Année 2023—2024

Lancer de Rayons orienté objet

Une bibliothèque pour générer des scènes en utilisant la technique du lancer de rayons

B. Monsuez

Lancer de Rayons orienté objet

Une bibliothèque C++ pour implanter des moteurs implantant des techniques de lancer de rayons.

But du projet

Ce projet consiste à :

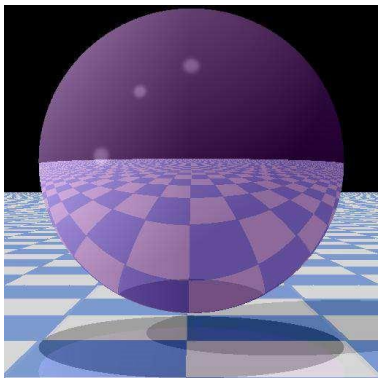
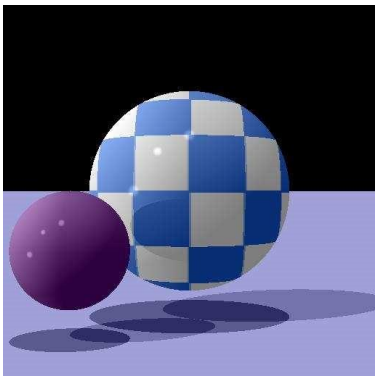
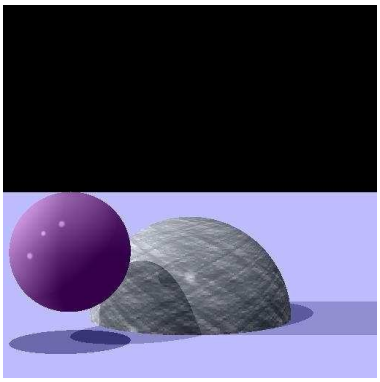
- Fournir une bibliothèque d'objets permettant de décrire les éléments composant une scène
- Fournir une bibliothèque d'objets permettant d'implanter différents moteurs implantant le rendu d'image par lancer de rayons

Et pour tester l'intégration de ces fonctions,

- Un environnement permettant de spécifier les scènes, de lancer l'exécution des moteurs et d'afficher le résultat :

Il s'agit de réaliser un code de calcul de lancer de rayons. Le lancer de rayons est une technique de rendu d'images de synthèse simple, mais relativement coûteuse en temps de calcul, connue pour ses résultats réalistes obtenus sur les ombres et les reflets.

Il s'agira donc, ici, de réaliser un moteur de lancer de rayons permettant de rendre des scènes composées d'objets géométriques simples tels que des sphères et des plans, au moins. On trouvera ci-dessous quelques exemples de scènes simples rendues à l'aide de cette technique.

Les scènes pourront être décrites par le biais d'un petit langage de description de scènes qui sera convertit vers les objets définis dans la bibliothèque. Ce langage peut-être soit basé sur de l'XML mais encore être un langage ad hoc que vous définissez de manière totalement libre.

De même, il faut faire attention dans les techniques de lancers de rayons. Une bonne introduction est présentée dans le document suivant : Techniques de lancer de rayons : un guide complet pour des visuels époustouflants - (lambdageeks.com). Ce document énumère les problèmes et les différentes techniques sans aller dans tous les détails mais en fonction du niveau de réalisme que vous souhaitez avoir, vous pouvez effectivement suivre les suggestions qui sont énumérées dans ce document.

Qu'est ce qui vous est demandé ?

Au niveau de la conception et de l'analyse

Vous vous trouvez dans la situation où vous devez réaliser une analyse quant à la faisabilité du logiciel, proposer une architecture de ce logiciel et ensuite implanter un démonstrateur reposant sur cette analyse.

1. Analysez le problème, sachant qu'il est souhaitable d'avoir une solution évolutive, permettant d'ajouter au fur et à mesure des fonctionnalités au logiciel, notamment progressivement étendre les formats supportés ainsi que les algorithmes de traitement d'image.

