

# Intelligenza Artificiale: Metodi ed Applicazioni

Gran Premio MIVIA 2025

## GRUPPO 03

Claudia Montefusco: 0612707404  
Rossella Pale: 0612707284  
Alessio Leo: 0612707279  
Emanuele Tocci: 0612707488



# Indice

01

Introduzione

03

Organizzazio  
ne codice

02

Scelte  
Progettuali

04

Problematic  
e

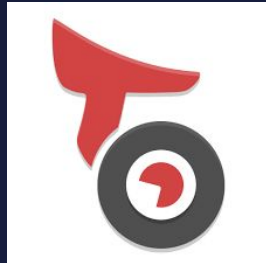


01

# Introduzione

# OBIETTIVO

- Sviluppare un sistema di guida autonoma all'interno dell'ambiente TORCS (The Open Racing Car Simulator)
- Completare un giro senza avversari, nel minor tempo possibile, mantenendo una traiettoria ottimale
- Requisiti: controllo del mezzo, nessun incidente



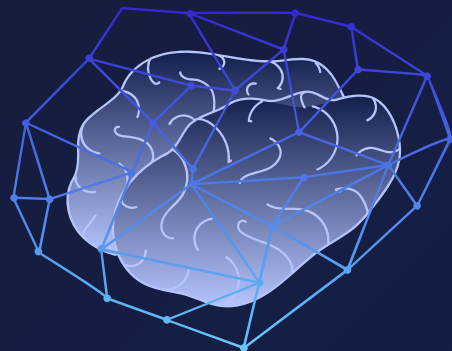
# APPROCCIO BEHAVIORAL CLONING

Il Behavioral Cloning é una tecnica di **apprendimento supervisionato** che imita le decisioni umane osservando le risposte del pilota a particolari input sensoriali.

Consente di addestrare un **classificatore** imitando le decisioni di guida di un pilota umano a partire dai dati sensoriali. In questo modo, l'intero processo decisionale viene appreso direttamente dai **dati**.

## Limitazioni:

- Alta dipendenza dalla qualità e varietà del dataset;
- Scarsa capacità di generalizzazione in situazioni nuove;
- Piccoli errori possono amplificarsi progressivamente, portando il veicolo fuori traiettoria



# FASI DELLA PROGETTAZIONE



## Raccolta dati

Un pilota umano guida nel simulatore TORCS mentre il sistema registra sensori e comandi.



## Creazione Dataset

I dati raccolti vengono puliti, normalizzati e salvati in file CSV.



## Implementazione agente

Con l'uso del classificatore KNN, l'agente confronta i dati in tempo reale con quelli del dataset per decidere come agire.



## Test

L'agente guida in autonomia: si valuta se riesce a completare i giri in modo fluido e senza intervento umano.



02

# Scelte Progettuali

# SCELTA DEL CLASSIFICATORE KNN E DEL PARAMETRO "K"

Il classificatore si basa su un approccio **supervisionato**:

- dato un nuovo campione da classificare, si cercano i k **campioni più vicini** presenti nel dataset di addestramento, utilizzando una struttura KD-Tree per ottimizzare la ricerca.
- La classe del campione in input viene determinata tramite **voto di maggioranza** tra le classi dei k vicini trovati.
- Fino a  $K=3$  si ottengono risultati accettabili



# SCELTA DELLE FEATURES

Abbiamo scelto solo le feature essenziali per garantire efficienza e prestazioni del classificatore:

- 7 Sensori di distanza dalla pista (da  $-30^\circ$  a  $+30^\circ$ ).
- Posizione laterale rispetto al centro della pista
- Angolo rispetto all'asse pista
- Velocità longitudinale
- Velocità laterale

track4	track6	track8	track9	track 10	track 12	track 14	track Pos	angle	speedX	speedY
--------	--------	--------	--------	-------------	-------------	-------------	--------------	-------	--------	--------

