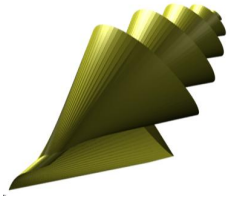
Master Informatique M1 - Université de Caen Basse Normandie

Projets d'Imagerie
POV-Ray & Blender

Grégoire BUREL

15 mai 2007



Résumé

Dans le cadre du module d'imagerie, nous avons réalisé deux projets distincts : un générateur de coquillages pour Blender (DreamSHELL) et une annimation POV-Ray représentant un partie du batiment « Science 3 ». Plusieurs difficultés ont été rencontrés pendant les projets : la modélisation d'objets non triviaux n'est pas adapté a POV-Ray même si l'usage des boucles est très pratique. Concernant la génération de coquillages, il n'est pas possible de généréer des coquillages plats en utilisant un algorithme classique de génération de coquillages hélicoidaux.

Mots clefs

Imagerie, POV-Ray, Coquillages, 3D.

Table des matières

Introduction

But du document

Le présent document explique comment les projets d'imagerie ont été implémentés ainsi que les difficultés rencontrés lors de la réalisation de « DreamSHELL », un générateur de coquillages 3D pour Blender et « Science 3 », une animation POV-Ray.

Documents de référence

Document	Auteur	Contenu
Projet d'imagerie	Jerzy Karczmarczuk	Sujet et cours.
POV-Ray	POV-Team	Introduction to POV-Ray.
Shells.sig92.pdf	D. R. Fowlery , H.Meinhardtz & P. Prusinkiewicz	Modeling seashells.

Site internet (sources)

http://users.info.unicaen.fr/~gburel/ue6/

1 Projet Blender : DreamSHELL

DreamSHELL est un générateur de coquillages pour blender, il possède une interface graphique, peut être facilement amélioré et être utilisé sans son interface graphique (par l'intermédiaire d'un autre script par exemple).

1.1 Manuel d'utilisation

1.1.1 Installation

Pour pouvoir utiliser correctement DreamSHELL, vous devez possèder blender (le logiciel a été testé sur la version 2.43 sous GNU/Linux avec succès et devrait normaelemnt fonctionné avec les versions ulterieures de Blender), DeamSHELL utilise la bibliothèque « Bsurflib » pour géner les surfaces en trois dimensions.

Le programme se trouve dans le répertoire « blender » de l'archive des projets (tar.gz) :

1. Copiez le contenu du répertoire qui contient « dreamshell.py » dans un endroit accessible faites aussi en sorte que les bibliothèques utilisés par dreamSHELL soit dans le « path » de blender.
2. Ouvrez le fichier « dreamshell.blend » puis exécutez le script Python (commande alt+p).
3. Utilisez l'interface pour construire votre coquillage.

1.1.2 Utilisation

L'interface graphique de dreamSHELL se divise en quatres parties (Fig.1.1, page 4) :

1. Courbes génératives.
2. Matériaux.
3. Paramètres.
4. Modèles prédéfinis.

La mise a jour du coquillage peut se faire automatiquement ou non par l'intermédiaire du bouton « Dessin auto ». Il est possible de quitter l'application via le bouton « Quitter » et de dessiner manuellement un objet via « Dessiner ».

Courbes génératives

Les courbes génératives existent sous cinq formats différents :

1. **Chemin (point) :** Ce mode de génération permet d'obtenir un visuel du chemin du coquillage.
2. **Carré :** Permet de générer un coquillage « cubique » (non réaliste).
3. **Demi-cercle :** La forme du coquillage est un demi-cercle.
4. **Coquillage 1 :** Un générative adapté au coquillages de type « Turitella ».
5. **Coquillage 2 :** Un générative triangulaire adapté au coquillages de type « Tatcheria ».

Il est possible d'améliorer facilement l'application en ajoutant de nouvelles courbes génératives (cf. Implémentation).

