Haute Ecole de Gestion & Tourisme
Hochschule für Wirtschaft & Tourismus

Filière Informatique de gestion
Studiengang Wirtschaftsinformatik

Travail de Bachelor 2014

Interface gestuelle pour recherche d'images médicales sur internet

Etudiant-e : Oliver Schmidt

Professeur : Dr. Henning Müller

Déposé, le : 28 juillet 2014

I. Résumé

Le domaine médical stocke de grandes quantités d'images et l'accès à celles-ci est fastidieux demandant beaucoup de temps et de réflexion. C'est pour cette raison qu'il est intéressant d'avoir une interface pour afficher et faire des recherches. Ce projet est destiné à donner une nouvelle expérience d'utilisation en intégrant la recherche d'images dans un espace en trois dimensions.

En plus de la recherche, le programme offre la possibilité d'effectuer une recherche d'images médicales rapide sans toucher à quoi que ce soit, ceci grâce à l'utilisation d'un capteur gérant la détection des mains dans un espace délimité en-dessus de lui. Ce système permet par exemple à un chirurgien de faire une recherche pendant une opération sans devoir toucher d'objets impliquant de ne pas devoir désinfecter à nouveau. Le projet se base sur une interface existante qui se nomme Shambala et qui s'apparente à une page web. Objectivement, celle-ci peut déjà faire une partie du programme comme l'utilisation du capteur pour affiner la recherche.

Pendant le processus de réalisation du programme, une partie de la méthodologie agile a été appliquée. Des rendez-vous hebdomadaires avec le professeur responsable ou son remplaçant ont été organisés. Pendant ces entretiens, il ressortait ce qui a été développé, les problèmes rencontrés et ce qui va être fait ou amélioré. Ces séances mettaient en évidence la progression du travail et son orientation avec les éventuelles erreurs à résoudre.

Le processus normal d'exécution d'une recherche avec cette interface est :

- Ouvrir le menu grâce au pointeur. Celui-ci s'affiche lorsque l'utilisateur tend l'index au-dessus du Leap Motion. Lorsque le bouton menu est survolé pendant un laps de temps, la première rubrique du menu s'affiche. Ensuite l'utilisateur survole les différents boutons jusqu'à ce qu'il trouve la recherche préliminaire qu'il aimerait effectuer. Il est possible d'accéder au menu à tout moment.
- Après que la recherche préliminaire s'est effectuée, le résultat s'affiche sur l'écran principal de la recherche. L'utilisateur peut regarder dans toutes les directions et effectuer un zoom pour mieux choisir l'image qui l'intéresse. Ensuite il peut manipuler celle-ci en l'ajoutant positivement ou négativement à la recherche, ce qui va relancer une recherche et l'afficher. L'autre possibilité est de l'attribuer comme objet

de comparaison. En regardant en haut de la sphère, l'utilisateur voit les images qui ont été assignées.

- L'écran de comparaison est affiché grâce à un geste simple depuis l'écran de recherche. Il se compose seulement des deux images sélectionnées. Il est possible de les agrandir, rétrécir et bouger dans toutes les directions pour les comparer. Un geste est disponible pour revenir à la recherche.
- Le dernier écran sert à afficher les éléments ajoutés à la recherche. Il est accessible comme le précédent grâce à un geste simple depuis l'écran principal. En choisissant une image, il est possible de l'enlever de la recherche, ce qui en relance une autre et l'affiche. Une fonction utile si l'utilisateur a ajouté par inadvertance une image ou qu'elle ne correspond pas à la recherche désirée.

Mots-clés : manipulation d'image, leap motion, shambala, unity

I. Avant-propos

Ce travail de bachelor met en place une recherche d'images depuis un programme utilisant une interface en trois dimensions. Celle-ci se fait grâce à l'utilisation de requêtes sur un serveur distant. Ce projet vise à améliorer l'expérience de l'utilisateur en proposant une vision agréable d'une recherche d'images.

Ce sujet a été proposé par mon professeur responsable, Monsieur Dr. Henning Müller et son but est de remplacer la recherche déjà existante qui se nomme Shambala. Qui est une interface web servant à la recherche d'images pour aider les médecins à l'identification des problèmes chez leur patient.

Le projet consiste à reprendre le même concept mais dans un environnement en trois dimensions. Ce qui implique dans un premier temps d'effectuer les recherches nécessaires permettant l'élaboration d'un prototype reprenant les mêmes principes que sur l'autre interface. Le fait que l'application tourne sur un navigateur est un plus.

Le programme en question doit être réalisé en trois dimensions pour profiter pleinement de tout l'espace fourni permettant ainsi l'utilisation de mouvement dans toutes les directions c'est-à-dire, la profondeur, l'axe horizontal et l'axe vertical. Il doit rester simple, intuitif pour l'utilisateur et la comparaison des deux images fait partie des fonctions en plus.

La difficulté majeure de ce travail était de trouver le visuel idéal pour la recherche d'images dans un espace en trois dimensions. L'interface devait aussi rester intuitive, simple et facile d'utilisation. Durant le projet, l'interface a changé plusieurs fois.

La démarche adoptée était de commencer par rechercher des informations sur les outils imposés et conseillés dans le projet, ainsi que les langages utilisés. Puis de comparer les possibilités de chaque outil, suivi d'une liste des avantages et inconvénients de chacun. Ensuite, il a fallu créer quelques exemples visuels, surtout pour l'interface de la recherche. Dès le choix effectué, le développement de l'application a pu commencer. D'abord par l'intégration des différentes fonctions de la recherche d'images, puis des autres écrans. La méthode de travail suivante a été adoptée : rechercher ce qui existe déjà ou créer la méthode nécessaire, l'intégrer, lui attribuer un geste selon les besoins, faire diverses adaptations en testant ce qui a été fait. Pendant l'intégration des nouvelles fonctions dans le prototype, des améliorations ou des changements des fonctions ont été effectués, soit pour s'adapter à la nouvelle interface soit parce que l'ancienne ne fonctionnait pas correctement.

III. Remerciements

Je souhaite remercier mon professeur responsable, le Dr. Henning Müller, et le chercheur de la HES-SO, Monsieur Antoine Widmer, qui ont suivi et encouragé mon travail par leurs conseils et idées. Un grand merci aussi à Monsieur Roger Shaer qui a conçu le service procurant le résultat de la recherche, pour ses explications claires et ses idées intéressantes.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

GLOSSAIRE

HTTP : Hypertext Transfer Protocol, protocole de communication client-server utilisé sur le web

REST : Representational State Transfer, architecture permettant de lire, écrire, mettre à jour ou supprimer des informations sur un serveur distant grâce à des requêtes http

JSON : JavaScript Object Notation, format d'écriture, ressemble au xml

Swipe : geste préenregistrer du Leap Motion, c'est un mouvement rapide dans une direction

Mockup : exemple de visuel pour une interface

Autocomplétion : proposition pour compléter le mot, très utile en programmation

Plugin : une extension d'un programme

Prefab : modèle de GameObject

OpenGL ES : Open Graphics Library for Embedded System, une librairie graphique ouverte utilisée pour la conception d'application générant des images en trois dimensions

Asset : une librairie ou extension dans Unity

Tag : suffixe pouvant être attribué à un GameObject

Caption : mots clés attribués à l'image

1 Introduction

1.1 Problématique

L'utilisation d'images médicales, pour le diagnostic et le traitement d'une majeure partie des patients, est en constante augmentation. Par exemple, l'hôpital de l'université de Genève produisait en moyenne 300'000 images par jour en 2013. Des systèmes existent pour archiver les images de manière optimale. Néanmoins, un accès rapide à ces images peut être fastidieux pour le personnel médical.

1.2 Objectifs

Le but de ce projet est d'améliorer l'expérience de l'utilisateur en proposant une nouvelle interface pour la recherche d'images médicales. Celle-ci utilise des gestes simples pour la navigation et autres manipulations. Cette interface utilisera comme périphérique d'entrée principal le Leap Motion. C'est un capteur de mouvements capable de suivre jusqu'à 10 doigts dans un hémisphère de 40 cm de rayon. Celui-ci arrive aussi à décrypter différents gestes.

Les fonctionnalités recherchées dans ce projet sont :

- d'effectuer une requête sur le serveur de données pour avoir un résultat de recherche
- de faire une recherche sans l'utilisation d'objets physiques, très pratique pour les métiers où il faut faire attention aux bactéries
- de naviguer dans le résultat de la recherche simplement
- d'affiner la recherche en ajoutant des images positivement ou négativement
- de comparer deux images sélectionnées parmi le résultat de la recherche
- d'enlever des éléments ajoutés à la recherche grâce à un visuel de ceux-ci.

1.3 Étapes

1.3.1 Étapes déjà implémentées

1.3.1.1 ParaDISE

C'est un acronyme pour Parallel Distributed Image Search Engine. C'est un web service utilisant des requêtes Hypertext Transfer Protocol (HTTP)[1] sur des principes de Representational State Transfer (REST)[2]. Les paramètres à introduire dans la requête et les réponses sont écrits dans le format JavaScript Object Notation (JSON). Ce service est la base utilisée pour la recherche d'images dans le prototype et dans Shambala ; elle retourne les données nécessaires à l'affichage. Le lien de base pour l'appel du service est http://faster.hevs.ch/ParaDISEWSKhresmoi/resources/searchResource/searchImages, ensuite il faut ajouter les paramètres et leurs contenus à cette requête. Voici la liste des paramètres, avec ce qu'il faut mettre comme contenu, utilisés dans le projet :

- caption-query : le texte à rechercher dans les captions des images
- maximum-results : nombre total maximum de résultats à retourner

La documentation avec les autres paramètres qu'il est possible d'ajouter est annexée. Deux paramètres intéressants, qu'il serait possible d'intégrer ultérieurement, sont results-page, qui permet d'avoir une page de résultats spécifique, et results-per-page pour indiquer le nombre de résultat obtenus par page. Ceux-ci pourraient être intégrés pour séparer le résultat en plusieurs écrans qui s'afficheraient grâce à des gestes spécifiques effectués par l'utilisateur.

La requête retourne le nombre total de résultats et un tableau de résultats qui contiennent chacun le lien de l'image, le positionnement dans le résultat, le score qui varie selon la correspondance avec la requête, le code de modalité, les captions correspondant à l'image, le lien vers l'article et le lien vers l'image dans une taille réduite. Dans le projet, ce sont le lien de l'image et ses captions qui sont surtout utilisés.

1.3.1.2 Shambala

C'est l'interface web en deux dimensions qui permet de faire une recherche d'images sur laquelle le projet se base, voir la figure 1. La première recherche est faite avec des mots-clés.

[1] HTTP : Protocole de communication client-server utilisé sur le web
[2] REST : Architecture permettant de lire, écrire, mettre à jour ou supprimer des informations sur un serveur distant grâce à des requêtes HTTP

En entrant un ou plusieurs mots et en validant la recherche, une liste d'images apparait. Puis il est possible de prendre une image et de la déplacer, soit dans la colonne positive, soit dans la colonne négative, ce qui engendre l'ajout de l'image positivement ou négativement à la recherche. Ce geste peut être effectué sans contact grâce à un petit appareil qui remplace la souris par la détection des mains et des gestes. Il est possible de sélectionner une image avec un geste spécifique, puis de la manipuler. Le projet de bachelor se base sur cette interface, qui est particulièrement utile aux chirurgiens parce qu'elle évite l'utilisation de la souris et du clavier réduisant le nombre de désinfections. La seule condition est de faire une recherche préliminaire avec un mot-clé pour avoir des images concernant le type d'opération ou le type d'image (radio, réel ou autres).

Figure 1 : Shambala

1.3.2 Étapes à réaliser

1.3.2.1 Recherches

Avant de débuter le développement du prototype, il a fallu rechercher comment créer le visuel de l'interface en trois dimensions, avec quelle technologie incluant quel type de langage ? Les recherches préliminaires furent utiles pour avoir les bases et prendre des décisions.

1.3.2.2 Le visuel

Le projet est parti de zéro pour l'affichage, car l'interface précédente étant en deux dimensions ne correspondait pas aux besoins. Pour décider quel visuel utiliser, des mockups ont été créés. Ceux-ci ont été d'une grande aide pour le choix du visuel qui a changé plusieurs fois au cours du projet, pour finir comme il est actuellement.

1.3.2.3 Les fonctions

Le fonctionnement de l'application se base sur Shambala, mais toutes les fonctions ont dû être réécrites ou adaptées. Soit elles n'intégraient pas l'interface en trois dimensions, soit elles n'allaient pas avec le fonctionnement du projet. Cette partie a demandé beaucoup de recherche car beaucoup de fonctions se basaient sur les données du Leap Motion. La documentation officielle a beaucoup contribué au projet.

1.3.2.4 Choix des gestes

Une utilisation optimale demande des gestes simples pour déclencher les fonctions. Tout au long du projet les gestes ont été changés. Au départ, il y en avait beaucoup ceci compliquant ainsi leur utilisation. Mais lors du processus de développement, les gestes utilisés se précisaient et se réduisaient. Certains furent enlevés car ils n'étaient pas bien détectés.

2 Méthodes

2.1 Outils & principales librairies

Pour développer le prototype, il a été conseillé de se diriger sur le logiciel Unity ou l'interface WebGL. C'est pour cela que dans les outils, il y a une explication pour chacun d'eux. Mais également pour d'autres raisons qui seront évoquées dans leur description.

2.1.1 WebGL

Figure 2 : Logo WebGL[3]

C'est une interface qui permet la visualisation d'objet tridimensionnel sur le Web en utilisant le standard Open Graphics Library for Embedded System (OpenGL ES)[4]. Elle tire profit de l'accélération matérielle des cartes graphiques, ce qui engendre des rendus performants de formes et de textures complexes. Pour ce qui est du développement, tout doit être écrit à la main que ce soit les fonctions, l'affichage graphique ou la forme des objets, mais il existe des librairies facilitant la tâche[5]. Il utilise le langage JavaScript.

