Champs Gravitationnel

Procédure du rendu

Start	Fin
Vendredi 10/02/2023 –13h	Lundi 27/03/2023 –16h

Nombre d'étudiants par groupe : 3 maximum (groupe de 2 ou 1 autorisé mais privilégiez 3)

Rendu :

- Sources : tag GOLD sur git (sur la branche master)
- Sur l'exercice, un zip NOM1_NOM2_NOM3.zip
  - Exécutable (Windows)
  - Screenshots : minimum 4 dans un dossier Screens, montrant les fonctionnalités.

Pénalités de retard :

- -1 point par tranche de 3h de retard (donc 3h01 min -> -2 points, 5h59 -> -2 points)

Faites donc attention au jour du rendu 😉

Objectifs

L'objectif de ce TP est de créer un simulateur de système solaire, ou de manière plus générale d'un ensemble de N corps célestes. On s'intéressera donc aux champs de forces gravitationnelles qui sont des champs vectoriels conservatifs.
Ce projet comptera pour **40%** de la note du module.

Compétences

Le projet sera évalué sur les compétences suivantes :

Hard Skills:

- ➢ Calculer et représenter des champs vectoriels
- ➢ Calculer et représenter des lignes de champs
- ➢ Calculer et implémenter en code des opérateurs de champs
- ➢ Intégrer les équations du mouvement en fonction du temps

Sujet

Vous réaliserez avec le moteur **Unity3D** un prototype de « jeu » (outil de simulation) qui permet de visualiser le déplacement d'objets 3D (ici des astres solaires, des planètes, des lunes, des astéroïdes, etc…) soumis à des champs de forces gravitationnelles.
Le jeu sera comme un jeu « bac à sable » qui permettra à l'utilisateur :

- Dans un premier temps d'instancier des objets 3D et d'en définir la *masse*, la *position initiale*, et la *vitesse de déplacement initiale*. Pour la *dimension* de l'objet 3D, vous la déduisez de sa masse à partir d'une relation simple (par exemple plus la masse est grande plus l'objet 3D est grand, à vous de choisir la forme exacte de cette fonction). Pour sa forme, l'utilisation de sphère est recommandée mais vous pouvez éventuellement utiliser des mesh3D différents (notamment pour améliorer le réalisme d'un astéroïde par exemple). Pour les textures et matériaux, à vous de choisir selon le visuel que vous souhaitez (un matériau brut avec une simple couleur sera accepté si vous n'avez pas le temps de chercher mieux).
- Dans un deuxième temps, le joueur pourra visualiser la trajectoire d'un objet 3D par le biais du calcul d'une ligne de champ au sein du champ gravitationnel total obtenu comme la somme des champs gravitationnels individuels générés par la masse de chaque objet 3D.
- Dans un troisième temps, l'utilisateur pourra lancer la simulation (évolution temporelle) et voir le déplacement de chaque objet 3D obéissant aux forces gravitationnelles (recalculées en chaque instant). Il faut donc intégrer les équations du mouvement à l'aide des lois de Newton.

