttamente le distanze quando l'auto era posta in senso opposto, quindi se pur

mantenendo lo stesso principio base di funzionamento, il calcolo delle distanze è stato modificato al fine di essere indipendente dal singolo senso di marcia.

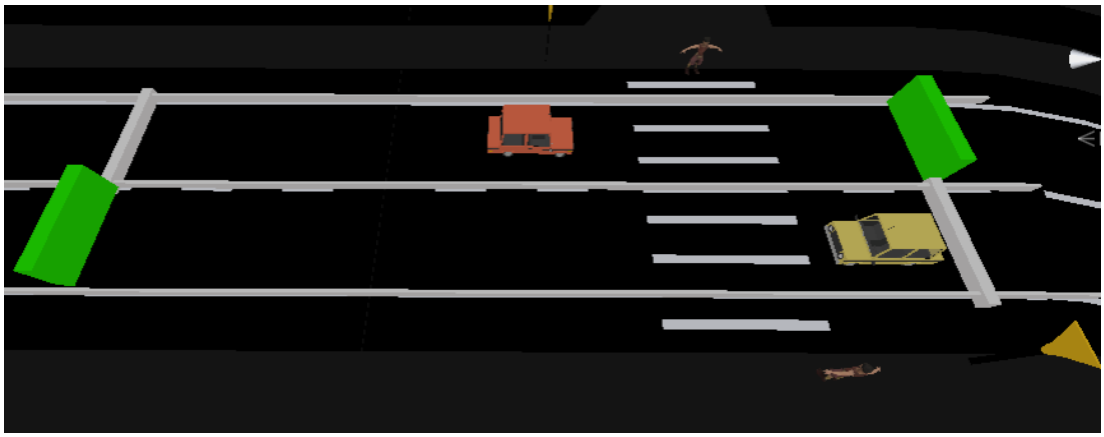


FIGURA 55 - AMBIENTE UTILIZZATO PER LE SESSIONI 6 E 7

SESSIONI DI ADDESTRAMENTO 6 E 7 – INTEGRAZIONE DEI MOVIMENTI DI STERZATA E AGGIUNTA DEL SECONDO AGENTE PER SIMULARE DUE SENSI DI MARCIA

La sesta e la settima sessione di training (la più lunga dell'intero iter progettuale), sono state le fasi che hanno effettivamente portato i maggiori risultati, ma con esse non sono mancate le difficoltà.

Come già anticipato con la riabilitazione dei comandi per i movimenti di rotazione dell'auto, si è presentata la necessità di sdoppiare il singolo fascio di raggi dell'auto e di progettarne due differenti, tale scelta progettuale è stata presa a metà della sesta sessione di addestramento dati gli scarsi risultati raggiunti dall'auto con un singolo fascio di raggi. Proprio lo sdoppiamento degli stessi, però ha portato alla perdita di tutte le informazioni precedentemente acquisite dall'agente (in quanto reti neurali già generate, non possono essere utilizzate come base di addestramento con un numero di osservazioni differente a quello già utilizzato).

Ricominciando da zero, si è facilmente riscontrato come l'auto, se pur con tempi di addestramento abbastanza lunghi, ha facilmente riappreso come muoversi su strada nella zona delimitata dai Walk Limit Walls, a dimostrazione che l'auto, pilotata da un

sistema di ricompense/penalità molto granulare, riesca facilmente ad acquisire le conoscenze corrette pur partendo da nessun'informazione pregressa. In particolar modo, con il nuovo sistema di monitoraggio delle ruote, l'agente ha migliorato gradualmente la sua traiettoria, fino a raggiungere una padronanza molto soddisfacente dei suoi movimenti, riducendo sempre di più le sterzate durante il prosieguo del cammino di marcia rispetto all'End Game Object.

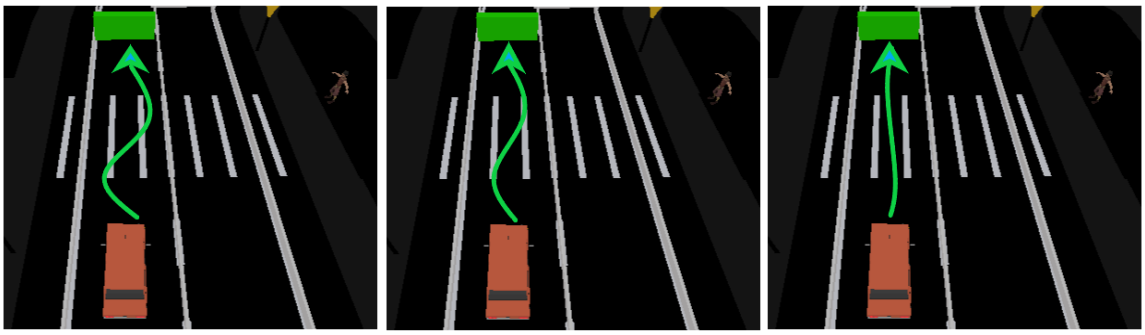


FIGURA 56 - PROGRESSIVI MIGLIORAMENTI DI TRAIETTORIA DURANTE LA SESTA SESSIONE DI TRAINING

Sul concludersi della sesta sessione di addestramento l'agente, quindi ha migliorato di molto il suo movimento verso l'End Game Object, evitando quasi del tutto le collisioni con i Walk Limit Wall, con velocità anche molto realistiche, però il corretto uso dei freni in vicinanza all'attraversamento pedonale ha presentato qualche difetto. Infatti, pur riuscendo ad essere molto più precisa nei suoi movimenti (quasi in modo tale da simulare il movimento lineare forzato delle precedenti sessioni 3,4 e 5) grazie al nuovo sistema di controllo. L'auto, però, non ha assunto sempre il comportamento ideale rispetto ai pedoni presenti sull'attraversamento, si è osservato infatti, come l'auto "preferisse evitare quando possibile l'utilizzo dei freni, continuando il suo cammino verso l'End Game Object, evitando il pedone davanti ad essa, tramite l'utilizzo dello sterzo, effettuando un movimento, sì molto preciso, ma anche potenzialmente molto pericoloso per l'incolumità del pedone (in questo caso però, paradossalmente, le collisioni con i pedoni venivano del tutto evitate).

