IFT3100H17

Projet de session

- Visualiseur interactif de scènes 3D -

(partie 2/2)

Sommaire

L'objectif du projet de session est de développer une application qui permet de construire, éditer et rendre des scènes 3D.

Une scène contient un ensemble d'entités géométriques transformées dans l'espace et rendues du point de vue d'une caméra.

Le contenu des scènes peut provenir soit de fichiers de ressources externes à l'application ou encore être créé interactivement par l'utilisateur avec l'application.

L'apparence visuelle des entités visuelles est déterminée en fonction de l'éclairage de la scène et des matériaux assignés à leur surface, en combinaison avec différentes techniques de rendu.

Les choix liés aux technologies, au design logiciel et à la thématique du projet sont laissés à la préférence des membres de l'équipe.

Votre travail pratique sera principalement évalué selon votre implémentation de différents critères fonctionnels en lien avec la matière des modules du cours.

Un total de 50 critères fonctionnels organisés en 10 catégories vous est proposée dans les pages suivantes.

La première partie du projet de session (TP2) porte sur les critères fonctionnels des 5 premières catégories. La seconde partie du projet de session (TP3) porte sur les critères fonctionnels des 5 dernières catégories.

Il est recommandé d'implémenter environ 3 critères sur 5 par catégorie.

Document de design

1. Sommaire

Description sommaire du projet en exactement une page.

2. Interactivité

Décrire en détail comment utiliser l'application et expliquer clairement toutes les formes d'interactivité.

3. Technologie

Présentation des principaux outils technologiques utilisés pour le développement de l'application.

4. Architecture

Un ou des diagrammes qui permettent d'avoir une bonne vue d'ensemble de l'architecture logicielle de l'application.

5. Fonctionnalités

Pour chacun des critères fonctionnels implémentés :

- ▷ Description sommaire de comment le critère fonctionnel a été réalisé dans votre projet.
- ▷ Présenter l'essentiel du code qui a permis d'implémenter le critère fonctionnel.
- ▷ Si pertinent, présenter une ou des images où le critère fonctionnel est mis en évidence.

6. Ressources

Liste des ressources produites pour le projet (ex : image, géométrie, shader, etc) et citation des références pour le reste.

7. Présentation

Présentation de l'équipe et de ses membres.

1 Image

1.1 Importation

L'application permet à l'utilisateur d'importer des fichiers images et elle les utilise à des fins de rendu graphique ou comme source de données externe.

□

1.2 Exportation

L'application permet d'exporter des images ou des séquences d'images en réponse à une action de l'utilisateur.

□

1.3 Espace de couleur

L'application utilise un espace de couleur qui n'est pas du ton de gris ou du RGB (ex : du HSB).

□

1.4 Traitement d'image

Présence d'au moins 2 types d'effets visuels réalisés par traitement d'image (ex : masque, teinte, opacité, composition, filtre).

□

1.5 Image procédurale

Au moins une image utilisée par l'application est générée de manière procédurale par un de vos algorithmes.

□

2 Dessin vectoriel

2.1 Curseur dynamique

L'application utilise au moins 3 différentes représentations visuelles du curseur en fonction des états de l'application, de ses modes d'interactivité ou des actions de l'utilisateur.

□

2.2 Primitives vectorielles

Il est possible pour l'utilisateur de créer au moins 3 types des primitives vectorielles de manière interactive.

□

2.3 Formes vectorielles

L'application permet de créer des instances d'au moins un type de forme vectorielle composée d'un ensemble d'instances de primitives vectorielles.

□

2.4 Outils de dessin

L'utilisateur peut modifier interactivement la valeur des outils de dessin vectorielle tel que l'épaisseur de la ligne de contour, la couleur de la ligne de contour, la couleur de la zone de remplissage et la couleur d'arrière-plan de la scène.

