Journal de Développement

Phase 1 : Construction d'un moteur élémentaire de gestion de particules

26/09/2022

Quentin Morel [MORQ22109800]
Gabriel Reboul [REBG22089901]
Clémence Clavel [CLAC30540107]

Difficultés rencontrées

OpenGL

La principal obstacle que nous avons rencontré lors de notre développement est l'utilisation d'OpenGL. Aucun membre de notre groupe n'avait utilisé OpenGL auparavant et nous avons eu des difficultés à le prendre en main, à comprendre sa syntaxe et son utilisation.

Matrices et Projection

Ne venant pas tous de la même filaire, certains membres n'ont pas étudié les domaines de la 3D notamment de la vision 3D avec les matrices de rotation, projection etc…

Architecture

Il nous a été difficile de faire des choix sur l'architecture du projet, sur la façon de l'ordonner, de mettre en relation les différents éléments et sur la façon de les utiliser au sein du projet.

Choix faits et justifications

Visual Studio

Visual Studio est un environnement de Développement en plus d'un compilateur (MSVC) pour le langage C++ . Sa grande popularité ainsi que sa puissance de debuggage (mode pas à pas, etc…), fait qu'il s'agit d'un très bon outil pour développer en C++.

CMake

CMake est un utilitaire pour la compilation de programmes.
Sa plus grande simplicité et sa facile implémentation de code pour du cross-plateforme a fait que notre choix s'est tourné vers ce dernier plutôt que "make".

ImGui

ImGui étant une librairie très optimisée avec une grande flexibilité et facile à implémenter dans un programme Vulkan et OpenGl. Pour notre moteur de physique de particule, il est très important de pouvoir contrôler/modifier des variables rapidement et sans avoir à redémarrer le programme incessamment.

OpenGL

Notre moteur physique devant être un moteur 3D, nous nous sommes orienté vers OpenGL plutôt que Vulkan. OpenGL étant très populaire et très documenté comparé à Vulkan qui lui est sorti en 2016.

GLFW

GLFW est une librairie pour créer des fenêtres, des contextes et des surfaces, et recevoir des entrées et des événements. Afin d'utiliser OpenGL il nous fallait déjà créer une fenêtre qui ensuite sera utilisée par OpenGL. GLFW étant facile d'utilisation et ayant les éléments de bases (fenêtres, entrées clavier, gestion de la souris) pour notre application, nous l'avons donc choisi.

Mathématiques et Physique

Vector3D

Implémentation d'un vecteur de 3 dimensions et des fonctions nécessaires (coordonnées, soustraction ,addition, multiplication par un scalaire, produit scalaire, produit vectoriel, affichage, …) .

Particule

Implémentation d'une particule et des éléments nécessaires : position, vitesse, accélération.

Integrator

Gestion de la mise à jour de la physique du jeu grâce en conservant un taux de rafraîchissement constant. (par exemple changement de la position selon la vitesse de l'élément)

Integrable

Élément qui subit la physique, par exemple Particule hérite d'Integrable.

Gestion du jeu et de la boucle

Physical Engine

La boucle de jeu est localisée dans la class PhysicalEngine qui est la classe mère du projet. C'est elle qui va créer le scène et gérer

Game

Gestion des actions enclenchées dans le jeu selon les entrées du joueur.

InputManager

Gestion des entrées du joueur grâce à des callbacks de GLFW (entrées claviers et mouvements de la souris).

Scene

Classe pour la gestion des GameObjects ainsi que de la caméra du jeu.