Chef d'œuvre
Equipe OSIRIX

Rapport de recette

15 Février 2012

Table des matières

Bibliographie

- « Interactive Reflection Editing »
  (SIGGRAPH Asia 2009), par Tobias Ritschel, Makoto Okabe, Thorsten Thormählen et Hans-Peter Seidel

  http://www.mpi-inf.mpg.de/resources/ReflectionEditing/

- « Interactive On-Surface Signal Deformation »
  (SIGGRAPH 2010), par Tobias Ritschel, Thorsten Thormählen, Carsten Dachsbacher, Jan Kautz et Hans-Peter Seidel

  http://www.mpi-inf.mpg.de/resources/OnSurfaceDeform/

I - Introduction

Le but de ce chef d'œuvre est de permettre l'édition interactive d'images de synthèses, en agissant sur les différents aspects visuels qui la composent. Pour cela, il faudra implémenter les différents algorithmes et techniques présentés dans les sources suivantes : « Interactive Reflection Editing » et « Interactive On-Surface Signal Deformation ».

La recette est le fruit de notre travail, ce que nous remettons au client, il est aussi le point de départ des personnes qui reprendront notre code en souhaitant l'améliorer, le corriger ou y ajouter des fonctionnalités.

Ce rapport décrit le code de notre équipe, la conception finale de notre programme ainsi que les fonctionnalités prévues depuis le cahier des charges remis au client le 5 Novembre 2011 et le rapport de conception générale du 20 Décembre 2011.

Nous y joignons un manuel d'utilisation complet de notre logiciel.

II – Conception finale

La conception finale varie un petit peu dans les noms et dans certains retours de méthodes suite à des erreurs remarquées lors de l'implémentation. La classe appelée scène dans les précédents rapports se retrouve dans *openglwidget* dont une partie était fournie par le client.

Notre conception utilise le patron d'architecture MVC (Modèle Vue Contrôleur), afin de garantir de bonnes interactions entre l'interface (IHM) et le logiciel. Le contrôleur s'assure donc de la bonne transmission des événements afin que les méthodes soient appelées au bon moment.

Le diagramme de classes d'implémentation qui suit permet de se rendre compte de notre conception, et permet ainsi de voir les relations entre les différentes entités du système.

