ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE AVEC MÉMOIRE EN GÉNIE LOGICIEL
M. Sc. A.

PAR
Alexandre MILLETTE

# DUALCAD : INTÉGRER LA RÉALITÉ AUGMENTÉE ET LES INTERFACES D'ORDINATEURS DE BUREAU DANS UN CONTEXTE DE CONCEPTION ASSISTÉE PAR ORDINATEUR

MONTRÉAL, LE 27 AVRIL 2016

**PRÉSENTATION DU JURY**

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Michael J. McGuffin, directeur de mémoire
Département de génie logiciel et des TI à l'École de technologie supérieure

M. Luc Duong, président du jury
Département de génie logiciel et des TI à l'École de technologie supérieure

M. David Labbé, membre du jury
Département de génie logiciel et des TI à l'École de technologie supérieure

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 6 AVRIL 2016

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

## AVANT-PROPOS

Lorsqu'il m'est venu le temps de choisir si je désirais faire une maîtrise avec ou sans mémoire, les technologies à Réalité Augmentée (RA) étaient encore des embryons de projets, entrepris par des compagnies connues pour leurs innovations, telles que *Google* avec leur *Google Glasses*, ou par des compagnies émergentes, telles que *META* et leur *Spaceglasses*. Beaucoup d'effort était mis dans la Réalité Virtuelle (RV) et les grands de ce marché voyaient déjà des résultats concluants. Or, j'imaginais personnellement un futur où la RA aurait davantage d'importance, plus particulièrement dans le domaine professionnel. Je voulais me détacher des visions s'approchant trop de la science-fiction ou du gadget pour identifier les utilités réelles qu'auraient ces technologies. L'idée originale était d'utiliser des lunettes à RA pour permettre à un utilisateur, assis devant son PC, d'étendre son écran à toute la pièce où il se trouve. Faisant usage des surfaces pour projeter des données ou bien faisant apparaître des écrans virtuels suspendus en l'air, révolues auraient été les limitations des écrans classiques.

Bien sûr ces ambitions furent refroidies lorsque je pus mettre la main sur de vraies lunettes, qu'on vantait être à la fine pointe de la technologie. Une résolution beaucoup plus petite que celle de nos téléphones, un champ de vision déplorable, un manque de précision quant au suivi de la tête et une difficulté à reconnaître les surfaces m'ont fait comprendre que, même si je croyais encore à mon idée originale, il me faudrait voir plus petit. J'entrepris alors de prototyper certaines des fonctionnalités imaginées, avec des résultats variant entre l'échec et tout juste la preuve de concept. Un visualisateur 3D de graphiques produits dans Excel, un outil permettant de scanner du regard des items pour une base de données… tout cela était intéressant, mais rien de bien révolutionnaire. Je gardais toutefois en tête une idée qu'un collègue m'avait donnée : la possibilité de voir les objets produits avec un logiciel de Conception Assistée par Ordinateur (CAO ou CAD en anglais) et même de les retoucher. Si cela avait déjà été fait par d'autres auparavant, très peu s'étaient intéressé à l'idée de basculer directement d'une interface à l'autre. Mais comment s'intégrer à un système de CAO existant, tel que *Maya* ou *Blender* ? Et serais-je limité par leurs fonctionnalités ?

Il me fallut un temps pour l'admettre, mais il était nécessaire que je fasse mon propre logiciel de CAO pour PC, aussi simple soit-il. Heureusement, j'avais déjà démontré, lors de mes précédents prototypes, que je pouvais faire marcher en parallèle 2 logiciels, l'un en tant qu'interface utilisateur classique, l'autre en tant que système pour les lunettes à RA, et même les faire communiquer entre eux. C'est ainsi que fut développé DesktopCAD, en tout juste plus d'un mois. C'était un défi intéressant et je considère ce laps de temps comme celui le plus intéressant de ma maîtrise.

Une fois le logiciel pour PC réalisé, toute l'infrastructure de communication en place et une base de faite pour interagir du côté du logiciel AR, le vrai défi, et surtout le travail pertinent à mon mémoire, pouvait débuter.