FIGURA 57 - COMPORTAMENTO ASSUNTO DALL'AUTO ALLA FINE DELLA SESSIONE

Tale condotta in prossimità delle strisce pedonali, se pur di buon effetto ai fini evitare collisioni con l'oggetto pedone, non è però da considerarsi ottimale, ai fini della simulazione, in quanto come ribadito più volte, questo genere di atteggiamenti, in un contesto reale può mettere in serio pericolo l'incolumità del pedone stesso.

La sessione 6, se pur con diverse sfaccettature, ha messo in luce le vere potenzialità del veicolo, infatti è stato riscontrato, se pur in modo molto fievole, il tanto agognato aspetto di coordinazione tra corretto movimento verso l'End Game Object e arresto di marcia per permettere l'avanzamento di uno o più pedoni cominciava a presentarsi, (nonostante l'utilizzo dei freni tendesse ad essere molto sporadico date le azioni errate pocanzi illustrate).

Restava ora la necessità di incentivare l'utilizzo dei freni in prossimità dell'attraversamento, e fortunatamente ciò è stato possibile, aumentando leggermente il raggio sferico del sensore RayCast per la percezione dei pedoni, in modo tale da far interferire i singoli raggi tra di loro, e non far percepire punti in cui poter svoltare all'agente.

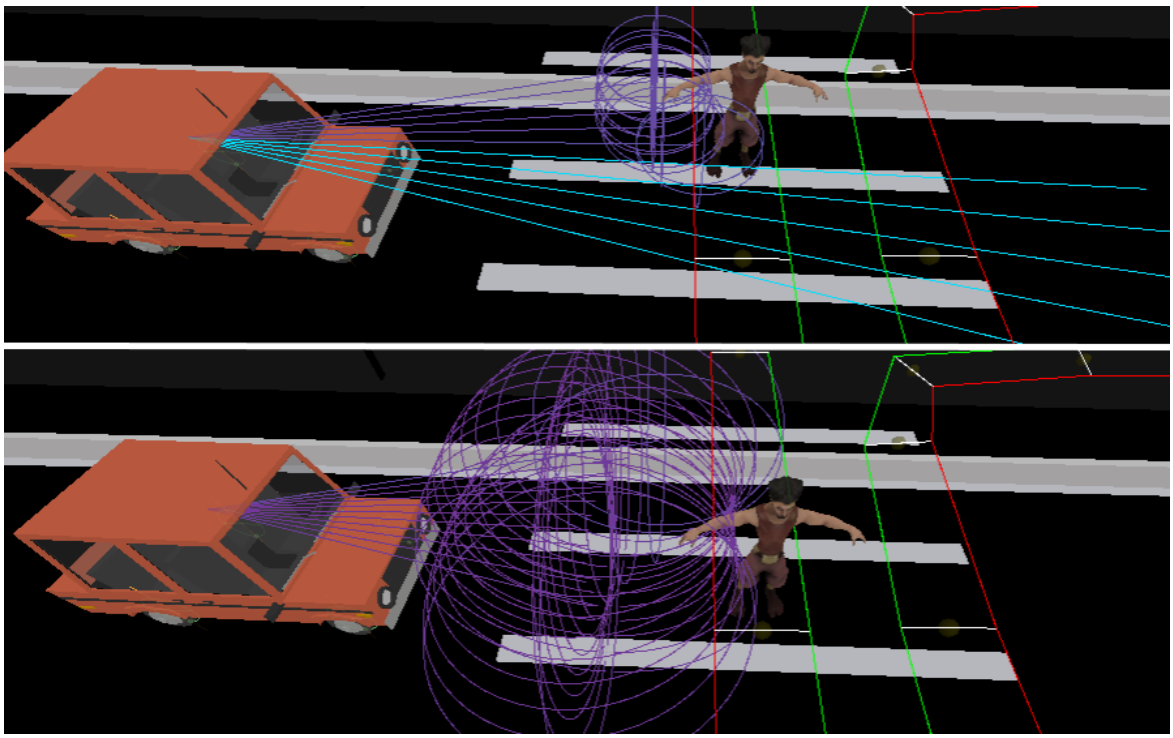


FIGURA 58 - COMPORTAMENTO DEL SENSORE DI PERCEZIONE PEDONALE, PRIMA E DOPO LA MODIFICA EFFETTUATA ALLA FINE DELLA SESSIONE 6

La figura 58, illustra precisamente il cambio di comportamento del sensore di riconoscimento pedonale prima e dopo l'ampliamento delle sfere, in un primo momento l'auto percepiva ancora la possibilità di sterzare a destra del pedone, avendo ancora visibilità oltre ad esso, nel secondo caso, essendo tutti i raggi sottoposti ad interferenza causata dal passaggio del pedone, non era possibile per l'auto valutare nessun altra possibilità se non frenare e attendere, per evitare la collisione con il pedone in addestramento. Tale modifica, ha fatto registrare fin da subito un comportamento molto positivo nell'utilizzo dei freni, infatti l'agente (anche senza essere sottoposto ad

addestramento) ha di molto incrementato l'utilizzo dei freni, anche se in modo ancora un po' instabile vicino all'attraversamento pedonale.

Dopo aver riscontrato tale miglioramento, sono stati avviati i primi training di addestramento della settima sessione, che quasi da subito hanno dimostrato quanto ipotizzato, l'auto riusciva facilmente ad assumere il corretto comportamento in vicinanza ad un pedone in attraversamento, coordinando anche i movimenti verso il singolo senso di marcia.

A questo punto avendo finalmente ottenuto una corretta alternanza tra i comportamenti basilari del mezzo, si è subito deciso di incrementare il grado di difficoltà della simulazione con l'aggiunta di secondo agente, il quale avrebbe dovuto replicare il comportamento del primo, nel senso di marcia opposto. A questa complicazione, sono stati dedicati ben dieci training, al fine di dimostrare quanto sia complesso, per un agente addestrato tramite tecnologie di Reinforcement Learning, assimilare nuove osservazioni e conseguentemente rispondere con nuove azioni con una rete neurale già pronta, e soprattutto già predisposta all'affrontare una serie di problematiche.

