

**DOCUMENTAZIONE**

Progetto di sistemi di Telemedicina

GRAN PRIX:

Bettini Rachele [rachele.bettini01@universitadipavia.it](mailto:rachele.bettini01@universitadipavia.it)

D’Amato Cristina [cristina.damato01@universitadipavia.it](mailto:cristina.damato01@universitadipavia.it)

Del Gatto Eleonora [eleonora.delgatto01@universitadipavia.it](mailto:eleonora.delgatto01@universitadipavia.it)

Draghi Barbara [barbara.draghi01@universitadipavia.it](mailto:barbara.draghi01@universitadipavia.it)

Pellegrino Federica [federica.pellegrino01@universitadipavia.it](mailto:federica.pellegrino01@universitadipavia.it)

Rotalinti Ylenia [ylenia.rotalinti01@universitadipavia.it](mailto:ylenia.rotalinti01@universitadipavia.it)

Sommario

[Descrizione 3](#_Toc517965861)

[Personalizzazione della pista 6](#_Toc517965862)

[Modalità UserControlledCar 8](#_Toc517965863)

[Help per l’esecuzione dell’applicazione 10](#_Toc517965864)

[Algoritmi degni di nota 12](#_Toc517965865)

[updateRanking() 12](#_Toc517965866)

[Agenti 13](#_Toc517965867)

[Supervisor 13](#_Toc517965868)

[CAR 13](#_Toc517965869)

[Diagramma delle comunicazioni 15](#_Toc517965870)

[Creazione agenti 15](#_Toc517965871)

[Start game 15](#_Toc517965872)

[Movimento 15](#_Toc517965873)

[Collisioni 16](#_Toc517965874)

[End game 16](#_Toc517965875)

[Resume game 17](#_Toc517965876)

[Diagramma delle classi 18](#_Toc517965877)

# Descrizione

L’obiettivo del nostro lavoro è stato quello di simulare, attraverso un’applicazione Java, una gara di Formula1 alla quale potessero partecipare da una a tre macchine.

La pista sulla quale queste gareggiano è stata modellizzata come una lista ordinata di coordinate continue di paletti (interni ed esterni), che delimitano la strada vera e propria.

Tutto ciò che non è compreso tra questi paletti è considerato fuoripista.

Sono state realizzate due piste predefinite: una più semplice mostrata in Figure 1 e una più complessa, in Figure 2.



Figure Pista più complessa

Figure Pista più semplice

Entrambe sono modificabili a discrezione dell’utente (Figure 3) che può cancellare a piacere dei paletti e aggiungerne successivamente di nuovi, creando tragitti inediti.

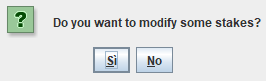


Figure Possibilità di modificare le piste predefinite

Cliccando il bottone “draw your racetrack” nell’interfaccia iniziale, è possibile disegnare una pista personalizzata; i vincoli da rispettare per disegnare la pista vengono riportati nella sezione Personalizzazione della pista. L’interfaccia iniziale richiede inoltre di definire il numero di macchine che si intende far gareggiare ed il numero di giri della gara.

Nel momento in cui si istanzia un nuovo veicolo, è necessario assegnargli un nome, in modo tale da identificarlo durante lo svolgimento del gioco (Figure 4).

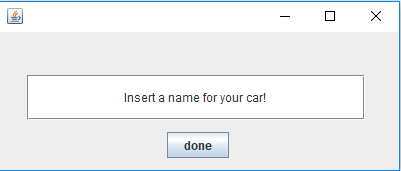


Figure Inserimento dell'identificativo della macchina

Quando vengono create le macchine, queste vengono posizionate ordinatamente dietro la linea bianca che indica il traguardo e gli viene associato l’identificativo (Figure 5).

Figure Posizionamento iniziale delle macchine

Ci sono due tipologie di macchina:

* Autonoma
* Controllata dall’utente

La macchina autonoma prende delle decisioni riguardo al punto target da seguire grazie ad un algoritmo illustrato in seguito. La macchina controllata, invece, riceve il punto target direttamente dall’utente.