Au niveau de la compatibilité, il est supporté par ces navigateurs :

- Firefox v4 +
- Chrome v9 +
- Safari si activé par le menu développement
- Internet Explorer v11+

Avec cette technologie, il est obligatoire de gérer soi-même les graphismes, ainsi que les objets, et leur rendu. Pour ce qui est du lancement de l'application dans un navigateur, contrairement à Unity, il n'y a pas besoin d'installer des extensions pour l'exécuter. Lors du choix du développement, WebGL constituait un plus long processus parce qu'il fallait consacrer plus de ressources à la partie graphique. Ceci impliquant plus d'attente avant

[3] http://www.khronos.org/legal/trademarks/#webgl (15.07.2014)
[4] OpenGL ES : une librairie graphique ouverte utilisée pour la conception d'application générant des images en trois dimensions
[5] http://www.alsacreations.com/tuto/lire/1572-webgl-3d-three-canvas-threejs.html (03.07.2014)

d'avoir un prototype mais aussi pour avoir des mis-à-jours. Finalement, à ce moment-là du projet, il n'était pas possible de savoir combien de temps prendrait la création manuelle des formes étant donné que le visuel de l'interface n'était pas encore défini.

2.1.2 Unity

Figure 3: Unity logo[6]

Unity est un logiciel permettant la création d'application en deux et trois dimensions, principalement utilisé pour la création de jeux-vidéos. Ce logiciel est gratuit mais il existe aussi une version pro avec des fonctionnalités supplémentaires telles que la possibilité de lire une vidéo, depuis le net ou en interne de l'application, des améliorations graphiques et de lumière, ainsi que la vente de l'application sans la limitation à 100'000 dollars de chiffre d'affaire. Pour ce projet l'utilisation de la version gratuite est suffisante. Unity permet le déploiement d'applications sur internet grâce à son Web Player. Celui-ci nécessite l'installation d'une extension sur le navigateur utilisé. Il est également à relever qu'il est possible de faire une version mobile de l'application.

En plus d'Unity, un autre logiciel s'installe avec le fichier d'installation, il se nomme MonoDevelop et permet d'écrire les différents scripts qui vont être utilisés dans le programme. Celui-ci est agréable à l'utilisation, permet d'écrire et de tester les scripts. Il inclut même un affichage des dossiers et scripts du projet. L'autocomplétion est aussi disponible pour le développement dans certains langages de programmation.

L'avantage principal d'Unity sur WebGL est sa simplicité de développement surtout pour la partie graphique. Unity propose des objets de bases comme des sphères, des cubes, des plateformes, des cylindres et autres objets. De plus, il est possible d'en acheter beaucoup d'autres dans sa boutique en ligne. Cet avantage permet d'avoir un prototype visuel rapidement, puis de le faire évoluer soit en y intégrant des fonctionnalités soit en modifiant

[6] http://forum.unity3d.com/threads/legality-on-using-unitys-logo.74057/ (15.07.2014)

son affichage à volonté. Cette manière de faire permet aussi de faire un développement agile avec un prototype qui est amélioré au fil du temps.

L'autre avantage est que cette technologie met de nombreux paquets et extensions à disposition qui couvrent les textures, les animations, les modèles, voire des exemples de projets et des tutoriels. Ceux-ci sont disponibles gratuitement ou par achat à la boutique officielle ou sur internet.

L'inconvénient majeur qui est ressorti pendant la phase de développement est le web. Unity Web Player ne peut pas accéder directement aux différents composants de l'ordinateur comme le micro ou le Leap Motion. Pour obtenir l'accès direct, il faut appeler un script externe qui accède à un de ces éléments.

En plus de cela, les dernières nouvelles du site d'Unity annoncent que la prochaine version contiendra une fonctionnalité pour déployer un projet directement en WebGL. Ceci pourrait enlever les contraintes imposées par le Web Player.

C'est en connaissance de ces différents avantages que mon choix s'est porté sur Unity. Ce logiciel correspondait aux besoins du projet, c'est-à-dire un développement permettant d'avoir un prototype rapidement et de faire des améliorations en fonction des résultats des tests. Mais aussi la possibilité de montrer un résultat fonctionnel dans un laps de temps plus réduit et de pouvoir tester une des versions à tout moment.

Il existe de nombreuses alternatives à Unity, mais celui-ci me semblait plus complet, même pour la version gratuite, et plus facile d'utilisation grâce aux nombreux tutoriaux disponibles sur internet et aux nombreux compléments disponibles sur la boutique et en-dehors.

2.1.2.1 Fonctionnement d'Unity

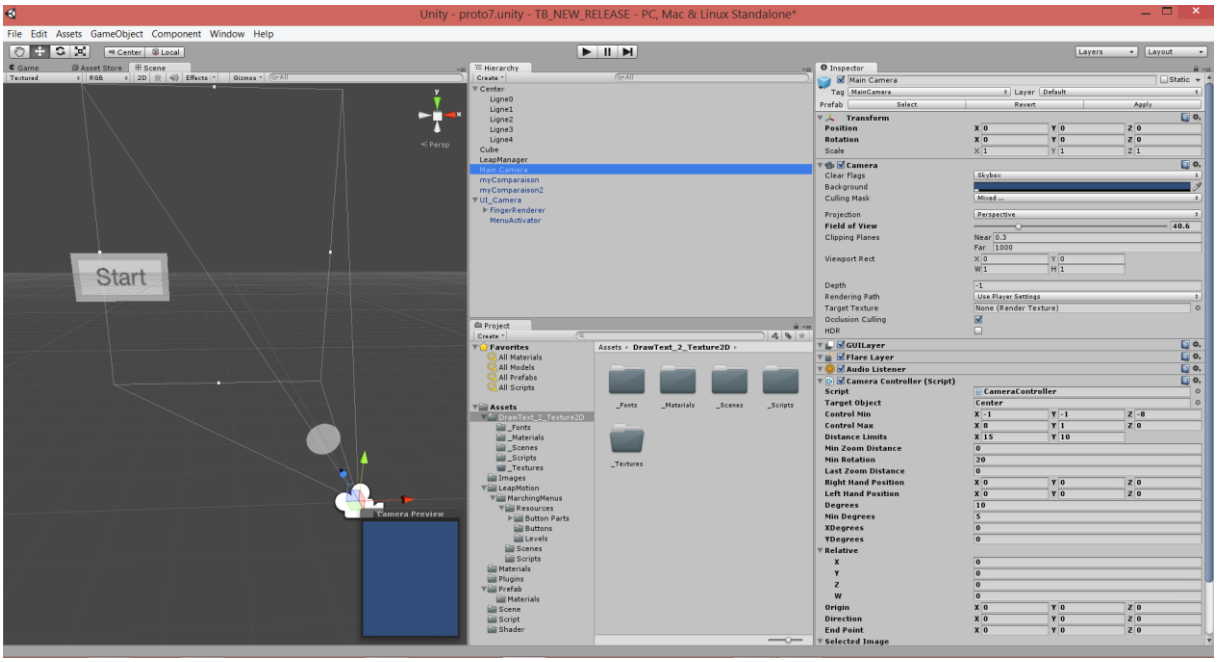


Figure 4 : Interface Unity

Pour mieux comprendre la suite du travail, une brève explication des différents éléments d'Unity est nécessaire. L'interface où sont affichés les objets se nomme une scène, c'est la partie gauche de la figure 3. C'est un espace en trois dimensions qui permet de voir les éléments graphiques qui lui sont attribués mais aussi de les manipuler par la taille, la rotation ou le déplacement dans l'espace. Ces éléments sont appelés GameObject et peuvent prendre diverses formes comme la caméra, des objets graphiques, la lumière et bien d'autres. Une application développée avec Unity peut comporter plusieurs scènes. La hiérarchie permet d'avoir la liste des objets attribués à une scène, cela permet aussi de gérer chaque élément enfant d'un objet ou de configurer les prefab, c'est-à-dire, des modèles de GameObject, que l'on peut introduire sur toutes les scènes du projet ou de les importer dans d'autres. Tout GameObject peut devenir un prefab, même avec des enfants. Grâce à l'inspecteur, il est possible de voir et configurer les composants attribués à un GameObject, tel que les scripts, les rendus, les collisions, la position, la rotation et bien d'autres. L'onglet projet permet de voir notre projet et de créer divers éléments tels que des dossiers, des scripts ou des prefab.

2.1.3 JSON

JavaScript Object Notation est un format qui ressemble à un tableau multidimensionnel écrit en texte. Il permet d'avoir de l'information structurée comme le langage eXtensible

Markup Language (XML)[7]. Dans ce projet, le script *SimpleJSON* s'occupe d'interpréter les textes en format JSON et de les transcrire en tableau.

Exemple de JSON avec une requête sur ParaDISE :


```
{"totalResults":100,
"results":[
{"imageURL":"http://fast.hevs.ch/images/other/imageclef2012/1757-1626-0002-
0000007968-
002.jpg","rank":1,"score":0.008841528104774777,"runname":null,"descriptor":
null,"modality":"DRCT","caption":"CT scan of carcinosarcoma of the left
lung.","articleURL":"http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2740266","
thumbnailURL":"http://fast.hevs.ch/images/other/imageclef2012_thumbs/1757-
1626-0002-0000007968-002.jpg"},
{"imageURL":"http://fast.hevs.ch/images/other/imageclef2012/1476-0711-3-16-
2.jpg","rank":2,"score":0.008695491895116005,"runname":null,"descriptor":nu
ll,"modality":"DRCT","caption":"CT showing cavitary lesions in both
lungs","articleURL":"http://www.ann-
clinmicrob.com/content/3/1/16","thumbnailURL":"http://fast.hevs.ch/images/o
ther/imageclef2012_thumbs/1476-0711-3-16-2.jpg"},…
```


2.1.4 Shambala3DUnity

Shambala3DUnity est un projet basic, fournit par Monsieur Antoine Widmer, qui envoie une requête avec comme critère « caption-query=lung+ct ». Puis le programme crée un mur d'images s'affichant successivement, voir figure 5. Il est possible d'interagir par la sélection d'une image grâce au Leap Motion et une fois l'image désélectionnée, elle tombe. Cette interface a été instructive pour le début du projet. Car grâce à elle, il est possible de voir ce que sont les images récupérées et d'interagir avec. Mais aussi de savoir comment fonctionne la requête avec le service ParaDISE.

[7] http://www.commentcamarche.net/contents/1332-introduction-a-xml.html (15.07.2014)

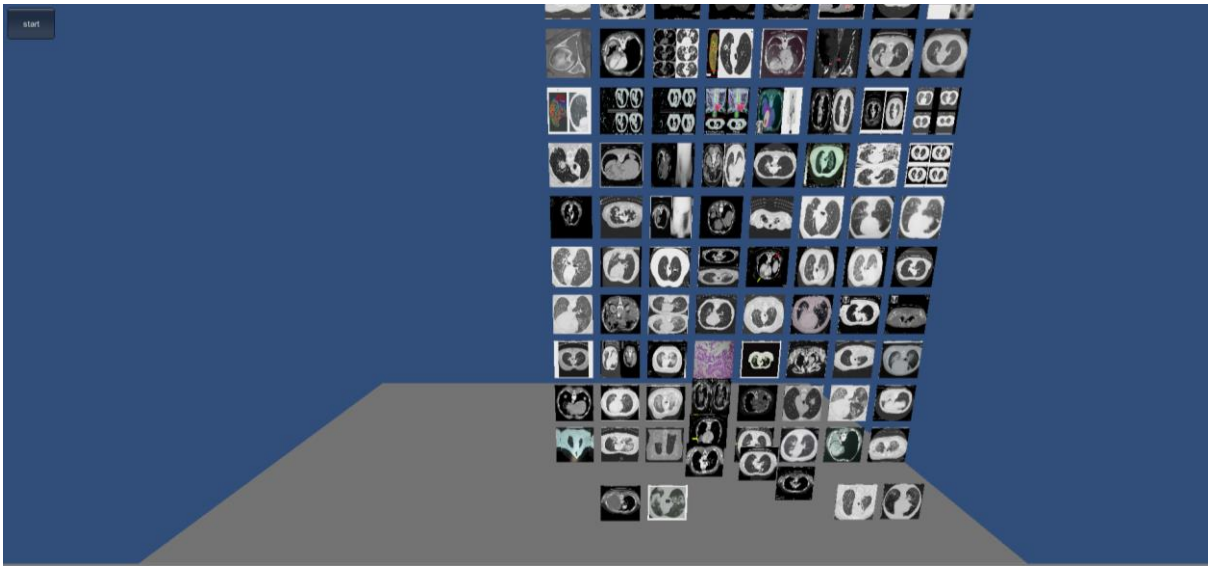


Figure 5 : Vue de base Shambala3DUnity

2.1.5 Leap Motion Boilerplate Asset[8]

Cet asset est proposé par l'équipe du Leap Motion. Il constitue les bases pour intégrer leur appareil dans un projet. Il comprend quatre exemples d'utilisation de celui-ci, dont un qui a été utilisé pour débuter le projet. Cet exemple met en scène une sphère représentant la terre, voir figure 6, et intègre un geste pour pivoter autour du globe et effectuer un zoom. C'est le script *RevolvingCamera* qui gère les gestes utilisés pour la manipulation. Il a été réutilisé et modifié dans le projet. C'est un élément clé pour le fonctionnement du prototype.

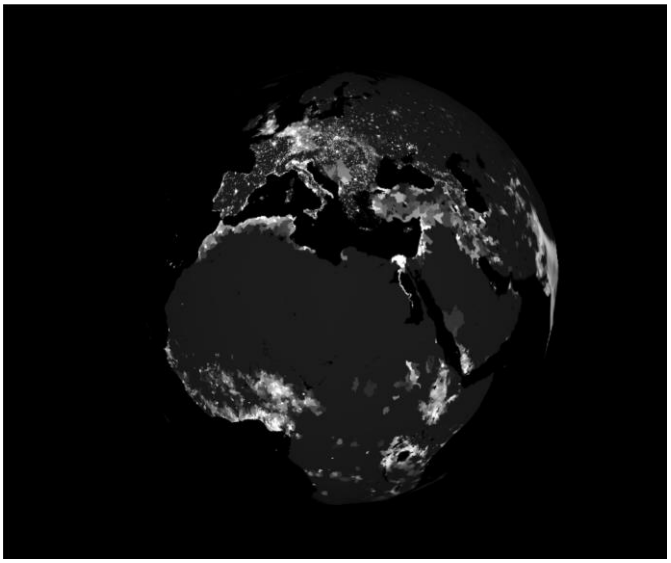


Figure 6 : Scène avec le globe

[8] https://developer.leapmotion.com/documentation/csharp/devguide/Unity_Boilerplate.html (15.07.2014)

2.1.6 DrawTexture_2_Texture2D[9]

Cette librairie permet d'écrire du texte sur une texture grâce au script *Texture2D_extended*. Elle est utilisée dans le projet pour afficher les informations de chaque image grâce à la classe *TextDrawing*. De base, la version gratuite d'Unity ne permet pas d'écrire du texte sur une texture. C'est pour cette raison qu'il fallait chercher un autre moyen de le faire. Une alternative trouvée pour cette librairie est *TextToTexture*[10].