A colpo d'occhio forse può sembrare banale, ma il cambio di senso di marcia ha posto l'agente in una prospettiva di osservazione totalmente nuova (anzi opposta a quella precedente), all'interno della quale non solo era necessario perfezionare i movimenti, ma anche interagire più assiduamente con i pedoni in attraversamento. Infatti, il secondo agente ha avuto il compito di determinare il corretto comportamento da avere rispetto al Game Object Pedestrian, già dall'inizio di ogni episodio, essendo il suo punto di spawn molto vicino all'attraversamento.

Per il secondo agente, è stato molto complicato soprattutto riadattare le informazioni acquisite in un singolo senso di marcia al fine di raggiungere correttamente il lato opposto della strada dove era posizionato il suo End Game Object di riferimento, evitando oltretutto collisioni con i Walk Limit Wall. Prima di arrivare a destinazione, infatti il secondo agente ha impiegato molto tempo per interagire bene con la nuova

disposizione, ma con il progredire dell'addestramento l'obbiettivo prefissato è stato ampliamente raggiunto, cioè ciascuno delle due autovetture si dirigeva verso il corretto End Game Object, avendo un corretto comportamento sia in termini di traiettoria che di corretto utilizzo del sistema frenante in vicinanza ai pedoni.

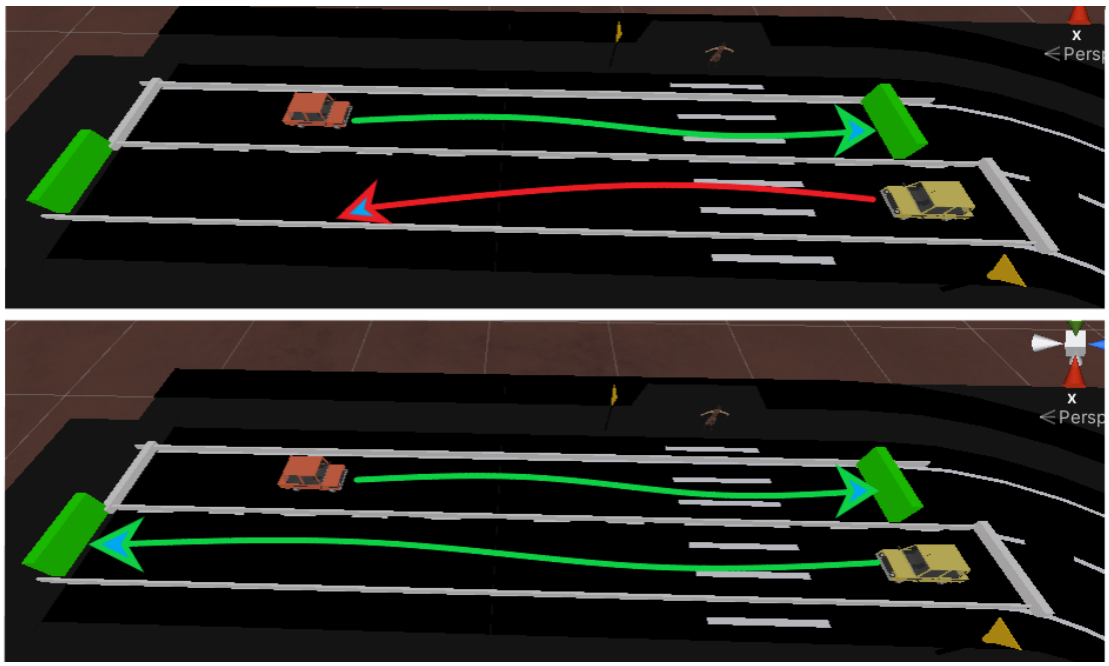


FIGURA 59 - MIGLIORAMENTO COMPLESSIVO DEL SECONDO AGENTE DURANTE LA SETTIMA SESSIONE DI TRAINING

OSSERVAZIONI CONCLUSIVE E LIMITAZIONI RISCONTRATE ALLA FINE DELLA SETTIMA SESSIONE DI TRAINING

Al termine della nuova fase di addestramento, il raggiungimento di un comportamento completo e corretto rispetto agli obbiettivi prefissati, ha permesso di mettere in luce anche alcune criticità nel comportamento dell'agente, in particolare, si è notato che la politica di assegnazione dei punteggi adottata, non rispettava molto la logica di bilanciamento tra premi e penalità alla base di un buon addestramento di Reinforcement Learning, infatti è facilmente riscontrabile, dai diagrammi di Report prodotti in fase questa fase di addestramento, come l'applicazione di forti penalità abbia creato forti oscillazioni nell'andamento generale di ogni singolo training, nonostante l'agente abbia acquisito un comportamento sempre migliore e adatto agli scopi della simulazione.