A lato del pannello nel quale si svolge la gara, è presente una classica delle macchine che si aggiorna in tempo reale (Figure 6). Il nome mostrato nella classifica è quello impostato all’inizio per il singolo veicolo a cui è aggiunta fra parentesi la tipologia della macchina

(A: automatica, U: controllata dall’utente).

Inoltre viene mostrato il numero di giri che mancano per terminare la gara, calcolato in base al giro della macchina in prima posizione.

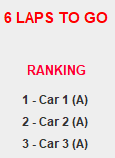


Figure Classifica in tempo reale e numero di giri mancanti alla fine della gara

Quando tutte le macchine hanno percorso il numero di giri indicato inizialmente da interfaccia, la gara termina e appare un pannello nel quale è presente la classifica finale (Figure 7).

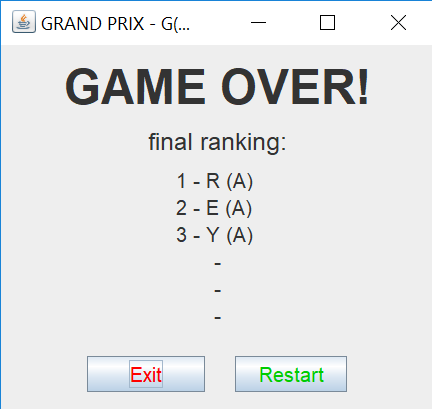


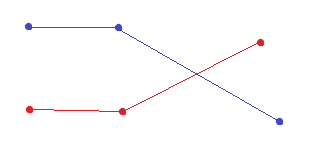
Figure Classifica finale

# Personalizzazione della pista

Per personalizzare la pista abbiamo implementato due funzionalità:

1. **Disegnare la pista da zero**: cliccando *Draw your Racetrack* nell’interfaccia iniziale, è possibile modellizzare la pista a proprio piacimento. Per farlo bisogna cliccare con il mouse sul pannello aperto nel punto in cui si vuole posizionare un determinato paletto, procedendo prima con quelli esterni e successivamente con gli interni.

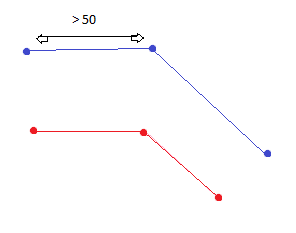
Per il corretto funzionamento del programma, l’utente deve disegnare una pista sulla quale le macchine riescano a gareggiare, quindi:

* Non devono esserci degli incroci.
* Non devono esserci dei punti nei quali la pista è troppo stretta (almeno 60 (unità di distanza definita da Java)).



* I paletti devono essere posizionati sufficientemente vicini: abbiamo ragionato in modo tale che una macchina abbia sempre almeno un paletto davanti a sé e uno dietro (quindi la distanza tra due paletti deve essere circa 40)

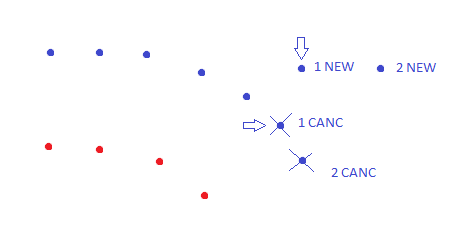
Inoltre i paletti devono essere inseriti in modo ordinato (sequenzialmente).



Per aiutare l’utente è stata disegnata una griglia fittizia, la cui risoluzione è 50.

1. **Modificare una pista di default**: dopo aver scelto una delle tue tipologie di pista proposte, l’utente può decidere se modificarla o meno in alcuni tratti.

Per farlo correttamente è necessario prima cancellare, a partire da un determinato paletto, in senso orario una sequenza di paletti, che si vorranno sostituire con nuovi. A questo punto è possibile aggiungere i nuovi paletti, sempre in sequenza. Procedere prima con i paletti esterni e poi con gli interni, utilizzando la stessa logica.



# Modalità UserControlledCar

È possibile giocare utilizzando la modalità UserControlledCar, ovvero con una macchina guidata dall’utente. Effettuata la run della CAR1U si apre, in aggiunta alla GUI che mostra l’intera gara, un’ulteriore interfaccia. Se la run del Supervisor e la run della UserControlledCar vengono lanciate dallo stesso computer, questa interfaccia appare come in Figure 8.