2.1.7 Leap Motion Marching Menus Asset

Ceci est le dernier asset utilisé dans le projet. C'est sur celui-ci que le menu du prototype final se base. La scène fournie avec l'asset permet de se rendre compte de son fonctionnement. Premièrement, un bouton avec « menu » écrit dessus apparait. Puis en utilisant une main, il suffit de survoler celui-ci pour que d'autres boutons s'affichent, ce sont les éléments de base d'un menu. Chaque bouton a deux possibilités distinctes, soit il fait apparaitre un sous-menu, soit il effectue une action. Pour la première exécution, il suffit de survoler l'élément avec le pointeur et pour la deuxième, il faut faire un mouvement vers la droite de l'objet. Dans la scène proposée, la main sert de pointeur.

[9] http://forum.unity3d.com/threads/drawtext-on-texture2d.220217/#post-1474196 (15.07.2014)
[10] http://blog.almostlogical.com/2010/08/20/adding-text-to-texture-at-runtime-in-unity3d-without-using-render-texture/ (15.07.2014)

2.2 Technologies

2.2.1 Leap Motion[11]

Figure 7 : Photo Leap Motion

Le Leap Motion est un petit appareil que l'on branche à l'ordinateur comme une souris. Il fait 8 cm de long et 2,9 cm de large pour 1,1 cm d'épaisseur, au-dessus une comparaison de la taille du doigt avec celle du Leap Motion. Il crée une interface en trois dimensions juste en-dessus du capteur. Celle-ci permet de détecter très précisément (au millimètre) les deux mains et les dix doigts en utilisant des infrarouges. Il détecte aussi différents gestes préprogrammés, qu'il faut activer préalablement dans le code. Il faut penser de nettoyer régulièrement la surface pour éviter de mauvaises détections.

Les 4 gestes[12] détectés sont :

- le *KeyTapGesture*, c'est le même mouvement que lorsqu'on tape sur une touche mais dans les airs, voir figure 8. Après des tests d'utilisation, celui-ci s'avère difficile à exécuter, surtout pour un débutant.

[11] Leap Motion Controller, Great Hardware in Search of Great Software July 24, 2013 NYT, http://www.nytimes.com/2013/07/25/technology/personaltech/no-keyboard-and-now-no-touch-screen-either.html?_r=0 (24.07.2014)

[12] https://developer.leapmotion.com/documentation/csharp/devguide/Leap_Gestures.html#id1 (15.07.2014)

Figure 8 : KeyTap

- le *CircleGesture*, c'est une rotation dans le sens de la montre et vice versa, voir figure 9. Il est très pratique d'utilisation et facile, voire trop facile à détecter. Pour éviter que ce geste soit détecté trop facilement, car une rotation incomplète suffit pour qu'il s'exécute, une variable est disponible pour connaitre la distance de rotation effectuée. Ce geste a été intégré dans le projet pour redémarrer le prototype. Il existe aussi des méthodes pour connaître la direction de la rotation, mais cette méthode n'est pas incluse dans le projet.

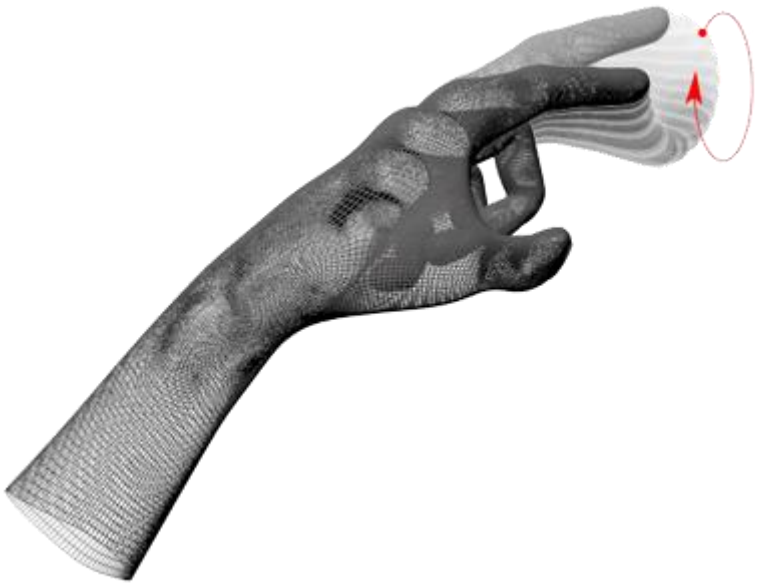


Figure 9 : Circle

- le *ScreenTapGesture*, c'est le déplacement en avant d'un doigt comme si l'on voulait toucher l'écran, voir la figure 10. Shambala l'utilise pour faire la sélection d'une

image. Après quelques essais, il est ressorti qu'il n'est pas facile à exécuter dans un espace en trois dimensions. Il ne sera donc pas utilisé dans le projet.

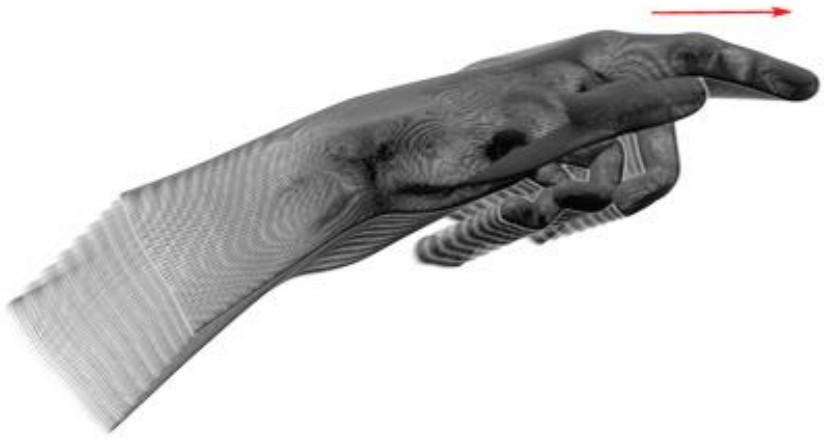


Figure 10 : ScreenTap

- le *SwipeGesture* : c'est un mouvement rapide dans l'axe horizontal ou vertical, voir figure 11, impossible à faire la main fermée, mais exécutable avec au moins un doigt tendu. Ce geste est utilisé pour plusieurs méthodes dans le projet.

Figure 11 : Swipe

La librairie du Leap Motion est constituée de plusieurs classes reliées les uns aux autres, voir la figure 12, la classe principale est la Frame, c'est le point de départ pour connaître le reste, c'est-à-dire les gestes, les mains, les doigts, les pointeurs, etc.

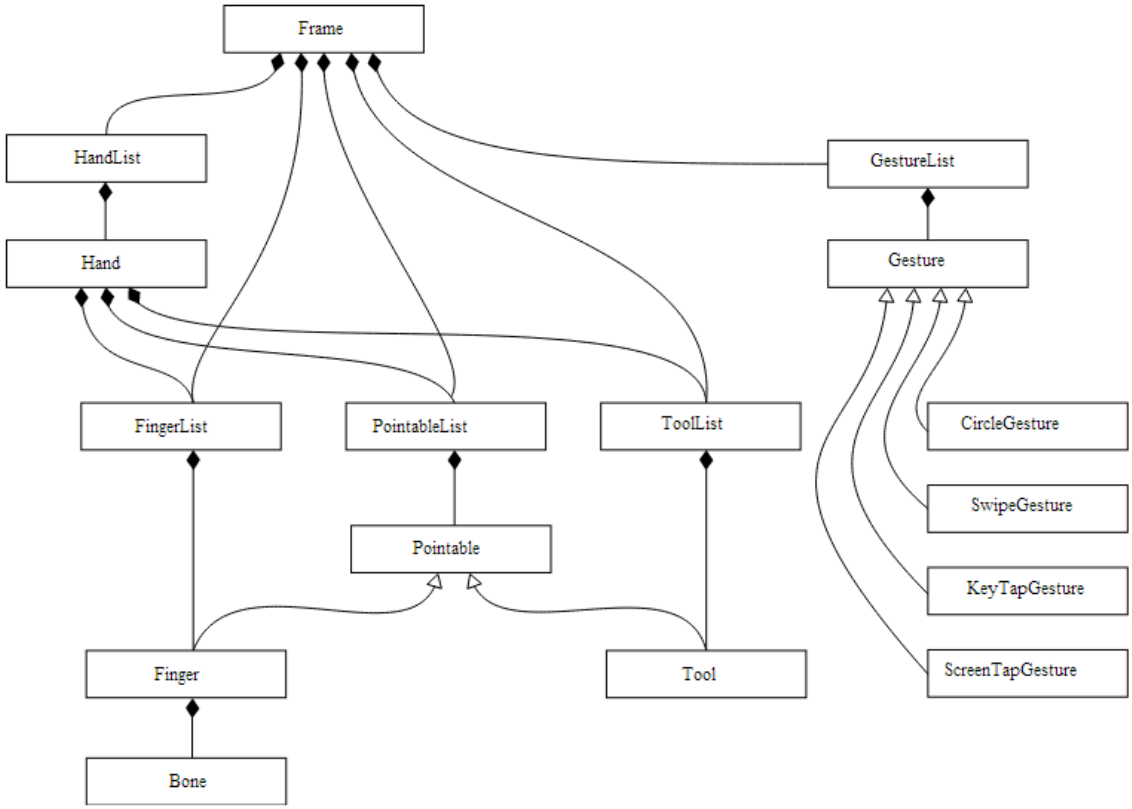


Figure 12 : Classes de la librairie du Leap Motion[13]

Les différents problèmes rencontrés avec cette technologie durant ce projet sont :

- la détection de la main inversée, c'est-à-dire que l'appareil peut confondre la main gauche et la main droite au début
- la rotation qui se détecte trop facilement
- la main n'est pas très bien détectée au début
- la position qui n'est pas stable parfois.

Le Leap Motion est une technologie imposée dans le projet, les alternatives potentielles, qui ont pu être trouvées, sont la kinect[14] version deux qui permet une détection d'une

[13] https://developer.leapmotion.com/documentation/skeletal/csharp/devguide/Leap_Tracking.html (16.07.2014)

[14] http://www.tomshardware.fr/articles/kinect-2-pc,1-47244.html (02.07.2014)

personne se situant à au moins 40 cm du boitier, le MYO[15] qui détecte des mouvements grâce aux muscles du bras et nimble UX[16] qui a le même fonctionnement que le Leap Motion en utilisant une caméra de profondeur.

2.2.2 Nouveau kit de développement pour le Leap Motion

Pendant le processus de création du prototype, une nouvelle version améliorée du kit de développement pour le Leap Motion est sortie. Le projet a donc été mis à jour à ce moment-là.

Différences avec l'ancien kit

Le nouveau kit incorpore de nouvelles fonctionnalités telle que :

- Main droite et gauche sont définies
- Un indicateur de force de fermeture de la main
- Les 5 doigts sont toujours présents
- Chaque doigt à un type (pouce, index, …)
- Un indicateur de force pour le pincement entre le pouce et l'index
- Définition pour chaque os des mains et la position des jointures des os
- La détection des mains est améliorée

Cette mise à jour permet une meilleur détection des mouvements ce qui est un grand plus.

En intégrant celle-ci au projet, quelques problèmes sont survenus. Le principal étant que les cinq doigts d'une main sont toujours présents, alors qu'avant, lorsque l'utilisateur en pliait un, il était considéré comme absent de la main. De plus, ce souci n'a pas été identifié tout de suite car le projet comportait d'autres erreurs à ce moment-là. Du coup, la façon de compter les doigts a dû être changée. À la place de vérifier combien sont présents, c'est le nombre de doigts tendus qui est compté. C'est une méthode simple à écrire, parce qu'avec la nouvelle librairie il est possible de reconnaître si un doigt est tendu ou non, grâce à une variable assignée à celui-ci.

C'est une bonne amélioration, introduisant des nouveautés qui ont pu être utilisées dans ce projet ; comme le fait de savoir si la main est fermée ou non, chaque main a ses fonctionnalités, etc.

[15] https://www.thalmic.com/en/myo/ (02.07.2014)
[16] http://www.pmdtec.com/nimbleux/ (02.07.2014)

Ce kit comprend un projet de démonstration qui a été testé. Mais il ne comportait rien de spécifique utile pour ce projet. Par contre cela a permis d'avoir un aperçu de ce que cette nouveauté pouvait faire.

2.2.3 Langage de programmation C#

Prononcé si charp, est un langage de programmation orienté objet créé par Microsoft. Il a été conçu pour leur plateforme de programmation, Microsoft .NET. Il est semblable aux autres langages et est orienté objet.

Unity supporte 3 langages, le JavaScript, le C# ou le Boo et le Leap Motion reconnait que le C# et le JavaScript. À la base, il a été décidé que le projet utiliserait le langage JavaScript parce qu'il était possible de reprendre, ce qui sera écrit, dans d'autres programmes ou sites internet. Mais sur le site d'Unity, le leap est utilisé avec du C#, la documentation est donc aussi dans ce sens. De plus, la plupart des programmes développés sur Unity, surtout pour le leap, qui ont été recherchés durant le projet, sont écrits en C#. Et puis, l'autocomplétion des différentes méthodes de la librairie du Leap Motion se fait en C# mais pas en JavaScript sur MonoDevelop. Donc, pour avoir un prototype rapidement, il a été décidé d'utiliser le langage C# dans ce projet. Par la suite il sera possible de le transcrire en JavaScript.

Voici brièvement les raisons de l'utilisation du C# dans le projet :

1. Facilité

Comme beaucoup de choses existe déjà en C#, il est plus facile de trouver du code déjà existant correspondant plus ou moins à ce qui sera introduit dans ce projet. Ce qui est un avantage, car par exemple, il est possible d'améliorer le prototype en intégrant certaines méthodes trouvées sur le net qui pourraient se comporter de la manière recherchée, puis d'appliquer des modifications selon les besoins.

