Minot Eva
Cavani Nicolas
Moënne-Loccoz Célestin

Journal de bord

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

I/ Démarrage du projet (2ème cours)

Au démarrage du projet, Eva et Célestin se connaissaient, mais ils ne connaissaient pas encore vraiment Nicolas. Cependant, il n'y a pas du tout eu de problèmes avec cela puisqu'il s'est avéré que nous avions la même mentalité par rapport à ce projet : nous étions vraiment là pour apprendre à faire les choses proprement, et pour pousser notre projet au maximum et en être fiers.
Nous avons donc commencé par créer un projet git sur lequel nous avons ajouté un fichier de configuration CMake pour être sûrs que notre projet compilerait de la même manière sur n'importe quelle machine sous Windows.
Une fois ceci fait, nous avons voulu commencer par la déclaration de la classe de base Vector3D pour nous assurer d'avoir une idée relativement claire dès le départ du comportement qu'elles devraient avoir. Cela nous semblait important puisqu'elle est réellement le fondement de tout ce que nous ferons par la suite.

II/ Première réunion (en dehors des cours)

Lors de notre première entrevue à la fin du deuxième cours, nous n'avons pas eu assez de temps pour compléter la classe Vector3D. Nous avons donc pu terminer sa déclaration proprement, en y ajoutant des détails auxquels nous avions pu penser entre les deux entrevues (comme par exemple la méthode "getNorm" ainsi que les opérateurs "+=", "-="... qui sont très redondants avec les opérateurs "+", "-"... mais qui pourraient s'avérer utiles par le suite). Nous avons également pu déclarer entièrement la classe Particle tous ensembles.
Nous avions fini cela relativement rapidement, et nous avons donc pu commencer à nous répartir le travail : Célestin et Eva ont commencé à définir les méthodes précédemment déclarées des classes Vector3D et Particle. La méthode centrale à ce début de projet, "integrate", a été commencée, mais il nous manquait encore plusieurs informations pour être réellement sûrs de ce que nous faisions.
Nicolas, lui, a commencé à se renseigner sur les différentes possibilités pour obtenir un rendu graphique. A ce moment là, les options étaient les suivantes : glfw ou ogre3D. Ogre3D semblait permettre plus de choses et nous avons donc opté pour cette option. Mais après plusieurs tests, nous nous sommes rendu compte qu'il serait très complexe de réussir à utiliser une librairie si complexe pour un petit projet comme le nôtre et étant tous novices avec openGL..

III/ Deuxième réunion (3ème cours)

Après discussion avec le professeur, nous avons effectivement conclu que l'utilisation des librairies glfw et glew serait plus adapté pour notre projet. Nous avons également pu demander de l'aide pour compiler cette librairie, et pouvoir commencer quelques tests.
Ayant reçu un cours sur la mécanique newtonienne, nous avons pu modifier la méthode "integrate" pour intégrer les éléments qui nous semblaient précédemment obscurs (comme notamment la notion de "inverseMasse"). Nous avons également ajouté la gestion de forces constantes plutôt que de décider directement de l'accélération.

IV/ Troisième réunion (en dehors des cours)

Lors de cette troisième réunion, nous avons d'abord terminé la gestion de la physique newtonienne de base pour nos particules. Cela n'a pas posé de problème particulier.
Ensuite, nous avons réfléchi ensemble sur les différents projectiles que nous voudrions ajouter. Nous avons décidé d'en créer trois:

- le projectile "bullet", une balle de pistolet: projectile rapide, soumis à une gravité plus élevée que la normale (g=15).
- le projectile "laser", un laser à la star wars: projectile très rapide, pas soumis à la gravité.
- le projectile "fireball", une boule de feu: projectile plutôt lent, soumis à une gravité négative (la chaleur fait remonter la boule de feu)

Pour l'implémentation des projectiles, nous avons décidé de créer une classe Projectile, qui servira d'interface pour les projectiles énoncés ci-dessus. Nous avons donc commencé sa définition, puis nous nous sommes réparti le travail à finir avant la deadline (Célestin et Eva sur les projectiles, Nicolas sur l'interface graphique).

V/ Complétion du travail assigné (personnel)

Du côté de Célestin et Eva, nous avons créé les classes des projectiles spéciaux: Fireball, Bullet et Laser. Nous les avons implémentées et avons créé un “main” permettant, via la console, de choisir le type de projectile à tirer, la position initiale et la direction du tir, et d’instancier le projectile avec les informations données. La console indique ensuite pendant un certain temps (indiqué par l’utilisateur) les informations du projectile (position et vélocité) toutes les secondes, ce qui permet de visualiser la trajectoire du projectile.
Grâce à ce “main”, nous avons pu vérifier le bon comportement de nos projectiles, le seul problème que nous avons rencontré concerne l’accélération des particules: elle a la bonne valeur lors de l’intégration, mais à l’affichage on ne récupérait pas la valeur actualisée (mais la valeur de base, qui est un vecteur nul). Nous n’avons pas réussi à corriger ce problème, c’est pourquoi nous n’affichons pas l’accélération du projectile, mais a priori les valeurs calculées pour l’accélération sont justes.
Avec une interface graphique fonctionnelle, nous n’aurions pas besoin de ce “main”, mais nous avons préféré le créer au cas où l’interface ne soit pas satisfaisante à temps.

Du côté de Nicolas, en ce qui concerne le retour graphique, il est possible d’afficher à l’écran des objets 2D qui possèdent une texture et que l’on peut déplacer via ImGui. Dans le code les principes de Renderer, Texture, Shader, VertexBuffer, IndexBuffer et VertexArray ont été abstraits dans des classes. Il a évidemment fallu comprendre comment fonctionne le pipeline graphique de manière simple.

VI/ Finalisation de la première phase (4ème cours)

Nous avons revu notre “main” permettant de lancer les 3 types de projectiles : à la place d’appliquer à chaque frame une force constante, plus ou moins forte suivant les projectiles (en plus de la force de gravité), nous allons juste donner aux projectiles une certaine vélocité initiale, dans la direction décidée par l’utilisateur, et laisser uniquement la force de la gravité agir. Ainsi, nous pouvons observer de belles paraboles.
Nous avons également essayé de finaliser un début d’interface graphique, mais nous avons conclu qu’il serait préférable de prendre le temps de maîtriser le rendu graphique avant de le lier à notre projet plutôt que de bâcler une interface plus ou moins fonctionnelle que nous ne comprendrions pas par la suite. Nous nous sommes donc contentés de la sortie console pour cette première partie du projet.
Nous allons maintenant nous pencher sur la liaison entre notre moteur physique et l’affichage pour pouvoir rendre à l’écran les particules qui se déplacent. Il nous faudra nous renseigner sur l’affichage d'objets 3D projetés sur une caméra.