□

2.5 Interface

Une ou des interfaces graphiques sont présentes pour offrir de la rétroaction informative à l'utilisateur et des contrôles interactifs pour influencer l'application.

□

3 Transformation

3.1 Transformation interactive

Il est possible de modifier interactivement la translation, la rotation et les proportions de toutes les entités géométriques présentes dans une scène.

□

3.2 Structure de scène

Toutes les entités géométriques présentes dans une scène sont organisées dans une ou des structures de données qui permettent de gérer les transformations et le rendu de chaque élément.

□

3.3 Sélection multiple

Il est possible de sélectionner plus d'une instance des entités géométriques présentes dans la scène et de modifier sur toute la sélection la valeur de certains attributs qu'elles ont en commun.

□

3.4 Coordonnées non-cartésiennes

Des entités géométriques sont transformées dans l'espace à partir de coordonnées non-cartésiennes (ex : coordonnées polaires, coordonnées cylindriques, coordonnées sphérique).

□

3.5 Historique

Il est possible pour l'utilisateur d'annuler ou de refaire (*undo* / *redo*) au moins les 5 dernières actions interactives qui ont un effet sur la transformation des entités géométriques présentes dans une scène.

□

4 Géométrie

4.1 Particules

L'application peut rendre un nuage de points avec représentation visuelle (ex : texture) avec une seule commande d'affichage.

□

4.2 Primitives

L'application permet de créer au moins deux types de primitives géométriques 3D générées à partir d'un algorithme sans aucune données externes à l'application (ex : solide de Platon).

□

4.3 Modèle

L'application permet de créer une ou des instances de modèles 3D à partir d'un maillage géométrique importé à partir d'un fichier externe (ex : un fichier *.obj*).

□

4.4 Texture

Il existe au moins un modèle texturé avec des coordonnées de mapping adéquatement distribuées sur la surface du maillage géométrique (n'importe quel modèle à l'exception de ceux-ci : triangle, plan, quad, cube).

□

4.5 Géométrie procédurale

Il est possible de générer prodéduralement un maillage géométrique dont l'élévation des sommets est en fonction de la couleur échantillonnée dans une texture (ex : *heightmap*, terrain, montagne, etc).

□

5 Caméra

5.1 Propriétés de caméra

Il est possible pour l'utilisateur de modifier les attributs des caméras les angles de champ de vision (horizontale et verticale), le ratio d'aspect et la distance du plan de clipping avant et arrière.

□

5.2 Mode de projection

Au moins deux type de mode de projection (ex : perspective et orthogonale) sont supportés par l'application et offrent à l'utilisateur un point de vue cohérent de la scène.

□

5.3 Caméra interactive

Il est possible de transformer une caméra par rapport à une ou des entités géométriques et l'utilisateur peut manipuler interactivement la caméra pour voir la position centrale de la sélection de différents points de vue, en plus de pouvoir s'en approcher et s'en éloigner.

□

5.4 Caméra multiple

Il est possible de voir une scène de plusieurs points de vue différents, en pleine fenêtre ou avec des sous-fenêtres d'affichage dans la fenêtre principale de l'application où le rendu est fait en simultané dans chaque fenêtre.

□

5.5 Caméra animée

Présence dans l'animation d'une séquence d'animation de la caméra d'un moins 3 secondes avec interpolation des valeurs d'au moins 3 attributs de transformation (ex : translation ou orientation sur un des 3 axes) ou propriétés de caméra (ex : champ de vision).

□

6 Pipeline de rendu

6.1 Portabilité

Pour au moins 2 paires de shaders (sommets + fragments), une déclinaison permet de supporter au moins 3 différentes versions de GLSL (ex : 120, 150, 330, 410, 450, ES1, ES2, ES3).

□

6.2 Shader de géométrie

Au moins un shader de géométrie est utilisé pour générer dynamiquement des données géométriques.

□

6.3 Rétroaction de transformation

L'application utilise d'une manière ou d'une autre des données retournées par le stade de rétroaction de transformation (*transform feedback*).