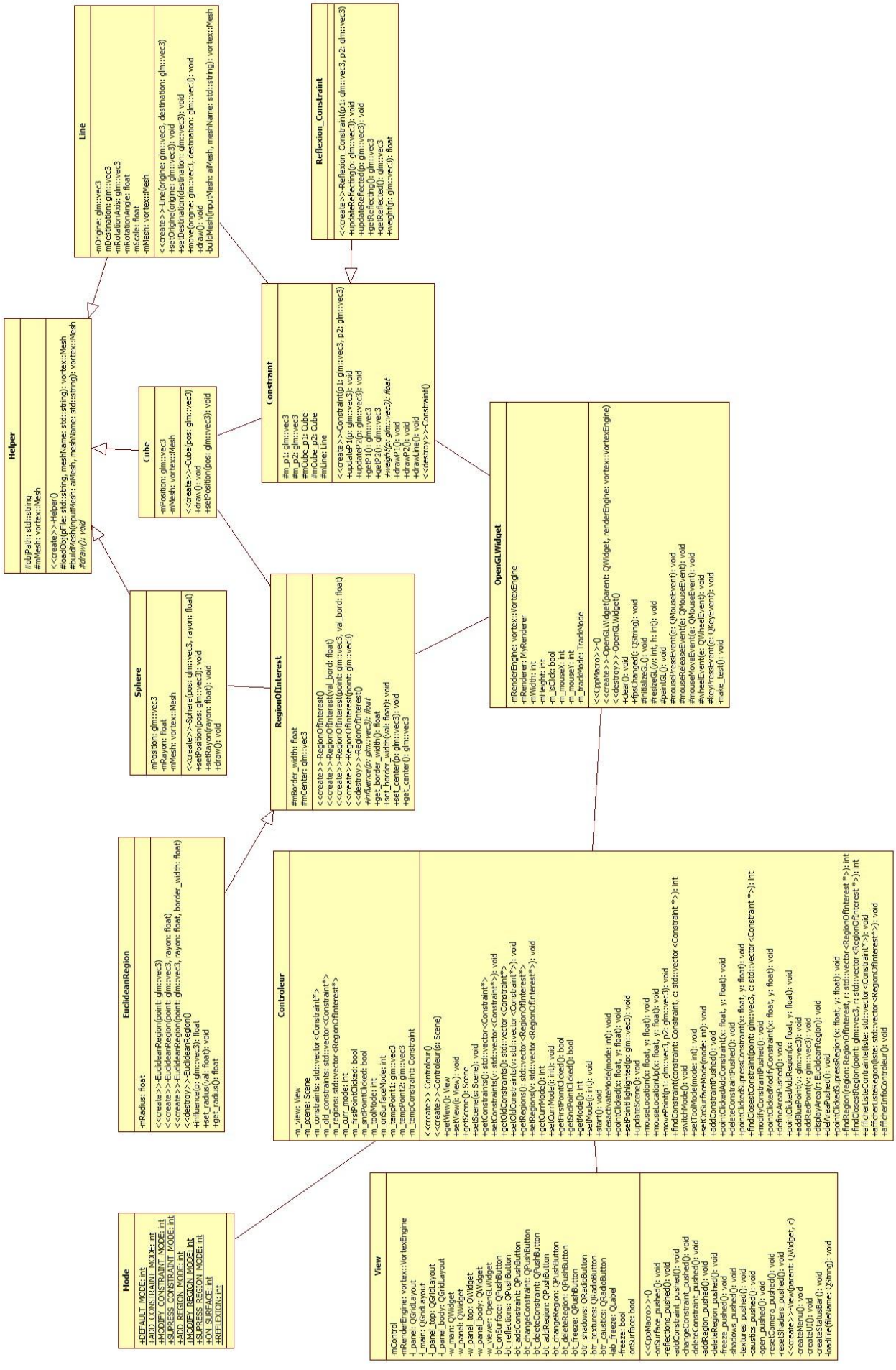

Figure 1 Diagramme de classes d'implémentation

Afin d'avoir une meilleure lisibilité, nous avons repris la partie modèle de notre diagramme, qui correspond aux classes spécifiques de notre logiciel.

