Journal de développement - Phase 3

Mathématiques et physiques pour le jeu vidéo – Groupe Sonic

Chronologie du projet

- **30 octobre**: Implémentation des classes de base (matrices 3 et 4, quaternions)
- **31 octobre** : Implémentaion des tests pour ces classes. Nettoyage du dépôt, suppression des éléments uniques à la phase 2.
- **2 novembre**: Utilisation de la camera 'EasyCamera' fournie par openframeworks, déplacements dans l'escpace avec les touches directionnelles.
- **4 novembre**: Classe de base pour les RigidBody.
- **7 novembre**: Implémentation des fonctions majeures (update et draw) des RigidBody. Ajout de quelques opérateurs manquants.
- **12 novembre**: Premier test de la rotation d'un RigidBody. Implémentation d'un raycast afin de viser un point pour lui appliquer une force plus tard.
- **14 novembre**: Implémentation des points d'application de forces ainsi que du torque.
- **16 novembre**: Liaison de toutes les fonctionalités entre elles dans le GameContext, notament l'application d'une force au point de contact du raycast. Ajout de la possibilité de faire apparaître des rigidBody de différentes masse devant la caméra. Quelques bugs ont été corrigés, et le README a été mis à jour avec l'ensemble des nouveaux contrôles.

Difficultés rencontrées

La phase 3 du projet a été plus facile à mettre en place que la seconde. Nous nous sommes mieux organisés suite à l'echec de la phase deux, avec une séance de travail en commun par semaine. Le travail était cependant compliqué à répartir de manière uniforme pour chaque membre de l'équipe. En effet, certaines taches sont préférablement faites par une seule personne pour éviter les dépendances de deux personnes ne travaillant pas au meme moment.

Astuces de programmation apprises

Comme il s'agissait de notre première phase avec un environnement 3D, nous avons du mettre a jour les controles vers un déplacment dans l'espace contrairement a un écran fixe pour les phase précédentes. Le raycast a également été implémenté pour cette phase du projet. Autrement, cette phase ressemblant beaucoup à la première, nous connaissions donc déjà les concepts de programmation nécessaires pour venir à bout de celle-ci.

Choix de développement

De la même manière que pour la masse des particule, nous avons choisis de stocker le tenseur d'inertie inverse pour éviter des calculs redondants. En effet, celui ci n'est utilisé que pour mettre a jour l'acceleration angulaire, on peut donc se permettre de le stocker de cette manière. De plus, comme nous n'utilisons que des RigidBody de forme parallépipède rectangle, le tenseur d'inértie est une matrice diagonale et peut être facilement inversée.

Nous avons également choisit de modèliser le RigidBody avec 3 axes réprésentant les demi-dimensions de la boite. Les axes de base (avant la rotation) sont également stocké dans la classe, premièrement pour avoir accès au dimensions de celles-ci sans calcul, mais surtout pour appliquer la rotation de manière absolue à partir des axes de départ, plutot que sur ceux de du tout de boucle précédents, ce qui résulterait en un perte de précision rapide des flottants, et donc une forme de boite qui se détériore avec le temps. Les axes après la rotation sont stockés dans la classe afin de les recalculer entre l'update et le draw.

Enfin, l'ajout de force a du etre modifié pour supporter les points d'applications de force, contrairement a ce que l'on se contentait de faire jusqu'à présent avec les particules.