2. Librairie

Pour coder en JavaScript, il est possible d'intégrer une librairie comme dans C# mais l'autocomplétion des méthodes et des variables de la librairie ne se font pas. Cela implique qu'il n'est pas possible de savoir directement si c'est juste ou faux ce qui a été fait, tant que l'application n'est pas lancée ou par la vérification des méthodes de la librairie sur le site du Leap Motion.

3. Test

Pour tester et savoir si ce qui est écrit est juste, avec le JavaScript, il faut lancer à chaque fois le programme alors qu'avec l'autre langage il est possible d'avoir une correction directe.

4. Possibilité ultérieure

Comme la librairie C# et JavaScript du leap sont presque les mêmes, le passage d'un langage à l'autre en est facilité et pourrait être envisagé si le projet a besoin d'être écrit dans l'autre langage.

5. Nouveauté

Avec la nouvelle mise à jour de Unity, il est possible de transformer un projet dans le langage de programmation C# ou UnityScript (JavaScript) en un projet WebGL sans problème avec une nouvelle fonction intégrée dans celui-ci.

2.3 Planning

Il a été convenu qu'une séance hebdomadaire devait se tenir soit avec Monsieur Henning Müller, soit avec Monsieur Antoine Widmer. Pendant ces rendez-vous les résultats obtenus et les problèmes survenus depuis la dernière séance sont présentés et les objectifs de la semaine suivante sont décidés. Ce planning permettait d'avoir un projet qui suit la bonne ligne directrice, d'avoir des avis sur ce qui a été accompli et des conseils sur ce qui doit être fait.

Chaque semaine le prototype a évolué, ce qui correspondait à une sorte de mise à jour. C'est partiellement une méthodologie agile, avec une liste de fonctions dans le cahier de charges et des séances régulières.

Pour que les personnes suivant le projet sachent où il en était, tout ce qui est écrit, rapports, un journal de ce qu'y a été fait et autres documents étaient sauvegardés dans un dossier commun sur internet.

3 Résultat

3.1 Recherches effectuées

3.1.1 Reconnaissance vocale

La reconnaissance vocale fait partie des solutions pour éviter de devoir entrer des mots-clés pour lancer la recherche ou pour en ajouter dans la recherche actuelle. Pour cette partie, une recherche préliminaire sur la boutique d'Unity a été effectuée, mais il n'existe rien, à part un asset pour la reconnaissance vocale sur le système d'exploitation Android[17]. Autrement, il y a deux façons de faire. La première est de créer soi-même un service et de définir sur le server une liste de mots qui vont être reconnus par le service de reconnaissance vocale[18] et la seconde est d'utiliser un service qui a déjà des mots-clés inclus comme celui de Google[19].

L'avantage de la première est qu'elle permettrait de prendre dynamiquement uniquement les captions contenus par la recherche actuelle et donc d'éviter de rechercher des mots non inclus dans cette recherche mais aussi des erreurs de compréhension. Cela implique aussi de ne pas dépendre d'un autre service. La deuxième solution est juste l'intégration du service dans l'application et donc demande moins de travail. Mais l'inconvénient est qu'elle ne détecte pas forcément les bons mots et pourrait ne pas intégrer le mot souhaité.

3.1.2 Leap Motion et Unity Web Player

Dans cette recherche, il a été constaté qu'Unity Web Player ne peut pas utiliser le hardware de manière directe alors qu'une application Unity indépendante le peut. Donc en cherchant comment contourner ce problème il est ressorti que la librairie JavaScript peut être appelée par Unity grâce à une fonction *ExternalCall*. Elle permet d'appeler une fonction d'un fichier externe de la même manière que JavaScript peut appeler une fonction sur le programme qui utilise Unity avec la fonction *SendMessage*. Ces messages ne peuvent inclure que des floats, strings, integer ou boolean. Ceci implique que les fonctions recevant les données du JavaScript pour interpréter des gestes doivent être toutes écrites. De plus il s'est avéré que le Web Player est limité en images par seconde donc il ne lit pas tous les mouvements du Leap Motion, ce qui peut engendrer des imprécisions.

[17] https://www.assetstore.unity3d.com/en/#!/content/13882 (17.07.2014)
[18] http://forum.unity3d.com/threads/windows-udp-voice-recognition-server.172758/ (17.07.2014)
[19] https://www.google.com/intl/fr/chrome/demos/speech.html (17.07.2014)

"The JavaScript Leap Motion player can receive hundreds of frames a second, while the Unity player can realistically render 30 frames a second."[20]

3.1.3 Unity à Three.js

Avant de savoir que le logiciel Unity allait intégrer un convertisseur vers WebGL, une autre solution, que d'avoir un script qui fait intermédiaire entre le Leap Motion et Unity Web Player, est ressortie durant les recherches. Cette solution propose d'exporter les modèles en trois dimensions d'Unity pour pouvoir les utiliser dans WebGL[21].

3.1.4 Les mockups

Les mockups sont des visuels servant d'exemple pour l'interface qui va être développée. La création de ceux-ci a été effectuée sur Unity directement. C'était une étape importante pour avoir une première idée pour se rendre compte comment afficher les images dans un espace en trois dimensions. Les premiers étaient juste des murs d'images, plus centrés sur l'aspect deux dimensions, mais les derniers étaient des sphères, cylindres et cubes. Celui qui avait été choisi comme point de départ de l'interface, est la disposition d'images à l'intérieur d'une sphère. Ce choix judicieux permettait de profiter pleinement de l'espace en trois dimensions. Après quelques recherches, le positionnement prédéfini des images lors de leur apparition semblait le mieux adapté.

3.2 Menu

Pour avoir le premier résultat servant de base pour affiner la recherche, il a été décidé de faire un menu. Celui-ci était nécessaire pour éviter d'entrer soi-même les mots-clés pour la première recherche comme c'était le cas dans Shambala.

3.2.1 Visuel

Le menu se base sur l'asset *marching menu* créé par l'équipe du Leap Motion et se trouvant sur le store d'Unity. Celui-ci a un affichage de gauche vers la droite. Le premier élément à être visible est un bouton où il est écrit « Menu » dessus. Lorsque le pointeur y reste un instant dessus, la première rubrique du menu est affichée. Lorsqu'un élément est survolé,

[20] https://www.leapmotion.com/blog/using-the-leap-motion-hardware-with-unityweb-player/ (17.07.2014)
[21] http://helloenjoy.com/2013/from-unity-to-three-js/ (17.07.2014)

soit la suite est affichée, soit un début de sélection se fait comme il est possible de le voir sur la figure 13. La sélection est un affichage dans les tons verts.

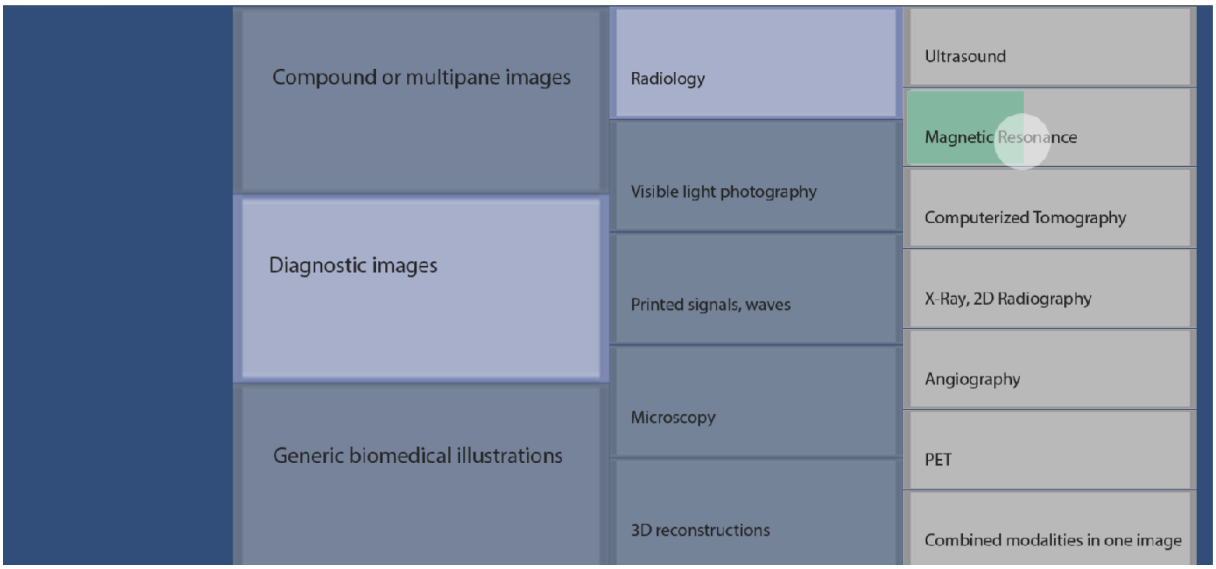


Figure 13 : Affichage du menu

3.2.1.1 Texte du premier menu

Le texte du menu et de ses rubriques se compose des diverses modalités faisant partie de la recherche. Sur la figure 14 ci-dessous, un arbre de toutes les modalités est affiché.

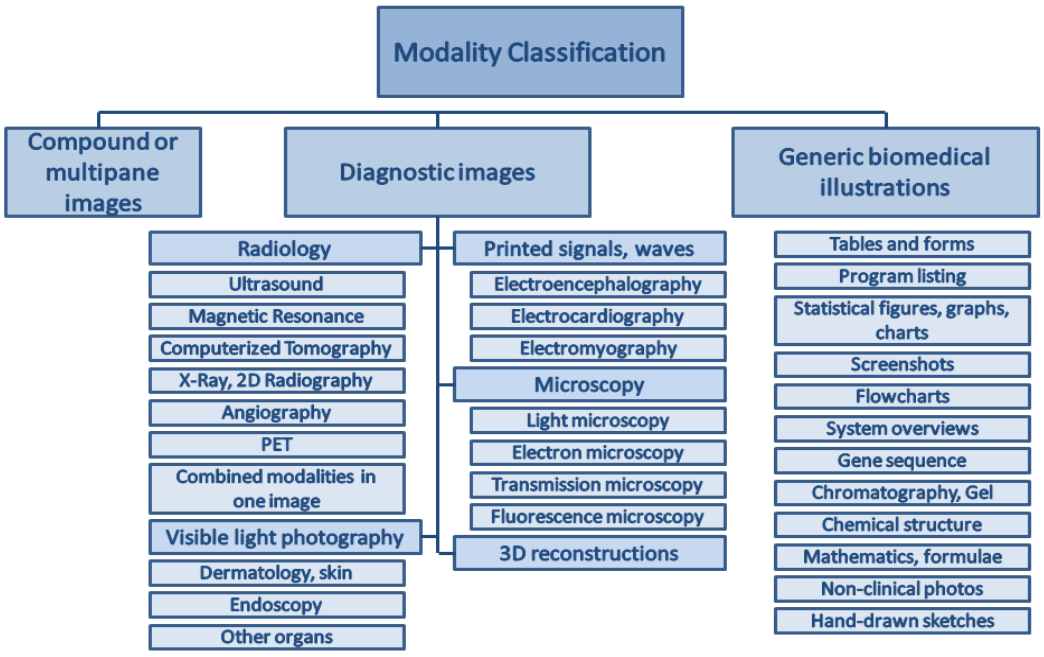


Figure 14 : Classement des modalités[22]

[22] http://imageclef.org/2013/medical (21.07.2014)

3.2.2 Etape pour créer le menu

Avec Photoshop, il faut modifier les différents éléments graphiques, comme le texte affiché, le visuel du bouton, la transparence et le visuel de la sélection. En ce qui concerne la taille, la hauteur se calcule selon le nombre d'éléments à afficher dans une rubrique et la largeur, par rapport à la longueur du texte.

Dans le logiciel Unity, il faut transformer les textures en sprite, élément graphique, en cliquant sur la liste déroulante dans l'inspecteur, sous la rubrique type de texture. Puis créer chaque rubrique en tant que prefab, en utilisant comme exemple un déjà existant. Une rubrique se compose d'un élément qui sera parent de tous les autres objets contenant le script *LevelBehavior* et ne sera pas affiché graphiquement. Puis chaque enfant de cet objet sera un des boutons affichés. Celui-ci contient le script *ButtonBehavior* auquel il faudra affecter les éléments graphiques créés précédemment avec Photoshop. Si le bouton doit afficher une autre rubrique il faut la lui attribuer grâce à la variable NextLinkage. Mais s'il doit faire une action, alors il faut lui attribuer la variable SelectionAction et Caption, pour le texte à rechercher. Il ne faut pas oublier de calculer et d'indiquer à quelle hauteur se trouve le bouton, grâce à la position y. En enfant de ce bouton, il y a trois éléments auxquels il faut affecter les sprites indiqués. La figure 15 représente une rubrique avec ses divers éléments et l'affichage du script ButtonBehavior dans l'inspecteur.

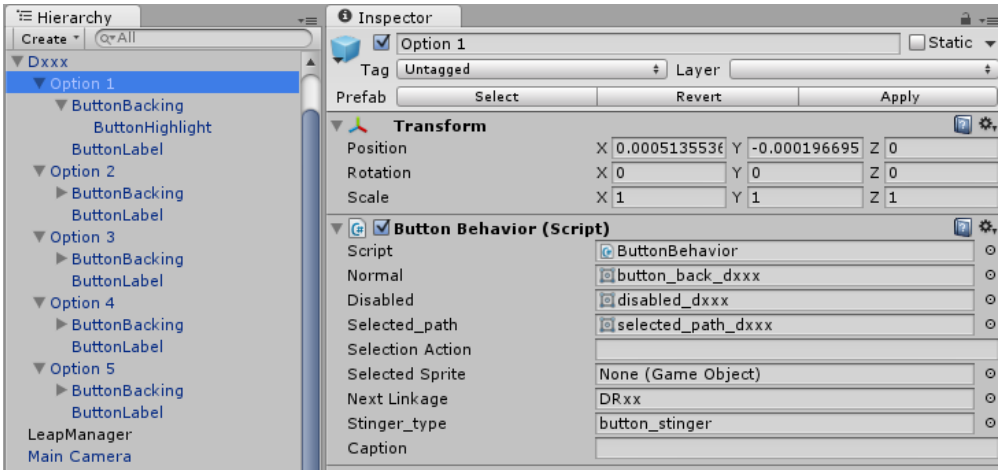


Figure 15 : Hiérarchie d'une rubrique du menu et dans l'inspector le script ButtonBehavior

3.3 Ecrans

L'application fonctionne grâce à des modes. Ceux-ci permettent d'afficher trois écrans, celui de la recherche, de la comparaison et des éléments ajoutés à la recherche. Grâce au

principe de mode, il n'y a pas besoin de changer la position de la caméra, mais juste faire un affichage des éléments nécessaires. Un écran est l'affichage des éléments nécessaires pour les différents modes, c'est aussi ce que l'utilisateur voit.

