Rapport d'avancement sur le stage du 2 Avril au 11 Juin 2012

Sommaire

Nouveautés :

Depuis wave_main, avec l'interface graphique on peut choisir un fichier dat pour en faire un inc avec Rendering > Dat2Inc.
Depuis un inc on peut obtenir une image via povray en utilisant Rendering > Picture.
Rendering > Movie permet d'obtenir une vidéo à partir d'un dossier de .dat ou .dat.gz
Rendering > 3D Modelisation permet d'obtenir une vue en 3D de l'objet décrit dans un .dat et de se déplacer dans cette scène, l'objet étant modélisé en fil de fer.

Ce projet wave peut être renommé à tout moment en utilisant le script change_noms.sh. Une fois lancé il est expliqué qu'il suffit de choisir le nouveau préfixe pour le projet, par exemple taper "truc" reviendrait à obtenir des truc_main.cc, truc_mainframe.cc, truc_mainframe.h et ainsi de suite. Dans les fichiers, les #include seront modifiés afin d'inclure les fichiers correspondant au nouveau nom choisi.
Une sauvegarde des fichiers avant le changement de préfixe est effectuée, les fichiers originaux se voient attribuer l'extension .backup. Si vous êtes sûr de ne pas avoir causé de problème en choisissant un mauvais préfixe vous pouvez supprimer ces sauvegardes en lançant efface_sauvegardes.sh.

Poissons

Maintenant que j'ai terminé la transition entre opengl et povray je trouve qu'il serait peut-être plus intéressant de ne pas effectuer une rotation de toute la scène sauf la caméra dans opengl et ainsi une fois la scène retranscrite pour povray on aurait juste un déplacement de la caméra et on pourrait garder un contexte fixe. Alors que là on a la caméra qui reste au même endroit et il faut recalculer la position de tous les éléments de la scène par rapport au poisson, pour que ça reste cohérent.
J'ai calculé une position relative au poisson pour que le fond marin reste sous le poisson mais si on veut aussi ajouter des plantes, des rochers et autres choses dans la scène il faudra alors recalculer leur position à tous dans le programme.
Cependant si on choisit de déplacer uniquement la caméra on pourrait déclarer tout les objets de la scène dans le .pov et leur emplacement resterait toujours le même, seule la caméra se déplacerait.
J'ai constaté que dans la partie "camera" sous povray, up ne sert pas à indiquer le ciel par rapport à la caméra, il faut utiliser sky. Par défaut up vaut 1y.
Up doit être utilisé avec right pour donner le ratio de l'image à la sortie. Par exemple
Up 9y
Right 16x
Donnera un ratio 16/9 pour l'image.

L'en-tête du .inc créé a été modifié et il faut utiliser ces valeurs accessibles grâce aux declare dans le .pov. Le fichier essai01.pov illustre une utilisation de ces données.

```
#declare PdV=<0.0000000e+00, 0.0000000e+00, 0.0000000e+00>;
#declare LAt=<0.0000000e+00, 0.0000000e+00, -3.8000000e+01>;
#declare fond_marin=
 plane {<0.0000000e+00, 9.2848587e-01, 3.7136841e-01> ,-3.0000000e+02 }
#declare lumiere= light_source {<-4.5044281e+01,1.3544740e+02,8.7273788e+01>  color White}
```

Dans la fonction ecritMesh2_UVmapping() du fichier fish_opengl.cc à la ligne 430, on peut voir la section dans laquelle il faudra intégrer l'ajout des points u et v de l'uv-mapping, le mieux serait de les écrire dans le même ordre que les points associés dessinant le poisson.

Povray

J'ai passé une bonne partie du stage sous povray à tester différents paramètres pour trouver un bon rendu de l'eau avec la caméra dans l'air ou dans l'eau tout en évitant des méthodes qui demanderaient un temps de calcul trop important.
Pour obtenir le meilleur rendu sous l'eau j'ai pu lire un compte-rendu en anglais d'une personne qui a fait du bon travail mais pour ça il a été forcé de tricher en créant plusieurs sources lumineuses "dirigées" (spotlight) au dessus de la surface. Pour donner un bel effet à la surface il n'a d'ailleurs pas utilisé d'IOR parce que le rendu n'était pas assez bon et trop long, il a superposé 2 plans, un blanc et un qui simule la réfraction. Pour donner un effet de particules dans l'eau il a aussi créé une "boîte" juste devant la caméra transparente à 99% et avec des particules à ce que j'ai compris, il a aussi dû la faire tourner sur elle-même pour que les rayons de lumières frappent correctement les particules de la boîte donnant ainsi un effet intéressant.
J'ai retenu de tout ça que pour donner un bon effet d'eau avec povray il faut éviter les cas "généraux" en bombardant la scène de photons mais envisager des cas particuliers et créer des pseudo-filtres devant la caméra mais déplacer la caméra et le filtre que l'on met devant pourrait ne pas s'avérer suffisant en fonction de l'angle entre les sources de lumière et la caméra.