Liste des fonctionnalités

- Une méthode qui instancie des objets 3D célestes. Vous pouvez créer une « prefab » mais attention la masse de l'objet, son matériau, sa position, et sa vitesse sont donnés en input. Attention pensez à mettre une limite au nombre d'objet maximum. [2 points]
- Un script attaché à l'objet 3D céleste qui permet de calculer sa position au prochain pas de temps. Le script a besoin de récupérer la position actuelle de l'objet, sa masse actuelle, sa vitesse actuelle, les forces gravitationnelles agissant en la position actuelle. [4 points]
- Un script qui dessine un champ vectoriel, c'est-à-dire qui affiche des vecteurs correspondant au champ vectoriel sur une grille régulière 3D. la densité du maillage de la grille est en input permettant à l'utilisateur de gérer l'affichage. La taille maximum d'un vecteur ne doit pas dépasser la diagonale d'un petit cube du maillage afin d'assurer une certaine visibilité (ne pas prendre en compte et donc ne pas dessiner les vecteurs dans la « bounding box » des objets 3D : le champ y sera de toute façon trop intense). [2 points]
- Un script qui calcul le champ vectoriel gravitationnel total et son rotationnel en n'importe quel point de l'espace. [2 points]
- Un script qui dessine une ligne de champ gravitationnel partant d'un objet 3D. A vous de gérer la fin du traçage de cette ligne : quand est-ce que j'arrête de prolonger cette ligne ? En effet une ligne champ peut être très longue il faut donc mettre une limite. [2 points]
- Un script qui attache la caméra à un objet et lui confère un nouveau mode de déplacement qu'on appellera ***mode orbite*** : les flèches permettent de faire varier la colatitude et la longitude, la molette de la souris permet de zoomer (faire varier le rayon) pour se rapprocher de l'objet 3D et inversement. La caméra pointe vers le centre de l'objet. [2 points]
- Un panneau latéral coulissant permettant de montrer/cacher les outils du « bac à sable ». Le panneau contient [2 points]:
    - Un bouton création d'objet 3D qui ouvre une fenêtre contextuelle permettant de renseigner dans des champs textuels les valeurs initiales de l'objet 3D (position, vitesse, masse). [Le matériau peut être attribué aléatoirement ou vous pouvez laisser le choix à l'utilisateur via par exemple un menu déroulant].
    - Une case à cocher (« toggle button ») activation/désactivation de la vue du champ vectoriel gravitationnel.
    - Une case à cocher (« toggle button ») activation/désactivation de la vue du **rotationnel** du champ vectoriel gravitationnel.
    - Une « slide barre » pour augmenter/diminuer la densité de la grille utilisée pour dessiner le champ vectoriel.
    - Une case à cocher (« toggle bouton ») activation/désactivation de la vue d'une ligne de champ partant de l'objet 3D actuellement sélectionné (si pas d'objet sélectionné ne rien dessiner)

    - Une case à cocher (« toogle bouton ») activation/désactivation du suivi caméra de l'objet 3D sélectionné. La caméra passe en ***mode orbite***.
- Un bouton start : pour démarrer la simulation temporelle [0,5 points]:
- Un bouton pause : pour interrompre la simulation temporelle [0,5 points]:
- Une caméra (« god view ») que l'utilisateur peut déplacer librement dans l'espace 3D. Vous avez le choix sur la méthode de déplacement libre qui est le mode par défaut. [0,5 points]:
- Sélection d'un objet 3D par le clic gauche de la souris (utilisation de « raycast »). La sélection fait apparaître une fenêtre contextuelle renseignant les valeurs de cet objet 3D (position, vitesse, masse). Un clic gauche dans le vide désélectionne l'objet. [1 points].
- Le travail est propre et soigné (code commenté, bien indenté, projet unity bien structuré, etc..) [1,5 points]

Contraintes

- **Utilisation du moteur Unity**
- Vous ne devez évidemment utiliser aucune librairie externe pour réaliser les fonctionnalités, ni copier du code de vos camarades, sous peine d'un 0

Tips

Afin d'améliorer votre rendu visuel voici quelques propositions (optionnelles) :

- Utiliser une « skybox » d'environnement spatial (fond étoilé).
- Ajouter un « trail » (voir « particle system ») aux objets 3D se déplaçant avec une vitesse plus grande qu'un certain seuil.
- Ajouter une vitesse angulaire aux objets 3D (aléatoire ou basée sur la masse)
- Ajouter des shaders pour la surface des objets 3D (par exemple une planète gazeuse pourrait avoir une surface avec des mouvements de fluides).
- Lorsqu'il y a collision de deux corps célestes, détruire le plus léger et augmenter la masse du plus lourd pour absorber la masse du plus léger (n'oubliez pas d'adapter la taille de l'objet). Un petit VFX (voir « particle system ») pour l'explosion.