KARTING / GO-KART GAME

Avel LE MÉE MORET

Informations

Type: time-trial racing game
Graphics: 3D
Engine: FUDGE

Units and Positions

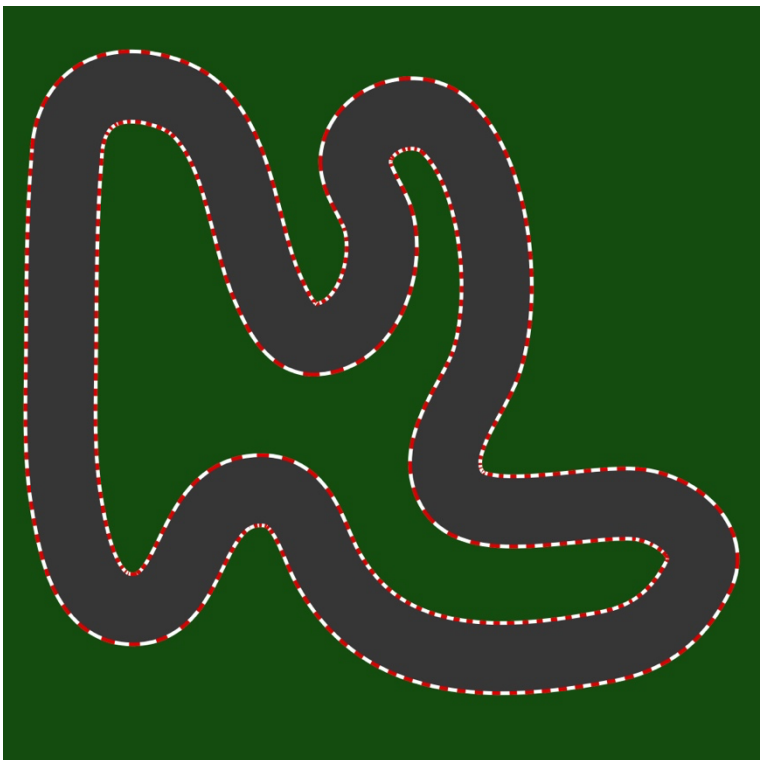
The 0 is the center of the track. Absolute positions of entities are not really important as all movement is done using physics.
Distance unit is meter (m) so a cube of 1 by 1 in Fudge is 1 meter by 1 meter in reality. That implies speed unit is meter per second (m/s).

Hierarchy

The kart, start and spectator nodes contain a geometry node which allows to group and apply transformations on the different subparts contained in it.

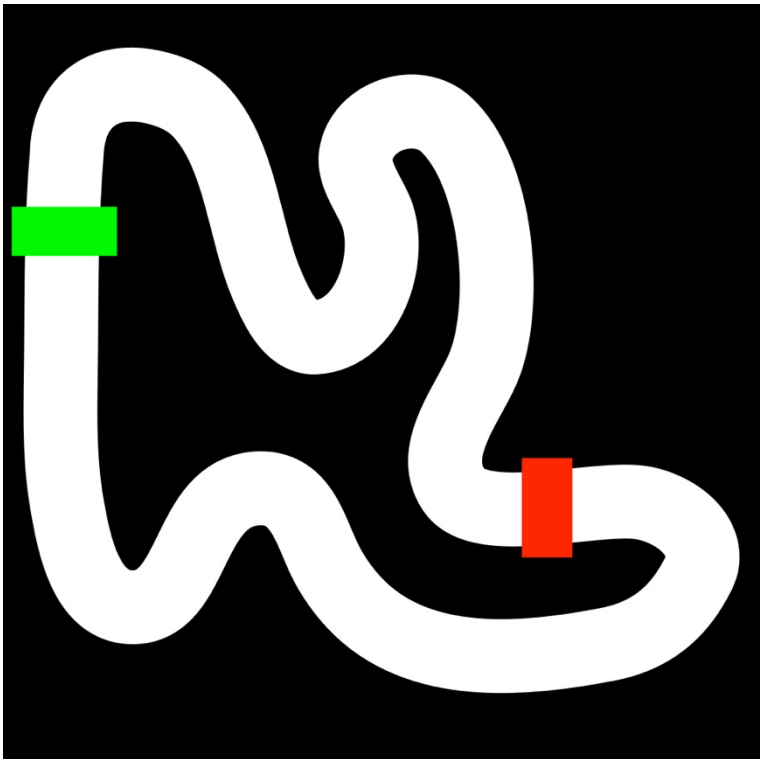
The kart ComponentRigidbody is directly applied to the kart node so it can use the kart ComponentTransform to make the kart move.

