|  |
| --- |
|  |
| Cahier Des Charges |
| Projet 3 : Laboratoire SYMME |
|  |
| **BERTHET Charly, CHENAL Vincent, DUPARC Audry, LETOURNEUR Léo, YOCCOZ Geoffrey** |
| **2013-2014** |

|  |
| --- |
|  |

**Sommaire**

[I. Historique du document 2](#_Toc372285922)

[II. Introduction 3](#_Toc372285923)

[III. Présentation de la société 4](#_Toc372285924)

[IV. Présentation générale du projet 5](#_Toc372285925)

[V. Organisation du projet 6](#_Toc372285926)

[VI. Etude de l’existant 7](#_Toc372285927)

[VII. Les besoins fonctionnels 8](#_Toc372285928)

# Historique du document

|  |  |  |
| --- | --- | --- |
| **Version** | **Travail effectué / Raison de la modification** | **Date** |
| 1.0 | Création/Mise en page document | 13/11/2013 |
| 1.1 | Correction des besoins | 4/12/2013 |
|  |  |  |

# Introduction

Le laboratoire SYMME (SYstèmes et Matériaux pour la MEcatronique) est un laboratoire de l'Université de Savoie, dont les activités sont tournées vers les avancées technologiques et la santé.

Le but du projet présenté est d’établir tout d’abord un état de l’art sur OpenGL et ensuite, de développer une application permettant de créer une représentation 3D d’un objet à partir de fichiers images sources.

Cette application sera utilisée pour l’automatisation de contrôle qualité de pièces mécaniques et doit s’intégrer à une structure logicielle déjà existante.

# Présentation de la société

**Raison sociale :** Laboratoire SYMME (SYstèmes et Matériaux pour la MEcatronique)

**Statut :** Individuel

**Capital :** NC

**Date de création :** 2006

**Situation géographique :** Haute -Savoie, Annecy

**Siège social :** 7 chemin de Bellevue,

74944 Annecy le Vieux.

**Activités :** Recherche en mécatronique

**Effectif :** 60 personnes

# Présentation générale du projet

* Le sujet

Le projet se déroule en deux étapes majeures, tout d’abord l’élaboration de l’état de l’art, puis la conception de l’application qui s’appuiera sur les différences, les méthodes, les concepts répertoriés dans notre rapport.

* Les objectifs principaux
* Réaliser un logiciel mettant en œuvre les techniques décrites dans l'état de l'art.

L'application doit permettre de d'afficher une représentation 3D calculée à partir d'une ou deux images afin de simuler des effets de lumière.

Nous gèrerons principalement l'aspect technique (utilisation d'OpenGL, fenêtrage, rendu par GPU).

* Position du projet dans la société
* Les services et personnes concernés

Les principaux utilisateurs du logiciel seront :

* Maurice Pillet, l'enseignant chercheur en charge du projet de contrôle qualité.
* Simon-Frédéric Desage, doctorant au sein du laboratoire SYMME.
* Les résultats attendus

Le but de l’état de l’art est pour M. Desage de s’appuyer sur celui-ci pour la méthode de rendu OpenGL, l’application, elle, a pour but d’être réutilisée à l’avenir , pour permettre le rendu d’une image.

# Organisation du projet

* **Mandant :**

M. Maurice PILLET, Professeur des Universités

* **Comité de pilotage :**

M. Simon DESAGE, Doctorant

M. Vincent COUTURIER, Enseignant suiveur

* **Equipe de projet :**

M. Simon DESAGE, Doctorant

M. Charly BERTHET

M. Vincent CHENAL

Mlle. Audry DUPARC

M. Léo LETOURNEUR

M. Geoffrey YOCCOZ

* **Maitrise d’ouvrage :**

M. Simon DESAGE, Doctorant

* **Maitrise d’œuvre :**

M. Vincent COUTURIER, Enseignant suiveur

M. Charly BERTHET

M. Vincent CHENAL

Mlle. Audry DUPARC

M. Léo LETOURNEUR

M. Geoffrey YOCCOZ

* **Prestataire :**

IUT Annecy-le-Vieux

M. Charly BERTHET

M. Vincent CHENAL

Mlle. Audry DUPARC

M. Léo LETOURNEUR

M. Geoffrey YOCCOZ

# Etude de l’existant

Le laboratoire SYMME possède un dispositif à la détection des anomalies d’aspect qui a été développé au sein même du laboratoire. Le but est de remplacer le mode de vérification actuel où les pièces sont inspectées de façon manuelle en faisant varier les angles d’observation et l’incidence de la lumière sur la surface en fonction de l’orientation de la pièce.

