Area di progetto

Andrea Peretti Ledi Salillari

I.T.I.S. “Q. Sella” biella

Air Race - descrizione

Il progetto consiste nella creazione di un videogioco di guida in 3 dimensioni. Bisognerà guidare quindi una macchina su delle piste appositamente create. Tali pista saranno delle strutture poste a diversi chilometri di altezza dal suolo e il giocatore dovrà cercare di non cadere in nessun modo.

OBIETTIVI DEL PROGETTO

Le modalità di gioco saranno 3:

- Time attack – Allenamento con vite infinite
- Arcade – Gioco con vite limitate
- Survive – Gioco con una sola vita.

  Il giocatore perde una vita quando cade.

Il giocatore dovrà cercare di raggiungere la fine di ogni pista per poter vincere. Le macchine inserite nel gioco sono due di diversa forma e colore. Quella verde è per i principianti, in quanto più semplice da utilizzare per l'accelerazione minore e la minore velocità massima, mentre quella blu andrà più veloce ma sarà più difficile da controllare. Ogni pista avrà la propria musica così come il menu ne avrà una. La fine della pista sarà identificata da un cubo rosso trasparente. La difficoltà e la lunghezza delle piste aumenta man mano che si avanza.

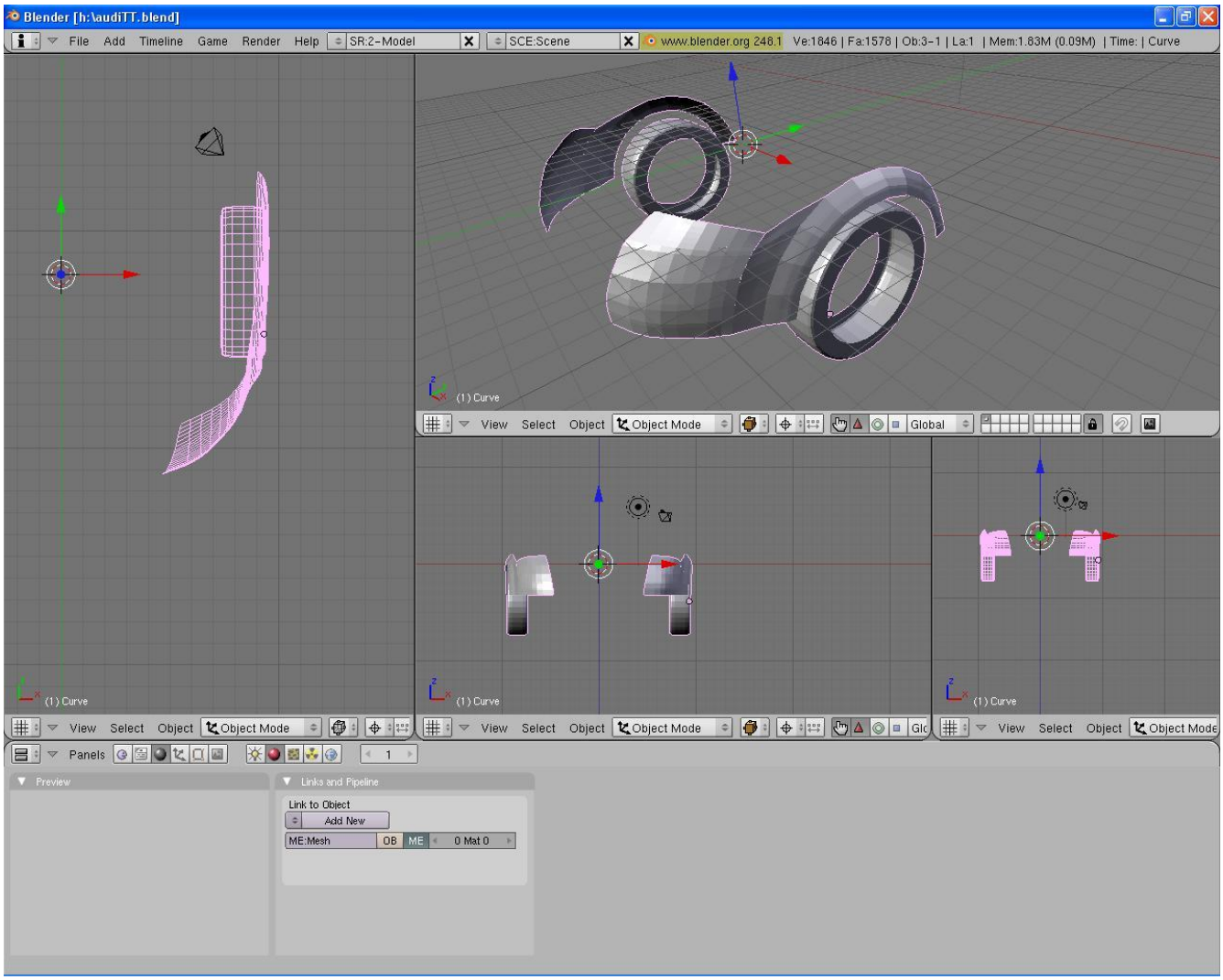
Strumenti utilizzati

Per questo progetto sono stati utilizzati:

- Blender
- Dev C++
- Irrlicht
- IrrKlang

Segue la descrizione di ognuno:

Blender è un programma open source per la modellazione grafica. È stato scelto questo programma per modellare le macchine e le piste in quanto è semplice da utilizzare ma dispone di strumenti e render molto potenti. Con Blender è possibile creare video di animazioni di modelli 3 dimensionali, modellare oggetti e ottenere degli ottimi render.



Questo è un esempio di Blender in azione. La macchina viene realizzata creando tutte le superfici della macchina (il paraurti, i fari, le ruote ecc) che vengono uniti in un unico modello (chiamato anche mesh) alla fine. Dopo la modellazione vengono applicati i materiali delle varie superfici (con colore, trasparenza nel caso dei vetri ecc)

Il risultato finale della modellazione è mostrato dalla seguente immagine. Il modello così creato è pronto per essere incorporato nel videogioco

IRRLICHT

Irrlicht non è né un applicativo né un ambiente di sviluppo. Si tratta di una raccolta di librerie dedicate alla gestione della grafica scritte e utilizzabili in C++ (oltre che da altri linguaggi).

Con Irrlicht è possibile:

- gestire delle mesh in programmi scritti in C++ e rappresentarli graficamente
- applicare delle texture (ovvero delle immagini) ai modelli 3D
- gestire le collisioni tra mesh differenti, l'illuminazione e la grafica applicata al cielo

Irrlicht è composto in 3 parti fondamentali alla creazione del videogioco:

- Scene manager – si occupa di gestire le mesh, la loro posizione, rotazione ecc.
- GUI environment – crea interfacce grafiche
- Video driver – fa funzionare il tutto occupandosi della gestione hardware della scheda video

Gli oggetti in irrlicht vengono definiti come nodi. Il nodo ha una mesh associata, una rotazione, una posizione, la texture della mesh e altri parametri come la trasparenza ecc.

La linea di principio che viene utilizzata per rappresentare tutto su schermo è la seguente:

- Lo scene manager e il gui environment disegnano in memoria tutte le informazioni necessarie.
- Viene creata una telecamera che grazie al video driver visualizza tutto sullo schermo.

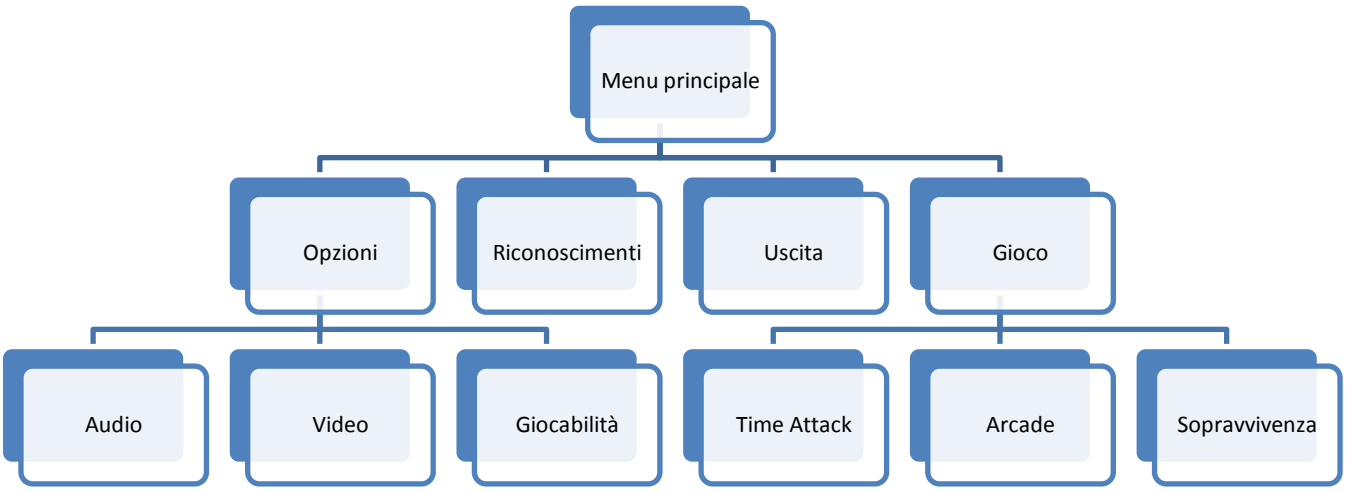
IRRKLANG

IrrKlang si occupa della parte audio del progetto. Irrlicht non è in grado di gestire l'audio e per questo si necessita di librerie appositamente scritte per questo scopo.

