Università degli Studi di Salerno



**Dipartimento di INFORMATICA**

**tesi di laurea**

**in**

**INFORMATICA**

***Intelligenza Artificiale per il rilevamento dei pedoni in ambiente simulato***

**Relatore: Candidato:**

***Prof. Andrea Francesco Abate******Ferrara Carmine***

***Matr.05121/05255***

**ANNO ACCADEMICO 2019/2020**

Ringraziamenti

quotidianamente

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 ML-Agents, 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 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 - 6 -](#_Toc46229808)

[La Problematica - 6 -](#_Toc46229809)

[Intelligenza Artificiale e Automazione del veicolo - 7 -](#_Toc46229810)

[Sistema Proposto - 8 -](#_Toc46229811)

[Struttura del Documento - 8 -](#_Toc46229812)

[Capitolo 1 – Paradigmi e Metodologia - 9 -](#_Toc46229813)

[1.1 Sistemi ADAS - 9 -](#_Toc46229814)

[1.2 Machine Learning e metodologie di approccio utilizzate - 11 -](#_Toc46229815)

[1.3 Neural Networks and Machine Learning - 12 -](#_Toc46229816)

[1.4 Reinforcement Learning e Reti Neurali - 13 -](#_Toc46229817)

[1.5 Reinforcement Learning e Sistemi ADAS - 14 -](#_Toc46229818)

[1.6 Game Engine e Sviluppo di Simulazioni - 15 -](#_Toc46229819)

[Capitolo 2 – Tecnologie e Strumenti utilizzati - 19 -](#_Toc46229820)

[2.1 Unity 3D - 19 -](#_Toc46229821)

[Bibliografia - 22 -](#_Toc46229822)

# 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 e proposte di soluzione a queste domande 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 ed in particolar modo dell’automazione rende possibile, da più di una decina, la creazione di sistemi automatizzati che rendono il mezzo di trasporto in grado di assistere il guidatore nelle scelte corrette da avere in qualsiasi situazione esso si possa trovare.

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 su strada 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, dalla corretta esecuzione di un parcheggio, una corretta valutazione in corrispondenza di un semaforo, il rispetto della segnaletica stradale, regolazione di velocità del mezzo in relazione alle condizioni atomosferiche fino al 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, l’utilizzo di sensori per il riconoscimento di 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 un 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 il corretto comportamento da avere sia per mantenere una corretta 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 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 un sensore ADAS per il corretto movimento su strada e il riconoscimento dei pedoni quando essi sono situati in prossimità dell’auto. Per 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 nella simulazione.

## Struttura del Documento

Il documento di tesi sarà articolato nel seguente modo:

* Capitolo 1 – Paradigmi e Metodologia utilizzata – Principali paradigmi e tecniche alla base del progetto implementato, e interconnessioni tra le stesse;
* Capitolo 2 – Tecnologie Implementative e strumenti di sviluppo utilizzati;
* Capitolo 3 – Progettazione e implementazione:
  + Creazione della scena, predisposizione dell’agente di guida e scelte progettuali;
  + Scena 1 – Predisposizione dell’agente in una scena semplificata;
  + Scena 2 – Movimenti realistici dell’auto e complicazioni;
  + Scena 3 – Miglioramenti finali e predisposizione al comportamento ottimale;
  + Considerazioni e Sviluppi futuri.
* Bibliografia.

# Capitolo 1 – Paradigmi e Metodologia

## 1.1 Sistemi ADAS

Al giorno d’oggi ogni auto di nuova fabbricazione sono dotate di si 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 alla guida 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].

Come suggerisce il significato più profondo del termine i sistemi ADAS (Advanved Driver Assistence System per l’appunto) sono sistemi elettronici di sensoristica 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 ma di rilevante importanza per il mio progetto, sono state le metodologie di approccio dei sensori:

Figura 1 - Rilevazione Automatica di un pedone su strada

* 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 comunicano al “cervello dell’auto” quando esse 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.

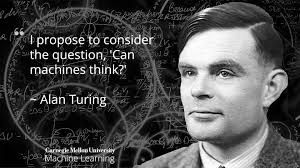
Tanti sono i progetti di applicazione futura per i sensori ADAS, in particolare sono in studio sistemi che permetteranno all’auto di eseguire valutazioni e azioni con livelli di autonomia, pur richiedendo la presenza del pur richiedendo la presenza del pilota a bordo, per intervenire in caso di necessità. Far avere maggiore autonomia ad un’autovettura significa spingerla sempre di più all’acquisizione di coscienza di se e dell’ambiente circostante, la sfida è molto ampia e spazia dal riconoscimento della tridimensionalità di pericoli o altri veicoli, fino alla pura valutazione autonoma delle situazioni di pericolo, a dimostrazione che questo discorso è sempre meno utopistico è il progressivo aumentare dei formalismi e degli studi fatti in materia 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).