Le logiciel utilisé pour récupérer les données par le dispositif est un programme édité sous Matlab. Le dispositif est basé sur l’analyse et la modélisation de l’interaction surface/lumière.

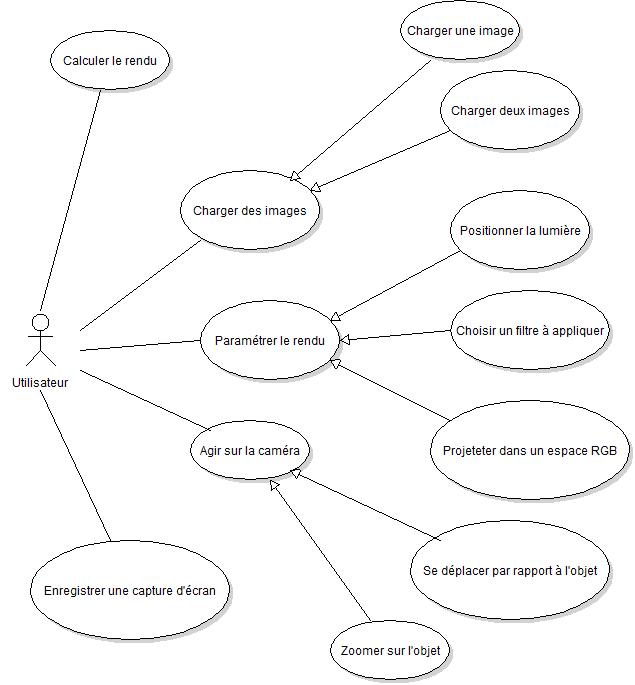
L’observateur est matérialisé par un système de prise de vue (un appareil photo numérique), des sources de lumières (LED) qui sont réparties sur toute la surface d’un dôme. L’objet est placé au centre de ce dôme, verticalement par rapport au système de prise de vue.

Des sources lumineuses s’allument progressivement et une photo est prise pour chacune d’entre elles. Le programme créé sous Matlab prend alors en charge l’ensemble d’images et génère un fichier afin de simuler le rendu de la pièce. L’utilisateur peut ensuite mettre en évidence ce qu’il souhaite voir en détail, quelle que soit la position de la source lumineuse qui éclaire.

Le dispositif attendu devra inspecter les pièces sans les manipuler, simuler de manière instantanée le rendu visuel d’une surface sous un éclairage choisi afin de mettre en évidence les anomalies.

Le programme que nous devons développer en C++ devra utiliser OpenGL pour reconstruire des objets à partir d’une ou deux images dont l’information sera ensuite traitée et rendu. Notre objectif est de montrer en quoi l’utilisation d’OpenGL et des shaders permettrait d’améliorer les performances du programme précédent.

# Les besoins fonctionnels



* **Cas d’utilisation** : Calculer le rendu

**Acteur primaire** : L’utilisateur

**Objectif** : Calculer le rendu 3D à partir des fichiers sources en tenant compte des différents paramètres de rendu.

**Pré conditions :** Avoir séléctionné une ou des image(s) d’un objet à étudier.

**Post conditions :** Le rendu est effectué.

**Scénario nominal :**

1. L’utilisateur clique sur le bouton de rendu.

2. L’application calcule le nouveau rendu en fonction des fichiers sources ainsi que des paramètres de rendu.

* **Cas d’utilisation** : Charger une image

**Acteur primaire** : L’utilisateur

**Objectif** : Charger l’image voulue pour la reconstruction 3D

**Pré conditions :** Avoir une image d’un objet à étudier.

**Post conditions :** L’image est chargée dans l’application

**Scénario nominal :**

1. L’utilisateur clique sur le bouton de chargement.

2. Le programme affiche la fenêtre d’exploration de l’ordinateur.

3. L’utilisateur sélectionne l’image.

4. L’utilisateur confirme son choix.

5. L’application enregistre le chemin absolu de l’image.

6. L’application charge l’image.

**Scénario Alternatif :**

2a1. L’utilisateur annule le chargement.