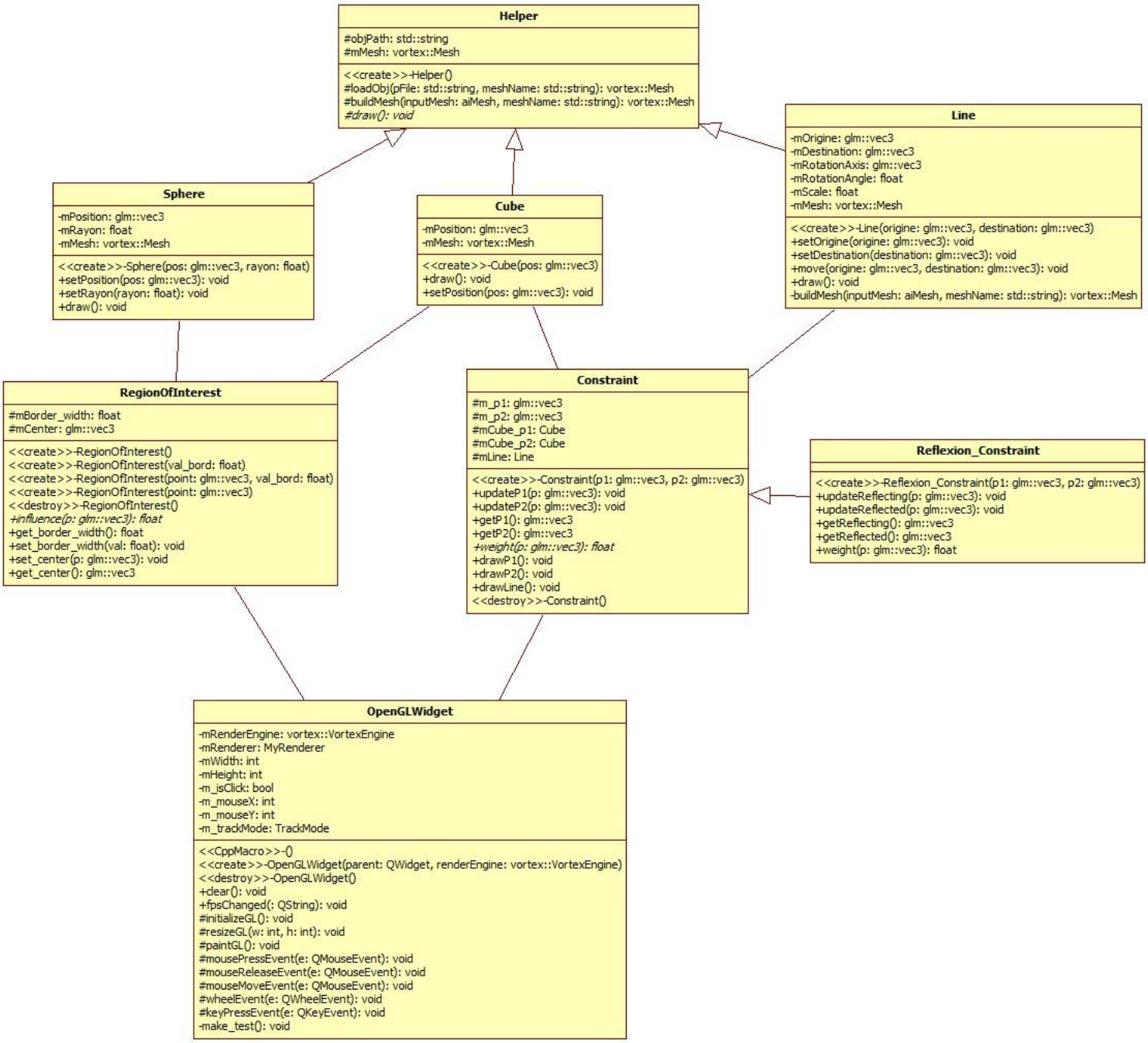

Figure 2 Partie modèle du diagramme de classes d'implémentation

III – Fonctionnalités principales

1) Sélection d'un mode

La fonction de sélection d'un mode est gérée par l'IHM de notre programme, lorsqu'elle se lance l'application OSIRIX permet à l'utilisateur de choisir entre le mode « Sur la surface » et le mode « Reflexions ». Chacune d'elles proposant à l'utilisateur d'effectuer les opérations propres au mode. Nous n'avons pas implémenté le mode « Sur la surface » mais notre application met à jour, suite au clic de l'utilisateur, les tableaux de contraintes dans notre classe gérant la scène (*openglwidget*). Il faudrait aussi changer le pipeline de rendu de l'application (ne faisant également pas la même chose selon le mode) changer les shaders à utiliser etc.

2) Créer une contrainte

La fonction créer une contrainte permet d'ajouter une contrainte sur la scène. Elle s'exécute après que l'utilisateur ait cliqué sur le bouton « ajout d'une contrainte » le curseur de la souris change et suite à un clic de l'utilisateur une contrainte est crée et ajoutée à la liste de contraintes de la scène (dans *openglwidget*). Une contrainte, comme spécifié dans nos précédents rapports, aura une influence inversement proportionnelle à la distance des points sur lesquels elle s'applique. Visuellement l'utilisateur voit un cube rouge apparaitre à l'endroit de son clic symbolisant la contrainte, qui est lié à un autre cube vert qui est le point réfléchi (point de destination de la contrainte de réflexion).

3) Créer une région d'intérêt

La fonction créer une région d'intérêt est implémentée, c'est le constructeur d'une région d'intérêt, l'utilisateur peut suite à ses clics déterminer une région d'intérêt euclidienne aillant un centre et un certains rayon. Visuellement le centre est de couleur bleu. Cette zone détermine la zone d'effet d'une contrainte, conformément aux précédentes spécifications l'influence des contraintes sur les points dans la région d'intérêt varie de 1 à 0 : proche de 0 aux bords et proche de 1 au centre.