Figura 2 - Tipico esempio di frenata automatica tramite ausilio ADAS

* 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. Il campo di applicazioni del machine learning è molto complesso in quanto esso stesso prevede differenti modalità, strumenti e tecniche per essere realizzato. Inoltre, le differenti tecniche di apprendimento e sviluppo degli algoritmi danno vita ad altrettante possibilità di utilizzo che allargano il campo dell’apprendimento automatico rendendone difficile una definizione specifica. Tuttavia si può dire che, quando si parla di machine learning, il fondamento comune è un insieme di **meccanismi che permettono ad un automa intelligente di migliorare le proprie capacità e prestazioni nel tempo**. Date quindi una serie di notazioni primitive (definite tramite algoritmo), l’agente autonomo saprà prendere una specifica decisione piuttosto che un’altra o effettuare azioni apprese nel tempo.

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 alla base di sofisticate forme di reti neurali), 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 una serie di 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 tra i diversi paradigmi fondanti del machine learning, si possono vedere come:

* 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 prevede invece che le informazioni inserite all’interno della macchina non siano codificate, ossia la macchina ha la possibilità di attingere a determinate informazioni senza avere alcun esempio del loro utilizzo. Dovrà essere la macchina stessa, quindi, a 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 metodologia 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 Neural Networks and 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.

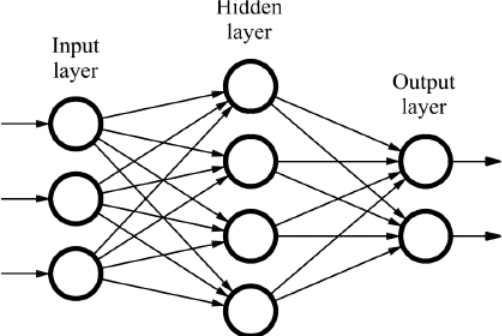
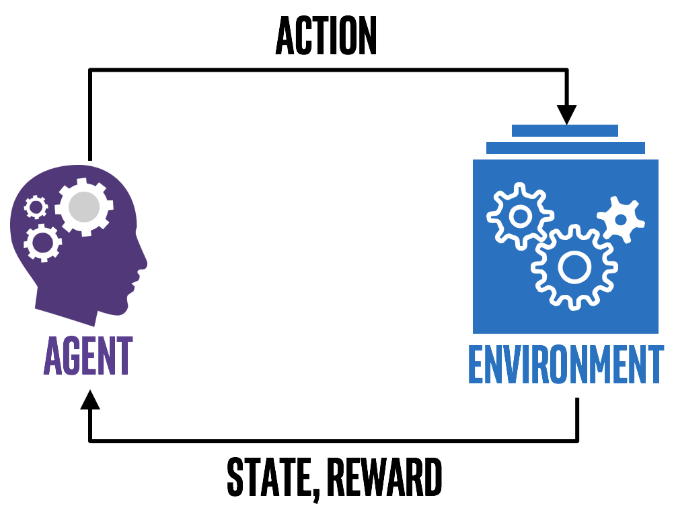
Le reti neurali possono essere viste come un insieme di strumenti e approcci usati in un algoritmo di machine learning, le quali vengono perfezionate solo tramite l’acquisizione di nuove esperienze derivanti dall’acquisizione di nuove esperienze tramite la fase apprendimento, nella la quale l’agente acquisisce coscienza ed esperienza dell’ambiente circostante, e sulla base delle informazioni in suo possesso ne elabora di nuove e agisce in maniera sempre più minuziosa e specifica.

Figura 3 - Esempio di Rete Neurale

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



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.

Figura 4 - Schema Base dell'apprendimento per rinforzo

* **Agente** – Sinonimo per indicare la macchina o il 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 su un attraversamento pedonale posto davanti alla traiettoria del mezzo. Scopo del training di Reinforcement Learning, 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.