The track is a bit different because it contains two different layers. One for the texture and one under it for the track limits. The texture node contains a static ComponentRigidbody in order to collide with the kart. It also contains a Material which is added on runtime (see below).



Track texture

The track limits node contains a static ComponentPick in order to receive the pick ray sent by the kart. It also contains a Material which is added on runtime (see below).



Track limits texture

Editor

The editor allowed me to create the 3D models of the kart and the spectator by grouping, transforming and applying materials on several meshes. Performing a visual task such as modeling, the editor allows you to work much faster and preview your work in real time.

Script components

The CustomScriptComponent was not really useful for me during this project but it allowed me to animate the spectator only if the best lap time is under 12 seconds.

Extend

The kart and track node extends the Fudge node. This allowed me to easily access the geometry node and the various components and keeping a good organization by creating a new class for each.

Sound

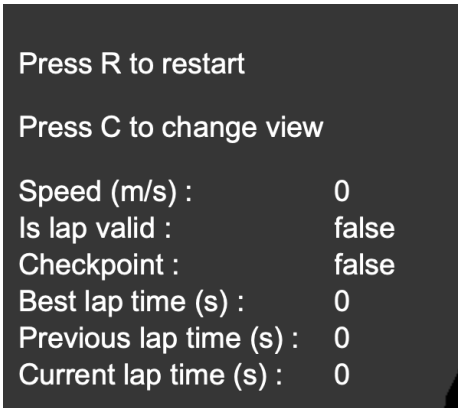
I had to create a motor sound in fudge. I use a simple motor idle sound and increase or decrease it rate in order to modify the pitch. This produces a higher pitch sound when driving fast and lower pitch sound when driving slowly.

The ComponentAudioListener is a child of the kart because the player is theoretically seated in the kart. The sound is also emitted by the kart because the engine is in theory on the kart.

VUI

In order for the player to see his current, previous and best speed and lap time, I created a VUI with all this information.

It also shows how to change cameras and restart the lap.



VUI

Event-System

I used the event system to know which key the player presses in order to detect if the player wants to restart the lap or toggle the camera position.

External Data

I used a configuration file in order to change the default camera position, the max velocity and other data useful for the movement of the kart. Having the ability to change its settings in the setup makes it easy to adjust the racing feel.

Light

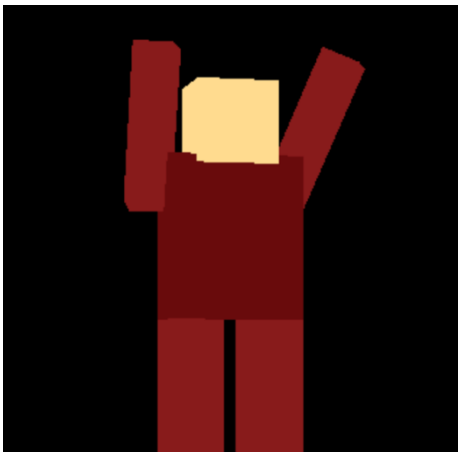
I have a simple environment light only used to see the track.