2a2. L’application retourne à la fenêtre graphique.

**Scénario d’exception :**

6a1. L’application ne parvient pas à charger l’image.

6a2. L’application affiche un message d’erreur.

* **Cas d’utilisation** : Charger deux images

**Acteur primaire** : L’utilisateur

**Objectif** : Charger les images de l’objet nécessaires à la reconstruction 3D.

**Pré conditions :** Avoir deux images d’un même objet à étudier, espacées de 6 cm (stéréocscopie)

**Post conditions :** Les images sont chargées dans l’application.

**Scénario nominal :**

1. L’utilisateur clique sur le bouton de chargement.

2. Le programme affiche la fenêtre d’exploration de l’ordinateur.

3. L’utilisateur sélectionne les images.

4. L’utilisateur confirme les images.

5. L’application enregistre les chemins absolus des images.

6. L’application charge les images.

**Scénario Alternatif :**

2a1. L’utilisateur annule le chargement.

2a2. L’application retourne à la fenêtre graphique.

**Scénario d’exception :**

6a1. L’application ne parvient pas à charger les images.

6a2. L’application affiche un message d’erreur.

* **Cas d’utilisation** : Positionner la lumière

**Acteur primaire** : L’utilisateur

**Objectif** : Changer la position de la lumière par rapport à l’objet.

**Pré conditions :** Avoir un rendu affiché dans la fenêtre graphique

**Post conditions :** Le rendu est mis à jour en fonction de la modification

**Scénario nominal :**

1. L’utilisateur clique sur le cercle de positionnement de la lumière.

2. L’application met à jour le rendu en fcontion de la nouvelle position de la lumière.

* **Cas d’utilisation** : Choisir un filtre à appliquer

**Acteur primaire** : L’utilisateur

**Objectif :** Appliquer un filtre sur la ou les images sources pour améliorer le rendu

**Pré conditions :** Avoir chargé des images

**Post conditions :** Le rendu est recalculé en fonction du filtre appliqué

**Scénario nominal :**

1. L’utilisateur selectionne le filtre à appliquer dans la liste déroulante.

2. Le programme recalcule le rendu en utilisant le filtre selectionné.

* **Cas d’utilisation** : Projeter dans un espace RGB

**Acteur primaire** : L’utilisateur

**Objectif** : Changer les composantes du repère 3D afin d’afficher le rendu 3D non pas dans un repère X,Y,Z (Coordonnées spatiales) mais R,G,B (Couleurs)