## REMERCIEMENTS

Je remercie tout d'abord mon directeur de maîtrise, Michael J. McGuffin, qui a non seulement contribué financièrement à mon travail et fourni du matériel de qualité, mais m'a beaucoup aidé lors de la conception et du développement de mon système, en plus d'assister pour la rédaction de ce mémoire et de l'article scientifique.

Je remercie les organismes suivant (présentés en ordre chronologique) pour leur contribution financière, sans lesquelles je n'aurais pas pu me permettre de réaliser mon mémoire : Décanat des Études de l'ÉTS, Conseil de recherches en sciences naturelles et en génie du Canada (CRSNG) et Fonds de recherche Nature et Technologies Québec (FRQNT).

Je remercie également mon collègue Jean-Nicola Blanchet pour son assistance lors de la conception du système, lors du tournage du vidéo accompagnant mon article et pour ses idées.

Je remercie le service des TI de l'ÉTS qui m'a fourni une licence professionnelle pour Unity3D, prêté divers composants et conseiller quant à l'achat de plusieurs autres.

Finalement, je remercie ma famille et mes amis, qui m'ont encouragé tout au long de mon projet et qui m'ont changé les idées lorsque j'en avais besoin.

# DUALCAD : INTÉGRER LA RÉALITÉ AUGMENTÉE ET LES INTERFACES D'ORDINATEURS DE BUREAU DANS UN CONTEXTE DE CONCEPTION ASSISTÉE PAR ORDINATEUR

Alexandre MILLETTE

## RÉSUMÉ

Une soudaine résurgence de l'engouement envers les technologies à Réalité Virtuelle (RV) et Réalité Augmentée (RA), liée à des avancées considérables dans le domaine des visiocasques, est très prometteuse pour de nombreux secteurs professionnels. Plus particulièrement, la Conception Assistée par Ordinateur (CAO) peut grandement profiter de ces technologies permettant la modélisation d'objets 3D dans un environnement immersif. Les systèmes de VR et de RA permettent aussi l'usage de dispositifs tenus en main ou simplement de gestuelle pour effectuer des opérations de CAO plus rapidement et simplement qu'avec une souris. Toutefois, les visiocasques actuels souffrent d'une faible résolution, d'un champ de vision limité et manquent généralement de précision dans leurs contrôles. Nous proposons donc trois contributions : (1) un système multimodal de CAO, nommé DualCAD, permettant de passer rapidement d'une interface classique de bureau avec clavier-souris (DesktopCAD) à une interface de RA avec visiocasque (ARCAD) en une seule commande et vice versa. Le premier mode permet des opérations précises alors que le second se veut plus intuitif, permettant un travail brouillon. Ensuite, pour améliorer le mode à RA, nous proposons (2) l'usage d'un téléphone intelligent et de techniques d'interaction qui en font usage. Le téléphone permet non seulement l'affichage d'informations utiles comme des menus, mais aussi de tracer avec un stylet et d'entrer des commandes sans ambiguïté. Celui-ci peut aussi être suivi en 3D pour servir d'accessoire de transformations directes ou indirectes. Deux techniques novatrices pour la manipulation et édition de modèles sont présentées et implémentées. Finalement, (3) le tout fut évalué qualitativement à l'aide d'une étude informelle et est comparé à l'état de l'art à l'aide de deux taxonomies. Le document se clôt sur des recommandations pour des travaux futurs.