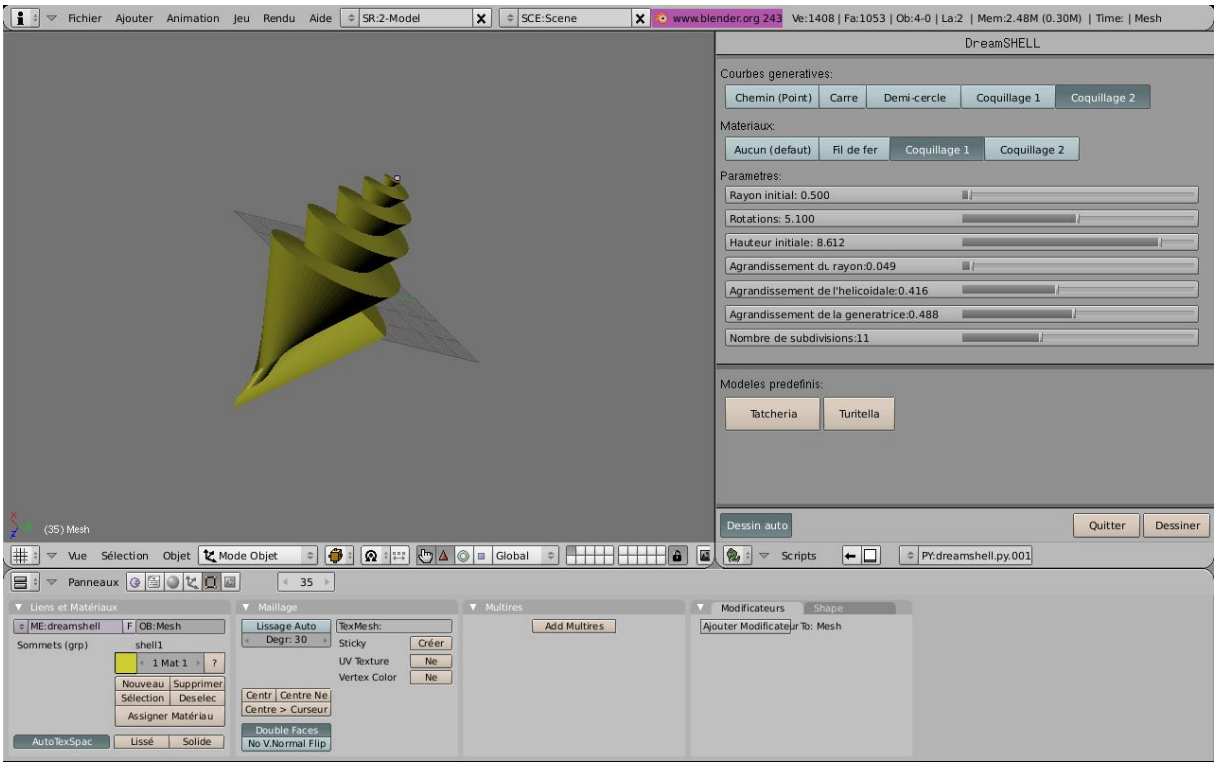


FIG. **1.1** – DreamSHELL en action

Matériaux

Il est possible de texturer les coquillages de quatre manières différentes : seul le mode de texturage « Fil de fer » est vraiment utile, il permet d'obtenir un rendu du maillage du coquillage. Les autre rendus se contentent de modifier la couleur de l'objet (la génération de texture est compliqué avec l'API blender et l'importation d'image ne semblait pas fonctionner).

Paramètres

Le coeur du programme ce situe dans cette partie. En faisant varier sept paramètres, il est possibles de fabriquer divers sortes de coquillages :

1. **Rayon initial :** La taille du rayon initial du coquillage.
2. **Rotations :** Le nombre de tours éffectués par la génératrice.
3. **Hauteur initiale :** La hauteur du coquillage à l'origine.
4. **Agrandissement du rayon :** Le coefficient d'agrandissement du rayon.
5. **Agrandissement de l'helicoidale :** Le coefficient d'agrandissement du chemin.
6. **Agrandissement de la generatrice :** Le coefficient d'agrandissement de la génératrice.
7. **Nombre de subdivisions :** Le nombre de subdivisions de l'objet (plus le nombre est élevé plus l'objet est lisse).

Modèles prédéfinis

Pour voir ce que peut produire le logiciel, des paramètres sont prédéfinis il permettent de construire deux coquillages réels (cf. ShellSig92.pdf) :

– Tatcheria.
– Turitella.

Il est bien entendu possible de créer d'autres coquillages réalistes manuellement.

1.2 Implémentation

DreamSHELL a été implémenté en Python/Blender. Pour construire les coquillages, nous nous sommes inspirés de l'article « ShellSig92 ». Le programme se divise en trois parties principales qui se chargent de générer le coquillage proprement dit :

- Une fonction générant le chemin hélicoidal.
- Une fonction construisant la génératrice.
- Une fonction ceffectuant la rotation de la génératrice ainsi que diverses opérations sur le chemin.

DreamSHELL peut être utiliser sans son interface graphique, grâce a son architecture Orientée Objet, par un autre script par exemple :

```
Shell = DreamShell("mySHELL")
Shell.height.val = 40
Shell.build()
```

L'interface graphique s'utilise très simplement :

```
Shell = DreamShell()
Shell.gui()
```

1.2.1 Chemin hélicoidal

Le chemin est un ensemble de points circulaire formant un hélice autour de l'axe Z. Celui-ci est afffecté selon plusieurs paramètres (rayon, hauteur...)