**Pré conditions :** Avoir un rendu affiché dans la fenêtre graphique

**Post conditions :** Une nouvelle fenêtre contenant un nouveau rendu 3D sur un repère RGB apparait

**Scénario nominal :**

1. L’utilisateur coche l’option d’affichage RGB.

2. L’application ouvre une nouvelle fenêtre contenant le rendu de l’image sur un repère RGB.

**Scénario alternatif :** Desactivation du rendu RGB

1. L’utilisateur décoche l’option d’affichage RGB.

2. L’application ferme la fenêtre contenant le rendu 3D sur le repère RGB.

* **Cas d’utilisation** : Se déplacer par rapport à l’objet

**Acteur primaire** : L’utilisateur

**Objectif** : Changer l’angle de vision de la camera par rapport à l’objet

**Pré conditions :** Avoir un rendu affiché dans la fenêtre graphique

**Post conditions :** Le rendu est recalculé avec le nouvel angle de vision

**Scénario nominal :**

1. L’utilisateur fait un « cliquer-glisser » sur la fenetre de rendu.

2. L’application met à jour le rendu en fonction du nouvel angle de vision.

* **Cas d’utilisation** : Zoomer sur l’objet

**Acteur primaire** : L’utilisateur

**Objectif** : Rapprocher ou Reculer la camera par rapport à l’objet

**Pré conditions :** Avoir un rendu affiché dans la fenêtre graphique

**Post conditions :** Le rendu est recalculé avec la nouvelle position de la camera

**Scénario nominal :**

1. L’utilisateur utilise la molette de la souris ou les boutons « + »/ « -» sur la fenetre de rendu

2. L’application met à jour le rendu en fonction de la nouvelle position de la camera.

* **Cas d’utilisation** : Enregistrer une image

**Acteur primaire** : L’utilisateur

**Objectif** : Enregistrer une capture d’écran de la fenetre de rendu à un moment donné

**Pré conditions :** Avoir un rendu affiché dans la fenêtre graphique

**Post conditions :** La capture d’écran est enregistrée dans le dossier choisi par l’utilisateur

**Scénario nominal :**

1. L’utilisateur clique sur le bouton pour enregistrer une image.

2. L’application ouvre un sélécteur de dossier.

3a.L’utilisateur choisi un dossier et valide

4a. L’image est enregistrée dans le dossier correspondant.

**Scénario alternatif :**

3b. L’utilisateur appuis sur le bouton d’annulation.

4b. L’application quitte le sélécteur de dossier et revient sur le fenêtre de rendu.

# Contraintes

* **Les contraintes organisationnelles :**
* Le projet est à vocation pédagogique (projet tutoré dans le cadre de la 2ème année de DUT)
* Accessibilité des locaux : le développement se fait à l’IUT.
* Disponibilité : suivant le planning des étudiants et du Maître d’œuvre.
* **Les contraintes Techniques :**
* Système d’exploitation Windows 7
* Langage de programmation C++
* Compilateur Visual Studio 2010 64bits
* Utilisation de l'API OpenGL pour le rendu ainsi que du framework QT pour créer l’interface

# La solution :

* **Architecture technique :**
* Langages : C++, GLSL, LaTeX pour l’Etat de l’Art
* Bibliothèques : OpenGL, QT, OpenCV
* Utilisation de Shaders
* IDE : Microsoft Visual Studio
* **Lots :**

L’application sera livré par lots.

Dans le cadre du projet, seul l’état de l’art sera remis pour la 1ière soutenance.

**1ier lot :**

La création d’une application en C++ utilisant les méthodes trouvées dans l’Etat de l’art.

Il comprend :

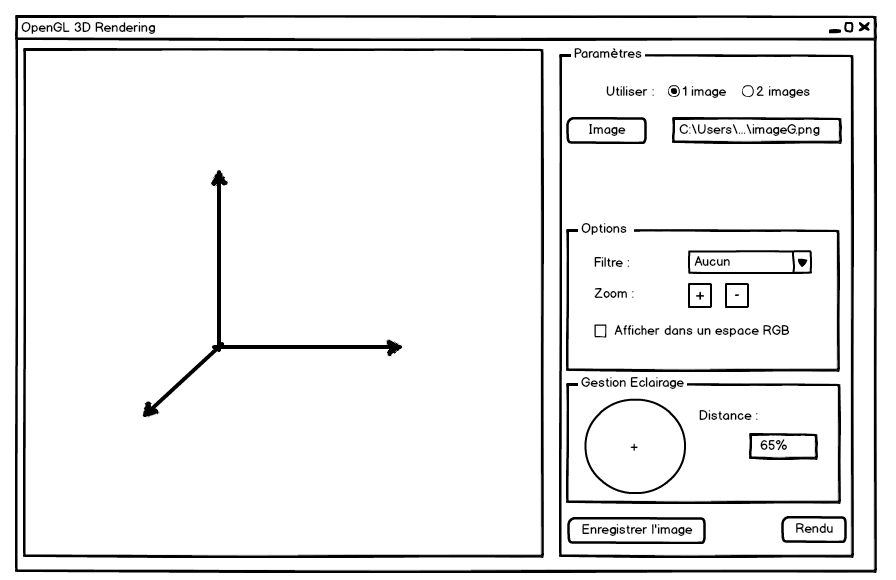
* Une IHM,
* La reconstruction à partir 3D d’une image,
* La possibilité de modifier l’éclairage de la scène,
* La possibilité de bouger autour de l’objet et de zoomer dessus,
* La mise en place de filtres permettant de changer le rendu de la pièce.

**2ième lot :**

Reprend les fonctionnalités du premier lot, mais permet de reconstruire le modèle 3D à partir de deux images (Principe de la stéréoscopie).