□

6.4 Passes de rendu

La boucle de rendu fait plus d'une passe de rendu pour générer le contenu d'un framebuffer.

□

6.5 Technique d'occlusion

L'application utilise un algorithme d'occlusion pour tenter de minimiser le nombre d'éléments du graphe de scène à rendre (ex : algorithmes de type BVH, BSP, quadtree, octree, kd-tree.

□

7 Illumination

7.1 Types de lumière

Il est possible d'avoir au moins une instance de chacun des 4 types de lumières classiques (ambiante, directionnelle, ponctuelle, projecteur).

□

7.2 Lumières multiples

Possibilité d'avoir au moins 4 différentes instances de lumières dynamique dont la couleur, la position et l'atténuation sont prises en compte lors des calculs d'illumination de surfaces présentes dans une scène.

□

7.3 Matériaux

Certains éléments visuels d'une scène ont un matériau sélectionné parmi un ensemble d'au moins 3 matériaux différents.

□

7.4 Modèle d'illumination

Le rendu des entités visuelles est fait à partir d'un modèle d'illumination qui supporte au moins une combinaison de lumière dynamique et de matériau selon au moins 2 modèles d'illumination parmi les classiques (ex : lambert, gouraud, phong, blinn-phong).

□

7.5 Volume de lumière

Il existe au moins un type de lumière qui permet d'émettre de la lumière seulement à l'intérieur d'un volume délimité par un modèle géométrique.

□

8 Lancer de rayon

8.1 Intersection

L'application est capable de calculer l'intersection entre un rayon et au moins 2 types de primitives géométriques.

□

8.2 Réflexion

Une technique de rendu inspirée des principes du lancer de rayon est utilisée pour rendre au moins 1 effet de réflexion (ex : une surface miroir).

□

8.3 Réfraction

Une technique de rendu inspirée des principes du lancer de rayon est utilisée pour rendre au moins 1 effet de réfraction. (ex : une surface en verre)

□

8.4 Ombrage

Une technique de rendu inspirée des principes du lancer de rayon est utilisée pour rendre au moins 1 effet d'ombrage.

□

8.5 Rendu graphique

Il est possible de rendre au moins 2 types de primitives géométriques qui sont rendues à partir d'un algorithme inspiré des principes du lancer de rayon (ex : *raycasting*, *pathtracing*, *raymarching*).

□

9 Topologie

9.1 Courbe cubique

Il est possible de rendre au moins 2 types de courbes cubiques avec 4 points de contrôle, comme par exemple une courbe de Hermite ou une courbe de Bézier cubique.

□

9.2 Courbe paramétrique

Il est possible de rendre au moins un type de courbe paramétrique avec plus de 4 points de contrôle, comme par exemple une spline de Bézier ou de Catmull-Rom.

□

9.3 Surface paramétrique

Il est possible de rendre une ou des surfaces paramétriques, comme par exemple une surface de Bézier bicubique ou une surface de Coons.

□

9.4 Shader de tesselation

Un shader de tesselation permet de sous-diviser la géométrie d'au moins 1 modèle, comme par exemple un plan ou une surface paramétrique.

□

9.5 Triangulation

Un algorithme permet de générer un maillage triangulaire à partir d'un ensemble de sommets, comme par exemple une triangulation de Delaunay un diagramme de Voronoï.

10 Techniques de rendu

10.1 Effet de relief

Il est possible de rendre un maillage géométrique où une texture est utilisée pour moduler l'orientation de la normale afin de simuler un effet de relief sur la surface, comme par exemple les techniques du *normal mapping* ou du *parallax mapping*.

□

10.2 Cube de réflexion

Un cube de réflexion (cube map) est utilisé comme source d'échantillonnage de données pour réaliser un effet visuel de surface, comme par exemple un effet de réflexion du cube map sur la surface d'un modèle.