   De plus, comme il s'agit d'un outil devant apporter un bon confort d'utilisation, il est notamment demandé de faire :
   a. Une analyse d'usage du logiciel,
      i. Contexte d'utilisation,
      ii. Fonctions souhaitées par les utilisateurs de la bibliothèque à la fois pour la description des « composantes » de la scène que pour les moteurs d'exécution.
      iii. La définition des interactions.
      iv. La définition des fonctions de base du programme qui va vous permettre de tester votre programme.
   b. Une analyse fonctionnelle du logiciel
      i. Les différents blocs fonctionnels du programme (le rendu des images, la navigation, le décodage et l'encodage des fichiers de scènes, …)
      ii. Les interactions avec l'environnement au niveau fonctionnel
   c. Une architecture gros grains du logiciel.
      i. Les différentes classes principales
      ii. Les concepts de modularité et d'extensibilité,
      iii. Les interactions entre ces éléments.

   avant même de commencer réellement à étudier l'implantation de la bibliothèque ainsi que du programme de test.

2. Réalisez un démonstrateur pour valider l'approche et les choix techniques et s'assurer que ces derniers sont pertinents. Dans le cas d'une approche évolutive, il est conseillé d'implanter un noyau applicatif :
   a. Prise en compte des formes simples, sphères, parallélépipèdes, plans,
   b. Prise en compte des sources de lumières en un point.
   c. Prise en compte des couleurs.
   d. Un moteur simple de calcule par lancer de rayons.

   Puis de faire évoluer ce noyau avec de nouveaux formats d'objets supportés mais aussi de en y ajoutant la possibilité de définir des textures, de définir la transparence, les objets par description de facettes.

   Proposer ensuite des évolutions pour accélérer le temps de calcul (soit par une implantation SIMD (GPU) ou alors multithread ou en mixant les deux approches).

   Il est notamment demandé de faire :
   e. Une description du modèle d'architecture
   f. De bien documenter les choix des classes,
   g. De réaliser une implantation documentée du logiciel.

Au niveau des fonctionnalités de l'outil

L'outil devra fournir au minimum les services suivants :

Posséder une bibliothèque d'objets relativement riches permettant de décrire une scène à générer la plus réaliste possible.

Supporter les textures, couleurs, transparences et réflexions (avec atténuation).
Définir un langage permettant de positionner les objets et créer de nouveaux objets par composition d'objets.

Proposer une bibliothèque proposant des classes génériques encapsulant différents moteurs de lancer de rayons. Cette bibliothèque devrait permettre d'ajouter progressivement des versions améliorées des moteurs de lancer de rayons.

Proposer au moins un moteur utilisant soit le parallélisme soit l'utilisation des GPU pour accélérer le calcul.

Implanter un programme de génération d'images permettant d'éditer les fichiers de description de scènes, de sélectionner un moteur et de lancer la génération de la scène et d'en afficher le résultat.

Présentation de la technique du Lancer de rayons

Le lancer de rayons est une technique simple sur son principe qui considère un modèle élémentaire de caméra consistant en un écran rectangulaire derrière lequel se trouve un observateur ponctuel O.

Pour chaque pixel P=(x,y,$z_{écran}$) de l'écran, on étudie alors l'intersection de la droite passant par O et P avec les différent objets de la scène.

Parmi tous les objets intersectant la droite, s'il y en a pas la couleur associée au pixel est le noir, on choisit celui donnant le point d'intersection le plus proche de l'observateur. On créé alors un nouveau rayon allant du point d'intersection I vers chacune des sources (supposées ponctuelles) lumineuses L1, L2, ... éclairant la scène. Si ce nouveau rayon n'intersecte aucun objet alors la source lumineuse en question, Li, contribue à l'illumination du pixel proportionnellement au produit scalaire entre la normale à la surface au point I et le vecteur <I,Li>.

Eventuellement, on relance un ou deux rayons selon les caractéristiques de la surface (réflexion et/ou transparence), rayons qui contribueront également à l'illumination du pixel courant.

Vous pouvez commencer par la page Wikipédia sur le lancer de rayons http://en.wikipedia.org/wiki/Ray_tracing_(graphics) pour acquérir les bases de cette méthode ainsi que vous référé à la page : Techniques de lancer de rayons : un guide complet pour des visuels époustouflants - (lambdageeks.com).

Quelques éléments de Réflexion sur les différentes Etapes

Étape 1 : Analyse du problème

Phase A :

Les phases importantes de votre logiciel sont (1) l'algorithmique de calcul de la scène, (2) la description de la scène.

D'une part, analysez les différentes méthodes pour aborder le lancer de rayons. Comment génère-t-on les sources de lumières, comment s'effectue la propagation des rayons de la source à l'œil. Comment s'effectue les calculs avec quelles précisions. Rechercher les différents algorithmes proposés dans la littérature.