03

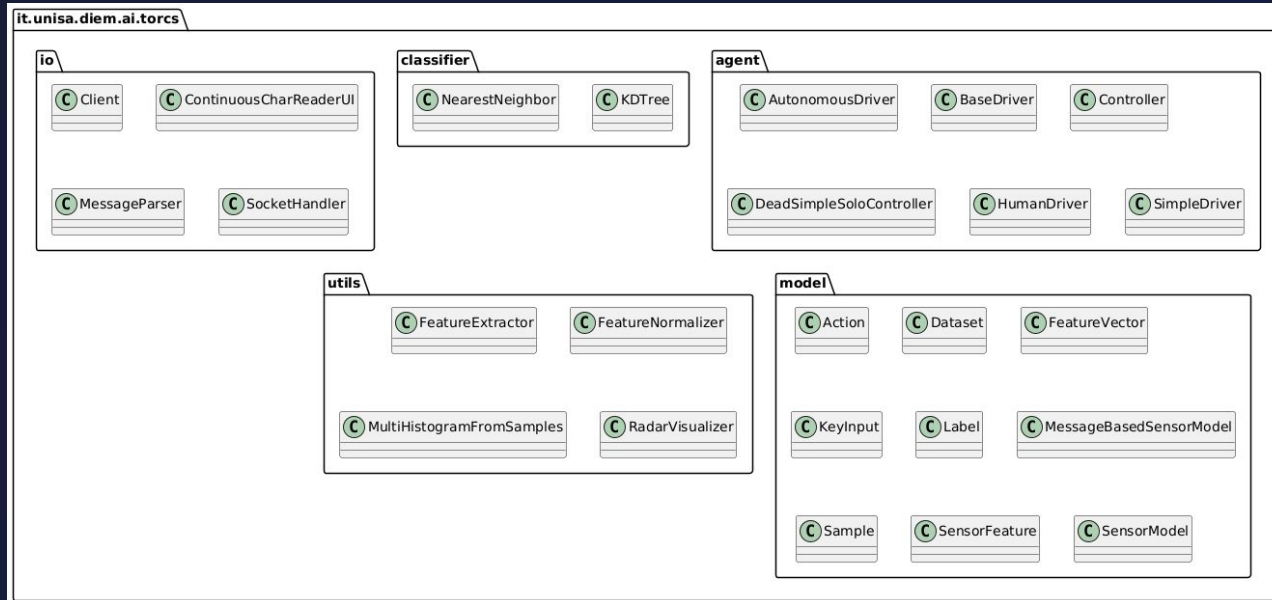
# Organizzazione codice



# ORGANIZZAZIONE DEL CODICE

Organizzazione modulare:

- Separazione in package funzionali
- Focus su leggibilità, manutenibilità e riuso del codice



# AGENTI DI GUIDA



## SimpleDriver

Guida autonoma mediante regole fisse



## HumanDriver

Guida manuale, essenziale per la raccolta dei dati



## AutonomousDriver

Guida autonoma mediante classificatore KNN

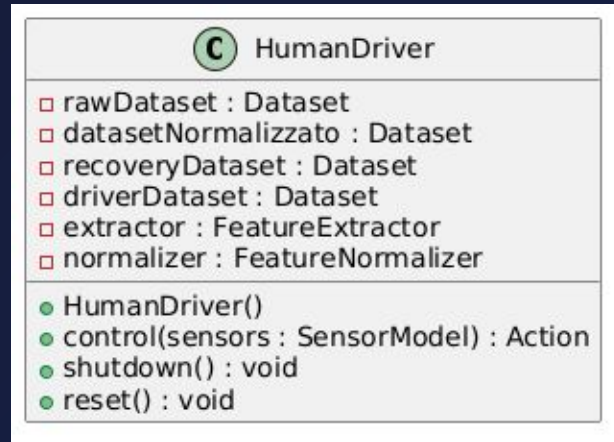
Tutti condividono una base comune, la classe `BaseDriver`, che fornisce infrastrutture per accelerazione, frenata, sterzata, cambio marcia, ABS, frizione automatica...

Questa struttura modulare facilita il riuso del codice nei diversi agenti.

# AGENTE DI GUIDA: HumanDriver

HumanDriver consente la guida manuale tramite tastiera e si occupa della raccolta dati per l'addestramento:

- Controllo manuale;
- Raccolta dati: crea sia dati grezzi che normalizzati, distinguendo tra guida normale e recupero.
- Salvataggio automatico



Questo agente è stato essenziale per generare i dataset che alimentano il modello di guida autonoma.

# RACCOLTA DEI DATI: HumanDriver

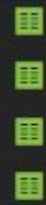
Il pilota umano controlla il veicolo manualmente tramite tastiera (WASD + spazio).  
Durante la guida:

- Vengono acquisiti i dati sensoriali del veicolo.
- Ogni azione viene etichettata con una label e registrata.
- I dati vengono normalizzati e salvati in 4 dataset.
  -

Questo rappresenta la base su cui si costruisce la conoscenza dell'agente.