3.3.1 La recherche

3.3.1.1 Le principe

Le principal but de ce programme est de pouvoir faire une recherche rapide grâce à un visuel simple et des gestes faciles, en utilisant le Leap Motion. Dans ce principe, l'écran sert à afficher le résultat de la recherche. La manipulation du résultat sert soit à faire une nouvelle recherche en ajoutant des éléments à la recherche ou en sélectionnant une image pour effectuer la comparaison de deux images. Dans ce but, l'interface doit être simple à manipuler et pratique à regarder.

3.3.1.2 La vue

La vue finale de la recherche, voir figure 16, c'est l'utilisateur qui se tient à l'intérieur d'une sphère d'images. Elle permet d'éviter l'excès de mouvement pour atteindre les images.

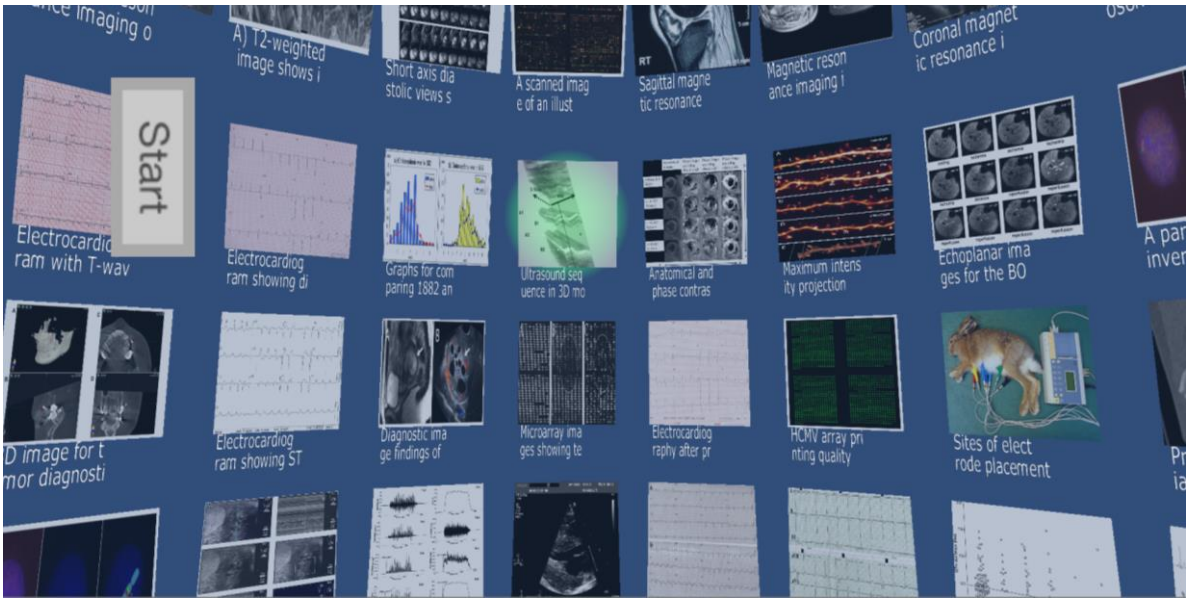


Figure 16 : Vue principale

Les premières vues étaient depuis l'extérieur et cela demandait des mouvements tels que la rotation de la caméra et de la vue, les déplacements sur tous les axes, … Alors que celle-ci demande juste la rotation de la caméra sur elle-même et un zoom. Elle permet aussi de voir

les deux objets sélectionnés pour la comparaison en regardant en haut de la sphère, voir figure 17.

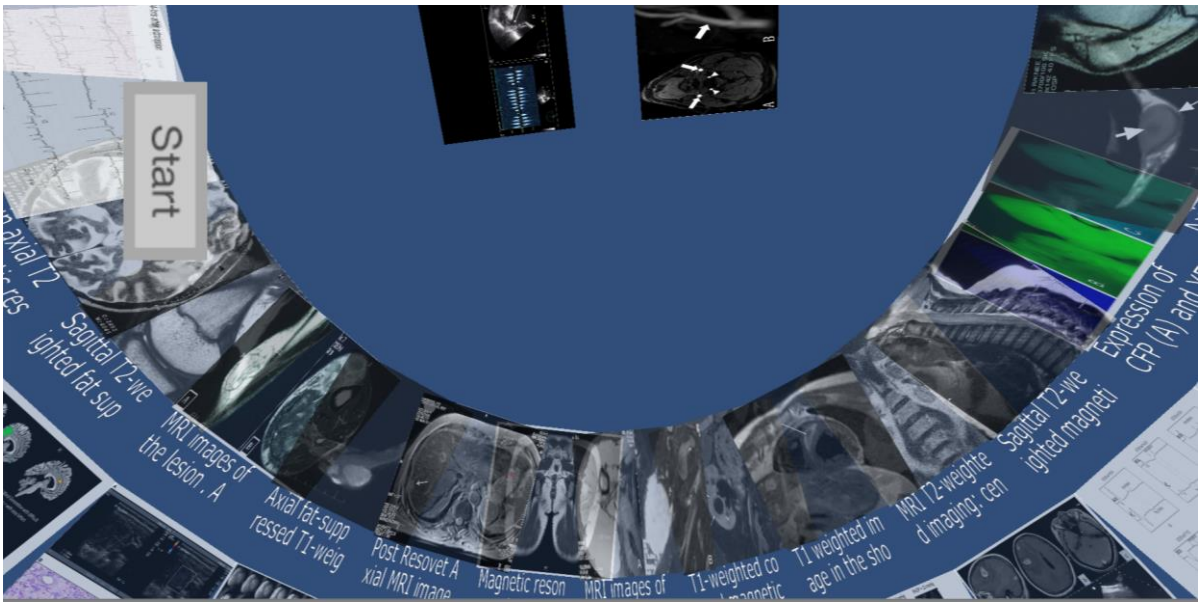


Figure 17 : Vue principale avec objets de comparaison

Mais avant d'arriver à ce résultat, le projet est passé par plusieurs affichages. La figure 18 reprend l'interface du mockup. Celui-ci consistait à prédéfinir les positions des images à l'intérieur d'une sphère. Il n'a pas été gardé à cause des problèmes mentionnés ci-dessus. Mais aussi parce que le nombre de positions est limité.

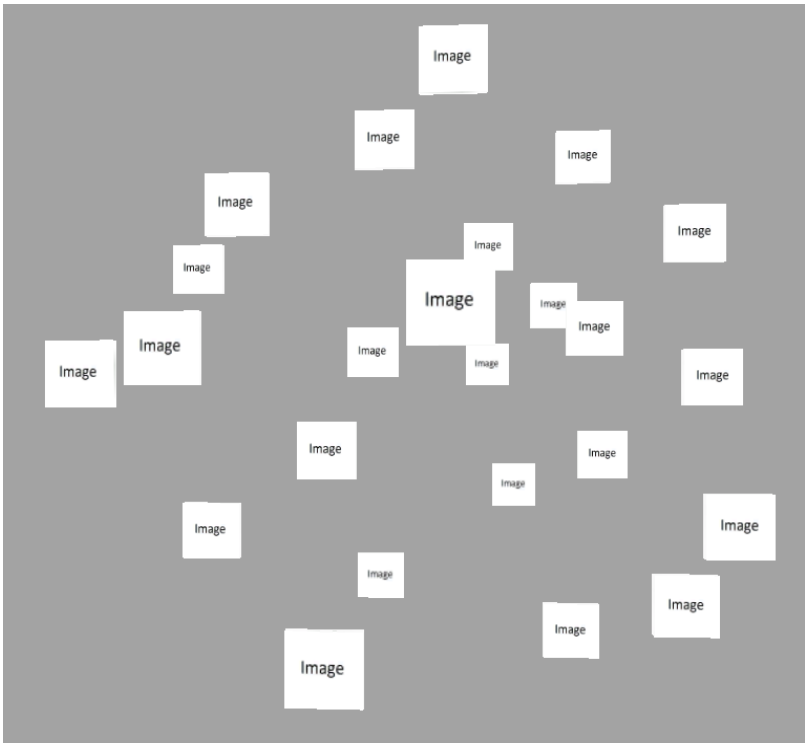


Figure 18 : Vue extérieure avec positions prédéfinies

Un autre affichage essayé pendant le développement est celui à l'intérieur d'un cylindre, voir figure 19. Il a été testé juste après celui de la sphère. Pour le créer, une méthode positionnant des objets dans le périmètre d'un cercle[23] sur plusieurs niveaux a été utilisé. Ce principe a été repris dans l'affichage final.

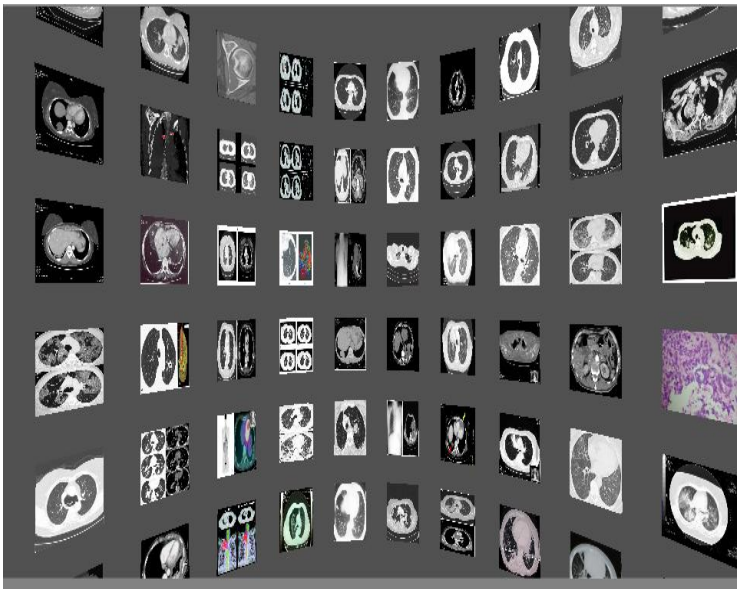


Figure 19 : Vue à l'intérieur du cylindre

3.3.1.3 Positionnement des éléments

La première méthode positionnant les objets sur une sphère[24] n'était pas totalement au point car les objets se trouvant à son sommet ou à sa base posaient problèmes. C'est pourquoi la méthode a été changée par celle utilisée dans le cylindre. Celle-ci consiste d'abord à placer un objet sur un périmètre d'un cercle en calculant l'angle où il va être placé. Puis une boucle s'occupe de le faire plusieurs fois à des hauteurs différentes et avec des rayons différents. Cette méthode pourrait être modifiée pour gérer dynamiquement la taille de la sphère par rapport au nombre d'objets à afficher. Mais dans le projet, il a été préféré de la laisser à un nombre fixe d'objets, ceci permet de réduire les ressources utilisées par l'application étant donné que la sphère est créée qu'une fois lors de son lancement.

3.3.1.4 Les images

Les images de la recherche sont récupérées depuis le web, puis affichées en tant que texture sur un élément de la sphère. Lors de l'attribution, la collision et l'affichage de l'objet

[23] http://forum.unity3d.com/threads/how-to-instantiate-objects-in-a-circle.10693/ (17.07.2014)
[24] http://entitycrisis.blogspot.ch/2011/02/uniform-points-on-sphere.html (17.07.2014)

sont activés, c'est-à-dire que l'élément devient visible et peut être sélectionné pour la comparaison ou pour l'ajout à la recherche. Pour différencier les objets activés de ceux qui ne le sont pas, un tag « Touchable » est ajouté à ceux-ci. Lorsqu'une recherche est relancée, soit par le menu soit par l'ajout d'une image à la recherche, les éléments sont remis par défaut avant l'attribution du résultat.

3.3.1.5 Le texte des images

Le texte des images correspond pour l'instant à un bout de la caption, les mots-clés qui identifient l'image. La fonction idoine a été récupérée sur internet. Elle utilise la librairie *texture2D* et l'extension correspondante. Elle se situe sur un script qui est ajouté sur l'objet sur lequel le texte va être affiché. Lorsque celui-ci est initialisé, le paramétrage pour le texte de la texture se fait. Puis en appelant la méthode *setText* du script, il est possible d'afficher un texte sur cet objet. Le fond de l'objet qui affiche le texte est transparent, cela permet une meilleure visibilité de l'image.

3.3.1.6 Objets regardant la caméra

Une des choses qui devait être ajoutée c'est les objets suivant la caméra, surtout lors de la phase où la vue était depuis l'extérieur, étant donné que c'est la caméra qui tournait autour de la sphère ; cela donnait une meilleure vue des images. Avec la vue de l'intérieur de la sphère, cette fonction est utile lors de la création des images et pour la comparaison des objets.

Dans le programme Unity, il existe une fonction qui commande à l'objet de regarder un autre objet *gameObject.transform.LookAt*. Originalement c'est celle qui a été utilisée dans le projet, d'abord en trouvant les objets qui doivent regarder la caméra, puis en utilisant le système de recherche par nom de tag *GameOjbect.FindGameOjbectsWithTag*. Cette méthode était effectuée à la fin de chaque *Update* de la caméra. Mais comme le prototype a changé et que la caméra ne bouge plus, il n'y a plus besoin de le faire à chaque fois, mais seulement lorsque les objets sont créés et placés autour de la sphère. Pour finir, l'utilisation de *LookAt*, était trop restreinte, il n'était pas possible de bouger uniquement l'axe des x, y ou z ou deux à la fois. C'est pourquoi une autre méthode a été écrite utilisant la fonction *Quaternion.LookRotation*. Avec celle-ci il était possible de changer les valeurs de la rotation avant de l'effectuer.

3.3.1.7 Choix des manipulations

Pour ce qui est des différentes manipulations qu'il est possible d'effectuer sur cet écran, elles ont été séparées en deux. La main droite gère tout ce qui concerne la caméra, c'est-à-dire la rotation, le zoom et le changement d'écran. La main gauche s'occupe des manipulations des objets qui ont un halo autour d'eux. Dans le projet, il est possible de sélectionner cet objet choisi pour la comparaison ou de l'ajouter à la recherche. Cette séparation a été instaurée pour éviter d'embrouiller l'utilisateur. De cette manière, il est plus facile de savoir quelle main s'occupe de quoi. Toutes les méthodes de manipulations sont dans le script *CameraController*. Ces gestes ont été choisis après l'essai de plusieurs autres.