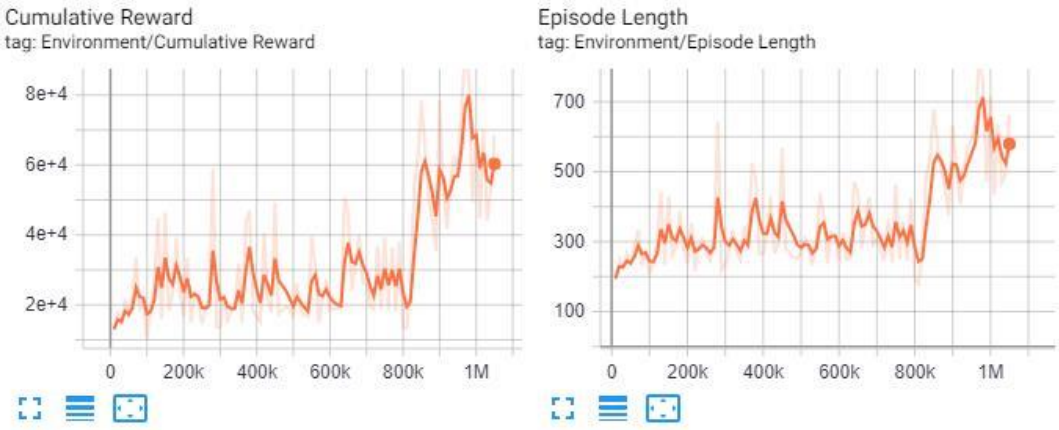


FIGURA 60 - TRAINING 7.17 TENSOR BOARD REPORT

Tali oscillazioni hanno creato talvolta reazioni di forte incertezza nei comportamenti generali dell'agente, esso infatti (indipendentemente dal senso di marcia) in alcuni episodi, pur riuscendo a portare a termine la simulazione, all'occasione tendeva a fermarsi con un'angolazione scorretta rispetto all'End Game Object, quando un pedone era davanti alla sua traiettoria di marcia. In particolare era possibile osservare come l'agente arrivato ad una corretta distanza di arresto rispetto al pedone, con le ruote leggermente rivolte verso uno dei due Walk Limit Walls, tendesse correttamente a frenare ed attendere il passaggio del pedone, ma una volta giunto il momento di ripartire, lo stesso non avendo spazio per riprendere la corretta traiettoria di marcia, tendeva a restare fermo per tempi troppo lunghi, o addirittura a collidere con i Walk Limit Walls in casi più estremi.

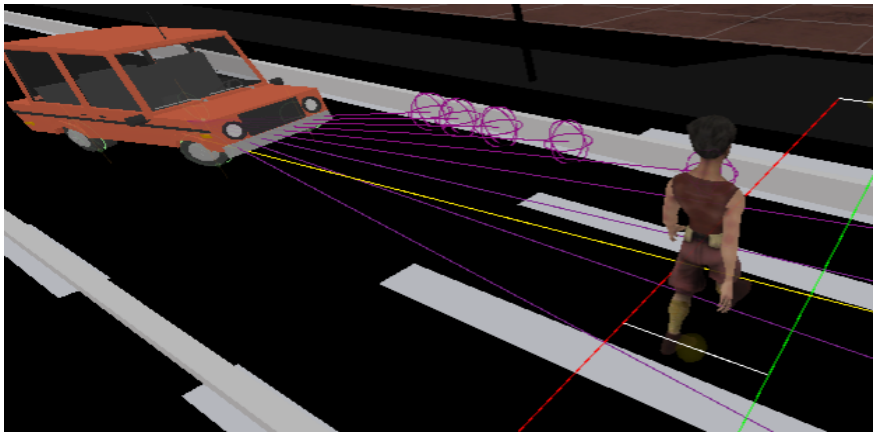


FIGURA 61 - FRENATA SCORRETTA DELL'AGENTE AL PASSAGGIO DEL PEDONE

Pur cercando di riposizionarsi correttamente di riposizionarsi e ripartire, essendo fortemente penalizzato per l'utilizzo della retromarcia e per la collisione con il Walk Limit Wall, l'agente era come intrappolato in uno stato di incertezza, il quale appunto sfociava nelle conseguenze sopra riportate. Si è ritenuto, quindi, necessario procedere con una nuova fase di addestramento finale con lo scopo di ottimizzare i comportamenti dell'auto anche nelle circostanze più critiche rilevate, come ad esempio quella appena riportata, e soprattutto per riportare i Reward positivi o negativi in una condizione di accettabile bilanciamento.

4.4 SESSIONI 8 E 9 – OTTIMIZZAZIONI NEL COMPORTAMENTO DELL'AGENTE

Come riscontrato dai comportamenti generali della macchina, in un addestramento per rinforzo, è cruciale assegnare premi e penalità in modo bilanciato in quanto forti penalità (o viceversa forti punteggi positivi) possono rendere l'agente incapace di rispondere correttamente alla criticità di alcune circostanze. Per risolvere la problematica di arresto erroneo riscontrata nel comportamento dell'agente, sono stati innanzitutto rivalutati tutti i valori numerici di premi e penalità assegnati nello script, in modo tale da bilanciarli e quindi avere un andamento più lineare in fase di addestramento, ma è stato anche predisposto in scena un secondo attraversamento pedonale con pedone singolo, al fine di sottoporre più spesso i due agenti in movimento, all'analisi comportamentale da effettuare in vicinanza ad un pedone.

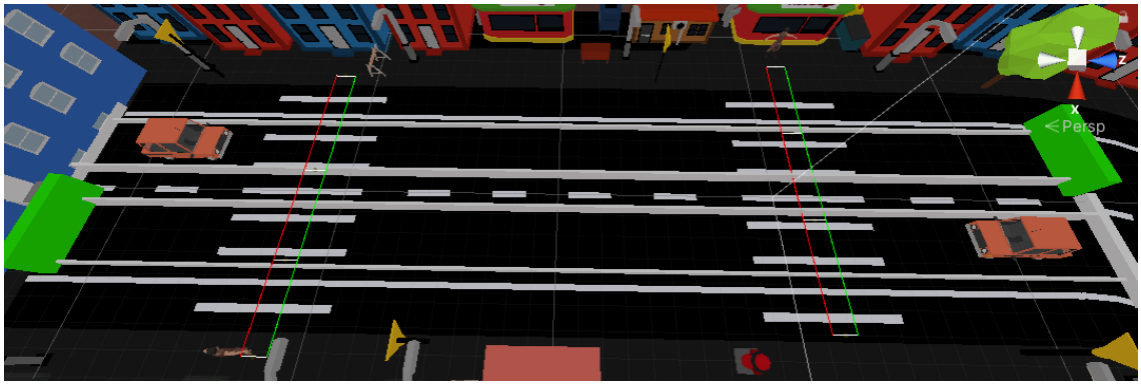


FIGURA 62 - ENVIRONMENT UTILIZZATO PER LA FASE FINALE DI ADDESTRAMENTO

Scelta cruciale che ha condizionato questa nuova fase di addestramento, è stato scegliere se continuare ad addestrare il veicolo sulla base degli addestramenti precedenti, (quindi impostando come base iniziale l'ultima rete neurale ottenuta dall'addestramento finale della settima sessione), o ricominciare da capo.