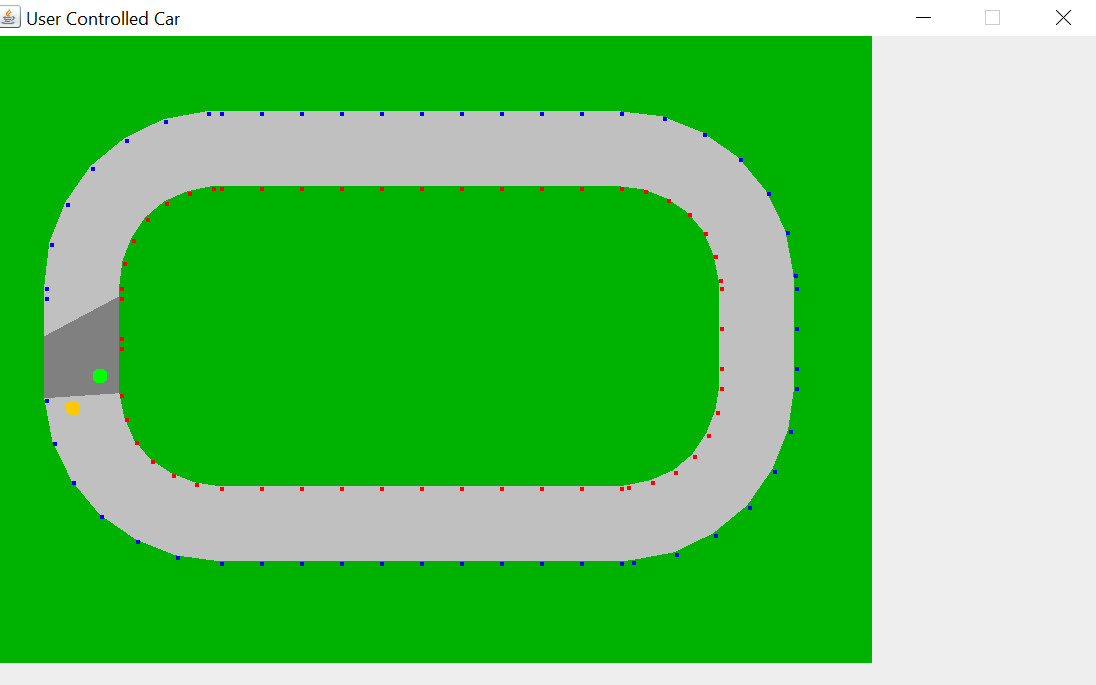
Per motivi grafici abbiamo deciso di mostrare comunque all’utente la pista per intero; per poterlo fare una volta lanciato il Supervisor e deciso la pista con cui giocare, facciamo uno screen della grafica della pista e salviamo l’immagine. Tale immagine viene richiamata nel momento in cui si crea la UserControlledCarGUI. In realtà l’effettiva visuale dell’utente, che corrisponde alla CarView, si limita alla zona evidenziata in grigio scuro.

Figure Interfaccia che appare all’utente quando lancio il supervisor e la macchina dallo stesso computer

Se il Supervisor e l’agente macchina telecomandata dall’utente vengono lanciati da due computer diversi bisogna fare una piccola modifica nel codice: in particolare in UserControlledCarPanel bisogna decommentare le prime due righe del metodo drawBoard() e commentare la terza (Figure 9). Questo perché lo screen dell’immagine della pista non può essere fatto lanciando solamente l’agente UserControlledCar, poiché esso non conosce la pista nel suo insieme.