J'ai testé différentes valeurs de bumping pour tenter "d'adoucir" les formes de l'eau parce que la vague tec17.dat n'est pas "lisse", j'ai cherché longtemps des méthodes pour obtenir un meilleur rendu telles un mesh de smooth_triangle ou appliquer un bumping mais le smooth_triangle prendrait trop de temps à mettre en place et le bumping n'est pas uniforme mais aléatoire ce qui fait que pour une vidéo on aurait des images qui se suivent avec des reflets très différents donc je n'ai rien fait pour adoucir les angles de la vague.

Film

Le fichier cng_wave.tcl a été réécrit en c++ et inclus au code dans le mainframe.cc, la création de .inc est située dans opengl.cc.
En utilisant Rendering > Movie, on peut désormais créer des png ou tga (selon l'option donnée au lancement du programme) via povray. Ces images et les .inc associés seront dans un dossier movie_tmp situé dans le même dossier que le programme. Les fichiers définissant la vague peuvent être des .dat ou .dat.gz.
Lorsque vous choisissez un dossier le programme vérifie la présence de .dat.gz, s'il y en a il ne s'occupera que d'eux même s'il y a aussi des .dat s'il n'y a pas de .dat.gz il cherchera des .dat.

Il n'y a pas de limite au nombre de fichiers donnés en paramètres (ou plutôt la limite est celle de la mémoire vive puisque les noms des .dat sont stockés dans la ram le temps de l'exécution de Movie).

On peut distinguer 3 étapes lors de l'exécution de Movie :

La génération de .inc à partir des .dat (passage par MarchingCube pour obtenir un mesh)

La génération de .png via povray à partir des .inc obtenus à l'étape 1.

La génération d'un mpeg à partir des .png de l'étape 2.

En exécutant le programme avec l'option -serveur, l'exécution de Movie sera différente. Seule l'étape 1 sera exécutée. Un script lance_povray.sh sera créé et les dossiers movie_tmp, pov et ce script devront être copiés sur le serveur imath01, il suffira alors de lancer le script pour lancer l'étape 2. L'étape 3 n'est pas prise en compte puisqu'elle ne donne pas de résultat stable, il m'est arrivé d'avoir l'ordinateur "immobilisé" pendant une demi-heure le temps d'avoir pour réponse que convert mpeg ne fonctionne pas. De plus sur le serveur la libffmpeg n'est pas disponible pour ImageMagick.

Avec l'option –mcDone on peut sauter l'étape 1 si jamais on l'a déjà terminée mais qu'on a besoin de faire la suite. Il faudra malgré tout sélectionner le dossier contenant les dat ou dat.gz utilisés lors de la création précédente de inc.

L'utilisation combinée de -mcDone et -serveur permet d'obtenir lance_povray.sh sans faire aucune des 3 étapes.

Blocages :

À chaque problème que je devais résoudre ou à chaque fonctionnalité que je devais implémenter j'imagine qu'en moyenne 2/3 du temps a été consacré à me documenter par internet.
Pour parler du programme en lui-même, c'est sur la partie consacrée à opengl pour fish où il fallait retranscrire la même vue dans povray que j'ai du bloquer le plus longtemps. D'une part à cause de cette inversion de vecteur (entre y et z) dans le .dat que constitue le poisson et d'autre part pour des petites erreurs quand je me suis trompé de vecteur de référence pour faire des rotations ou encore quand je multipliais chaque point par la matrice d'opengl que je récupérais, j'avais fait la multiplication en imaginant que la matrice était construite ligne par ligne mais j'ai mis du temps à me rendre compte qu'elle était construite colonne par colonne, ce qui faisait que la vue de face sous opengl était presque identique à ce que j'obtenais sous povray, en pivotant selon un axe j'obtenais le poisson à l'envers et j'imaginais

que c'était dû à un oubli dans une inversion entre y et z alors que c'était tout simplement la matrice qui se lisait en colonne et non en ligne.
Le temps de m'en rendre compte j'ai pu lire et apprendre par cœur plusieurs fois le code du programme…