I 4 dataset sono:

- dataset contenente l'insieme completo dei dati grezzi;
- dataset contenente l'insieme completo dei dati normalizzati;
- dataset contenente un sottoinsieme di dati utilizzati per la guida all'interno della pista;
- dataset contenente un sottoinsieme di dati utilizzati per la gestione della modalità Recovery;



```
dataset_normalizzato.csv
driver_dataset.csv
raw_dataset.csv
recovery_dataset.csv
```

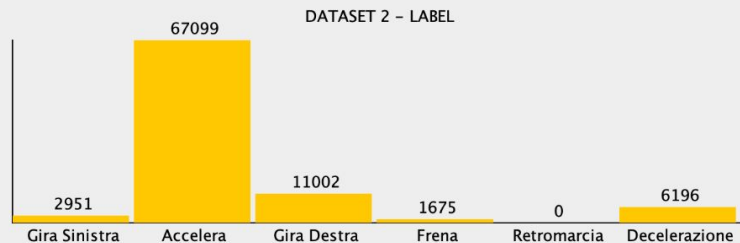
# DATASET USATI

## Dataset 1:

recovery\_dataset.csv,  
una raccolta di dati che mirano  
a riprendere il controllo  
dell'auto fuoripista, gestire  
situazioni critiche e tornare  
in pista

## Dataset 2:

driver\_dataset.csv,  
un dataset basato su  
uno stile di guida  
normale per quando  
l'auto si trova in pista



# AGENTE DI GUIDA: AutonomousDriver

L'agente autonomo consente la guida automatica della vettura mediante i classificatori implementati:

- **Condizioni Normali:** utilizza il classificatore driverKNN e il dataset driver\_dataset.csv
- **Fuori Pista:** utilizza il classificatore recoveryKNN e il dataset recovery\_dataset.csv





# CLASSI UTILITARIE

Il package `utils` raccoglie strumenti di supporto progettati per facilitare il preprocessing dei dati sensoriali e migliorare la robustezza del sistema.

**FeatureExtractor**, si occupa dell'estrazione ordinata delle feature rilevanti dallo stato sensoriale del veicolo (**SensorModel**), utilizzando un set predefinito di variabili selezionate tramite **SensorFeature**.

**FeatureNormalizer**, che applica una **normalizzazione Min-Max** ai dati estratti, riportando ogni valore numerico in un range  $[0,1]$ .

Questo processo include anche un controllo sui valori fuori range, che vengono "**clippati**" per evitare distorsioni nei dati.

Il risultato è un **FeatureVector normalizzato**, compatibile con i classificatori KNN.

$$x_{norm} = \frac{x - min}{max - min}$$

# CLASSI UTILITARIE

Durante la progettazione abbiamo riscontrato diverse difficoltà legate alla gestione dei dati, motivo per cui è stato incluso anche un sotto-package **debugging**.

- **RadarVisualizer** rappresenta graficamente i sensori di bordo pista.
- **MultiHistogramFromSamples** consente un'analisi comparativa delle distribuzioni delle feature in più dataset.

- 

Questi strumenti si sono rivelati **molto utili** durante le fasi di raccolta e validazione dei dati, permettendoci di rilevare facilmente **anomalie** e correggere eventuali discrepanze nel comportamento del veicolo



04

Problematiche

# SOLUZIONI AI PROBLEMI



## Scelta dei sensori

- Utilizzare 7 sensori di track;
- Aggiunta della velocità laterale per risolvere l'instabilità in curva



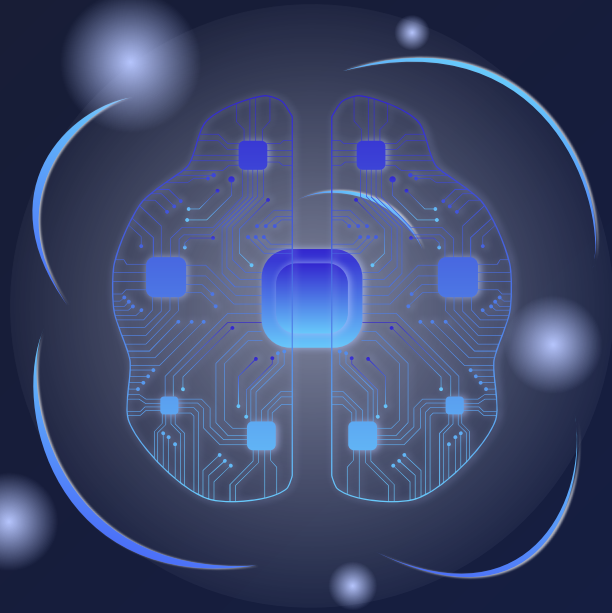
## Dataset misto

- Separazione dataset
- Due classificatori KNN distinti: uno per la guida normale uno per il recovery



## Performance scarse

- Test iterativi e selezione dataset bilanciato



# FINE

GRUPPO 03