D'autre part, analysez les méthodes pour représenter un scène d'objet. Notamment, réfléchissez à comment définir un format de fichier permettant de décrire la position des objets dans la scène. Regarder éventuellement si un format de type XML ne pourrait pas être intéressant. De plus regardez aussi ce que réalise certains logiciels comme Blender.

Phase B :

Définissez l'interaction de votre logiciel avec l'utilisateur. Cette interaction est nécessaire pour sélectionner d'une part les scènes à calculer, pour sélectionner aussi les paramètres de calcul ainsi que de procéder aux affichages des scènes qui ont été calculées.

L'analyse architecturale du projet étant terminée, voici une introduction rapide aux quelques difficultés algorithmiques de ce projet.

La représentation de la scène

La scène est composée d'un ensemble d'objets et de sources lumineuses.

Les objets sont caractérisés par une géométrie. Les objets les plus simples sont les plans, les cubes, les sphères. Mais il est possible d'avoir des objets plus complexes comme notamment des objets définis par des maillages. Les objets peuvent éventuellement être définis comme une intersection ou une union de deux objets.

Les objets sont aussi caractérisés par une surface. Cette surface est caractérisée par une réflexion ainsi qu'un niveau de transparence. Ceci va donner lieu à une réflexion et une diffraction du rayon incident. De plus, la surface de l'objet n'est pas obligatoirement uniforme, elle peut être texturée.

Un intérêt du projet est de concevoir une architecture d'objet qui permet de supporter de manière extensive l'ajout de nouveau types d'objets sans avoir à redévelopper le moteur de calcul. Par exemple, un objet doit être en mesure de déterminer si une intersection existe avec un rayon et si c'est le cas, de générer le rayon réfléchi et le rayon difracté et ce en prenant en compte la surface de l'objet. La surface de l'objet pourra être une surface transparente uniforme ou une surface colorée. La surface pourra être aussi texturée, avec des points ayant des couleurs, des niveaux de réflexions et de diffractions différents.

Nous pouvons donc imaginer une architecture objet où nous avons une classe de base SCENE_BASE_OBJECT qui fournit un ensemble de services attendus par le moteur d'exécution comme par exemple être capable de déterminer si un rayon intersecte l'objet et de générer les rayons réfléchis, réfractés et diffractés. Des classes d'objet qui hériteront de cette classe de base permettront

de définir les calculs nécessaires pour une sphère, un cube ou tout objet plus complexes. La surface de l'objet est caractérisée par un objet OBJECT_BASE_SURFACE par exemple qui pourra ensuite être dérivée en une surface colorée uniforme, une surface transparente, une surface texturée, etc.…

Il faudra faire de même pour les différentes sources lumineuses. Ces sources peuvent être colorées, elles peuvent être omnidirectionnelles ou au contraire avoir un cône de diffusion. Ce cône de diffusion peut être homogène ou non homogène.

Enfin les rayons eux-mêmes peuvent aussi être représenté par un objet.

L'efficacité algorithmique

Les algorithmes de lancer de rayons nécessitent beaucoup de calculs. Cependant, ils peuvent être parallélisés pour utiliser au mieux les différents cœurs disponibles sur les plateformes actuelles.

Il sera intéressant d'améliorer tant que possible, notamment en essayant de paralléliser tant que possible l'algorithme pour pouvoir générer la scène dans le minimum de temps.

Étape 2 : Réalisation du démonstrateur

Phase A :

Procédez à une première implantation de l'algorithme en ne considérant qu'un nombre réduit d'objet, par exemple, un plan, une sphère et un cube ainsi que des surfaces homogènes. De même, pour les sources lumineuses, limitez-vous à des sources simples.

Phase B :

Procédez ensuite à définition du format de description des scènes et réalisez un lecteur vous permettant de charger les scènes en question.

Phase C :

Augmenter le nombre d'objets supportés, introduisez les textures ainsi que les sources de lumières plus complexes.

Phase D :

Procéder aux améliorations des fonctionnalités en traitant dans l'ordre :

- La parallèlisation de l'algorithme de rendu d'image,
- Une interface graphique légère permettant de charger un fichier, de lancer le calcul du rendu d'image et d'afficher l'image,
- La possibilité de sauvegarder les images dans différents formats comme par exemple le format .jpg et le format .png.

Étape 3 : Amélioration de l'outil

Vous avez le choix pour améliorer votre outil en augmentant notamment le nombre d'objets supportés, introduire de nouvelles sources de lumière, prendre en entrée éventuellement des fichiers de Blender, mais n'oubliez pas de conserver le résultat de l'étape 2.