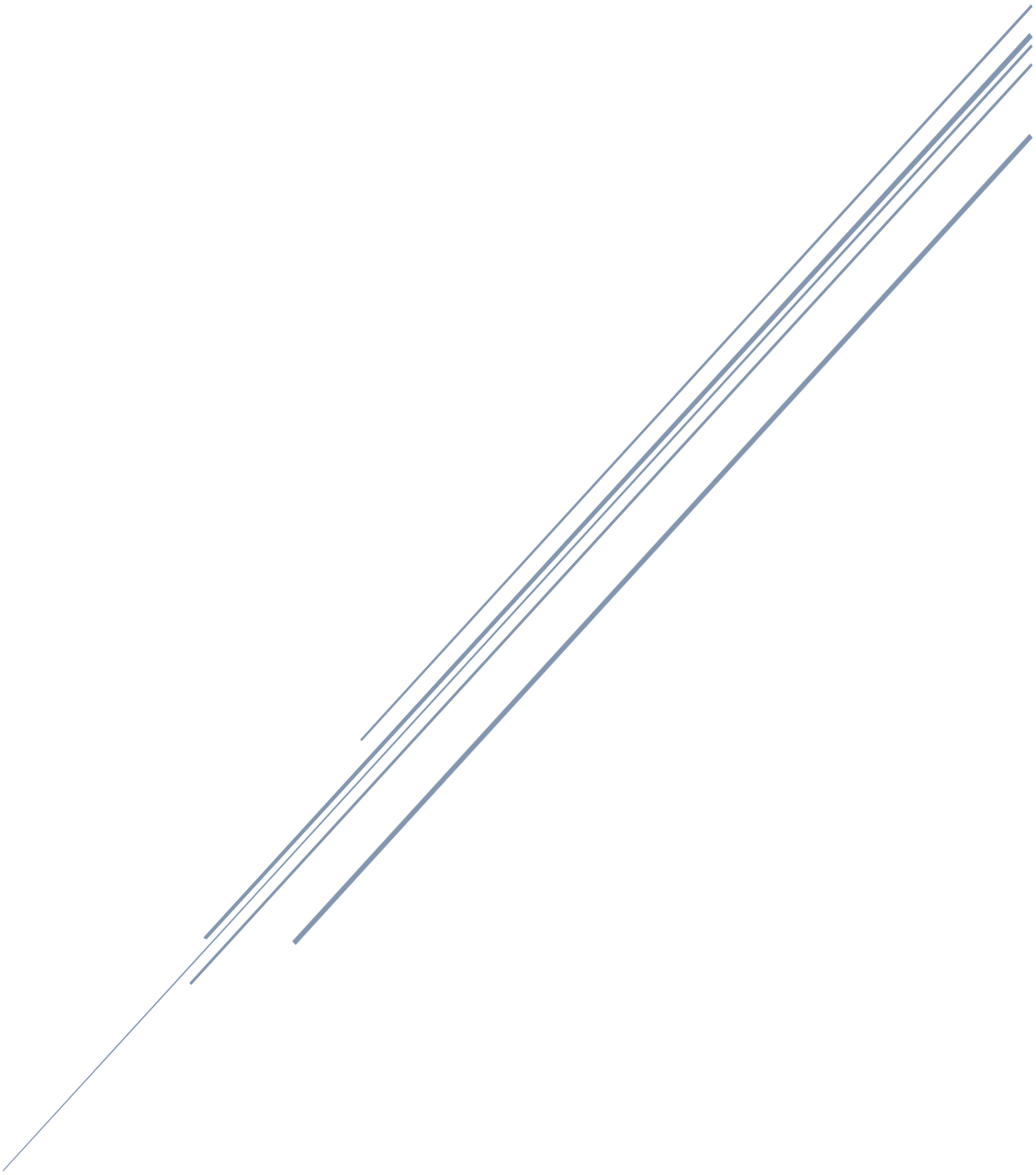
PROJET TUTEURE N°4

Réalité Augmentée en C++

Tuteur : M. Pierre-Frederic VILLARD



DUT Informatique – Année 2015/2016

Auteur : Alexis GEROME

I. Table des Matières

II. Introduction

A. La réalité augmentée

La réalité augmentée est un principe de plus en plus répandu. En effet, avec l'apparition de lunettes « intelligentes » (comme par exemple les Google Glass) permettant de pouvoir afficher des informations supplémentaires en plus de réalité. Je cite les Google Glass, mais ce n'est pas un très bon exemple. En effet, ces dernières ne sont capables que de faire des choses simples (comme afficher l'itinéraire GPS).