Ogni suono viene gestito all'interno del programma come un oggetto 3D invisibile. Questo oggetto ha diversi parametri come la direzione per la propagazione del suono, la sua potenza e logicamente il puntatore al file audio da eseguire

Struttura dei menu

Il menu di gioco è l'interfaccia che permette di scegliere la modalità di gioco e modificare le impostazioni della giocabilità. Per navigare nel menu, si dovranno utilizzare le frecce della tastiera per muovere la selezione, il tasto back per tornare al menu precedente ed esc per uscire dal gioco.
Di seguito la struttura ad albero dei menu:



Naturalmente, quello principale sarà il primo ad essere caricato. In esso saranno disponibili i collegamenti per il gioco, dove si deciderà la modalità tra quelle prima elencate, per le opzioni, per visualizzare i riconoscimenti e per uscire dal gioco.
Il menu opzioni è diviso in tre sottocategorie: Audio, Video, Giocabilità. Nel primo si potrà modificare il livello della musica, degli effetti sonori ed il bilanciamento. In video la risoluzione, la presenza di ombre ed il fullscreen. E poi possibile scegliere se utilizzare le frecce direzionali o i tasti WASD per pilotare la macchina. Modificate le opzioni volute, si possono salvare su file. Nel caso non si sia soddisfatti delle impostazioni settate, è possibile caricare quelle di default. In riconoscimenti, verranno visualizzati i nomi dei creatori ed i loghi degli strumenti utilizzati. Ed infine la parte più importante: il gioco. In esso sarà possibile selezionare la modalità, e così l'automobile da utilizzare.

Struttura del gioco

Come primo passo per sviluppare il videogioco si devono includere le librerie di Irrlicht ed irrklang. Bisogna includere anche la libreria matematica per delle operazioni descritte in seguito.

Come prima cosa, verrà riprodotta la musica del menu, poi verranno caricate tutte le immagini del menu. Visto che questa operazione non è immediata in quanto il caricamento in memoria di tutte queste immagini richiede tempo, è stata creata una schermata di caricamento in cui viene visualizzata graficamente una barra che aumenta progressivamente ogni volta che una immagine è caricata.

Il menu è strutturato in modo da visualizzare immagini che riproducono automobili per sfondo, poi una parte laterale a destra che riporta l'immagine di sfondo scurita con le varie opzioni. Ogni menu sarà strutturato in questo modo, apparte la schermata delle opzioni. Ogni volta che si premerà il tasto SU od il tasto GIU, la selezione si muoverà nella voce apposita. Alla pressione del pulsante INVIO il menu cambierà in base alla voce selezionata.

Per creare i menu si usano semplicemente delle immagini statiche, caricate appena il programma viene lanciato. Queste immagini vengono gestite da una funzione di scelta multipla. Ogni opzione corrisponde sia ad un'immagine che al valore di una variabile che verrà utilizzata per le altre funzioni de gioco.
All'interno di un menu se ne può caricare un altro allo stesso modo creando così una serie di menu in cascata. Tutte le variabili presenti nei menu più esterni sono visibili da tutti quelli interni. Per esempio: la variabile esci viene dichiarata nel menu principale ma è visibile da tutte le altre funzioni del gioco e si può decidere in qualsiasi momento di uscire dal gioco, in qualsiasi menu vi si trovi.

MENU OPZIONI

In questo menu si modificheranno le impostazioni prima descritte:

VOLUME: Si utilizza la apposita funzione di irrklang per settare il nuovo volume.
MUSICA: Se deselezionata si imposta il volume a 0.
EFFETTI SONORI: Se deselezionato una variabile verrà settata a false, e durante il gioco tramite controlli su di

essa si riprodurranno o meno gli effetti.
OMBRE, FULLSCREEN, RISOLUZIONE, RENDER GRAFICO: è molto semplice applicare queste modifiche, in quanto funzioni del device ne permettono la modifica facilmente.
COMANDI: Modificando queste impostazioni, verranno modificati i codici ASCII associati ai nomi dei pulsanti nel vettore dei pulsanti del ricevitore di eventi.