□

10.3 BRDF

Il existe au moins un modèle d'illumination basé sur une fonction BRDF et qui n'est pas simplement un modèle d'illumination classique mis sous forme de BRDF.

□

10.4 Effet en pleine fenêtre

Il est possible d'appliquer un effet visuel sur toute la surface de la fenêtre d'affichage (ex : anti-crénelage, occlusion ambiante, filtres de convolution, post-process, etc).

□

10.5 Style libre

Réaliser un effet visuel à partir d'une technique de rendu de votre choix.

□

Évaluation

Voici les informations par rapport à l'évaluation du projet de session :

- ▷ Le projet de session est séparé en deux livrables : le TP2 et TP3.
- ▷ Le travail pratique 2 et 3 ont une valeur de 15 % de la session.
- ▷ Ils seront corrigés sur 100 %.
- ▷ Le premier 5 % sera pour le document de design
- ▷ Les 85 % suivants sera pour les critères fonctionnels.
- ▷ Chaque critère fonctionnel sera évalué de 0 à 3 étoiles, pour un maximum de 35 étoiles.
  ★★★ au-delà des attentes
  ★★ au niveau des attentes
  ★ en bas des attentes
- ▷ Tous les critères fonctionnels sont optionnels.
- ▷ Une implémentation de bonne qualité d'une de ces fonctionnalités vaudra généralement 2 étoiles.
- ▷ Le 10 % restant sera d'ordre qualitatif (ex : qualité du code source, structure de projet, respect des consignes et des nomenclatures, ampleur du projet, complexité technique, originalité, etc)
- ▷ Pénalité de 10% par jour de retard.

⋆bonus⋆

Des points bonus peuvent être accordés pour une application où entre autres :

- ▷ L'application fonctionne sur plus d'une plate-forme sans limitations.
- ▷ Des animations sont présentes et rendent l'application moins statique.
- ▷ Le projet atteint un niveau de qualité visuel au-delà des attentes.
- ▷ Le projet se démarque par son professionnalisme.
- ▷ Le projet se démarque par son originalité.

⋆malus⋆

Des points malus peuvent être retranchés lorsque entre autres :

- ▷ Le projet ne compile pas.
- ▷ L'application plante au démarrage.
- ▷ L'application plante en cours d'exécution.
- ▷ La configuration du projet et sa procédure de compilation sont complexes et mal expliquées.
- ▷ L'application est ergonomiquement déficiente.

Exemples

Voici des exemples de critères fonctionnels avec ses détails d'implémentations, tel qu'ils auraient pu être présentés dans le document de design d'un des projets. Les exemples sont présentés avec trois variantes de l'implémentation, qui auraient été évaluées respectivement à 1, 2 et 3 étoiles.

3.2 Structure de scène

La scène est implémentée sous forme d'une structure de données de liste qui contient de références vers tous les éléments de la scène. À chaque fois qu'une nouvelle instance d'objet est créée dans la scène, une référence est ajoutée à la fin de la liste. Dans la boucle de mise à jour de la scène, les entités visuelles sont rendues dans l'ordre où elles ont été ajoutées à la liste.

1 étoile

3.2 Structure de scène

La scène est implémentée sous forme d'une structure de données d'arbre qui contient des références vers tous les éléments de la scène. La relation parent-enfant entre les éléments de l'arborescence permet d'implémenter la notion d'hiérarchie, où la transformation d'un parent affecte la transformation de ses enfants. Dans la boucle de mise à jour de la scène, les entités visuelles sont rendues en parcourant en profondeur la structure de l'arborescence.

2 étoiles

3.2 Structure de scène

La scène est implémentée sous forme d'une structure de données de graphe dirigé acyclique qui permet de partager des éléments entre différentes instances d'entités, comme par exemple des matériaux et des maillages géométriques. La structure du graphe est éditable de manière interactive à partir d'une interface graphique. Dans la boucle de mise à jour de la scène, les entités visuelles sont rendues selon leur type et leurs options de visibilité en parcourant en profondeur la structure de l'arborescence.