3.3.1.7.1 La rotation

Celle-ci sert à voir tous les objets de la sphère avec les objets de la recherche. Pour cette manipulation, une méthode s'occupe de calculer la rotation de la caméra sur elle-même par rapport à la position de la main, dans notre cas la main droite. Pour activer cette fonction, il faut que quatre doigts soient tendus. Comme conseil pour pratiquer la rotation, c'est de plier le pouce ainsi il est plus facile de l'arrêter, simplement en le dépliant. La méthode qui s'occupe de cela est *ProcessLook.*

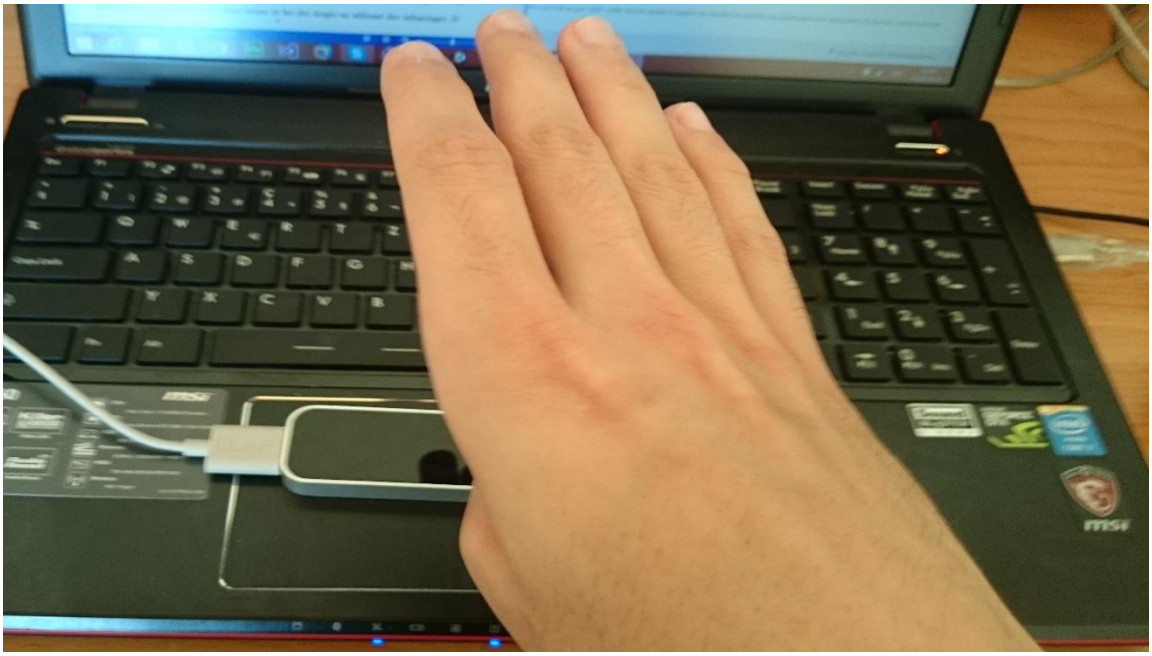


Figure 20 : Utiliser Leap Motion avec quatre doigts

3.3.1.7.2 Autres essaies pour la rotation

Durant le projet d'autres gestes ont été utilisés. Voici un tableau récapitulatif de ceux-là avec leurs points positifs et négatifs.

Geste	**Positif**	**Négatif**
Les deux mains	Bien pour une rotation depuis l'extérieur de la sphère	Utilisation des deux mains Problème pour les autres gestes
La main ouverte (5 doigts tendus)	Intuitif	Problème pour bien cibler l'image souhaitée et la stabilité n'est pas idéale
Un doigt tendu	L'impression de cibler l'objet	La rotation est calculée par rapport à la main et pas aux doigts donc c'est moins adapté
Poing fermé	L'impression d'attraper notre sphère pour la faire tourner	Fatigant pour l'utilisateur à la longue et moins facile pour cibler l'image
Utilisation du *swipe*	Ne fait pas de mouvement en trop	Pas assez précis pour l'utilisation de la direction de la caméra comme pointeur

Tableau 1 : Autres gestes essayés pour la rotation

Pour ce qui est des rotations, au début du projet, la vue été depuis l'extérieur de la sphère, de cette manière la rotation a été essayée de deux manières, la première avec la caméra qui tourne autour de la sphère et la deuxième la sphère qui tourne sur elle-même. Puis la vue a changé pour une à l'intérieur de la sphère, ce qui implique une rotation de la caméra sur elle-même. Parce qu'il est plus difficile de gérer la rotation de la sphère que celle de la caméra.

3.3.1.7.3 Le zoom

Il sert à agrandir ou réduire la taille du champ de vision permettant ainsi de voir les objets plus grands mais en plus petit nombre ou vice-versa. La méthode effectuant cette manipulation qui agrandit ou réduit le champ de vision de la caméra dépend de la valeur mise

en paramètre. La méthode se nomme *moveCameraToZoom*. Le geste s'occupant de zoomer ou dézoomer est la main droite fermée tournant à gauche ou à droite.

Figure 21 : Main droite fermé sur le Leap Motion

3.3.1.7.4 Autres essaies pour le zoom

Au début du projet, les deux mains géraient la rotation et le zoom en même temps. Le zoom s'effectuait en changeant la variable du champ de vision de la caméra. Celui-ci n'était pas agréable à utiliser parce qu'il s'exécutait tout le temps et donc dérangeait l'utilisateur. Il a été changé par deux gestes utilisant toujours la distance entre les deux mains pour calculer le zoom, l'un en utilisant les deux mains et l'autre avec une seule. La différence avec l'ancien calcul est que la caméra avançait ou reculait pour effectuer un zoom. C'est le centre de la sphère qui a été utilisé pour connaître la direction. Cette approche était nécessaire parce que le choix de l'image à manipuler se basait sur une recherche de l'objet le plus proche de la camera. Par la suite, le geste a été changé et il n'était plus nécessaire que la caméra bouge, c'est pour cela qu'actuellement le champ de vision de la caméra est de nouveau utilisé pour le zoom. La méthode qui l'exécute est simple à modifier parce qu'elle utilise la valeur de la variable *fieldofView* appartenant à la caméra pour changer la vue. Un autre geste qui a été testé est le *swipe* en avant et en arrière pour le zoom, mais ce geste est plus difficile à effectuer et moins précis.

3.3.1.7.5 Pointeur et objet de sélection

La première façon pour choisir l'objet qui va être manipulé pour l'ajouter à la recherche ou à la comparaison, était de chercher celui qui se trouvait le plus proche de la caméra. Par la suite, un pointeur a été introduit pour définir l'objet à manipuler. Il s'effectuait grâce à la direction du doigt, mais il n'était pas très précis à cause de la mauvaise détection des mains et des doigts. Par contre la méthode pour faire cela a été gardée. Elle utilise la fonction *Physics.Raycast* qui permet de savoir si un objet est touché dans la direction indiquée et surtout de récupérer celui-ci. Cette fonction a comme paramètre un point d'origine et une direction. Lorsqu'un objet est touché, la méthode teste si celui-ci est une image, si oui alors il est stocké dans une variable et pourra être utilisé pour les diverses manipulations dans l'écran de recherche ou dans celui des éléments ajoutés à la recherche.

3.3.1.7.6 Halo

Pour mieux distinguer sur quelle image la caméra pointe, il était nécessaire d'avoir quelque chose pour la différencier. Le plus simple à faire était de rajouter un halo à l'objet en question et de le rendre visible ou invisible lorsque cela est nécessaire. L'autre manière qui a été essayée, mais abandonnée, est le changement de couleur de l'image, parce qu'il était plus difficile de revenir à l'état initial.

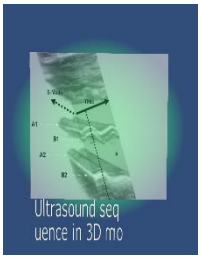


Figure 22 : Halo sur image

3.3.1.7.7 Ajout à la recherche

Pour ajouter des éléments à la recherche que ce soit positivement ou négativement, il a fallu choisir un geste. Le mieux a été d'utiliser un *swipe* gauche pour le positif et droite pour le négatif. Une fois ce mouvement détecté, le programme lance la méthode *addCaption* ou *removeCaption* du script JSONAccess.cs qui a pour paramètre le *caption* de l'image. Ces méthodes s'occupent de faire une requête sur le serveur avec les nouveaux mots-clés en ajoutant un + ou un – devant ceux-ci. Puis elles cachent les objets de la dernière recherche, désactivent la collision pour éviter qu'un objet caché soit sélectionné et assignent la nouvelle

recherche aux objets désactivés. Au début l'utilisateur ne savait pas si le résultat de la recherche avait changé ou si l'affichage des images correspondait vraiment au résultat car les anciennes images restaient affichées, ce qui faussait le résultat de la recherche. C'est pourquoi le nouveau processus inclut la partie où les objets sont cachés les faisant réapparaître uniquement au besoin. Grâce à cela le nombre d'images affichées correspond réellement à la recherche. Au niveau des gestes, pas d'autres ont été testés.

3.3.1.7.8 Sélection pour la comparaison

Dans le même principe que l'ajout à la recherche, il est nécessaire d'introduire un geste pour les objets de la comparaison. Il y a aussi deux gestes qui y sont assignés, pour attribuer une image soit à l'objet un de comparaison, soit à l'objet deux. De cette manière il est plus facile de gérer les images qui seront comparées. C'est à nouveau le *swipe* qui gère cela mais cette fois-ci avec un doigt de la main gauche. Lorsque l'utilisateur effectue le mouvement, l'objet est sélectionné et il est possible de le voir en haut de la sphère. Ceci permet d'avoir un aperçu des objets sélectionnés pour la comparaison.

3.3.1.7.9 Changement d'un écran à l'autre

Dans cette application, il y a pour l'instant trois modes, chacun doit être accessible grâce à un geste. Depuis l'écran de recherche c'est *swipe* gauche ou droite pour afficher les autres écrans. Puis à partir de ceux-ci, un *swipe* pour revenir à la recherche. Il n'est pas possible d'aller d'un écran à l'autre sans passer par le principal, parce qu'il a été décidé que l'utilisateur retournerait plutôt au résultat de la recherche.

3.3.2 Les éléments ajoutés à la recherche

3.3.2.1 Le principe

Pour ce qui est des images que l'utilisateur rajoute positivement ou négativement à la recherche, il a fallu faire un affichage servant à la gestion des images pour le cas où l'utilisateur aimerait enlever une image de la recherche ou simplement voir ce qu'il y a dans celle-ci. Par exemple, l'utilisateur ajoutant par mégarde une image à sa recherche, il va simplement dans cette vue et enlève l'image d'un simple mouvement au lieu de devoir tout recommencer.

3.3.2.2 Visuellement

La vue en elle-même se compose de deux tableaux, l'un affichant les images ajoutées positivement et l'autre négativement. Comme il est possible de le voir sur la figure 23 ci-dessous, à gauche ce sont les éléments ajoutés positivement et à droite ceux ajoutés négativement. Chaque tableau a une largeur de deux images.

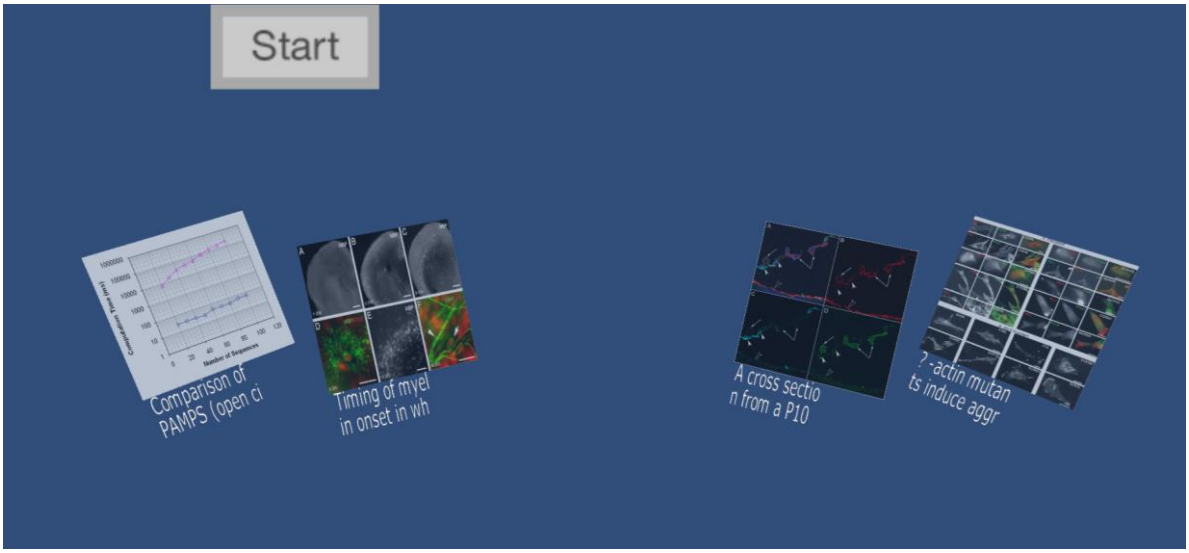


Figure 23 : Vue de l'écran des éléments ajoutés à la recherche

3.3.2.3 Méthode d'affichage

Pour changer de vue il suffit de faire un geste précis depuis la recherche. Celui-ci va lancer une méthode remplaçant le visuel par les deux tableaux en cachant les éléments de l'un et en rendant visible ceux de l'autre. Pour afficher ou masquer un objet, il faut utiliser la variable *renderer.enabled* de l'objet, disant à Unity d'afficher ou de cacher un objet de l'écran. C'est le *swipe* vers la gauche qui est le déclencheur.

3.3.2.4 Manipulation

Dans cet écran, la main droite effectue les mêmes manipulations que dans l'écran de la recherche, c'est-à-dire la rotation, le zoom et le pointeur. La main gauche s'occupe d'enlever l'élément de la recherche en fermant la main.