Physics

I use physic on the kart to move it and create collision between the kart and the track. Physics makes it easy to reproduce the behavior of a kart. It was just necessary to solve the problem of "drift" if the kart turns faster than it advances (rotation on the spot).

Animation

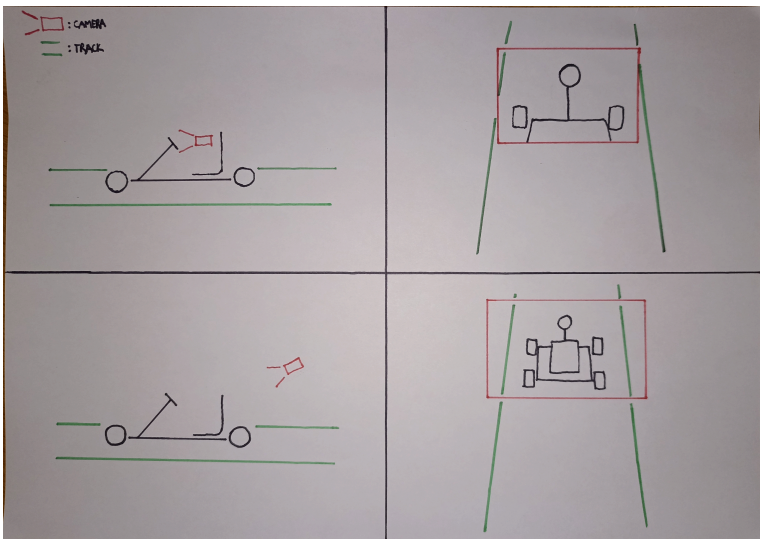
I created a little animation so the spectator says hello when the best time is under 12 seconds.



Spectator

Camera

There are two main camera position usable: first and third person.
In order to follow the kart, both are children to the it.
The difference is just the location and angle of the camera relative to the kart.
You can switch the camera position in game or in the configuration.



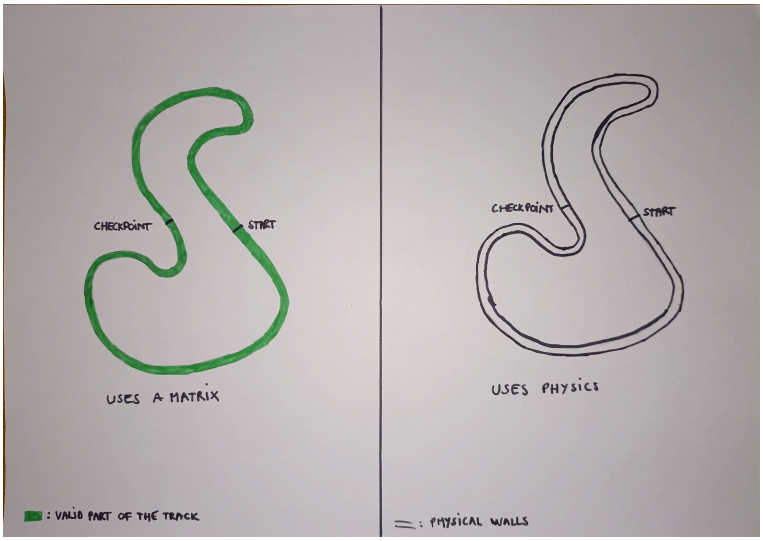
First and third camera view

Track limits

At the beginning I taught using one of the two possibilities:

1. A matrix of 0 and 1 defining valid positions on the track.
2. Walls physically blocking the kart from leaving the track.

After the professor told me that I could use a system of rays to detect whether the kart is out of the track or not I changed my mind to follow his advice.
I created to plane meshes one on the top of the other.
The top one on which the kart is has a track texture with red borders and black track. The other one bellow is only used to detect if the kart is on the track, out of the track, on the checkpoint or on the start line.
It works using a ray starting from the kart to the bottom (-y), depending on the color of the track limit textures where the ray crosses it, the game knows what to do.



Track limits system