MENU SELEZIONE GIOCO

Alla pressione di INVIO con una delle tre modalità selezionate, si creeranno quattro variabili:

- Ris: che memorizzerà il risultato restituito dalla funzione di gioco
- Macchina: memorizzerà il numero della macchina selezionato
- Pista: settato ad uno per indicare di caricare la prima pista
- Vite: le vite del giocatore, settate a 0 in time attack, a 3 in arcade e ancora a 0 in survivor. In time attack è settata a quel valore perché la funzione provvederà a decrementare il numero, ma ignorare il comandi di interruzione della funzione. Per cui il valore continuerà a scendere in caso di perdita. Alla fine delle tre piste questo valore sarà moltiplicato per -1 in modo da ottenere le vite sprecate per completare il gioco. Per le altre modalità questo valore indica le ulteriori vite oltre quelle attualmente in uso dal giocatore.

La variabile macchina riceve il valore dalla funzione scegli:
Questa ha il compito di:
creare un nodo macchina nella posizione (0,0,0) e caricare la mesh e la texture della prima macchina. Successivamente si creano gli elementi necessari alla scena, ossia le luci, e la telecamera. Per mostrare la macchina, la telecamera sarà animata da una funzione di irrlicht, che la farà muovere seguendo una circonferenza immaginaria attorno alla macchina. Alla pressione dei tasti SINISTRA o DESTRA verranno cambiate le mesh visualizzate tramite la funzione apposita. Alla pressione di invio verrà restituito il numero della macchina visualizzata.

Successivamente si entrerà in un ciclo che terminerà quando il gioco verrà concluso arrivando fino alla fine delle tre piste, in caso di perdita di tutte le vite o quando il giocatore premerà ESC. In base al valore di pista verrà riprodotta la canzone apposita, poi si richiamerà la funzione, ed il valore restituito da essa sarà memorizzato in ris, e si agirà in base a quel valore.

Obiettivi

TIME ATTACK

Questa modalità di gioco può essere considerata come un allenamento del giocatore. Egli dovrà scegliere una macchina prima di poter iniziare a giocare. Non c'è un numero massimo di tentativi in cui finire la pista. Semplicemente, ogni volta che la si porta a termine, viene visualizzato il numero di sconfitte ed i complimenti per aver finito la modalità.

La funzione di gioco

Il gioco vero e proprio è stato implementato in una funzione. Essa ha come parametri la macchina scelta, la pista da giocare, il numero di vite, e la modalità di gioco. In caso di vittoria (il giocatore è arrivato in fondo alla pista) verrà restituito 1, in caso di sconfitta 0 e se il giocatore premerà esc il valore 2. Una volta inizializzate le variabili e creati i nodi in base ai parametri, si esegue un ciclo che continua fino a che il giocatore non preme il tasto ESC. Ogni volta che viene eseguito questo ciclo, il programma genera un fotogramma che verrà visualizzato nella finestra di gioco.

Come primo compito, il programma genera il Frame Delta Time, che come già detto, è il tempo che intercorre da un fotogramma al successivo, necessario per calcolare ogni formula. Infatti le formule utilizzate devono essere tutte moltiplicate per un tempo, per poter essere valide. Questo sistema, è necessario inoltre per fare funzionare il gioco su macchine di potenza diversa, perché un computer più veloce non deve rendere la macchina pilotata nel gioco una scheggia, mentre uno più lento non la deve rendere veloce quanto una lumaca.

Successivamente, viene creata una linea che inizia dalla posizione della macchina e finisce esattamente 4 unità sotto la posizione della macchina. Si dovrà ora verificare se questa linea interseca uno o più dei triangoli della pista. Per fare ciò si utilizza una funzione che passati come parametri il selezionatore da utilizzare, ossia quello assegnato precedentemente alla pista, e la linea, restituisce true in caso di collisione e false se essa non avviene. Se non avviene la collisione viene settata a true la variabile Cadendo, che segnerà l'inizio della discesa della macchina verso il terreno. Questa variabile inoltre impedirà ulteriori input ai comandi della macchina, siccome non sarebbe logico che la macchina si muova come sulla pista mentre sta' cadendo.
Al termine della pista, viene mostrato un cubo rosso trasparente. In ogni ciclo di gioco si verifica se le coordinate della macchina sono comprese nel volume del cubo, in caso affermativo si rimuovono tutti i nodi e la funzione restituirà il valore 1.