Tale bivio a portato all'avvio di due sessioni di training "parallele": l'ottava sessione di training, che si poneva l'obbiettivo di riaddestrare l'agente sulla base delle conoscenze apprese dopo la settima sessione e la nona, la quale rinunciava all'apprendimento già a disposizione del veicolo, ma si poneva l'obbiettivo di riaddestrarlo completamente, per step incrementali, supponendo che Rewards più stabili permettessero all'agente di meglio valutare anche situazioni generalmente più critiche.

Come previsto dai presupposti di avvio della nona sessione, durante i training di addestramento della sessione 8, l'auto non ha per nulla migliorato le tendenze negative assunte durante la sessione precedente, infatti la problematica citata alla fine del paragrafo precedente, si è presentata ugualmente, se non in modo forse più persistente rispetto a prima. Dopo tali osservazioni si è deciso quindi di abbandonare questa scelta e di proseguire con il secondo criterio formulato.

ACCORGIMENTI E CONSIDERAZIONI EFFETTUATE PRIMA DELLA NONA SESSIONE DI TRAINING

Dopo aver assunto constatato che un addestramento per rinforzo, guidato da Reward troppo sbilanciati, risultasse ostico da migliorare, è stato deciso di percorrere l'intero Iter di addestramento, partendo senza alcuna base pregressa, però fare ciò a questo punto

del progetto, appariva molto ostico, soprattutto per i lunghi tempi di apprendimento necessari all'auto per imparare correttamente ad utilizzare i freni vicino ad un pedone.

Al fine di velocizzare leggermente tale caratteristica, si è deciso di forzare leggermente via script il comportamento dell'auto, imponendole di fermarsi laddove la vicinanza al pedone era tale da mettere a rischio quest'ultimo se l'auto avesse proseguito nel suo cammino. Più precisamente, l'agente aveva ancora la facoltà di frenare quando lo ritenesse opportuno, ma se ciò non fosse stato fatto in autonomia, lo script di controllo gli imponeva l'arresto, permettendo il corretto attraversamento del pedone.

Prima di procedere con l'addestramento finale, come già accennato nel terzo capitolo, è stato aggiunto, per questa fase di addestramento anche un terzo fascio di raggi, dedito a visualizzare in ogni circostanza la posizione dell'End Game Object, in modo tale da non far perdere la visibilità dello stesso anche nelle posizioni più critiche. Tale strategia prende spunto dalla simulazione di parcheggio, nella quale la continua visualizzazione dell'End Game Object, è risultata di fondamentale importanza al fine di garantire il corretto posizionamento della vettura in manovra.

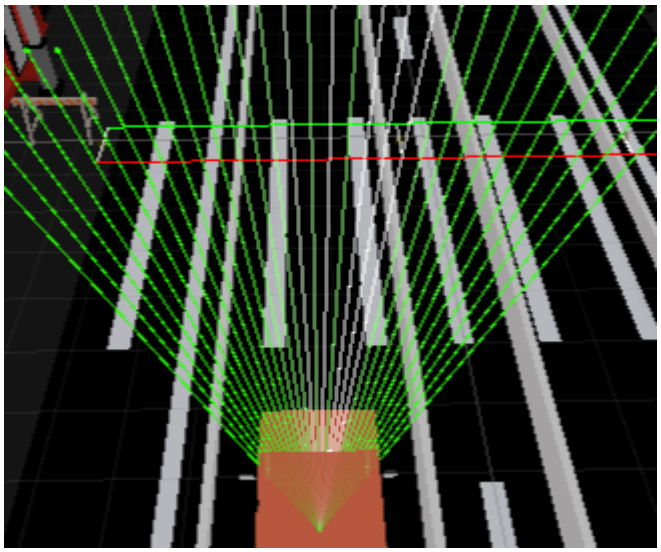


FIGURA 63 - SENSORE PER LA RICEZIONE DELL'END GAME OBJECT

SVOLGIMENTO E RISULTATI DELLA NONA SESSIONE DI TRAINING

Come anticipato la sessione di addestramento 9 ha avuto lo scopo di far assimilare nuovamente tutti i comportamenti appresi dall'agente durante le precedenti sessioni progettuali:

- I primi quattro addestramenti sono stati effettuati da un singolo agente, senza la presenza di pedoni in attraversamento sul tratto stradale, come presumibile il corretto movimento in avanti dell'agente è stato ripristinato senza nuovi accorgimenti, anche se complessivamente sono stati necessari più di 5 milioni di step per il ripristino complessivo di traiettorie e velocità ottimali ottenuti alla fine della sesta sessione di addestramento;
- I successivi cinque training di addestramento sono serviti per par interagire nuovamente il singolo agente con il Game Object Pedestrian, soprattutto con l'ausilio di un'assegnazione di premi e penalità più omogenea e l'ausilio via script per garantire in ogni caso la frenata. Come previsto, l'auto in questa fase ha assunto un comportamento ottimale, potendo contare su una buona base di movimento, preventivamente acquisita, e soprattutto un'ottima risposta all'attraversamento di un numero di pedoni variabili sulle strisce pedonali.

    - Durante questa sotto-sessione di training, è importante considerare il ruolo avuto dallo script rispetto al comportamento dell'agente pur essendo vero che l'agente venisse "limitato" nelle sue scelte e costretto alla frenata, però proprio la frenata automatica, ha fatto scattare in automatico i premi destinati all'agente per il corretto comportamento rispetto ai pedoni, invogliando quindi di conseguenza l'auto a frenare in autonomia. Dai messaggi di Log stampati in fase di addestramento, è stato infatti possibile notare come l'agente tendesse sempre di più ad azionare i freni anche in autonomia. In ogni caso, questa tecnica ha permesso di garantire in modo efficace come e quando l'auto dovesse ricorrere al supporto

dei freni, garantendo una buona dose di sicurezza per il pedone in addestramento (obbiettivo principale per la simulazione).

- Per concludere poi la sessione di training, con gli ultimi 3 addestramenti, è stato riattivato anche il secondo agente, utilizzato durante la settima sessione, per il movimento in senso di marcia opposto, anche in questo caso, in cui i pedoni erano posti anche in posizioni differenti rispetto a prima data la presenza di un doppio attraversamento pedonale, i risultati sono stati molto soddisfacenti, infatti in entrambi i sensi di marcia l'agente ha risposto bene agli obbiettivi prefissati per l'intera simulazione, portando così a compimento l'intero iter di addestramento richiesto dalla simulazione.