3 étoiles

5.3 Caméra interactive

La caméra a été réalisé en intégrant la classe *ofEasyCam* de *openFrameworks* dans notre projet.

1 étoile

5.3 Caméra interactive

Nous avons réalisé notre propre caméra à partir de rien et elle est pleinement fonctionnelle. Par contre, les bornes de transformation ne sont pas validées et la caméra peut donc s'inverser ou accumuler une variation dans les angles qui rend la manipulation un peu difficile.

2 étoiles

5.3 Caméra interactive

La caméra est toujours orientée en direction de la position du point de focus, avec un seuil de distance minimale et maximale, et la possibilité de faire tourner la caméra autour de l'entité en Y (*yaw*) à l'infinie dans les deux directions et en X (*pitch*) dans un intervalle borné.

Des touches de clavier ou des actions à la souris permettre de faire un zoom ou un *dolly* en profondeur, tourner à gauche et à droite, et de haut en bas, toujours par rapport au point de focus observé.

L'animation du mouvement est fluide, avec une accélération du mouvement qui est intuitive et agréable pour l'utilisateur.

3 étoiles

Livraison

Voici les notes par rapport à la livraison du travail pratique :

- ▷ Livrer le travail pratique dans la boîte de dépôt de l'équipe sur le site web du cours sous forme d'archive de format ***.zip***.
- ▷ Tout le contenu du livrable doit se trouver à l'intérieur d'un répertoire qui respecte cette nomenclature où le 'XX' est substitué par votre numéro d'équipe dans la boite de dépôt.
  (ex : **IFT3100H17_TP3_09** pour le TP3 de l'équipe 9)
- ▷ Votre livrable doit contenir le document de design, les sources de l'application ainsi qu'une version compilée pour chaque plateforme supportée.
- ▷ Prière d'éviter les espaces et les caractères accentués dans tous les noms de fichiers et de répertoire.
- ▷ La date de livraison est le dimanche **12 mars** avant minuit
- ▷ La date de livraison est le dimanche **23 avril** avant minuit

10.6 Prix Pierre-Ardouin

Le projet de session est éligible au Prix Pierre-Ardouin, dont voici les principales informations :

Depuis l'automne 2013, le Département d'informatique et de génie logiciel a mis en place un concours récompensant l'équipe qui aura produit le meilleur TP/projet dans le cadre d'un cours. Ces travaux de session ont l'envergure d'un mini-projet qui est admissible par rapport aux normes fixées par le Département.

À la suite des évaluations des travaux, l'enseignant du cours détermine l'équipe gagnante ; chaque membre de l'équipe gagnante reçoit alors une bourse de 50$ ainsi qu'une attestation remises par le Département. De plus, le Département d'informatique et de génie logiciel a mis en place une bourse Élite, appelée bourse « Pierre Ardouin », qui vise à récompenser le meilleur projet de session, tous cours confondus.

Deux principaux critères guident le choix des évaluateurs dans l'identification du lauréat : l'excellence du travail (par rapport à ce qui est demandé dans l'énoncé) et l'aspect créativité/innovation. Il est actuellement prévu une bourse de 200$ pour récompenser chaque membre de l'équipe « élite » gagnante (pour un maximum de 1000$ pour toute l'équipe).

Aussi, le Département veille à publier l'information sur un site Web dédié : http ://www.ift.ulaval.ca/vie-etudiante/prix-pierre-ardouin.

À la deuxième moitié du mois de mai de chaque année universitaire, le Département organise une cérémonie pour honorer les finalistes et le lauréat du prix «Pierre Ardouin» des sessions d'automne et d'hiver, et leur remettre une attestation.»

Si vous souhaitez participer au concours, vous devez présenter votre projet sur le forum 'Présentation des projets' avant le 25 avril.