INIZIALIZZAZIONE DELLE VARIABILI

Vengono inizializzate le variabili relative al funzionamento della macchina. Queste variabili sono:

- V – rappresenta la velocità attuale della macchina. All'inizio della gara logicamente sarà uguale a 0
- Vmax – è la velocità massima della macchina
- Vmin – la velocità minima. Questo parametro serve quando la macchina sta andando in retromarcia
- Fren – rappresenta l'indice di freno della macchina, ovvero quanto riduce la velocità in un istante di tempo
- Acc – l'accelerazione, la variazione di velocità nel tempo
- Man – ovvero la manovrabilità della macchina
- CoeffFM – rappresenta l'indice di freno a mano

Oltre a quelli della macchina bisogna definire anche alcuni parametri della telecamera:

- MaxRot – è la differenza massima espressa in gradi che può esserci tra la rotazione della macchina e quella della telecamera. Serve per dare un effetto migliore al gioco
- Bdist – la distanza tra camera e macchina

- Alt – rappresenta l'altezza della camera rispetto al piano orizzontale. Tale altezza resta sempre fissa
- Dist – è la distanza della macchina dal punto focalizzato dalla telecamera. Se questa puntasse solo sulla macchina e non oltre, le curve ad alta velocità risulterebbero molto difficili da superare.
- Alfadiff – è la differenza in gradi tra la rotazione della macchina e quella della telecamera
- Allin – rappresenta la velocità con cui la telecamera si riallinea con la macchina. Questo effetto rallentato serve a migliorare l'esperienza di gioco.

Infine bisogna inizializzare alcune variabili di ambiente:

- Grav – rappresenta la gravità
- Attr – è l'attrito esercitato dalla pista sulla macchina
- Cadendo – variabile booleana per controllare se la macchina è ancora in pista o no
- TempoCaduta – indica da quanto tempo sta cadendo la macchina. Serve per calcolare l'altezza della macchina in base alla gravità.

INIZIALIZZAZIONE DEI NODI

Ogni oggetto che dovrà essere posizionato nella scena del gioco dovrà essere inizializzato tramite un nodo. Il nodo include la posizione dell'oggetto (espresso in 3 dimensioni), la sua rotazione e altri parametri.

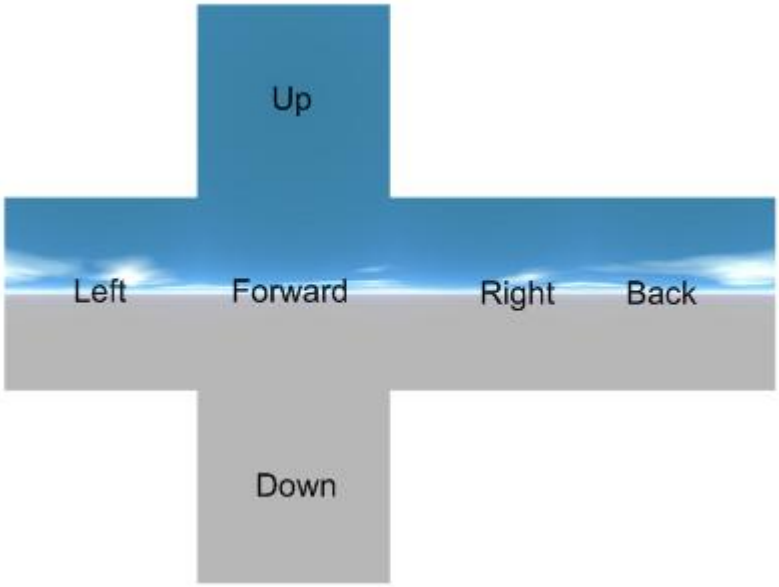
Prima che cominci il ciclo del gioco dovranno quindi essere inizializzati i nodi della macchina, della telecamera, della pista e anche del sole, ovvero la fonte di luce nella scena. I cubi di fine livello sono già memorizzati in modo da apparire al punto esatto al momento del caricamento. Per ciascuno sono state memorizzate le coordinate e l'ampiezza in modo da calcolare l'area in cui se la macchina si trova avviene la vittoria.

Collisioni

Per poter gestire le collisioni tra le mesh di un gioco bisogna utilizzare un "selezionatore di triangoli", ovvero una funzione che, passato come parametro un modello, ne calcola tutti i triangoli da cui sono composte le sue facce e ne memorizza la posizione. Verranno fatti dei controlli in seguito per decidere se la macchina si sta scontrando con qualche oggetto o no

Gestione dello sfondo e del cielo

Ci sono diversi modi per gestire la grafica dello sfondo. Il più semplice da realizzare è chiamato anche skybox. Consiste nel disegnare tutta la scena all'interno di un cubo. Le 6 facce del cubo saranno poi delle semplici



immagini. Se i lati delle facce del cubo

combaciano perfettamente con quelli adiacenti si avrà l'effetto di uno spazio tridimensionale.