## Game Engine e Sviluppo di Simulazioni

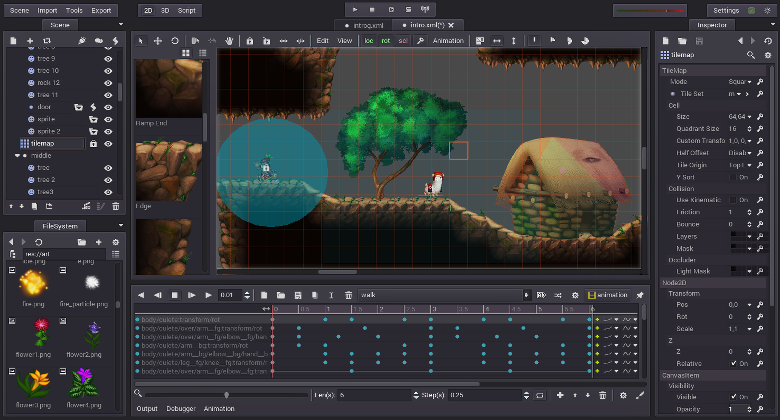
 Prima di introdurre e analizzare gli strumenti e le tecnologie utilizzate per il lavoro progettuale è doveroso fare una breve digressione su come è stato possibile creare un agente virtuale e un ambiente 3D da utilizzare come Environment per la simulazione di Machine Learning. In particolare, è doveroso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

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:

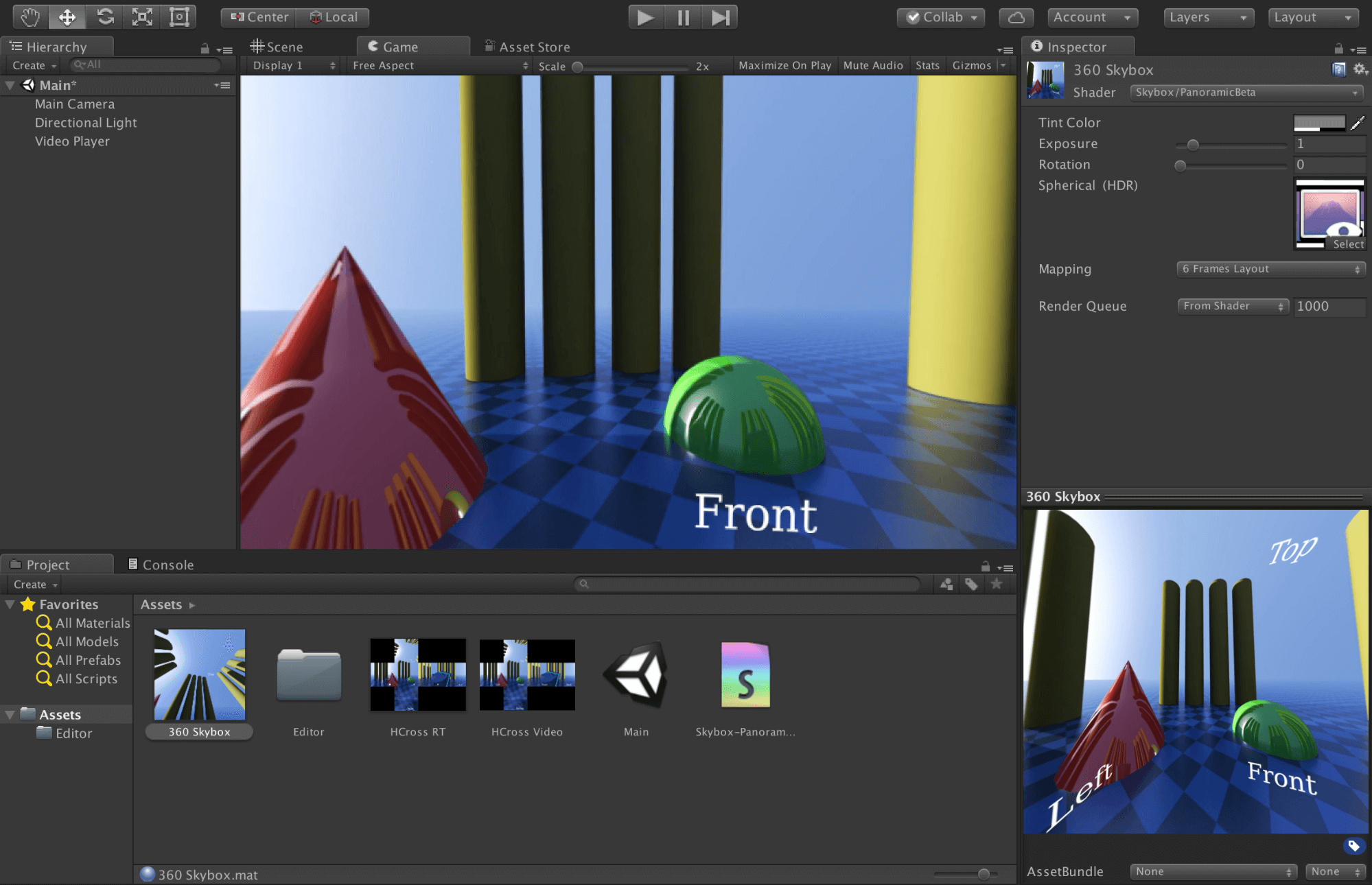
* La gestione della grafica: i moderni motori grafici (Unreal Engine, Unity, …) forniscono API molto potenti per l’organizzazione e l’interconnesione di contenuti grafici (modelli 3D ad esempio) in una scena a due o tre dimensioni, la cui resa grafica non pecca di basse prestazioni, data le complesse capacità di rendering visuale di cui i moderni game 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 una 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;