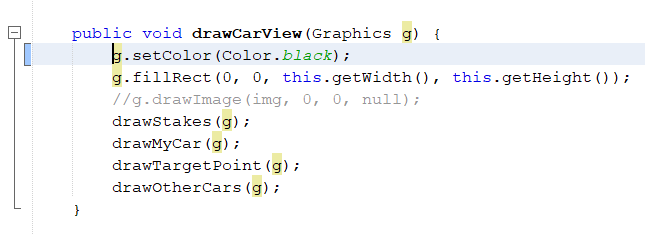


Figure Parte di codice da modificare

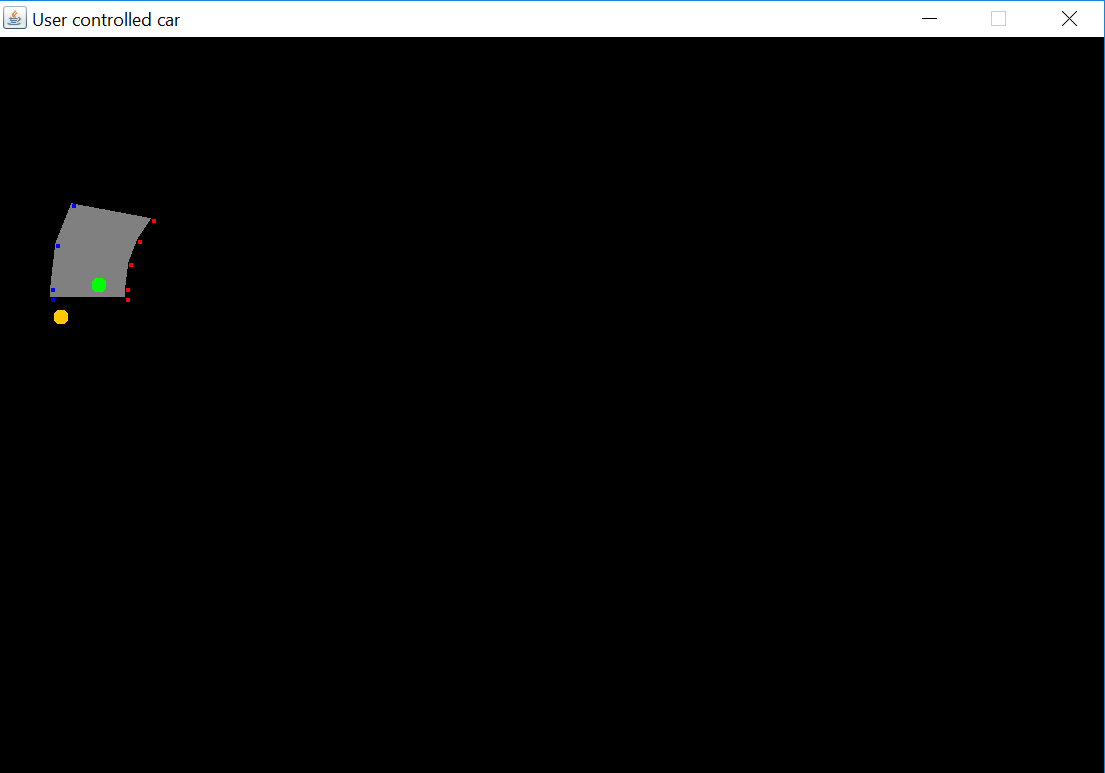
L’interfaccia quindi apparirà in questo modo (Figure 10).

Figure Interfaccia che appare all’utente quando supervisor e agente macchina sono lanciati da due computer diversi

La macchina guidata dall’utente corrisponde al pallino giallo, mentre il pallino verde rappresenta in questo caso l’unica macchina avversaria che si trova all’interno del suo raggio di visione.

Giocando con questa modalità, l’utente può decidere quale sarà il prossimo punto target da raggiungere, selezionandolo con un click del mouse. Il punto selezionato deve però stare all’interno del raggio di visione.

Per come abbiamo scelto di impostare la modalità UserControlledCar, l’utente può solo scegliere il punto target, ma non può decidere né come arrivare lì né il grado di accelerazione/frenatura.