Gestione della macchina

Successivamente, inizia la sezione che calcola la nuova posizione e la nuova rotazione della macchina. Prima di tutto, se la variabile Cadendo è uguale a true, si somma alla variabile TempoCaduta il Frame Delta Time, in modo da sapere da quanto tempo la macchina sta' cadendo, poi si calcolerà la nuova posizione sull'asse delle Y (per irrlicht l'asse delle Y è l'altezza) in base al moto uniformemente accelerato:

PosY = PosY + (Grav*TempoCaduta) * frameDeltaTime

Successivamente, se la variabile Cadendo è uguale a false, si controlla per ogni tasto lo stato. Se un tasto è premuto, si eseguono le seguenti operazioni:

Per l'accelerazione ed il freno, viene sommata o sottratta una velocità alla velocità attuale.

Durante l'accelerazione, ossia durante la pressione della freccia 'SU', viene sommata alla velocità l'accelerazione moltiplicata per il frame delta time:

V= V + (Acc * FrameDeltaTime)

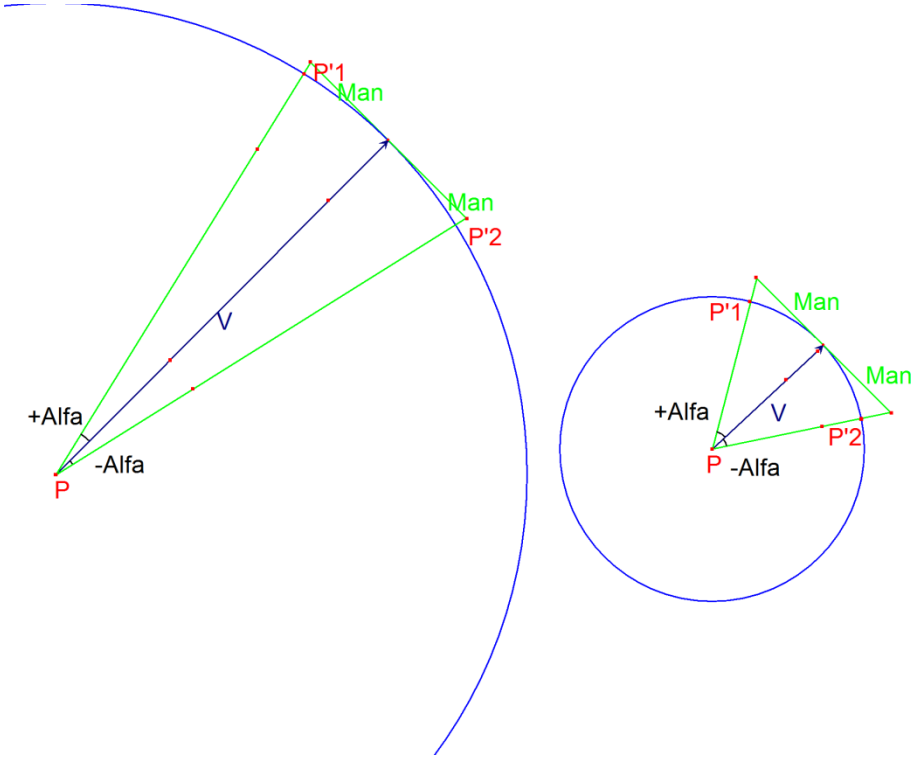
Se la velocità è negativa, la macchina dovrà prima fermarsi, per cui si utilizzerà la variabile Fren al posto di Acc

Alla pressione della freccia 'GIU' verrà sottratta alla velocità il valore associato alla variabile Freno:

V= V – (Fren * FrameDeltaTime)

Se la velocità è negativa, si sta andando in retromarcia,

Per gestire lo sterzo della macchina, è necessario calcolare il fatto che a velocità elevate la macchina fatica maggiormente a curvare. Se la velocità è minore di un certo valore, la macchina non può sterzare.



Alla pressione della freccia “SINISTRA”:

Rot= Rot – [Arctg(Man/V) * FrameDeltaTime]

Alla pressione della freccia “DESTRA”:

Rot= Rot + [Arctg(Man/V) * FrameDeltaTime]

Ed infine il freno a mano: se viene premuto il tasto “SPACE”, l’algoritmo opera in questo modo:

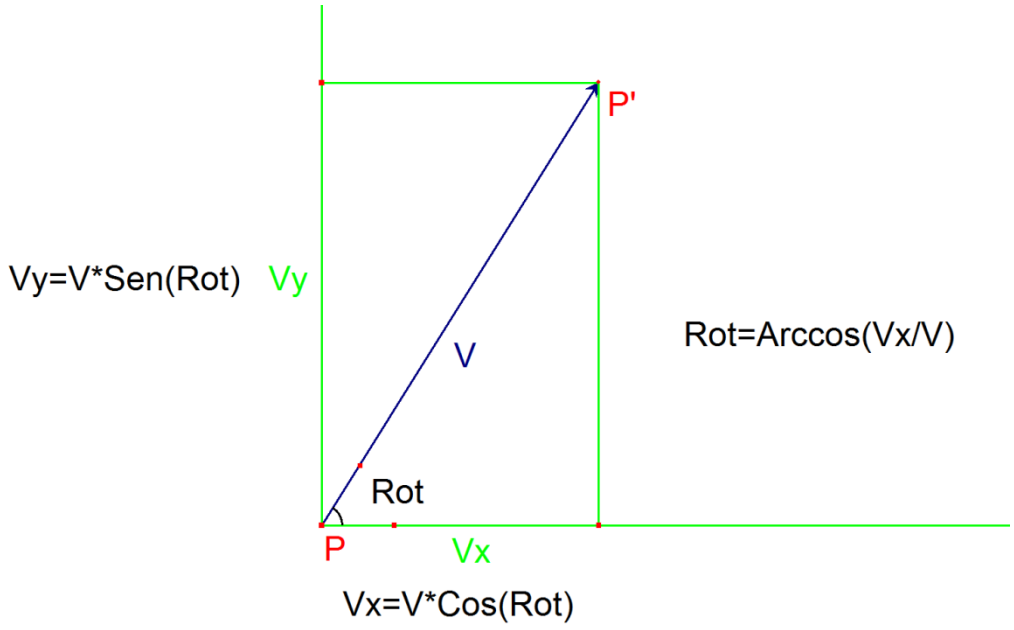
V = V - (Fren * CoeffFM) * frameDeltaTime;

Il freno a mano è stato implementato perché si è notato che durante il gioco rallentare con la freccia “GIU” può risultare macchinoso e snervante, siccome se si preme troppo a lungo si inizia ad andare in retromarcia e la telecamera si pone di fronte alla macchina. Il freno a mano, ferma la macchina più dolcemente, ed in caso di velocità uguale a 0, la macchina non va in retromarcia. Allo stesso modo di tutti gli altri, se la velocità è negativa, si somma il valore del freno, in modo da raggiungere lo 0.

Successivamente viene calcolato l’attrito, una velocità negativa sommata alla velocità della macchina.

Per tutte queste formule, deve essere applicato un criterio in base al quale se la velocità in seguito ad un'aggiornamento cambia segno, si deve porre la velocità uguale a 0, questo per evitare vari scatti della telecamera o dell'automobile, siccome se non si applica questo criterio la velocità oscillerà sempre intorno allo 0, il che non è corretto e causa problemi.

Il passo successivo è calcolare la nuova posizione nello spazio della macchina.



Si dovrà calcolare una velocità X, una velocità Y da sommare all'attuale valore di x e y della macchina.

PosX = PosX + V * sen(RotY) * frameDeltaTime;

PosZ = PosZ + V * cos(RotY) * frameDeltaTime;

Infine si assegnano i valori trovati al nodo della macchina, in modo che lo scene manager possa poi disegnarlo.

Prima di passare alla gestione della telecamera, di deve verificare che la macchina non sia caduta troppo in basso, ma solamente il necessario affinchè esca dal campo visivo della telecamera. Si verifica perciò se la posizione Y della macchina è inferiore a 50, in tal caso si riinizializzano le variabili della velocità, del tempo di caduta e della rotazione precedente della camera, poi si posiziona la macchina nella posizione iniziale con la rotazione iniziale. Infine se setta a false la variabile Cadendo.