1.2.2 Génératrice

Un courbe génératrice est un ensemble de points formant une courbe fermé (le dernier et le premier point sont communs). Il est simple d'ajouter une nouvelle courbe au logiciel :

```
shell.generatives['nom'] = [Vec(0,0,0).....Vec(0,0,0)]
```

1.2.3 Rotation de la génératrice

Pour chaque point du chemin on translate et tourne un exemplaire de la génératrice en lui appliquant diverses transformations. Ensuite on relie chacun de ces points pour former les faces du coquillage.

1.3 Remarques

Les difficultés rencontrés pour créer le logiciel ont été de plusieures natures :

- Nous ne connaissions pas le langage de programmation Python il a donc fallut apprendre a s'en servir.
- Il nous a fallut comprendre comment fonctionnais les fonctions de création des faces avec l'API Blender.

La création des textures avec l'API blender est compliqué, certains résultats quant a la génération de courbes générative n'a pas semblé fonctionner totalement correctement (génération de demi-cercle par exemple).

2 Projet POV-Ray : Science 3

2.1 Modélisation

2.1.1 Terrain et Ciel

Le ciel a été réalisé grace 3 plans :
- Un pour le ciel bleu foncé.
- Deux autres pour les nuages plus bas que celui du ciel permettant d'avoir des nuages réalistes.

La lune a été crée en tant que source de lumière apparentée à une sphère et placé entre le ciel et les nuages et texturée par nos soins. Le terrain quant à lui est un simple plan texturé.

2.1.2 Objets de la scène

Le parc

Les arbres ont été crées grâce a une macro prise sur internet permettant de généréer les différents arbres (ce sont les seuls éléments de la scène qui n'ont pas étés crées par nous mêmes). Le plan d'eau est texturé de manière a obtenir une eau réaliste. Les dalles du sol utilisent du « bump mapping » pour parraître plus réalistes.

Le batiment

La facade du batiment est construite avec aplusieurs unions et boucles afin de créer chaque partie de la facade. Celles-ci sont texturés en conséquence.

L'escalier

L'escalier est une représentation de l'escalier de la faculté situé dans le batiment « science 3 ». Pour pouvoir créer cette escalier on a utilisé une boucle while avec différents paramètres dont :
- Le nombre de marches.
- La longueur de la marche.
- L'angle de rotation.
- La hauteur de la marche.
- La distance entre chaque marche.
- La profondeur de la marche.

Rambarde et balcons

Les rambardes sont réalisés a partir d'une « spline » et de nombreux éléments assemblés entre eux avec la méthode de composition union/différence. A partir d'un de cet objet, on réalise les balcons par l'intermédiaire de boucles.

L'intérieur de batiment

Le batiment est réaliser avec de nombreuses boucles permettant de dupliquer les étages ou divers autre éléments.
Les textures ont étés réalisés par nos soins, certaines utilisent des photographies de l'université retouchés avec « Gimp ».

2.2 L'animation

La trajectoire est une spline qui a pour type « natural_spline ». La camera est dirigé de manière a obtenir une vue correcte.

2.3 Remarques

En ayant voulu reproduire de manière relativement réaliste l'interieur du batiment « Science 3 », ainsi qu'en réalisant tous les objets nous même, nous nous sommes heurté a de nombreuses difficultés :

- Les objets étaient plus compliqués a modèliser que prévus.
- Relier les travaux de chacun n'était pas très simple (correspondance de l'interieur et exterieur).

Concernant l'animation, plusieurs difficultés sont aussi apparues :

- Trouver une trajectoire et un timing pour les points de passage satisfaisant pour créer la video.
- Créer un éclairage naturel et réaliste (on a par exemple dû supprimer l'éclairage interne des spots du batiment qui rendait les calculs d'une image extrêmement long et donc impossible a utiliser dans la video).

Conclusion

La réalisation de ces deux projet a été très instructive en nous permettant d'approcher des aspects différents de la réalisation d'images en trois dimensions. Les deux outils (Blender et POV-Ray) ont leurs avantages et inconvénients :

- Blender permet de voir en tant réel ce qui est modèlisé et de créer des scripts interactifs.
- POV-Ray permet de réaliser des boucles facilement ce qui est très utile pour réaliser des objets architecturaux. Néanmoins, dès que l'on souhaite modèliser des objets compliqués il est prèférable d'utiliser un modeleur interactif, qui permet de voir en tant réel les modifications (il a été souvent frustrant de metre plusieures heures a modéliser des objets en POV-Ray alors que cela aurait été beaucoup plus rapide avec Blender).

Même si certains résultats ne sont pas exactement proche de la réalité, globalement il est facile de reconnaître que ce que nous avons modèlisé est l'interieur du batiment « Science 3 ». Et que l'on peut générer beaucoup de sortes de coquillages avec Blender même si il faut un certain doité pour obtenir un résultat réaliste.

Table des figures