Al termine della nona sessione di training, è senz'altro possibile affermare che gli obbiettivi di ottimizzazione sono stati rispettati, e l'agente ha assunto un comportamento davvero molto positivo mantenendo al contempo anche un ottimo equilibrio nelle sue azioni, infatti durante questa sessione il tool di monitoraggio Tensor Board, ha mostrato i risultati più equilibrati dall'inizio dell'intera fase di training, facendo evincere come per i movimenti verso l'End Game Object, l'agente abbia mantenuto un costante miglioramento.

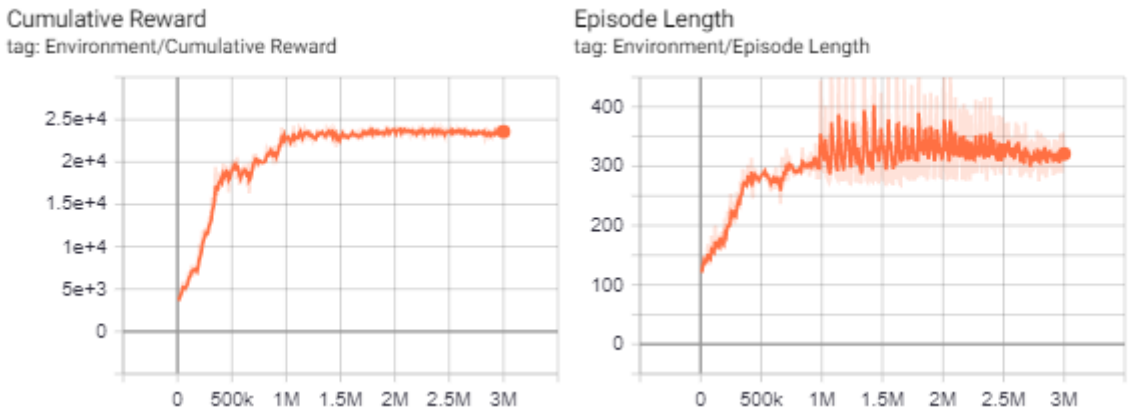


FIGURA 64 - TRAINING 9.2 TENSOR BOARD REPORT

Per quanto riguarda poi i successivi addestramenti, dove la figura del pedone è stata reintrodotta in simulazione, nonostante la continua alternanza tra premi e penalità che il sistema di analisi del rischio ha generato, è stato possibile constatare che con un equilibrio numerico tra punteggi acquisiti e punteggi persi, l'agente è riuscito in ogni caso ad equilibrare il suo comportamento, e a mantenere comunque un margine di miglioramento costante.

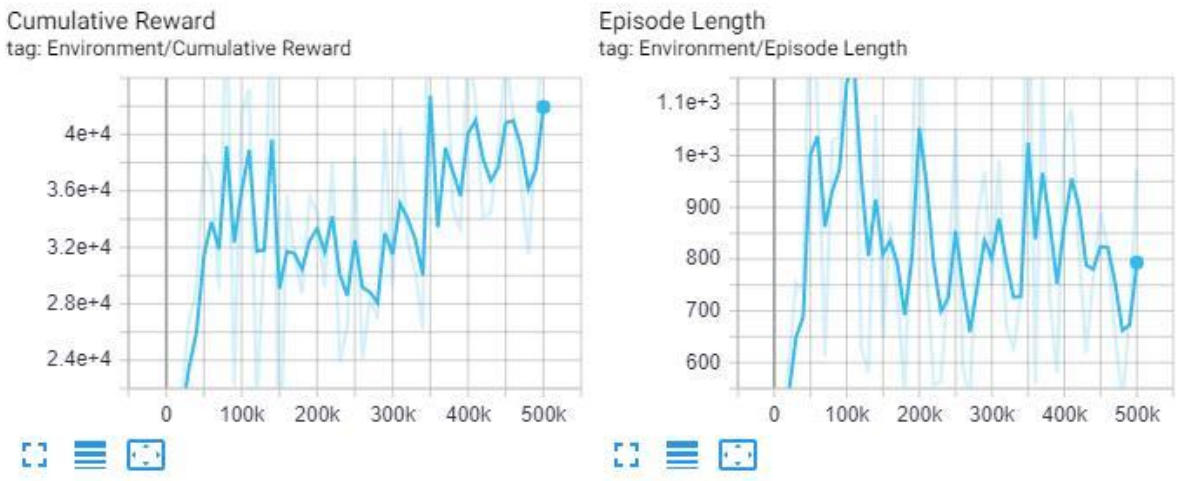


FIGURA 65 - TRAINING 9.12 TENSOR BOARD REPORT

4.5 CONCLUSIONI E CONSIDERAZIONI AL TERMINE DELL'ADDESTRAMENTO

Con la conclusione della nona sessione di training, l'agente ha risposto davvero molto bene agli obbiettivi prefissati, garantendo così la buona riuscita della simulazione e del progetto, in particolar modo con i risultati ottimali avuti con la nona fase di addestramento, si può senz'altro affermare che un addestramento per rinforzo, se fatto con dovute accortezze e soprattutto per piccoli passi mantenendo sempre un costante equilibrio tra premi e penalità, è un ottimo strumento per lo sviluppo di un'intelligenza artificiale atta al garantire sicurezza in una qualsiasi realtà stradale, sia essa un attraversamento pedonale o meno.

In termini della simulazione sviluppata, è importante considerare che la chiave di lettura vincente è stata senz'altro il concetto di ***equilibrio***, in quanto ogni eccesso, come il lasciare troppa autonomia all'agente (come fatto nella prima sessione) oppure

penalizzare o premiare troppo e in modo sbilanciato, non porta ad altro se non garantire risultati scarsi o poco affidabili. Guidare l'autovettura per passi incrementali, e affinando man mano tutte le scelte progettuali introdotte, ha reso possibile rendere un semplice modello di autovettura statico, non solo autonomo nei suoi movimenti garantendo un buon sistema adatto alla circolazione su strada, ma soprattutto capace di rispondere in modo autonomo e corretto ad un contesto molto complesso e imprevedibile quale un attraversamento pedonale. Le complicazioni introdotte (dal numero variabile di pedoni in posizioni e cammini differenziati al doppio senso di marcia) e le successive risposte positive dell'agente conseguenti ad un arduo e lungo processo di addestramento, hanno reso possibile raggiungere un grado di complessità non trascurabile, e soprattutto molto affine alla realtà quotidiana di un attraversamento stradale, pur garantendo la possibilità di futuri miglioramenti alla simulazione.