En revanche, un très bon exemple faisant déjà des choses incroyables et qui en fera certainement d'autres dans un futur assez proche, est un masque signé Microsoft. Je veux bien entendu parler du HoloLens. Cet outil bien fabuleux peut déjà effectuer énormément en termes de réalité augmentée. Une démonstration avait été faite au public lors des présentations du 6 Octobre 2015.

Pour faire simple, la réalité augmentée est simplement un ajout d'images en deux ou trois dimensions avec lesquelles on peut interagir. L'image ci-dessous illustre exactement ce qu'est cette technologie.

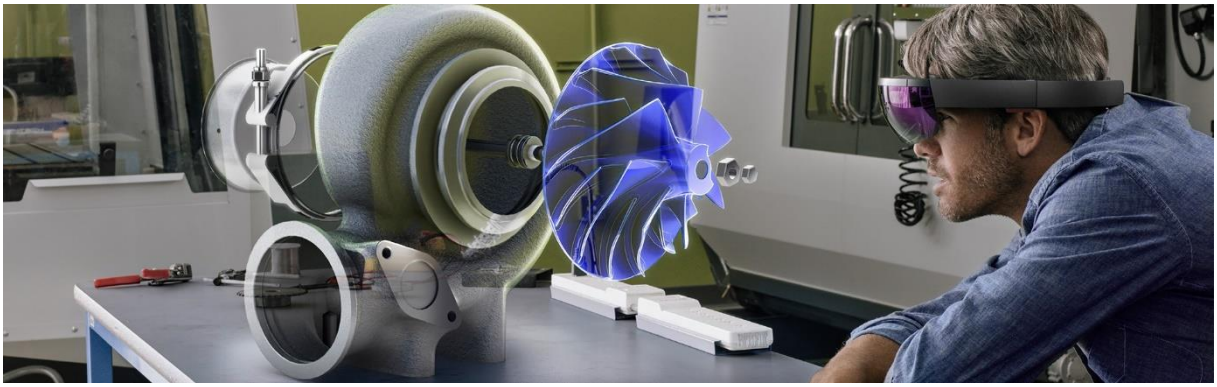


Figure 1 - Réalité augmentée avec le HoloLens

B. La bibliothèque PoLAR

PoLAR est une bibliothèque pour le langage de programmation C++. Il faut la compiler avec d'autres bibliothèques sur lesquelles elle s'appuie (OpenSceneGraph et Qt). Sa signification est **Po**rtable **L**ibrary for **A**ugmented **R**eality soit bibliothèque portable pour la réalité augmentée en français.

Comme son nom l'indique, cette bibliothèque est faite pour pouvoir faire de la réalité augmentée. De plus, elle est portable, c'est-à-dire qu'elle est compatible sur tous les ordinateurs, sous tous les systèmes d'exploitation.

Elle a été créée pour faciliter le développement d'application faisant appel à de la réalité augmentée, du visionnage d'image ou encore de l'imagerie médicale. C'est une bibliothèque très complète est relativement simple à utiliser.

C. L'objectif du Projet

Ce projet a pour but premier de faire découvrir la réalité augmentée et le langage C++. Le résultat sera une application permettant d'afficher une photo au choix (contenant un rectangle bien visible), mais dans laquelle il faudra inclure un ou plusieurs objets en trois dimensions. Pour que l'implémentation soit visuellement correcte et le plus réaliste possible, il faut procéder par étapes bien distinctes suivies par le tuteur.

III. Installation de PoLAR

J'ai fait le choix de travailler sous le système d'exploitation Windows 10. Pour coder, j'utiliserai Microsoft Visual Studio 2015. L'installation de la bibliothèque passe par plusieurs étapes : l'installation des programmes et bibliothèques dépendantes et ensuite la compilation de PoLAR.