* **Les Maquettes :**

**L’IHM par défaut :**



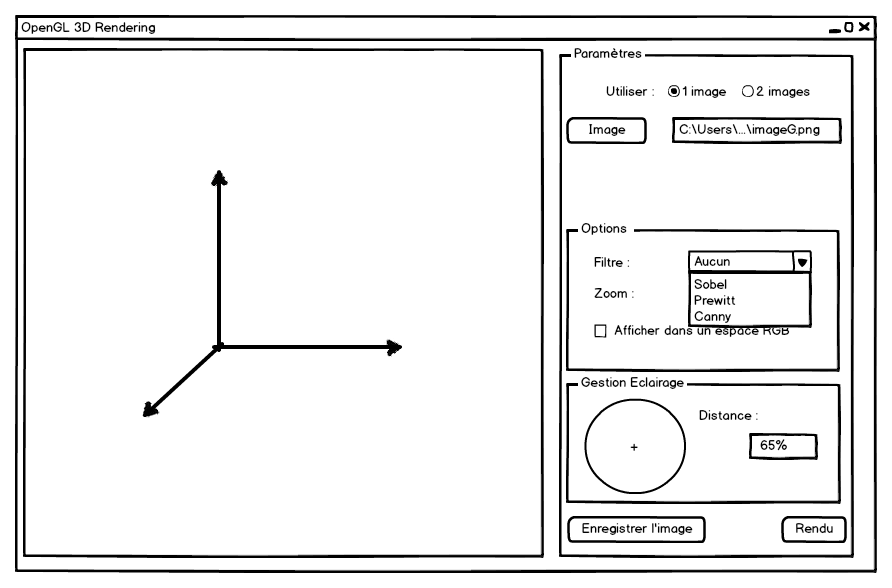
Utilisation de l’IHM sans paramétrer le rendu.

Charger une image en appuyant sur le bouton « image ».

Appuyez sur le bouton « Rendu » pour lancer le rendu dans le repère XYZ.