CAPITOLO 5 – PROTOTIPO E SVILUPPI FUTURI

5.1 VERSIONE PUBBLICA DIMOSTRATIVA

Come precedentemente ribadito, il progetto AI Car Kineton è composto di due scene indipendenti, le quali trattano reciprocamente due problematiche molto differenti tra loro, quali sono il parcheggio automatizzato e il rilevamento automatizzato di pedoni su strada. La versione pubblica del progetto permette all'utente di avere una visione completa di ciò che lo compone tramite un piccolo menù, il quale permette di visualizzare i risultati finali dei training su entrambe le scene, quindi di vedere come l'automobile compie scelte intelligenti in entrambe le situazioni.

Altra funzionalità messa a disposizione per l'utente è la guida manuale in una scena completa in cui sono compresi il parcheggio e molteplici attraversamenti pedonali. In questa scena, l'automobile non sarà dotata di alcun sensore e sarà l'utente ad averne il pieno controllo tramite comandi da tastiera.

5.1.1 FUNZIONAMENTO DEL MENÙ

Il menù segue una struttura semplice e lineare e permette all'utente di visualizzare le scene di parcheggio automatizzato e di rilevamento automatico dei pedoni e di provare l'auto manualmente in una scena dedicata.

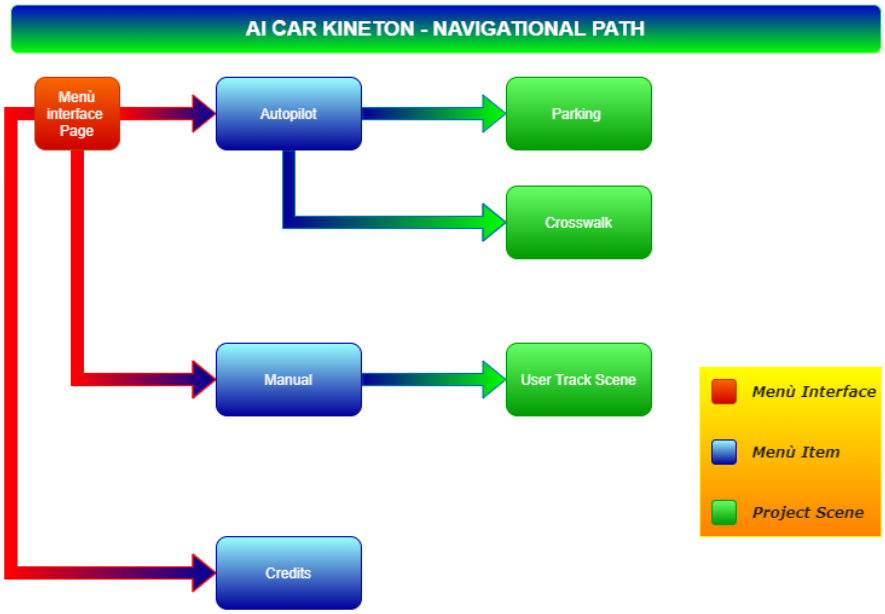


FIGURA 66 - PATH DI NAVIGAZIONE DEL MENU

Il menù è molto dinamico siccome fa utilizzo di piccole e simpatiche animazioni che rendono il tutto molto più gradevole. Le animazioni sono state realizzate giocando sulla sottolineatura del testo, sulla trasparenza e sulle posizioni, quindi su attributi primitivi del testo; il tutto viene gestito tramite due strutture create internamente al menù, quali sono il DecoButton e la DecoAnimation, i quali, rispettivamente, facilitano la gestione dei pulsanti e delle animazioni.

La logica del menù, utilizzando queste due strutture, risulta abbastanza semplice:

- Se l'utente naviga in una sezione del menù, il DecoButton sottolinea il testo su cui l'utente si trova;
- Se l'utente clicca un pulsante, vengono create delle DecoAnimation che permettono la visualizzazione dei pulsanti della nuova sezione del menù, facendoli comparire rimuovendo la trasparenza da essi;
- Se l'utente torna nella precedente sezione, vengono create delle DecoAnimation che nascondano i pulsanti della sezione da rimuovere, applicando la trasparenza ad essi.

```
class DecoButton {
    // Internal attributes
    private string text;
    private TMP_Text guiText;
    // Constructor
    public DecoButton(string text, TMP_Text guiText) ...
    /*
        Functions
    */
    public string getText() ...
    public TMP_Text getGUI() ...
    /*
        updateGUI(bool underlined)
            Manages the underlining of the text
    */
    public void updateGUI(bool underlined = false) ...
    /*
        changeAlpha(float alpha)
            Apply a new transparency to the text
    */
    public void changeAlpha(float alpha) ...
    public float getAlpha() ...
    /*
        changeAlphaOfIcon(float alpha)
            Apply a new transparency to the text's icon
    */
    public void changeAlphaOfIcon(float alpha) ...
    public float getAlphaOfIcon() ...
}
```

FIGURA 67 - STRUTTURA DECOBUTTON

```
class DecoAnimation {
    // Internal attributes
    private DecoButton button;
    private bool start;
    public AnimationType type;
    // Constructor
    public DecoAnimation(DecoButton button, bool start, AnimationType type) ...
    /*
        Functions
    */
    public DecoButton getButton() ...
    public bool isStart() ...
    public void setStatus(bool visible) ...
}
```

FIGURA 68 - STRUTTURA DECOANIMATION

```
foreach (DecoButton b in modeButtons) {
    b.changeAlpha(0f);
    b.changeAlphaOfIcon(0f);
    b.updateGUI();
    toAnimate.Add(new DecoAnimation(b, false, AnimationType.FADE));
    backupToAnimate.Add(new DecoAnimation(b, false, AnimationType.FADE));
}
```

FIGURA 69 - ESEMPIO DI UTILIZZO DI DECOBUTTON E DECOANIMATION

La struttura DecoAnimation supporta anche altre animazioni oltre alla comparsa e la scomparsa del testo, come lo spostamento verso l'alto e verso il basso di quest'ultimo. La tipologia di animazione da eseguire viene scelta alla creazione di un'istanza della struttura.