3.3.2.4.1 Enlever un élément de la recherche

La main gauche sert à enlever un élément de la recherche en fermant le poing. Ce geste appelle la méthode *removeFromSearch* qui va enlever le *caption* de l'image de la requête et faire une recherche sans celle-ci. Puis elle assigne le résultat aux objets de la recherche.

3.3.3 La comparaison

3.3.3.1 Le principe

En plus de pouvoir faire une recherche, l'application donne la possibilité de comparer deux images. Dans le cas de la médecine, l'utilisateur peut voir si par exemple les images comportent les mêmes symptômes. Le médecin peut aussi être plus sûr de son choix. Dans cet écran il doit être possible de mouvoir les images, de les agrandir et de les rétrécir.

3.3.3.2 Visuellement

L'affiche est très simple, deux images côte-à-côte que l'on peut manipuler grâce aux mains. Chaque main manipule une image.

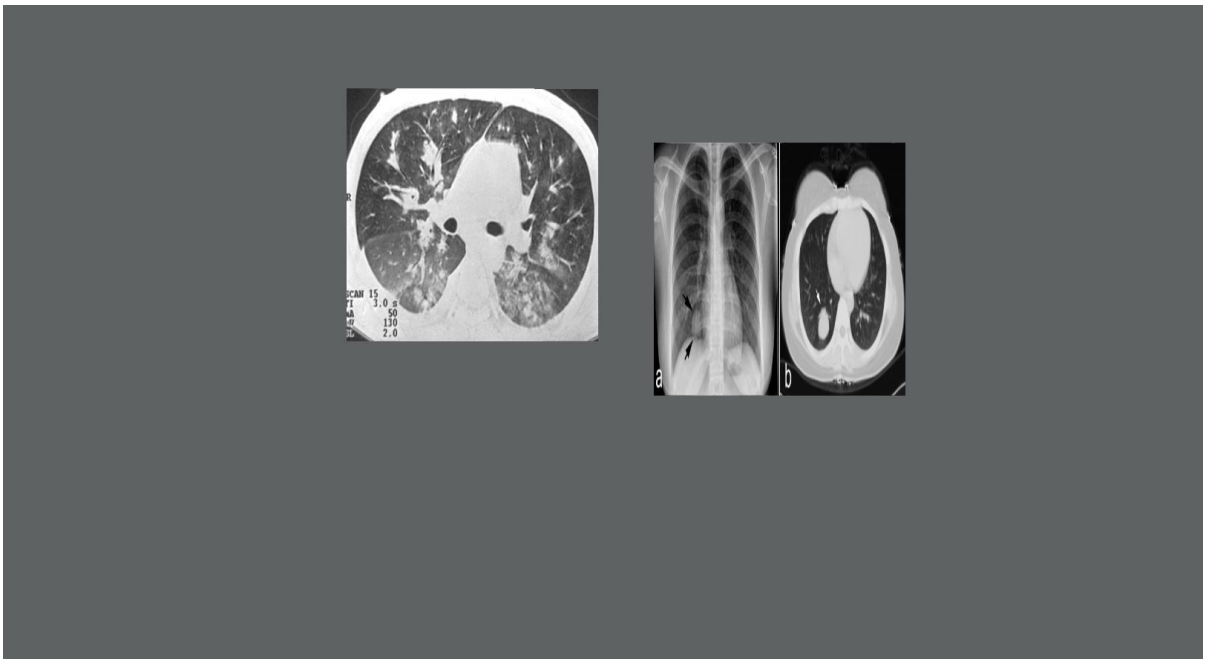


Figure 24 : Écran de comparaison

3.3.3.3 Méthode d'affichage

La méthode pour afficher cet écran est similaire aux autres. Pour ce faire, il faut passer par la recherche, puis une méthode s'occupe de cacher les objets de la recherche et d'afficher

ceux de la comparaison. Puis de positionner les deux images en face de la caméra, l'une à droite et l'autre à gauche. Il faut aussi penser à les tourner dans le bon sens.

3.3.3.4 *Manipulation*

La manipulation est similaire pour chaque main. L'objet suit le mouvement de la main assignée lorsqu'elle a quatre doigts tendus. En plus de faire des mouvements dans l'axe horizontal et vertical, il gère la profondeur, mais seulement sur deux positions prédéfinies. Il a été décidé que l'utilisateur n'a besoin de celle-ci uniquement pour mettre une image devant l'autre. De plus l'image se trouvant devant devient un peu plus transparente pour aider à la comparaison. Pour finir il est aussi possible d'agrandir ou réduire la taille des images. Ce n'est plus le champ de vision qui est réduit comme sur les autres écrans mais la taille de l'objet qui est agrandie ou réduite. Ceci s'effectue comme avec l'écran principal en fermant le poing et en effectuant une rotation vers la gauche pour agrandir et la droite pour rétrécir.

3.3.4 Gestes en bref

Écran	**Main et doigts**	**Geste**	**Fonction**
Recherche	Droite, quatre doigts tendus	Mouvement de la main	Rotation
	Droite, cinq doigts tendus	Swipe gauche	Afficher écran comparaison
	Droite, cinq doigts tendus	Swipe droite	Afficher écran éléments de recherche
	Gauche, plus de trois doigts tendus	Swipe droite	Ajouter négativement à la recherche
	Gauche, plus de trois doigts tendus	Swipe gauche	Ajouter positivement à la recherche
	Gauche, un doigt tendu	Swipe droite	Affecter à l'élément de comparaison un
	Gauche un doigt tendu	Swipe gauche	Affecter à l'élément de comparaison deux

Comparaison	Droite et gauche, quatre doigts tendus	Mouvement de la main	Mouvoir l'image attribuée à chaque main
	Droite et gauche, poing fermé	Rotation droite et gauche	Agrandir ou rétrécir l'image attribuée
	Droite	Swipe droite	Revenir à la recherche
Éléments de la recherche	Droite, quatre doigts tendus	Mouvement de la main	Rotation pour cibler une image
	Gauche	Fermer la main	Retirer un élément de la recherche

Tableau 2 : Gestes et fonctions du prototype

4 Conclusion

4.1 Bilan technique du projet

4.1.1 Performances

L'exécution du programme se fait rapidement. L'affichage du résultat de la recherche par la sélection dans le menu est direct, mais seulement si l'utilisateur dispose d'une bonne connexion. Sinon les images s'affichent de manière aléatoire les unes après les autres. L'utilisation des écrans est de manière générale fluide, même si parfois elle saccade. L'écran avec les éléments ajoutés à la recherche manque de précision au niveau du choix de l'image. Ajouter ou enlever un élément de la recherche fonctionne bien et la recherche qui s'en suit aussi. Les manipulations sur l'écran de comparaison sont faciles à effectuer.

4.2 Principaux obstacles rencontrés

Un des problèmes fréquents survenus pendant ce projet est le calcul de la rotation. Elle était souvent trop dynamique pour une utilisation optimale. La cause principale étant la position spatiale des mains détectées par le Leap Motion. Finalement une solution s'est avérée judicieuse, la distance parcourue par les mains dans l'espace a été réduite. Certaines difficultés et incompréhensions ont pu être résolues grâce aux sites officiels qui ont une documentation très complète. Mis à part cela, les problèmes étaient mineurs, dont certain qui devait être réglés. Voici quelques problèmes et leur solution :

+ **Problème**	**Solution**
L'intégration du Leap dans le logiciel Unity	Utiliser un asset qui intègre tout
Comment utiliser l'espace en trois dimensions avec des images en deux dimensions	Faire des exemples d'interface et s'inspirer de ce qui existe déjà
Affichage correct des images et dans la bonne position	Recherche sur internet et comprendre comment fonctionne l'espace en trois dimensions

Tableau 3 : Problèmes survenus dans le travail

Certains problèmes ont été résolus en abordant différemment la manière de faire de certaines fonctions ou d'utiliser le prototype.

4.3 Respect du cahier des charges

En ce qui concerne le cahier des charges, dans les fonctions obligatoires, une seule n'a pas pu être implémentée, c'est l'accès de l'application par le web. Cela étant il a été décidé que celle-ci n'était plus urgente parce que le logiciel Unity intègrera prochainement une fonction pour déployer l'application sur le web.

Dans les points optionnels du cahier des charges, certaines fonctionnalités n'ont pas été implémentées :

- L'utilisation de la reconnaissance vocale : la fonctionnalité n'a pas été implémentée mais les recherches nécessaires ont été effectuées. Et une alternative sous la fonction d'un menu avec des mots proposés a été introduite.
- Voir l'article concernant une image : l'avancement du projet ne permettait pas d'intégrer cette fonction.

De plus certaines autres fonctionnalités ont été introduites comme le menu que l'utilisateur peut utiliser à tout moment.

4.4 Améliorations futures possibles

Voici une petite liste non exhaustive des modifications possibles à effectuer sur l'application.

4.4.1 Reconnaissance vocale

L'intégration de la reconnaissance vocale serait un plus. Ceci permettrait une utilisation plus rapide pour la recherche par mots-clés. Le mieux serait de définir dynamiquement les mots qui peuvent être reconnus pour éviter les erreurs de compréhension.

4.4.2 Accès depuis internet

Pour l'accès de l'application par le web, il faudrait utiliser une des solutions proposées dans ce projet. Le mieux serait d'utiliser la nouvelle fonction qui va être incluse prochainement dans Unity. Celle-ci permet de passer le projet en WebGL simplement.

4.4.3 Menu dynamique

Actuellement pour ajouter un objet au menu, il faut créer tout ce qui est graphique. L'idéal serait de faire en sorte que les boutons soient toujours les mêmes, d'ajuster la taille et d'écrire le texte avec la fonction qui a été utilisée pour la partie texte sous les images.

4.4.4 Article

Une des fonctions qui n'a pas pu être intégrée mais qui serait intéressante, surtout pour les médecins, serait d'afficher l'article dont l'image fait partie.

4.4.5 Options de recherche

Sur l'ancienne interface, il existe des fonctionnalités qui n'ont pas été réintroduites dans ce projet. Celles-ci pourraient être analysées pour voir si elles devraient intégrer ce projet.

4.4.6 Vue différente pour l'écran de recherche

Durant le projet, différentes vues ont été réalisées. Il a été décidé d'utiliser l'intérieur d'une sphère, mais la possibilité d'avoir plusieurs vues différentes permettant ainsi d'autres rendus avantageant l'utilisateur. De manière à enrichir le résultat de la recherche. Mais cela implique aussi de devoir changer la position des objets.

4.4.7 Performances

Le projet n'a pas été pensé performance parce que l'important était d'avoir un prototype qui respecte le cahier des charges. C'est pourquoi il faudrait analyser ce qui a été fait et voir si c'est possible de faire mieux.

4.5 Possibilités d'utilisation

4.5.1 Réalité augmentée

Une utilisation de la réalité augmentée comme les Occulus Rift augmenterait l'expérience de l'utilisateur. C'est un casque qui permet de voir un espace en trois dimensions de manière réelle. Celui-ci permet l'immersion dans le programme et d'effectuer la rotation grâce aux mouvements de la tête.

4.5.2 Google Glasses

Les Google Glasses sont des lunettes permettant un affichage sur un petit écran devant l'œil droite. Une possibilité d'intégration de ceux-ci dans le projet est de les utiliser pour afficher des informations par rapport au résultat de la recherche. Cela permettrait par exemple d'avoir des informations affichées sur l'écran pendant une opération chirurgicale.

4.6 Conclusion personnelle

Ce travail a été très instructif. Il m'a permis d'avoir une expérience inédite grâce au développement d'une application avec les dernières technologies. Pouvoir utiliser un programme de conception de jeux-vidéos pour créer ce prototype m'a donné une vague idée du travail effectué par un développeur dans ce domaine. L'apprentissage de l'espace en trois dimensions était nouveau pour moi et a été très intéressant.

Le Leap Motion est une avancée technologique parfois frustrante à utiliser parce qu'elle ne fait pas toujours tout ce que l'utilisateur aimerait. Mais globalement il était captivant de chercher comment simplifier l'utilisation pour éviter les complications. C'est aussi intéressant de voir l'intégration d'une technologie avec une autre.

Les personnes me suivant dans ce projet étaient de bon conseil. Grâce aux rendez-vous hebdomadaires, je faisais le point avec eux et définissais les objectifs suivants.

5 Références

Commentçamarche.net. (2014, Juillet). *Comment ça marche.net Dossiers*. Récupéré sur commentcamarche: http://www.commentcamarche.net/contents/1332-introduction-a-xml.html

Edelhart, D. (2013, Novembre 8). *Using Leap Motion Hardware with Unity Web Player*. Récupéré sur Leap Motion: https://www.leapmotion.com/blog/using-the-leap-motion-hardware-with-unityweb-player/

Engineer, D. P.-L. (2014). *developer leapmotion*. Récupéré sur leapmotion: https://developer.leapmotion.com/documentation/csharp/devguide/Unity_Boilerplate.html

Google. (2014). *Web Speech API Demo*. Récupéré sur google: https://www.google.com/intl/fr/chrome/demos/speech.html

Henning Müller, J. K.-C.-F. (2013). *AMIA : Medical task*. Récupéré sur ImageCLEF: http://imageclef.org/2013/medical

IzzySoft. (2014, Janvier 6). *forum unity3d*. Récupéré sur unity3d: http://forum.unity3d.com/threads/drawtext-on-texture2d.220217/#post-1474196

Jessy. (2008, Mai 9). *How to instantiate objets in a circle*. Récupéré sur Forum Unity3D: http://forum.unity3d.com/threads/how-to-instantiate-objects-in-a-circle.10693/

KHRONOS. (2014). *KHRONOS Group*. Récupéré sur Khronos Group: http://www.khronos.org/legal/trademarks/#webgl

Lamelot, M. (2014, Mars 28). *Tom's Hardware Articles*. Récupéré sur tom's Hardware: http://www.tomshardware.fr/articles/kinect-2-pc,1-47244.html

Leap Motion. (2014). *Leap Motion Documentation*. Récupéré sur Leap Motion: https://developer.leapmotion.com/documentation/csharp/devguide/Leap_Gestures.html#id1

moust. (2013, Septembre 2). *alsacréations tutoriels*. Récupéré sur alsacréation: http://www.alsacreations.com/tuto/lire/1572-webgl-3d-three-canvas-threejs.html

nimble UX. (2013, Décembre 10). *pmdtec*. Récupéré sur pmdtec nimbre ux: http://www.pmdtec.com/nimbleux/

