GMINT317 - Moteurs de jeux – TP7
Gestion de scène

Rémi Ronfard  *remi.ronfard@inria.fr*  *https://team.inria.fr/imagine/team/*

Objectifs

Dans ce TP, nous allons nous intéresser à étudier la gestion efficace de scène 3D. Pour cela, nous allons modifier notre application précédente pour intégrer :

- Un gestionnaire de scène
- Une structure d'octree
- Un gestionnaire de niveau de détails
- Des imposteurs
- Un peu de physique … avec une sphère

Bonus :

- Réaliser tous les calculs de physiques de manière parallèle : GPU, CPU
- Un peu plus de physique … avec des géométries plus complexes
- Mettre en place un BSP-tree, Kd-tree
- Afficher une scène infinie
- Garder un rendu temps réel

Gestionnaire de version

Cloner et forker le TP7. Au plus tard la semaine prochaine, vous devrez créer un pull request avec votre code source et votre compte-rendu.

Un gestionnaire de scène

Sur votre application précédente, mettre en place un graphe de scène pour structure vos données. Ce graphe de scène sera partagé avec toutes les fenêtres.

Prévoir pour chaque feuille un ensemble de paramètres spécifique pour chaque fenêtre.

Intégrer un Octree

Maintenant, nous allons appliquer un octree sur notre terrain. Une fois ce travail réalisé, nous allons afficher ce terrain dans chaque fenêtre.

A l'aide du clavier, nous allons limiter la profondeur de lecture de notre arbre pour chaque fenêtre. Ainsi nous avons réalisé notre première solution de simplification (LOD) !

Sur la camera principale (première fenêtre), mettre en place une solution de multi-résolution progressive en fonction de la distance du sommet à la camera (le réaliser en temps réel).

Attention, à correctement trianguler votre surface.

Un gestionnaire de niveau de détails

Maintenant, nous allons nous intéresser un peu plus au niveau de détails. Pour cela, nous allons appliquer cette méthode à notre chargement d'objet 3D. Proposer une méthode de simplification d'objets 3D ainsi qu'une structure de donnée adapté.

Afficher l'objet sur la scène, et permettre le déplacement ce celui-ci. En fonction de la distance avec la camera, afficher un modèle plus ou moins simplifié.

Des imposteurs

Afin, d'habiller votre scène, mettre en place des imposteurs à la surface de votre terrain. Limiter dynamique ces imposteurs pour conserver un rendu temps réel.

Un peu de physique … avec une sphère

Enfin, nous allons expérimenter les enveloppes sur une sphère. Implémenter un comportement physique réaliste, et effectuer les calculs de collisions sur trois enveloppes englobantes (sphère, AABB, K-DOP). Quelle est la méthode la plus rapide ? Mettre en place une méthode multi-résolution pour le K-DOP.

Compte rendu

Présenter vos fonctionnalités
Expliquer votre démarche de développement.
Présenter votre structure de données.
Expliquer comment vous vous y prendriez pour les parties bonus.

Bonus

- Réaliser tous les calculs de physiques de manière parallèle : GPU, CPU

Réaliser tous les calculs de physiques grâce à une méthode d'accélération que nous avons vu en cours.

- Un peu de physique sur des modèles plus complexes

En plus de la sphère, charger des modèles avec une géométrie plus complexe, calculer les différentes enveloppes, et simuler leur comportement physique grâce à celle-ci.

- Mettre en place un BSP-tree, KD-tree

Pour votre terrain, nous allons stocker les données sous la forme d'un KD-tree et d'un BSP-tree. Appliquer pour chaque fenêtre une arborescence spécifique, et proposer la meilleure stratégie.

- Afficher une scène infinie

Elaborer la meilleur structure de données, ou la meilleur composition de structure de données pour réaliser une scène infinie. Appliquer des méthodes de multi-résolution, imposteurs, etc. en fonction de la position de la caméra.

- Garder un rendu temps réel