FIGURA 70 – MENÙ INTERATTIVO

5.2 SVILUPPI FUTURI

Il progetto di tirocinio AI Car Kineton sì è fin da subito distinto per Il progetto di tirocinio AI Car Kineton sì è fin da subito distinto per la sua ottima scalabilità, infatti anche durante l'intero iter progettuale, molte sono state le aggiunte e complicazioni apportate alle simulazioni, basti pensare il progressivo ampliamento di veicoli, con il quale l'agente si è misurato nella scena di parcheggio automatico, oppure all'aggiunta di più pedoni o il doppio senso di marcia per la scena di attraversamento pedonale. Oltretutto, il compito fondamentale di un sistema di sensoristica ADAS, è proprio quello di fornire un supporto pronto a rispondere alle più svariate e complesse eventualità riscontrabili in una quotidiana esperienza di guida urbana o extraurbana.

In quest'ottica è molto semplice ragionare in termini di sviluppi progettuali futuri per il progetto implementato, basti pensare alle innumerevoli possibilità di dinamizzazione che ancora possono essere implementate su entrambe le simulazioni:

- Per la scena di parcheggio, ad esempio, è possibile considerare la possibilità di effettuare altre manovre come ad esempio quelle utili alla realizzazione di un parcheggio in colonna. Altra possibilità consiste nel rendere l'autovettura ancora più libera nel suo percorso, avviando la simulazione su strada invece che dall'interno del parcheggio stesso, addirittura rendendo, magari, la locazione del posto libero randomica tra una serie di posti auto precedentemente definiti. Ciò rende la ricerca del posto auto più dinamica;
- Per la scena di attraversamento pedonale, invece, un'ottima possibilità di espansione è ancora presente nell'interazione tra pedoni e agente, ad esempio rendendo dinamiche e casuali le coordinate spaziali o temporali di attraversamento di ogni singolo pedone, rendendo l'agente ancora più versatile e pronto a rispondere correttamente alla casualità con cui un pedone può trovarsi su strada in un contesto reale.

Inoltre, è utile precisare che AI Car Kineton è stato concepito e progettato al fine di rispondere alle più svariate esigenze stradali, in cui un'autovettura è dotata di supporti basati sull'intelligenza artificiale.

Non a caso, come visibile dalla scena adibita alla guida manuale, è stato predisposto anche un incrocio stradale, nel quale sarebbe davvero molto interessante provare a specializzare i sensori di cui l'agente già è dotato, ad esempio implementando un sistema per il riconoscimento della segnaletica stradale, un meccanismo di precedenze tra varie autovetture, oppure una simulazione adibita alla circolazione rotatoria.

Infine, è importante dare uno sguardo alla possibilità di esportare e adattare le Neural Networks prodotte in un contesto di simulazione differente rispetto a quello utilizzato: infatti le Neural Networks prodotte da ML Agents possono essere facilmente esportate e adattate a piccoli prototipi di veicoli reali. Insomma, sarebbe addirittura possibile considerare l'ipotesi di uscire dall'ambito di simulazione e di passare all'ideazione di un piccolo progetto di automazione in ambito robotico, con l'ausilio di sensoristiche di vario tipo atte a simulare sempre di più il comportamento di un reale sensore ADAS da installare su un'autovettura.

Nonostante le proposte riportate, oggigiorno la sicurezza stradale necessita davvero tanto di supporti di sensoristica ADAS per varie realtà che un conducente può affrontare e dell'applicazione di Machine Learning e Intelligenza Artificiale, le quali garantiscono un ottimo grado di affidabilità. A tale scopo, AI Car Kineton lascia ampli spazi di libertà a chiunque voglia cimentarsi in questo enorme mondo, che è la sicurezza urbana, dando la possibilità di integrare, tramite una vasta gamma di scelte, le più svariate situazioni che necessitano di automazione e sicurezza nel complesso mondo dell'automotive.

BIBLIOGRAFIA

- [1] Cosa sono i sistemi di sicurezza ADAS? - Articolo Illustrativo automobile.it https://www.automobile.it/magazine/come-funziona/adas-sistemi-avanzati-assistenza-guida-3382
- [2] Adas, quali sono i sistemi di assistenza alla guida - Articolo Illustrativo https://www.newsauto.it/guide/adas-quali-sono-significato-2020-111281/
- [3] Machine Learning - Definizioni Generali e tipologie di learning maggiormente conosciute - https://www.intelligenzaartificiale.it/machine-learning/
- [4] Neural Network – Deep AI https://deepai.org/machine-learning-glossary-and-terms/neuralnetwork#:~:text=An%20artificial%20neural%20network%20learning,output%2C%20usually%20in%20another%20form.
- [5] Wikipedia – Deep Reinforcement Learning https://en.wikipedia.org/wiki/Deep_reinforcement_learning
- [6] Game engines—how do they work? - https://unity3d.com/what-is-a-game-engine
- [7] Jared Halpern, Developing 2D Games with Unity - https://www.amazon.it/Developing-Games-Unity-Independent-Programming-ebook/dp/B07FKFVTML
- [8] Sue Blackman, Unity for Absolute Beginners - https://www.amazon.it/Unity-Absolute-Beginners-English-Blackman-ebook/dp/B01HXKK5DG
- [9] Coding in C# in Unity for beginners - https://unity3d.com/learning-c-sharp-in-unity-for-beginners
- [10] Juliani, A., Berges, V., Teng, E., Cohen, A., Harper, J., Elion, C., Goy, C., Gao, Y., Henry, H., Mattar, M., Lange, D. (2020). Unity: A General Platform for Intelligent Agents. arXiv preprint arXiv:1809.02627. https://github.com/Unity-Technologies/ml-agents.
- [11] Tensor Flow Docs - https://www.tensorflow.org/