# Help per l’esecuzione dell’applicazione

In tabella sono riportati i parametri che è necessario settare per le diverse configurazioni l fine di garantire il corretto funzionamento dell’applicazione.

|  |  |  |
| --- | --- | --- |
| **Agente** | **Configuration** | **Arguments** |
| Supervisor | SUP | id1 SUP 0 |
| AutonomousCar 1 | CAR1A | id2 CAR A |
| AutonomousCar 2 | CAR2A | id3 CAR A |
| AutonomousCar 3 | CAR3A | id4 CAR A |
| UserControlledCar | CAR1U | id5 CAR U |

Per distinguere le due tipologie di macchina, quella automatica e quella guidata dall’utente, utilizziamo per la prima l’argomento A e per la seconda l’argomento U.

**File di properties**

**Run del Supervisor**

Per il corretto funzionamento del gioco è necessario per prima cosa lanciare il Supervisor (SUP), che causa l’apertura dell’interfaccia iniziale (Figure 11) dalla quale è possibile scegliere il numero di macchine, il numero di giri ed il tipo di pista da utilizzare. Scelta una delle opzioni, si apre la pista senza ancora nessuna macchina. Scelta una delle opzioni, si apre la pista senza ancora nessuna macchina.

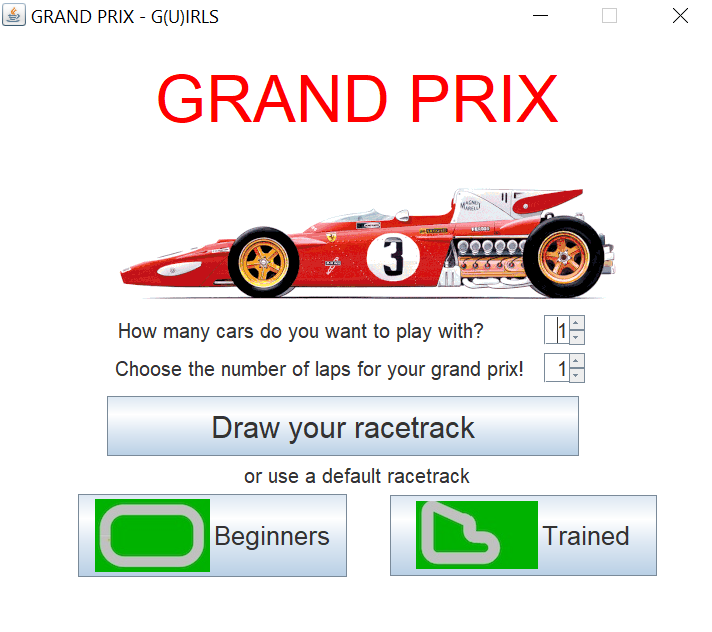


Figure Interfaccia iniziale

**Run degli altri agenti**

Dopo aver lanciato il Supervisor, è possibile lanciare sia l’agente AutonomousCar che l’agente UserControlledCar; una volta assegnato un nome alla macchina, questa verrà posizionata sulla pista.

**Partenza della gara**

Lanciate le diverse macchine, per cominciare la gara bisogna premere il bottone Start. Se durante la corsa si vuole ravviare la gara basta premere il bottone Resume, le macchine verranno quindi riposizionate dietro la linea di partenza. A questo punto sarà necessario cliccare nuovamente il bottone Start.

# Algoritmi degni di nota

# updateRanking()

Tale algoritmo si occupa di aggiornare in tempo reale la classifica in base alla posizione della macchina. Per prima cosa salviamo in una lista le macchine aventi lo stesso currentLap; quindi per ognuna di queste andiamo a selezionare il paletto interno più vicino e ci salviamo il suo indice. A parità di giro, la macchina salvata in prima posizione, sarà quella avente il paletto interno più vicino con indice maggiore.

# Agenti

Analizziamo ora nel dettaglio gli agenti e le rispettive activities.

## Supervisor

Il supervisor ha il compito di gestire la maggior parte delle dinamiche del gioco. Per fare ciò è a conoscenza della posizione di tutte le macchine in gara e della conformazione della pista. Tutte le macchine comunicano al supervisor i loro cambiamenti di velocità e direzione, e lui in risposta, applicando le leggi della fisica, ogni 3 secondi restituisce loro le nuove posizioni e i nuovi raggi di visione.

È il supervisor che, conoscendo le posizioni e il raggio di occupazione di ogni macchina, è in grado di rilevare eventuali collisioni.