A. Installation de Qt

La bibliothèque Qt permet de créer des fenêtres. Elle est portable et surtout, est compatible avec pas mal de langage informatique (Python ou encore Java par exemple).

L'installation de la bibliothèque Qt fut la plus simple parmi les autres. En effet, il a suffi de télécharger puis d'exécuter un fichier. Ce dernier a installé tout ce qui est nécessaire pour pouvoir faire ce que l'on souhaite. La partie un peu plus délicate, seulement si l'on ne connait pas du tout, fut de bien ajouter manuellement le chemin menant vers le dossier Qt dans les variables d'environnement de Windows.

B. Installation et compilation de OpenSceneGraph

La bibliothèque OpenSceneGraph est très puissante. Elle permet de faire des rendus en trois dimensions. Elle n'est compatible qu'avec le C++, et, s'appuie sur la technologie OpenGL.

L'installation de cette bibliothèque fut plus complexe. Il n'y a pas de fichier exécutable ou bien juste une commande à taper dans une invite de commande. En effet, il faut faire appel à un petit logiciel appelé CMake et au compilateur de Visual Studio (VS). CMake permet de préparer la compilation. En effet, à partir des fichiers sources de la bibliothèque et des dépendances que l'on lui donne, CMake crée un projet VS. En ouvrant et en lançant la compilation grâce à ce dernier, on génère ainsi des fichiers dont PoLAR aura besoin pour fonctionner. Il a fallu aussi installer un module complémentaire à OpenSceneGraph permettant de pouvoir lire une image au format JPG.

C. Compilation de PoLAR

Lorsque Qt et OpenSceneGraph ont été compilé avec succès, on peut passer à la compilation de la bibliothèque PoLAR en elle-même. Pour ce faire, il faut utiliser CMake pour spécifier le chemin d'accès d'OpenSceneGraph et de Qt. Si les étapes précédentes ont été un succès, on peut alors lancer la compilation avec VS ce qui donne au final les fichiers exécutables des exemples inclus avec PoLAR.

IV. La matrice de Projection

Pour la suite du projet, la photo ci-contre servira de fond pour y implémenter un ou plusieurs objets. Elle a été prise dans le salon. Le calcul de la matrice de projection s'effectuera grâce au carrelage du sol. Cette matrice servira à intégrer les objets virtuels proprement, comme s'il était vrai.

A. Mesure des points sur l'image

Un programme présent dans PoLAR s'appelant calib.exe permet de sortir les coordonnées des points créés avec la souris sur une image. En se servant de ce programme, il faut obtenir les coordonnées des quatre coins du rectangle présent comme spécifié auparavant.

B. Calcul de la distance focale

Ces points permettent de calculer la distance focale et ainsi définir le point de fuite de l'image. Grâce à cette distance focale, on peut calculer la matrice intrinsèque permettant de calculer la matrice de projection plus tard.

C. Calcul de la matrice d'homographie

Quelques opérations sur des matrices et celle-ci est calculée. Il suffit de suivre les formules indiquées dans le sujet de ce projet pour établir cette matrice.

D. Calcul de la matrice de projection

La matrice de projection souhaitée peut alors être calculée en faisant simplement le produit de la matrice intrinsèque avec la matrice d'homographie.

V. Utilisation de PoLAR

Pour utiliser PoLAR dans un projet, il faut suivre le manuel d'utilisation de la bibliothèque. Le codage en C++ n'est pas compliqué grâce à la documentation très détaillée.

VI. Problèmes rencontrés

Plusieurs problèmes sont survenus tout au long de ce projet.

A. Utilisation de Windows

Ce projet a été effectué sous le système d'exploitation Windows 10 du fait de l'incompatibilité de Linux avec la machine (processeur 6ème génération non pris en charge par le Kernel actuel). Avec Windows, il faut faire attention à bien ajouter toutes les variables d'environnement correctement. Aussi, il y a eu des problèmes pour la compilation du code en dehors du dossier d'installation de PoLAR. Il a fallu compiler le code du projet avec les exemples inclus dans PoLAR.

B. Ombrage non visible

Pour une raison qui n'est pas connue, l'ombrage ne fonctionne pas sur l'objet fantôme, malgré qu'il soir présent sur l'objet ajouté.