Création d'une vidéo à partir de fichiers images :

Je n'ai pas réussi à résoudre un problème à la création d'un mpeg depuis plusieurs .png.
S'il n'y a pas trop de .png (environ 100) la conversion semble bien se dérouler même si le lecteur a parfois un peu de mal à commencer la lecture de la vidéo à partir du début ce qui me laisse penser que l'encodage effectué par ImageMagick via la commande convert ne respecte pas les standards. Mais s'il y a trop de png (600 par exemple) la commande convert ne s'effectue pas, j'ai le message d'erreur suivant :
échec de délégation à ffmpeg.
Sur internet j'ai trouvé une méthode pour résoudre ce problème de délégation consistant à enlever "-flags +4mv+aic" de delegates.xml dans /usr/lib/ImageMagick-6.2.8/config/delegates.xml.
Ceci nécessite les droits super utilisateur donc ce n'est pas envisageable de l'implanter dans le programme, de plus ça n'a pas marché pour moi et je ne connais pas les possibles effets indésirables qui pourraient advenir à la suppression de ce flag.

Pour résoudre ce problème je n'ai trouvé que 2 solutions :

-Utiliser "windows live movie maker", c'est assez rapide mais le résultat n'est pas parfait et le seul encodage disponible est wmv (windows media video). Pour réduire le poids de la vidéo et utiliser un codec utilisée presque partout de nos jours sans altérer la qualité de la vidéo, j'ai utilisé un programme que j'utilise souvent sous windows pour l'encodage de vidéo, MeGUI, pour transformer le wmv en mp4.

-Utiliser un logiciel payant, adobe premiere pro. Le résultat est plus lent, la prise en main du programme est difficile mais les formats de sortie sont assez varié, la qualité est bien meilleure en choisissant un codec h264 avec un conteneur mp4.

Césures entre les domaines:

Lorsque la vague est créée à partir de plusieurs domaines on constate des césures qui peuvent être très fines aux bords des domaines, le phénomène devient évident lorsque le bord d'un domaine coïncide avec les bords d'un volume à un instant t, alors on constate un grand "trou", une absence de maillage dans le volume d'eau sur toute la surface qui est sur la limite du domaine.
Je n'ai pas pu trouver de méthode simple pour résoudre ce problème puisque les dimensions des domaines ne sont pas fixes, sur une face un domaine peut avoir plusieurs voisins ainsi ces domaines adjacents n'auront qu'un seul domaine voisin sur cette face.
Vu les nombreux changements apportés aux .dat il n'est pas impossible de voir un jour des domaines qui définissent juste des points contigus qui ne forment pas pour autant un parallélépipède donc inutile de penser à créer un quadrillage sur les bords des domaines pour chercher les points les plus proches.

Le mieux serait que les personnes qui créent les .dat fassent un dernier domaine décrivant les points aux extrémités des domaines avec les hexaèdres que forment ces points.

Dépendances de WxWidget :

Sur le serveur et avec ubuntu au-delà de la version 10.xx, il manque au programme une bibliothèque libgio-2.0.so.0 et en cherchant sur internet j'ai pu voir que cette bibliothèque appartient à gtk et pour ajouter ces bibliothèques en statique il faut télécharger les sources de gtk (>=2.0.0 pour wxwidget).
Je n'ai pas réussi à résoudre ce problème avec la version 11.10 de ubuntu et pour le serveur en tentant es chercher et compiler les sources on a pu constater qu'il faut d'autres versions plus récentes de libgc or beaucoup de programmes sur le serveur imath ont des dépendances sur la libgc installée et la mise à jour de cette dernière provoquerait des problèmes sur beaucoup d'autres programmes.
C'est pourquoi on a choisit de modifier le programme en lui ajoutant des options pour exécuter la partie nécessitant wxwidet sur un ordinateur personnel avec wxwidget, puis pour la partie génération d'images qui prend le plus de temps on peut effectuer cette seconde partie avec le serveur.