Si occupa di gestire le informazioni inviate dall’utente tramite interfaccia, come ad esempio il numero di giri, il numero di macchine e il click dei pulsanti.

Inoltre si occupa di decretare la fine della gara quando tutte le macchine hanno percorso il numero di giri totali e di stilare la classifica che si aggiorna in tempo reale.

Il supervisor comunica con gli agenti macchina tramite la messaggistica e ogni volta che riceve l’activity di SUP\_DO\_BIRTH crea un’istanza di CarPerception. La classe CarPerception rappresenta la percezione che il supervisor ha degli agenti Car ed è di fatto una sorta di “gemello” dell’agente. CarPerception viene utilizzato sia per questioni grafiche sia dal supervisor per tenere traccia delle activity precedentemente inviate dall’agente.

|  |  |  |
| --- | --- | --- |
| **ACTIVITY** | **MITTENTE** | **DESCRIZIONE** |
| SUP\_ACCELERATION | CAR | La macchina notifica al supervisor la sua accelerazione. |
| SUP\_BRAKING | CAR | La macchina notifica al supervisor che sta frenando (solo AutonomousCar). |
| SUP\_STEERING | CAR | La macchina invia al supervisor l’angolo di rotazione e il grado di accelerazione. |
| SUP\_GO\_ON | CAR | La macchina avvisa il supervisor l’intenzione di continuare a muoversi senza modificare velocità e accelerazione. |
| SUP\_DO\_BIRTH | CAR | La macchina notifica la sua nascita e richiede al supervisor la posizione iniziale. |
| SUP\_NOTIFY\_INITIAL\_VIEW | CAR | La macchina notifica al supervisor la ricezione del raggio di visione iniziale. |

## CAR

L’agente AbstractCar è una classe astratta, estesa dagli agenti AutonomousCar e UserControlledCar.

AbstractCar implementa BaseAgent che a sua volta implementa Runnable e contiene il metodo run() che si occupa di gestire le activity in entrata, provenienti dal Supervisor. Quando la macchina nasce, riceve una activity di startup dal broker che la spinge a notificare la sua nascita al supervisor. Il supervisor calcola per la macchina la posizione di partenza e le trasmette il raggio di visione iniziale. A questo punto la macchina attende che le arrivi la notifica di inizio gara, che è inviata dal supervisor quando l’utente clicca il bottone Start nella GUI principale. Una volta ricevuta l’activity di start, la macchina accelera o sterza e individua il suo punto target.

Nel caso della macchina autonoma, come già detto, il target viene individuato tramite un algoritmo mentre la macchina controllata dall’utente osserva il suo UserControlledCarPanel (pattern Observer-Observable) e rileva il punto target indicato dall’utente con un click del mouse. Entrambe le macchine decidono autonomamente, in seguito ad ogni nuovo update della posizione e del raggio di visione inviato dal supervisor con l’activity CAR\_UPDATE\_POSITION\_VIEW, se accelerare o sterzare.

Se avviene una collisione, la macchina riceve la notifica di CAR\_COLLISION, tramite la quale il supervisor le invia la posizione post scontro calcolata tramite un algoritmo pseudo-randomico.

|  |  |  |
| --- | --- | --- |
| **ACTIVITY** | **MITTENTE** | **DESCRIZIONE** |
| CAR\_INITIAL\_POSITION\_VIEW | SUP | Prima activity ricevuta dall’agente da parte del supervisor che le notifica il raggio di visione iniziale. |
| CAR\_START\_RACE | SUP | Il supervisor notifica l’inizio della gara e la macchina, in base al suo raggio di visione, decide cosa fare. |
| CAR\_UPDATE\_POSITION\_VIEW | SUP | Il supervisor invia il nuovo raggio di visione alla macchina in base alla sua nuova posizione. |
| CAR\_END\_GAME | SUP | Il supervisor notifica alla macchina il termine della gara. Di conseguenza essa si deregistra dalle activities e muore. |
| CAR\_COLLISION | SUP | Il supervisor notifica alla macchina circa l’avvenuta collisione, e la macchina aggiorna la sua posizione di conseguenza. |
| CAR\_RESUME | SUP | Il supervisor avvisa la macchina che la gara deve riniziare e le invia nuovamente la posizione iniziale. |