4) Relation entre un point 2D et un point 3D

La fonction *unProject* (également appelée findPoint dans des rapports précédents)a été implémentée quasi totalement par le client, elle permet à partir de deux coordonnées (un pixel) d'obtenir sa position dans la scène (un point en 3D). Comme décrit dans les rapports précédents cette fonction intervient fréquemment dans notre programme pour permettre à l'utilisateur d'effectuer ces actions avec la souris directement sur la scène qu'il souhaite modifier.

5) Figer les réflexions

La fonctionnalité figer les réflexions n'a pour l'instant pas été implémenté, nous ne savons pas si elle le sera lors de la présentation. Car pour l'instant nous n'avons pas tout à fait terminé l'édition des réflexions. Cependant nous avons prévu un bouton pour cette fonctionnalité dans l'interface ainsi que placé les méthodes à remplir dans notre code pour son fonctionnement.

IV – Fonctionnalités secondaires

Comme écrit précédemment nous n'avons pas implémenté le second algorithme, présenté dans « Interactive On-Surface Signal Deformation », nous ne nous sommes pas non plus attaquées aux fonctionnalités secondaires, vu que nous désirions avant de faire cela que nos fonctionnalités principales soient implémentées.

Cependant vu que le but de notre projet ne se limitait pas seulement à un travail d'implémentation nous avons grâce à notre conception, implémenté nos fonctionnalités de façon à ce qu'elles fonctionnent si les secondaires existaient. Cela grâce entre autres à des listes de pointeurs sur des classes abstraites qui spécifient les attributs et les méthodes qui devront être implémentés par les classes héritantes.

V – Organisation du travail

Comme préconisé par les enseignants nous avons utilisé au maximum l'outil Redmine et son svn. Ainsi nous avons dès que possible ajouté un GANTT sur Redmine, reprenant celui créé pour le rapport de conception détaillée. Malheureusement celui-ci ne s'est pas avéré exact, en effet durant notre travail nous nous sommes rendu compte que des choses manquaient dans notre conception, dès que nous nous en sommes rendu compte nous avons réagi au plus vite en éditant le GANTT de Redmine. Cela a eu pour conséquence de nous faire perdre du temps alors que nous avions déjà un planning plutôt serré pour réaliser notre application.