Plyson, J. (2014, Mai 6). *assetstore unity3d*. Récupéré sur unity3d: https://www.assetstore.unity3d.com/en/#!/content/13882

Pogue, D. (2013, Juillet 24). *Leap Motion Controller, Great Hardware in Search of Great Software*. Récupéré sur The New York Times: http://www.nytimes.com/2013/07/25/technology/personaltech/no-keyboard-and-now-no-touch-screen-either.html?_r=0

Reimer, D. (2010, Août 20). *blog almostlogical*. Récupéré sur AlmostLogical: http://blog.almostlogical.com/2010/08/20/adding-text-to-texture-at-runtime-in-unity3d-without-using-render-texture/

runevision. (2007, Novement 28). *Forum Unity3d*. Récupéré sur Unity3d: http://forum.unity3d.com/threads/legality-on-using-unitys-logo.74057/

THALMICLABS. (2014). *ThalmicLabs et Myo*. Récupéré sur ThalmicLabs: https://www.thalmic.com/en/myo/

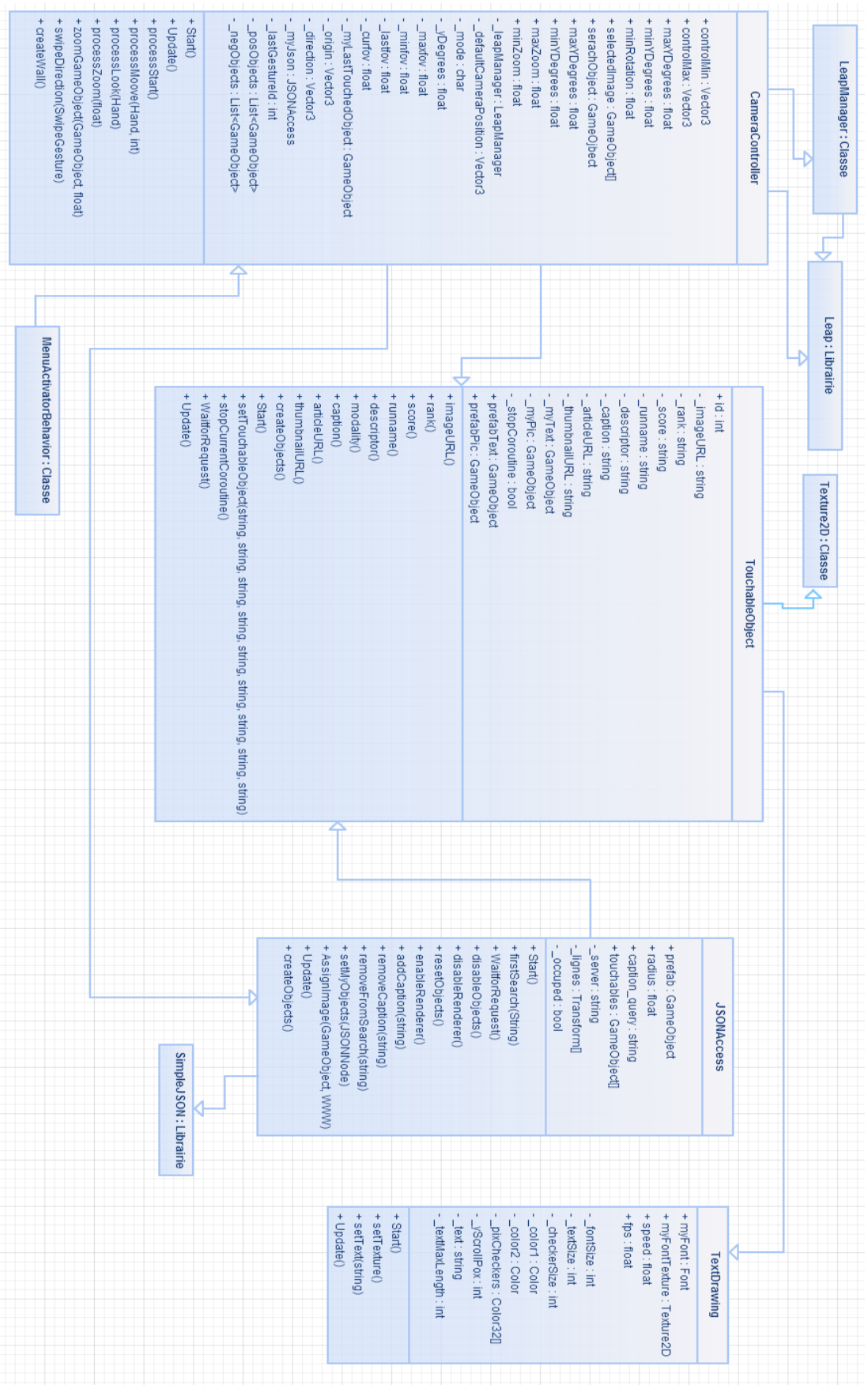
Ulloa, C. (2013, Octobre 30). *From Unity to Three.js*. Récupéré sur HELLO ENJOY: http://helloenjoy.com/2013/from-unity-to-three-js/

Wittber, S. (2011, Février 8). *Uniform points on sphere*. Récupéré sur entitycrisis blogspot: http://entitycrisis.blogspot.ch/2011/02/uniform-points-on-sphere.html

ZJP. (2010, Janvier 22). *windows udp voice recognition server*. Récupéré sur Forum Unity3D: http://forum.unity3d.com/threads/windows-udp-voice-recognition-server.172758/

6 Annexe

Annexe I : Diagramme de classes



Annexe 2 : Cahier de charges

Must have

Recherche sur la technologie et le langage à utiliser

Utilisation du Leap Motion pour effectuer des actions

Définir les gestes à intégrer pour chaque fonction

Comparer deux images visuellement en grand

L'utilisateur peut faire une recherche initiale grâce à un menu (initialement pas dans le cahier de charges)

L'utilisateur peut ajouter des éléments à la recherche

L'utilisateur peut gérer les éléments ajoutés à la recherche

L'utilisateur navigue dans une interface 3D

L'utilisateur accède à l'application grâce au web

Could be good

L'utilisateur peut faire une recherche grâce à la reconnaissance vocale

L'utilisateur à la possibilité de voir l'article concernant une image

L'utilisateur modifie l'apparence de l'image pour mieux comparer (taille et visuel)

L'utilisateur peut manipuler deux images en même temps

Annexe 3 : Journal de travail

Date	Temps (h)	Quoi	Problème	Solution	Liens utiles
26/05/2014	2	Spawn object			http://www.unity3dstudent.com/2010/10/creating-random-spawning/ https://www.youtube.com/watch?v=HiBBInQfAy8
26/05/2014	2	Draw text on an object			http://blog.almostlogical.com/2010/08/20/adding-text-to-texture-at-runtime-in-unity3d-without-using-render-texture/ http://forum.unity3d.com/threads/220217-DrawText-on-Texture2D?p=1474196&viewfull=1#post1474196
26/05/2014	3	Créer le prototype 1	Intégration leap		
27/05/2014	3	Inclure leap dans le proto		utiliser shambala	
27/05/2014	5	1 proto avec rotation et 1 autre avec sélection			
28/05/2014	4	essaie d'intégration de divers gestes	contrôle de la rotation		
28/05/2014	3	écrire rapport			
02/06/2014	1	Bricks suivent la caméra		chercher mes objets par tag	http://docs.unity3d.com/ScriptReference/GameObject.FindGameObjectsWithTag.html
02/06/2014	3	KeyTap pour sélectionner une image			
02/06/2014	3	Tester un système de highlight	Ne fonctionne pas correctement		
03/06/2014	2	Nettoyage du code			
03/06/2014	5	Ecrire un rapport sur le proto 1 et 2			
05/06/2014	4	Amélioration du zoom	Le zoom ne se		

			marie pas bien à l'utilisation		
08/06/2014	3	Améliorer la lisibilité du code		utilisation de switch case à la place des if else	
09/06/2014	4	Organisation du code + ajout d'un mode zoom			
10/06/2014	10	Réajustement du code	Programme ne fonctionne plus correctement	recommencer tout en reprenant le code existant et en réajustant	
11/06/2014	2	Crash d'Unity au lancement du projet	Pas les bonnes librairies	mettre les dernières librairies du sdk2	
11/06/2014	1	Mis à jour du code	Problème avec le count des doigts, dans le sdk2, toujours 5 doigts	Faire une méthode qui compte les doigts tendus	
11/06/2014	4	Tenter d'améliorer la rotation	La rotation est trop rapide	comprendre l'utilisation des quaternions	
12/06/2014	4	Ajout des mouvements directionnels			
12/06/2014	2	Ecrire rapport			
14/06/2014	4	Créer un pointeur, ajout d'un halo qui s'active ou désactive lorsque je pointe sur un objet			https://www.youtube.com/watch?v=dJTu-8RdP80
14/06/2014	2	Compléter le rapport			
14/06/2014	1	Amélioration du pointeur	origine du pointeur à bien définir		

14/06/2014	2	Ajout JSON			
17/06/2014	3	Amélioration zoom	Intuitivité		
18/06/2014	4	Mettre en place une rotation de la camera			
23/06/2014	6	Nouveau visuel "demi cylindre" + nouvelle façon de créer les objets			http://forum.unity3d.com/threads/how-to-instantiate-objects-in-a-circle.10693/
23/06/2014	3	écrire rapport			
24/06/2014	2	transparence alpha	alpha dans unlit shader	custom unlit shader	http://forum.unity3d.com/threads/unlit-with-adjustable-alpha.115455/
24/06/2014	2	Rotation de la camera sur elle-même	trop de vélocité		
24/06/2014	1	Pointeur suit la direction de la camera à la place du doigt			
24/06/2014	3	visuel et modification et sélection d'image			
25/06/2014	2	comparaison			
25/06/2014	3	rapport			
27/06/2014	6	ajuster affichage, ajuster affichage comparaison	affichage comparaison, un petit souci pour les mettre au bon endroit	utiliser des points par defaut	
30/06/2014	8	Ajouter add/remove à la recherche	url n'était pas bonne	utiliser la fonction EscapeURL	
01/07/2014	2	Recherche pour ajouter un objet pour le texte	l'objet n'est pas présent lorsque je le mets en enfant de mon objet d'image		
01/07/2014	5	Rapport			
02/07/2014	3	Rapport final, lire les exemples de rapport			

02/07/2014	2	Ecrire le rapport final, la partie méthode			
02/07/2014	5	Ecrire le rapport final : introduction, méthode, résultat			
03/07/2014	3	Ecrire le rapport final, approfondir certaines sections			
03/07/2014	5	Ecrire le rapport final			
04/07/2014	2	Ecrire méthode pour texte			
04/07/2014	4	continuer la méthode pour texte			
05/07/2014	4	rapport			http://gamedev.stackexchange.com/questions/10865/unity3d-or-webgl-where-should-i-go
07/07/2014	2	Transparence du fond texte			http://codeflow.org/entries/2013/feb/02/why-you-should-use-webgl/
07/07/2014	1	Rendez-vous			
avant le 26.05.14	20				
08/07/2014	3	Ecran positif, négatif + amélioration texte			
08/07/2014	7	Rapport final			
09/07/2014	4	Méthode pour le positif/négatif	pas de swipe en fermant le poing		
09/07/2014	4	Changement d'écran	problème pour faire disparaitre toutes les images		
09/07/2014	2	Tente de comprendre l'affichage	problème avec la hauteur		
10/07/2014	2	affichage	Problème d'affichage		
10/07/2014	1	améliorer le pointeur	myText qui posait des		

			soucis		
10/07/2014	2	écran comparaison, mouvement des objets qui suit les mains	ne fonctionne pas correctement		
10/07/2014	5	Faire le point sur ce qu'il faut faire + nettoyer code + rapport	enlever les variables pas nécessaires		
11/07/2014	2	2e ligne texte	taille		
11/07/2014	1	modifier json + modification de code	EscapeURL met des + à la place de l'espace	Utiliser replace pour mettre des -	
11/07/2014	3	déplacement et rotation dans comparaison	position des mains pas cohérentes, rotation pas bon	déplacement calcule spécifique, rotation mauvaise manipulation de ma part	
11/07/2014	2	zoom dans comparaison	zoom ne fonctionne pas correctement, revoir geste ou fonction		
	7	Séance hebdomadaire pas noté			
11/07/2014	3	Ecran comparaison et positif/négatif	Affichage, afficher texte, mouvement		
12/07/2014	4	Ecran positif-négatif	Soucis pour afficher correctement texte et image		
12/07/2014	0.5	Geste pour restart l'app			
12/07/2014	0.5	Position départ pour comparaison	à l'envers	changer la rotation Y à 180	
12/07/2014	3	Afficher texte dans pos/nég	Problème de coroutine	stopallcoroutine	
12/07/2014	2	Fonction			

		removefromsearch, enlever halo dans pos/nég après fin, zoom comparaison			
13/07/2014	10	Faire des recherches et l'interface du menu		utiliser MarchingMenu (asset store) puis faire les éléments graphiques avec Photoshop	
14/07/2014	9	Fusion menu + prototype recherche + modifier fonction + ajout fonction			
15/07/2014	10	Rapport final			
16/07/2014	6	Rapport final			
17/07/2014	9	Rapport final			
18/07/2014	10	Rapport final			
19/07/2014	6	Rapport final + modification de la méthode de déplacement dans écran comparaison			
20/07/2014	10	Rapport final			
21/07/2014	12	Rapport, modification et nettoyage du code			
22/07/2014	10	Rapport final, uml classe			
23/07/2014	10	Test performance, rapport et nettoyage code			
24/07/2014	12	Rapport, rendez-vous, code			
25/07/2014	8	Rapport final, code, rendu du travail			
Total	359				

Je déclare, par ce document, que j'ai effectué le travail de Bachelor ci-annexé seul, sans autre aide que celles dûment signalées dans les références, et que je n'ai utilisé que les sources expressément mentionnées. Je ne donnerai aucune copie de ce rapport à un tiers sans l'autorisation conjointe du RF et du professeur chargé du suivi du travail de Bachelor, y compris au partenaire de recherche appliquée avec lequel j'ai collaboré, à l'exception des personnes qui m'ont fourni les principales informations nécessaires à la rédaction de ce travail et que je cite ci-après : Monsieur Antoine Widmer, Monsieur le Dr. Henning Müller, Monsieur Roger Schaer et les personnes ayant corrigé mon travail.