Projet C++ - Synthèse d'images

IMAC 2

—2015-2016—

Un parcours interactif

Ce projet vous permettra d'aborder les notions clés d'une part de la programmation objet (via le C++) et d'autre part du rendu 3D (via OpenGL). L'enjeu est ici de construire une architecture logicielle cohérente, efficace et qui réponde aux enjeux de visualisation demandée. Le thème est celui d'une visite virtuelle - un parcours interactif - dans un univers 3D, dont le thème est laissé à votre appréciation.

---

1 Introduction

1.1 Positionnement du problème

On souhaite créer une "plateforme" permettant de suivre une "visite" interactive d'un environnement virtuel. La plateforme permettrait de recréer, par exemple, une visite virtuelle de musée ou un jeu d'aventure dans l'esprit de Myst. Il s'agit d'une "balade" entre différents sites ou lieux qui présente une scène ou un objet particulier (et rendu de manière particulière).

Ce qui est attendu est :

— De pouvoir naviguer (automatiquement) d'un "site" à l'autre
— Chaque site doit permettre de naviguer vers d'autres sites
— Les "sites" et les choix sont définis dans des fichiers de configuration chargés à l'initialisation
— Chaque "site" doit représenter un objet ou une scène précise. Cet objet ou cette scène doit avoir une apparence à chaque fois particulière, différente.
— L'utilisateur doit pouvoir choisir la suite de son parcours.

1.2 Gestion de projet

Le projet devra être réalisé par groupe de 2 ou 3 et sera à rendre pour le XX décembre 2015. Une soutenance aura lieue afin, d'une part, de vérifier la réalisation projet final, et d'autre part d'exposer les compétences acquises par chacun des deux binômes pour ce projet. **Les projets devront pouvoir tourner en C++ sur les machines de l'université (en particulier celles de la salle 1B077)**

En terme d'architecture, chaque binôme utilisera un gestionnaire de projet (GIT) et nous donnera accès aux différents relevés (commit) du projet.

2 Les différents éléments de la plateforme

La réalisation logicielle demandée contient plusieurs modules ou éléments.

2.1 Le moteur de rendu (3D)

La plateforme doit contenir un moteur de rendu permettant de visualiser les différents sites, ceux-ci contenant une scène ou un objet à rendre.

Chaque site doit avoir un rendu (et donc un ou plusieurs shaders) particulier.

Le nombre minimal de site à réaliser est 3.

2.2 Le chargement de modèles et de texture (3D)

A l'aide de la bibliothèque tierce Assimp, vous devrez être capable de charger différents modèles complexes 3D. Afin d'alléger la consommation mémoire utilisés par les objets 3D, les modèles devront être chargés puis déchargés.

Imposer un manager de texture ou de scène ?

2.3 IHM (3D / C++)

Pour assurer l'interface avec l'utilisateur, vous exploiterez la bibliothèque SDL (vue en TD). L'utilisateur devra pouvoir d'une part interagir avec les sites, et d'autre part, naviguer entre les sites.

L'interaction avec les sites peut être minimale et, par exemple, permettre la navigation à l'intérieur du site (en mode FPS ou autre), ou bien répondre (via le clavier) à des questions. Bien entendu, des modes plus complexes d'interactions sont possibles (action sur un élément du site par exemple)

La navigation entre les sites doit être possible grâce à un choix de l'utilisateur (par exemple via une interaction avec le site mais cela peut être plus simple). Le passage d'un site à l'autre se fait instantanément.

2.4 Architecture (C++)

Vous devrez concevoir une architecture simple de classes qui décrivent et organisent les différents constituants de l'univers de votre parcours interactif. On pourra par exemple retrouver des classes comme Scene, Object, Room, Timeline, etc. Vous utiliserez l'héritage et la composition vu en cours pour modéliser l'univers ainsi que le scénario. Vous serez sûrement aussi bloqués sur des problèmes de conception et pour vous aider il est fortement conseillé de regarder du coté des design patterns. Voici quelques liens :

- https://sourcemaking.com/design_patterns
- http://gameprogrammingpatterns.com/contents.html
- http://www.amazon.com/Design-Patterns-Elements-Reusable-Object-Oriented-ebook/dp/B000SEIBB8

Sans dévoiler les questions que vous aurez à la soutenance, attendez vous tout de même à devoir expliquer dans quel cas vous avez eu recours à un design pattern.

3 Travail demandé

3.1 Éléments de rendu

Le code source sera contenu dans un repository GIT accessible par les enseignants.

Vous devrez produire également un très court rapport (environ 1 à 2 pages) présentant la description de l'architecture des classes et les problèmes rencontrés, ainsi que les questions en suspend. Pourras être rajouté des indications de résultats (snapshot, FPS) et/ou de performances.

Enfin il y aura une soutenance.

3.2 Critères de notation

— Respect des spécifications données
— Robustesse du code (résistance aux crash, gestion de la mémoire)
— Originalité et diversité des rendus graphiques (shaders)
— Lisibilité du code source
— Intérêt et originalité de la plateforme

4 Documentation technique

4.1 Bibliothèques utilisables

— Bien entendu la bibliothèque standard C++ (y compris C++11)
— SDL
— OpenGL, GLEW
— ASSIMP
— glimac et glm

Si vous souhaitez utiliser des bibliothèques autres, merci de nous le signaler.

Pour le gestionnaire de code, vous utiliserez GIT ou Mercurial.

5 Options & Bonus

Il est conseillé de rajouter à votre programme quelques bonus et options :
— Navigation entre les sites plus élaborée (vrai déplacement le long de courbe spline, fondu enchaînés...)
— Modèles massifs (plus de 100000 triangles)
— Modélisation par points

6 Conseils et remarques

— Ce sujet de projet constitue un cahier des charges de l'application. Tout changement à propos des spécifications du projet doivent être validée par M. Biri ou M. Pichard
— Le temps qui vous est imparti n'est pas de trop pour réaliser l'application. N'attendez pas le dernier moment pour commencer à coder.
— Il est très important que vous réfléchissiez **avant de commencer à coder** aux principaux modules, algorithmes et aux principales structures de données que vous utiliserez pour votre application. Il faut également que vous vous répartissiez le travail et que vous déterminiez les tâches à réaliser en priorité.
— Le projet est à faire par groupe de 2 ou 3. Il est **impératif** que chacun d'entre vous travaille sur une partie et non pas tous "en même temps" (un qui regarde l'autre travailler) sinon vous n'aurez pas le temps de tout faire.
— Utilisez des bibliothèques ! Notamment pour les types abstraits de listes et de piles.
— Nous sommes là pour vous aider. Si vous ne comprenez pas un algorithme ou avez des difficultés sur un point, n'attendez pas la soutenance pour nous en parler.
— Ce document peut changer. Je vous dirais lorsque de nouvelles versions dues à d'inévitables précisions ou corrections apparaîtront sur le site `igm.univ-mlv.fr/~biri/`.

Contacts divers :
`G+ : Venceslas Biri, mail : biri@u-pem.fr, Twitter : @vence_biri`
`G+ : Cyril Pichard, mail : cyril.pichard@gmail.com`
Bon courage.