**Les Options :**

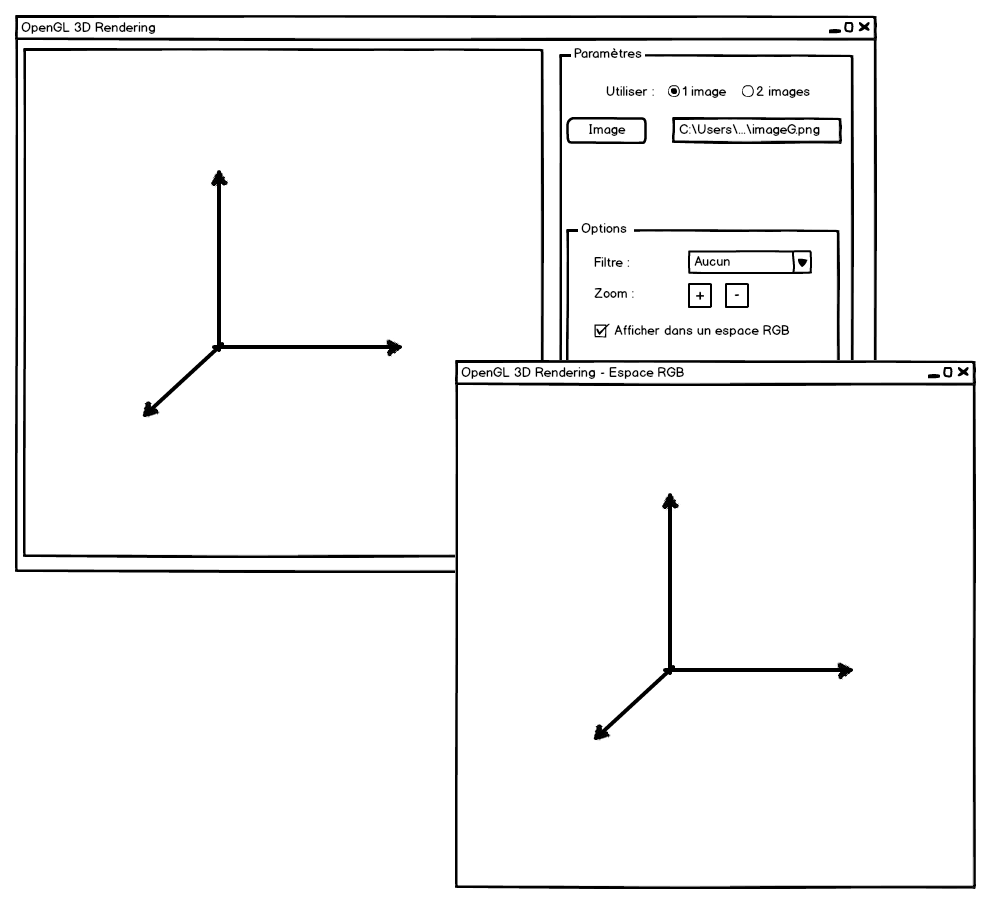
**Filtre :**



Il y a possibilité d’utiliser filtre pour mettre en avant certains aspects de l’image.

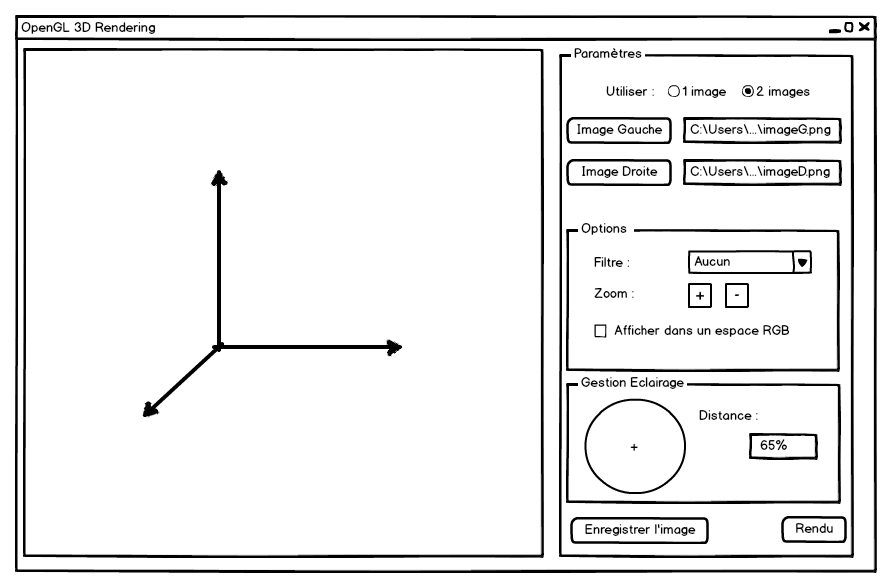
Allez dans le carré « Options », ouvrez la liste déroulante « Filtre » et choisissez le filtre désiré.

**Afficher dans un espace RGB :**



Pour voir apparaitre la fenêtre de rendu dans un espace RGB positionnez-vous dans le carré « Options » et cochez la case « Afficher dans un espace RGB ».

