# Phase 1 - Projet 8INF935

Alaoui, Mohamed-Wajih, Delort Tristan

Septembre 2021

## 1 Introduction

Ce projet rentre dans le cadre de l'unité d'enseignement "Mathématiques et physique pour le jeu vidéo" au sein de l'UQAC. Le but est de réaliser un moteur Physique en 4 Phase. Ce document explicites les difficultés technique, les solutions, et les choix rencontrés durant la réalisation de la première phase pour chacun des membres du groupe.

## 2 Delort Tristan

Ayant déjà développé en C/C++, le développements des classe $Vector3D$ et $Particle$ n'a pas beaucoup posé de difficulté. Par contre, le gestionnaire de build CMake que je n'avais jamais utilisé s'est avéré plus complexe, surtout pour ajouter d'autre bibliothèques.

J'ai aussi profité de la première phase plus simple pour me familiariser avec OpenGL (et GLFW). J'avais déjà un peu d'experience en GLSL/Cg et des connaissances sur le pipeline graphique, mais je n'avais jamais développé en utilisant OpenGL (ou toute autre bibliothèque graphique de très bas niveau). Après une longue lecture de la formation OpenGL fournie ici : `https://open.gl/`, de la documentation de GLFW :`https://www.glfw.org/docs/latest/quick.html`, et des informations sur le dépôt Github de ImGUI :`https://github.com/ocornut/imgui`. J'ai réussi à créer une fenêtre graphique utilisant OpenGL pour faire le rendu 3D, GLFW pour gérer la fenêtre, et ImGUI pour contrôler l'application avec une interface graphique utilisateur.

En chemin, j'ai aussi découvert l'utilité de GLAD, un loader pour OpenGL permettant au programme d'avoir accès au fonctions OpenGL. Enfin, j'ai du ajouter la bibliothèque GLM (OpenGL Mathematics) pour créer des matrices de transformation (notamment les matrices Model, View, et Projection) [1].

Ne voulant pas encombrer l'utilisateur de l'installation de ces bibliothèque, j'ai expérimenté avec les sous modules de git. Ainsi, quand on clone le projet, les bibliothèque sont clonées en même temps. Elle sont ensuite build et liées au projet grâce à CMake.

Durant la prochaine phase, il faudra développer une bibliothèque englobant les appels au fonctions de OpenGL pour factoriser en minimiser l'encombrement du code. Dans notre code actuel, on peut remarquer qu'une très grande partie est occupée par l'initialisation et l'utilisation de OpenGL, seulement pour dessiner un point.

# 3 Alaoui, Mohamed-Wajih

Pour ma part, n'ayant jamais codé sous C++, cette première phase a été pour moi l'occasion de prendre en main le langage ainsi que ses particularités.

J'ai commencé par suivre un tutoriel très détaillé et bien expliqué de la chaine Youtube The Cherno disponible à l'adresse suivante : `https://www.youtube.com/watch?v=18c3MTX0PK0` pendant environ 3 heures, tout en expérimentant et en essayant de reprendre tout seul le code. J'avais ainsi une base avant de pouvoir me pencher sur le sujet proposé. Le C++ étant par ailleurs un langage très documenté. (Notamment CPP reference : https://en.cppreference.com/w/ ). Je pouvais ensuite essayer de refaire les programme des classes $Vector3D$ et $Particle$. C'est alors que j'ai rencontré la difficulté majeure de ce langage : la synthaxe. Mais je pense qu'avec le temps et l'entraînement je pourrais bien assimiler les caractéristiques et les subtilités de ce langage.

J'ai également eu l'occasion de beaucoup échanger avec Tristan qui est très pédagogue dans ses explications et qui de plus est assez à l'aise avec ce langage, ce qui m'a alors été d'une grande aide. Nous avons d'ailleurs eu l'occasion de faire du "pair programing" ce qui m'a été d'une grande aide dans mon apprentissage car je pouvais à la fois réfléchir à la logique derrière la résolution d'un problème mais également discuter avec Tristan à propos de la traduction de celle-ci en C++.

[1] Cette bibliothèque implémente des vecteurs et des matrices. Je n'ai pas utilisé ces classes pour réaliser la partie Moteur Physique du projet car ce serait bien sur de la triche.