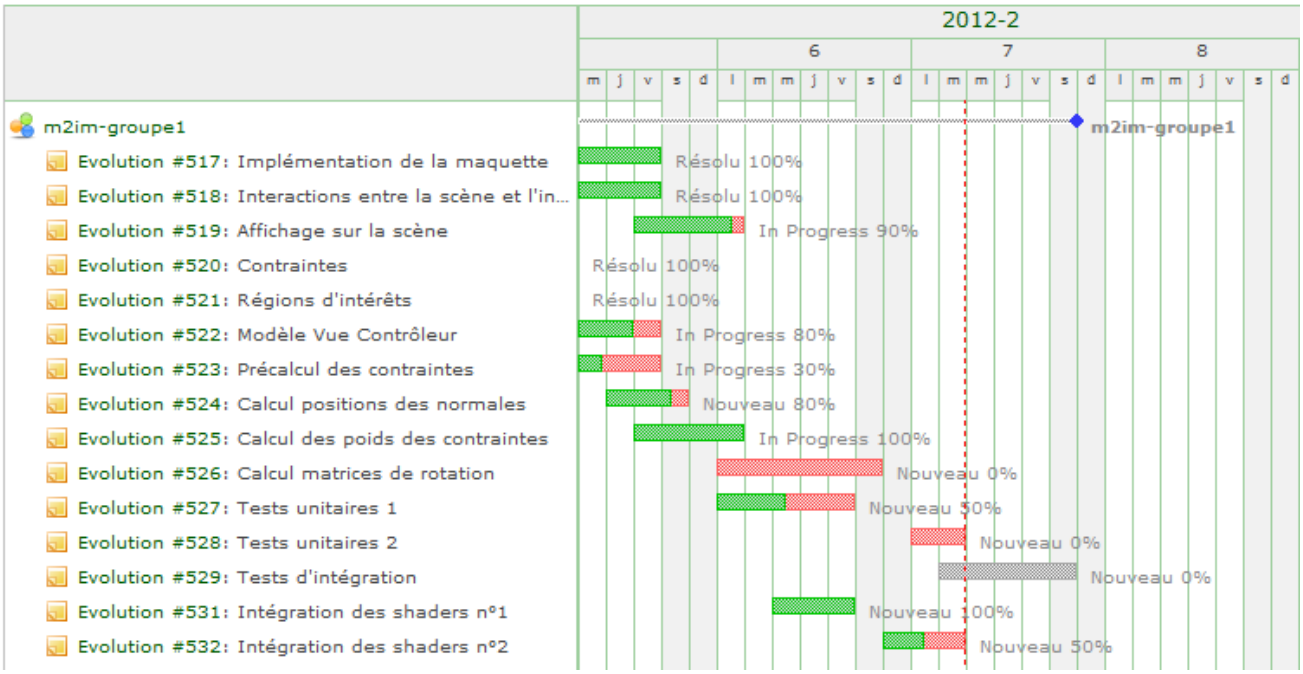

Figure 3 GANTT du projet sur Redmine

L'outil svn nous a servi pour regrouper le code et permettre de travailler plus rapidement et dans des lieux différents. Cependant le fait que personne dans le groupe ne le maitrisait très bien nous a également causé des problèmes, cet outil capricieux dans la salle disponible pour les étudiants (112 en U3) est désormais assez bien connu par tous les membres de l'équipe. Selon les conseils de notre client nous avons séparé notre projet en fonctionnalités indépendantes (d'après nos analyses) et nous avons ainsi créé des branches pour chacune d'elle. Une fois la fonctionnalité terminée elle était ajouté au programme principal.

Notre projet n'est pas terminé, dû à ces retards et aussi au fait que le travail nous a paru complexe, nous avons butté sur des choses que nous ne pensions même pas devoir faire. Malgré tout nous sommes satisfaits du travail effectué et de l'investissement des membres de l'équipe. Nous remercions

également notre client et enseignant Monsieur David Vanderhaeghe pour son aide et sa disponibilité. L'équipe pense qu'il lui aurait fallu au moins 3 voir 4 semaines de plus pour terminer le programme.

VI – Tests Unitaires

Pour s'assurer du fonctionnement du logiciel pas à pas, notre équipe à effectuer plusieurs tests unitaires dès lors qu'une fonctionnalité semblait terminée.

Les tests unitaires présentés dans le rapport de conception détaillée interviennent après l'implémentation d'un des papiers publiés de la bibliographie, ce qui n'est pas encore totalement le cas à l'heure de la remise du rapport.

Nous avions également prévu d'implémenter une méthode permettant de charger une scène déjà éditée (avec des régions d'intérêts et des contraintes) pour pouvoir faciliter ces tests, fonction qui n'a pas était réalisée.

VII – Produits

1) Logiciel

Le principal produit que nous avons réalisé pour ce projet est le logiciel permettant d'effectuer l'édition interactive de rendu et qui met donc en œuvre les différents algorithmes implémentés.

Conformément aux spécifications et aux maquettes qui ont été réalisées pendant la phase d'analyse, la maquette permet de charger une scène 3D et naviguer à l'intérieur de celle-ci grâce à la partie visualisation. L'autre partie permet d'interagir avec la scène afin d'y apporter les modifications souhaitées.

Figure 4 Interface du logiciel créé

2) Site web

Afin de mettre en avant le travail réalisé tout au long de ce projet, nous avons réalisé un site web. Etape faisant partie intégrante du travail demandé pour ce module, la construction du site internet a été réalisé afin de rendre compte de l'importance du projet.

Les différentes pages du site sont ainsi représentatives du travail possible à réaliser en sortant de la formation que nous avons suivie, le master 2 Image & Multimédia. Le site web est donc une vitrine pour la formation, ainsi que pour nous-même afin d'avoir des références concrètes à mettre en avant.

Figure 5 Page d'accueil du site web

Le site web permet également de connaître complètement le projet ainsi sa mise en place grâce à la mise à disposition des différents livrables que nous avons réalisés au cours de l'analyse et de la conception de notre logiciel.

Le site web sera mis à jour à la fin du projet afin de montrer le résultat final que nous avons réussi à atteindre.

Nous avons mis à disposition les pages web de notre projet à l'adresse suivante : http://dev.fakeornot.eu/osirix/

Index des illustrations