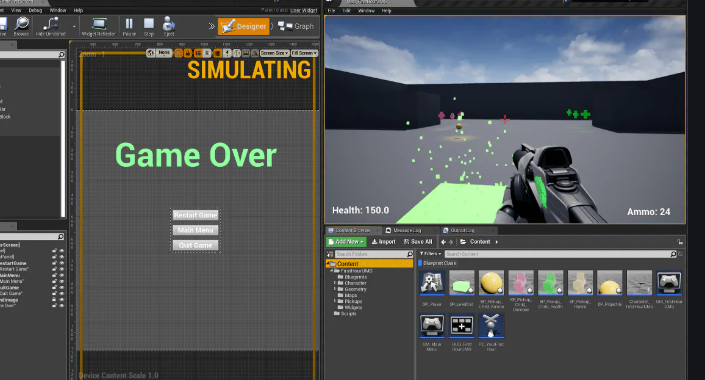


Figura 8 -Unreal Engine UI Definition

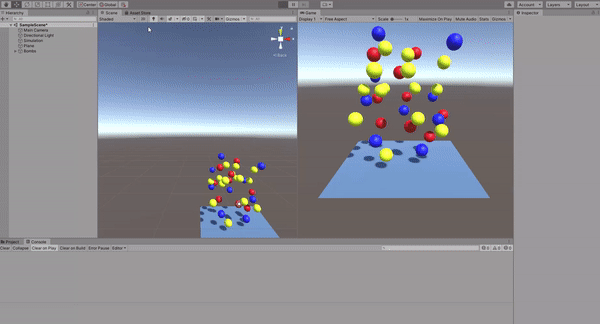
* Un aspetto cruciale di un game engine è rappresentato dagli script che definiscono la logica comportamentale delle componenti di gioco tramite l’ausilio di elementi di programmazione. Tramite lo scripting di un Game Engine, è facilmente possibile intercettare qualsiasi parametro o componente di un 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 le 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 base alle quali l’oggetto reagirà in maniera differente a contatto 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 game engine permettono di creare e manovrare comportamenti 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 ottimo Game Engine per lo sviluppo videoludico, il quale offre molteplici funzionalità gratuitamente. Esso permette ai Game Creators di sperimentare le proprie idee, imparare e vendere giochi.

Unity permette la modellazione di videogiochi fondamentalmente tramite oggetti detti GameObjects, ai quali può essere assegnato un comportamento ben definito dal programmatore tramite script. Ogni GameObject è caratterizzato da una serie di attributi comuni utili alla sua disposizione nello spazio virtuale, come la posizione su coordinate e la sua rotazione; può essere definito e personalizzato tramite l’utilizzo di textures e materiali, scripts e componenti inerenti alla fisica [8].



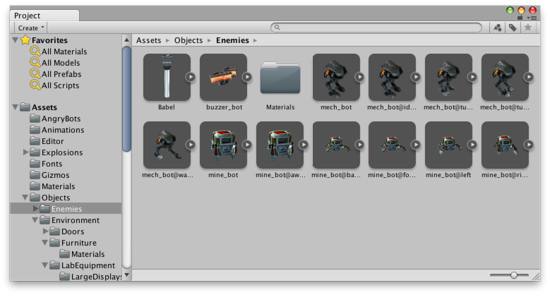
*Figura 10 - Unity logo*

Unity, tramite la sua semplice interfaccia, permette al Game Creator di gestire tutte le sfaccettature del proprio videogioco. Ogni sottosezione dell’interfaccia si dedica ad una particolare caratteristica di ciò che si vuole modellare:

* Gerarchia: visualizza tutti gli oggetti di cui la scena corrente è composta in formato gerarchico; permette la gestione di GameObjects esistenti e la creazione di nuovi tramite apposito menu a tendina. I GameObjects sono raggruppabili anche in strutture padre-figlio, in modo da ereditare il comportamento di GameObjects generici;

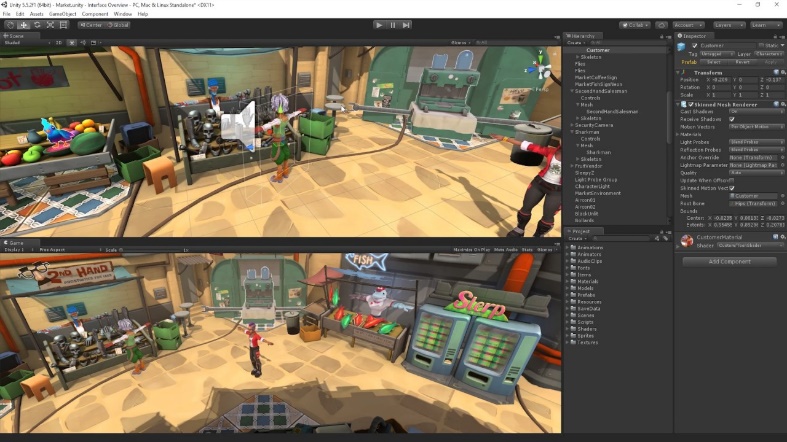
Figura 11 - Gerarchia

* Finestra Progetto: offre un’anteprima, permette l’organizzazione e la gestione di tutti i contenuti della cartella Assets, quindi di tutte le risorse che comporranno il videogioco. Inoltre, mostra tutte le librerie importate nel progetto;



*Figura 12 - Project Window*

* Scene view: finestra dell’editor che consente la modellazione di un ambiente simulato, detto anche scena, interagendo dinamicamente con i GameObjects presenti, modellandone caratteristiche e dimensioni in maniera manuale tramite appositi strumenti;
* Game view: finestra che renderizza in tempo reale ciò che accade nella scena durante l’esecuzione del videogioco, dal punto di vista della camera di gioco attualmente attiva. In parole povere, simula nell’editor ciò che accadrebbe durante l’esecuzione della build del gioco;
* Inspector: permette la visualizzazione e la gestione di tutte le caratteristiche inerenti ad un preciso GameObject, quindi permette la gestione degli scripts, dei colliders, della posizione, della rotazione e così via;



*Figura 13 - Scene view, Game view e Inspector*

* Asset Store: negozio integrato in Unity che permette l’acquisto o il download gratuito di risorse per il videogioco create dalla comunità [7].

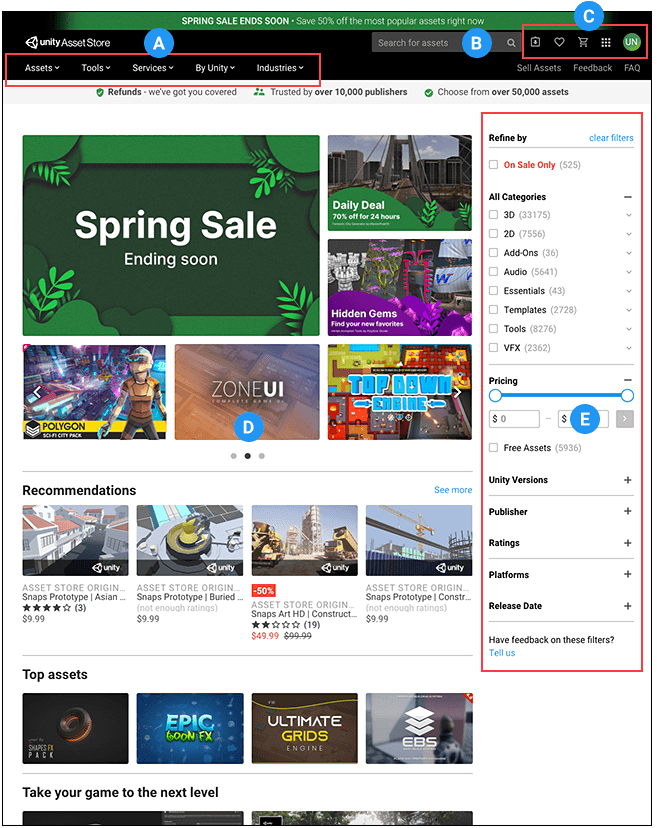


Figura 14 - Unity Asset Store

# 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>