**Charger deux images :**

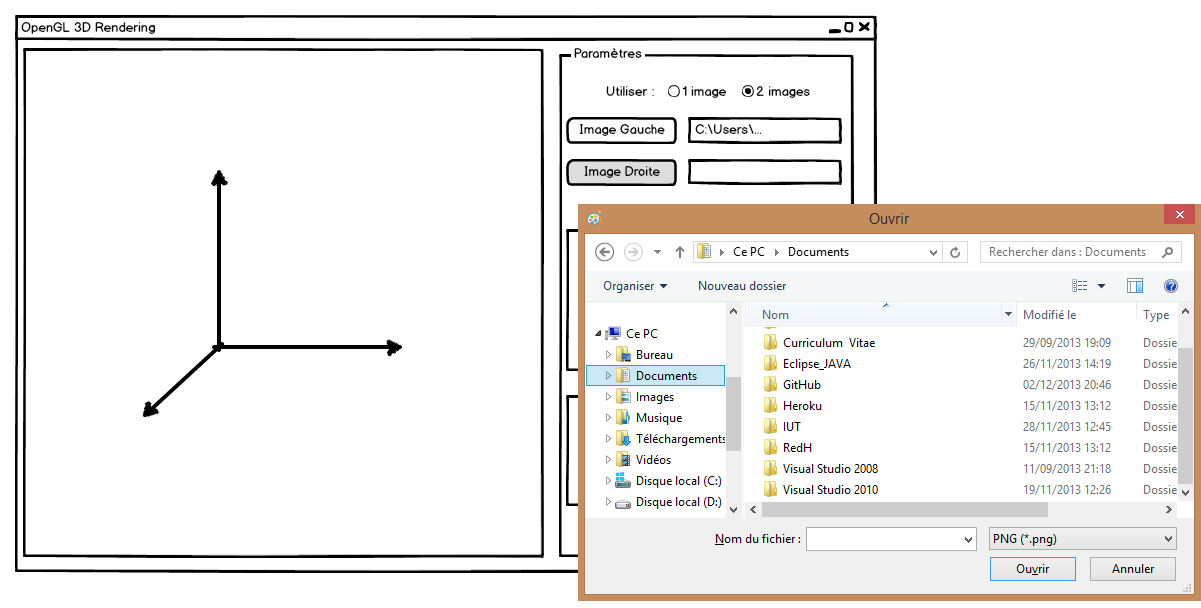


Pour utiliser deux images sources (avec un écart de 6 cm pour un rendu optimum) cliquez sur le bouton radio « 2 images ».

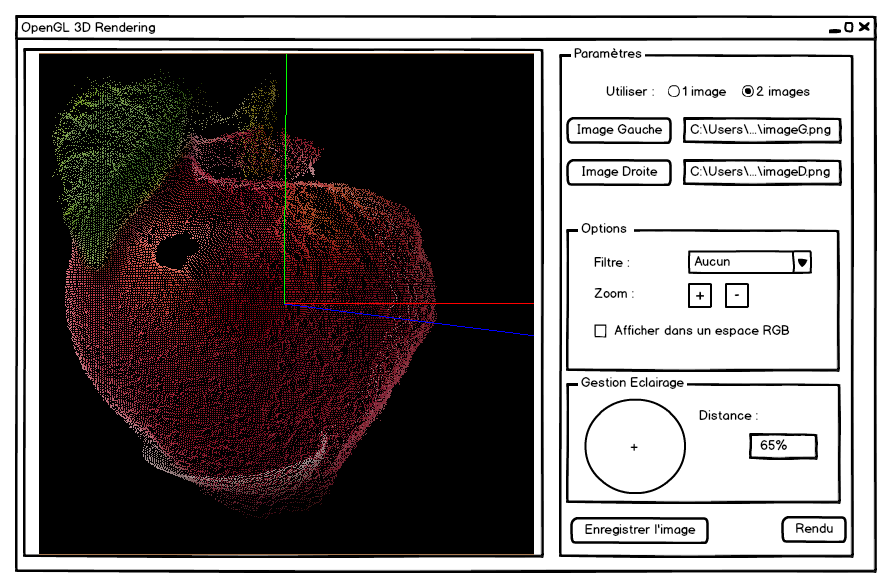
Un deuxième bouton devra apparaître « Image Droite », c’est à partir de ce bouton que l’on pourra aller chercher la deuxième image.

N’oubliez pas d’appuyer sur le bouton « Rendu » pour relancer le rendu à partir de ces deux images.

**Aller chercher une image :**



Lors de l’appui sur le bouton « Image », « Image Gauche » ou « Image Droite » un sélecteur de fichier devra apparaitre. Il vous permettra d’aller chercher votre image source à partir de l’arborescence de votre ordinateur.



Pour lancer le rendu appuyez sur le bouton « Rendu ».

# Le Planning

**Lundi 09 décembre 2013 :** Soutenance d'avancement du projet.

**Vendredi 13 décembre 2013 :** Validation définitive du Cahier des Charges.

**Mi-décembre 2013 :** Remise du dossier de gestion de projet.

**Début janvier 2014 :** Début de la phase de développement.

**Fin mars 2014 :** Soutenance finale du projet.

* Ce planning est approximatif et donc susceptible d’être modifié.

# Glossaire

**Shader :** est un programme informatique utilisé pour paramétrer et personnaliser une partie du processus de rendu réalisé par la carte graphique.

**GLSL :** (OpenGL Shading Language) langage de programmation utilisé pour la définition des shaders avec l’API OpenGL.

**GPU :** (Graphics Processing Unit) est un circuit intégré présent sur une carte graphique et assurant les fonctions de calcul de l'affichage.