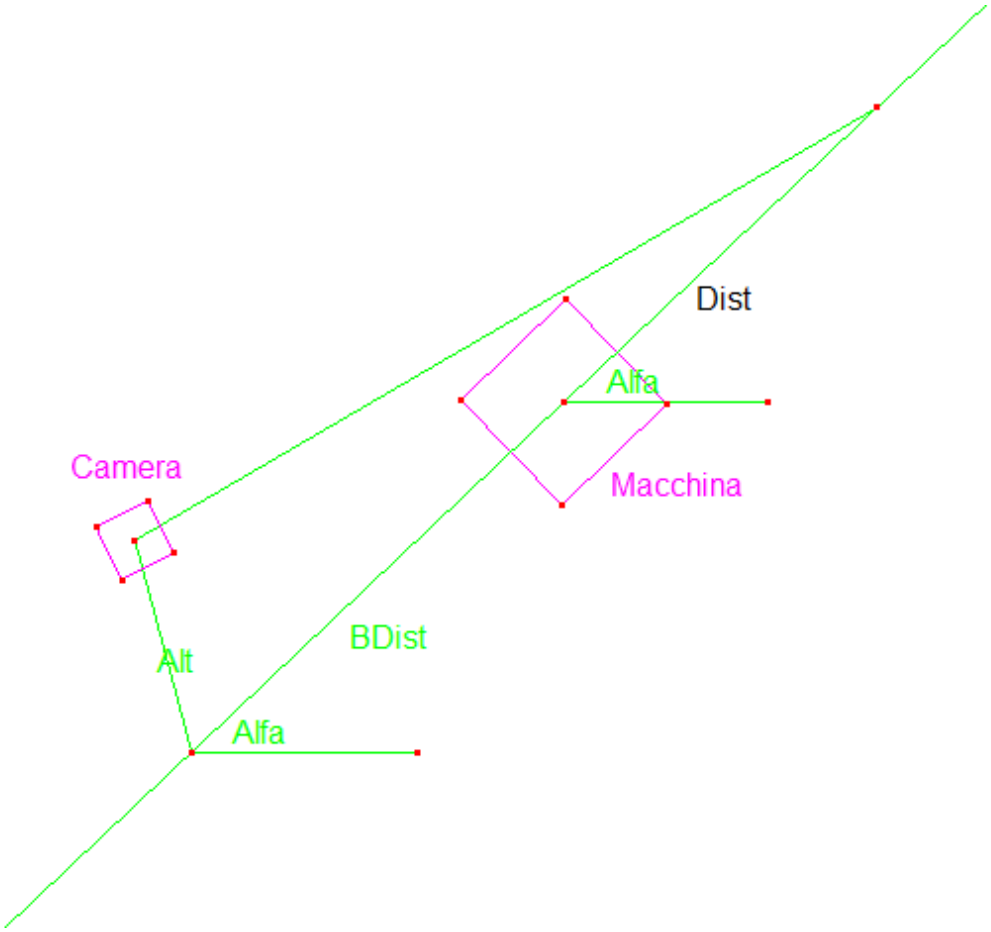
Gestione della telecamera

Il passo successivo è il calcolo della posizione della telecamera, ed invece della rotazione, il target, ossia dove la telecamera punta. La posizione verrà calcolata in base alla posizione della macchina, su una circonferenza di raggio BDist, e con una differenza di rotazione dalla macchina non superiore a MaxRot. Questa differenza è stata inserita per rendere la telecamera più elastica ai movimenti della macchina.

La differenza di rotazione, chiamata Alfadiff, è un valore che tande fotogramma dopo fotogramma ad annullarsi, ossia la camera tende ad allinearsi alla macchina con una velocità definita nella variabile Allin. Quando la macchina curva, la camera non lo fa immediatamente, ma più lentamente. Per calcolare Alfadiff, per cui è necessario sommargli la differenza della rotazione attuale della macchina dalla rotazione precedente della macchina. Se a questo punto Alfadiff è < di 0, gli sommiamo Allin moltiplicata per il frame delta time, se invece è maggiore di 0 gli sottraiamo la stessa moltiplicazione. Nel caso Alfadiff cambi segno in seguito di una di queste operazioni, si pone alfadiff = 0. Se Alfadiff è > di MaxRot si pone = a MaxRot, se è < di – MaxRot, si pone = -MaxRot.

Se la velocità è minore di -0.5, sommo 180 alla rotazione della camera, e pongo = 0 Alfadiff.

Ora si può calcolare la nuova posizione della telecamera:



nuovaPos X = macchinaPos X - (Bdist * sen(macchinaRot.Y+Alfadiff))

nuovaPos Y = Alt

nuova Pos Z =macchinaPos Z - (Bdist * cos(macchinaRot Y+Alfadiff));

Ed infine il target:

nuovoTarget X = macchinaPos X + Dist * sin(macchinaRot Y + Alfadiff)

nuovoTarget Y = 0

nuovoTarget Z = macchinaPos Z + Dist * cos(macchinaRot Y+ Alfadiff)

Creazione del fotogramma

Infine, definite le variabili, lo scene manager tramite una funzione genererà il fotogramma, poi disegnerà la GUI.

Infine salvo la rotazione della macchina in lastCamRot.

Conclusione

Questo progetto è stato realizzato come area di progetto per gli esami di Stato 2009 da Andrea Peretti e Ledi Salillari, dell'istituto tecnico industriale I.T.I.S. "Q. Sella" di biella. Il lavoro è molto curato nella stesura del codice con numerosi commenti, siccome un progetto di informatica. È stata invece trascurata la parte grafica visualizzando solo l' essenziale, in quanto non è stato il design e lo stile il nostro obiettivo. E poi, come ogni vero giocatore sa, non è la grafica di un videogioco che conta ☺