# Diagramma delle comunicazioni

# C:\Users\rachi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\CREAZIONEAGENTI.PNGCreazione agenti

Figure

Subito dopo essere stato creato l’agente Macchina notifica al supervisor la sua nascita tramite l’activity SUP\_CAR\_BIRTH. Il supervisor crea la CarPerception corrispondente all’agente e ne calcola la posizione e la direzione iniziali; quindi aggiunge la CarPerception alla mappa myCars e alla lista carsList presente nella classe RaceTrack; infine le invia il raggio di visione iniziale tramite l’activity CAR\_INITIAL\_POSITION\_VIEW. Una volta ricevuto il raggio di visione, la macchina conferma l’avvenuta ricezione tramite l’activity SUP\_NOTIFY\_INITIAL\_VIEW.

## C:\Users\rachi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\STARTRACE.PNGStart game

Figure

Quando l’utente preme il bottone Start nella GUI principale, il supervisor notifica l’inizio del gioco a tutti gli agenti macchina creati tramite l’activity CAR\_START\_RACE. In base al raggio di visione iniziale precedentemente ricevuto, la macchina decide la prima azione da fare che può essere sterzare o accelerare. Se si trova sul rettilineo invia l’activity SUP\_ACCELERATION, se invece rileva un’inclinazione della pista invia l’activity SUP\_STEERING grazie alla quale oltre ad aumentare la sua velocità modifica anche la sua direzione.

## C:\Users\rachi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\MOVIMENTO.PNGMovimento

Figure

Il supervisor ogni 3 secondi aggiorna la posizione della macchina e il suo raggio di visione, inviando l’activity CAR\_UPDATE\_POSITION\_VIEW. In base al suo nuovo raggio di visione la macchina decide se accelerare o sterzare (SUP\_ACCELERATION e SUP\_STEERING); inoltre l’AutonomousCar, se nel raggio di visione è presente un’altra macchina, può anche decidere di frenare (activity: SUP\_BRAKING). La UserControlledCar non ha questa possibilità, perché spetta all’utente modificare la traiettoria per evitare collisioni con gli altri veicoli. Se la macchina ha raggiunto la sua massima velocità ed è allineata con la direzione della pista, invia l’activity SUP\_GO\_ON; il supervisor quindi calcola la nuova posizione della macchina senza modificarne velocità e direzione.

## C:\Users\rachi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\COLLISIONE.PNGCollisioni

Figure

Quando avviene la collisione tra due macchine, esse ricevono l’activity CAR\_COLLISION dal supervisor. Con questa activity il supervisor notifica loro le nuove posizioni e i nuovi raggi di visione. Dopo lo scontro la macchina è ferma e può inviare l’activity di accelerazione o di sterzatura.

## C:\Users\rachi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\ENDGAME.PNGEnd game

Figure

Nel momento in cui la gara termina, cioè quando tutte le macchine hanno tagliato il traguardo, o quando si chiude l’interfaccia principale, il supervisor notifica a tutti gli agenti macchina la fine della gara attraverso l’activity CAR\_END\_GAME. Quando la macchina riceve questa activity si deregistra e muore; anche il supervisor si deregistra ogni volta che viene interrotto il gioco.

L’activity CAR\_END\_GAME viene anche utilizzata per uccidere un agente macchina lanciata in sovrannumero rispetto al numero di macchine inserito inizialmente.

## C:\Users\rachi\AppData\Local\Microsoft\Windows\INetCache\Content.Word\RESUME.PNGResume game

Figure

Quando l’utente preme il bottone Resume, il supervisor notifica a tutti gli agenti macchina che la gara deve ricominciare attraverso l’activity CAR\_RESUME; rinvia dunque il raggio di visione iniziale nel payload della activity e, come accadeva nella creazione degli agenti, la macchina invia la notifica di avvenuta ricezione del raggio di visione (SUP\_NOTIFY\_INITIAL\_VIEW).

# Diagramma delle classi