**Mots-clés :** Conception Assistée par Ordinateur, Réalité Augmentée, Multimodal, Visiocasque, Téléphone Intelligent

# DUALCAD: INTEGRATING AUGMENTED REALITY AND A DESKTOP GUI FOR COMPUTER AIDED DESIGN (CAD)

Alexandre MILLETTE

## ABSTRACT

The increasing performance and falling price of head-mounted displays (HMDs) create new opportunities for virtual reality (VR) and augmented reality (AR). Computer aided design (CAD) in immersive 3D environments will be accessible to more users and make it easier to perceive and understand 3D scenes. AR and VR will also allow the user to point directly in 3D with a device or with their fingers, allowing certain tasks to be performed faster than with a mouse. However, HMDs often suffer from limited field-of-view and/or limited pixel density. Also, users become quickly tired of holding their arms up, and pointing in midair is less precise than a mouse. This dissertation investigates two novel solutions to these problems for CAD and 3D modeling. First, we propose allowing the user to quickly and easily switch between two modes: a desktop mode ("DesktopCAD") for precise work, where a keyboard, mouse, and external LCD monitor are used; and an augmented reality mode ("ARCAD") for rougher but more natural work, with HMD for output and the use of fingers and a hand-held device for pointing. Second, to improve the augmented reality mode, we leverage the use of a smartphone that supports multitouch input and stylus input. The smartphone can display information such as menus but can also be used to register touch or stylus input. The phone's position and orientation can be tracked to turn it into a physical prop for transformations. We present two novel interaction techniques for defining and for modifying 3D objects using the stylus and smartphone while in ARCAD mode. Finally, we compare our system to the state of the art through two taxonomies and we report the results of informal testing with expert users.

**Keywords:** Computer-Aided Design, Augmented Reality, Multimodal, Head-Mounted Display, Smartphone

## TABLE DES MATIÈRES

Page

## LISTE DES TABLEAUX

Page

## LISTE DES FIGURES

Page

## LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

| | |
|---|---|
| CAD | *Computer-Aided Design*, voir CAO |
| CAO | Conception Assistée par Ordinateur |
| CLR | *Common Language Runtime* |
| PC | *Personal Computer*, ordinateur personnel |
| RA | Réalité Augmentée |
| RGB | Red-Green-Blue, permet de générer toutes les couleurs |
| RV | Réalité Virtuelle |

# INTRODUCTION

Lorsqu'on parle de l'utilisation d'ordinateurs, de nos jours, il est généralement question d'un unique utilisateur placé devant un écran non-stéréoscopique (c'est-à-dire à deux dimensions) et interagissant avec l'ordinateur avec un clavier et une souris ou autre dispositif de positionnement en deux dimensions. Et, même si depuis 2014 on recense plus d'utilisateurs d'appareils mobiles (tablettes, téléphones intelligents, etc.) que d'ordinateurs de bureau ou laptops (Bosomworth, 2015), ces derniers sont encore considérés par la majorité comme étant plus fiables, plus stables et plus essentiels que leur contrepartie mobile (Sung et Mayer, 2012). Offrant beaucoup plus de fonctionnalités, de logiciels spécialisés et de performance, les ordinateurs de bureau dominent encore largement le domaine professionnel de ce qu'on connait sous le terme « travail de bureau ». Allant du travail de comptabilité jusqu'au design industriel en passant par la programmation et le multimédia, les ordinateurs sont utilisés de maintes façons et chaque amélioration quant aux interfaces utilisées pour accomplir le travail pourrait résulter en une hausse significative de productivité et donc de profits pour une entreprise. En termes de logiciels ou de performance de la machine, les gains acquis dans les 50 dernières années sont immenses. Mais au niveau du matériel d'input, les choses n'ont pas vraiment changé depuis l'introduction de la souris d'ordinateur vers la fin des années 60 (Engelbart, 1970). Bien sûr la combinaison traditionnelle du clavier-souris a ses avantages : le clavier offre des contrôles sans ambiguïté et permet à des utilisateurs experts d'accélérer la vitesse de leur travail à travers des raccourcis, tandis que la souris filtre les tremblements de la main pour produire des déplacements contrôlés et précis, et permet même un contrôle relativement aisé d'opérations en 3D, surtout lorsque celles-ci requièrent de la précision (Bérard et al., 2009). Par exemple, dans un contexte de Conception Assistée par Ordinateur (CAO) où un utilisateur désire modéliser des objets 3D et les assembler, la souris déplacée en 2D peut s'étendre à un monde en 3D à l'aide de widgets et de plans de caméra.

Par opposition, il y a eu dans les dernières années beaucoup de progrès dans le domaine de la Réalité Virtuelle (RV), par exemple avec le *Oculus Rift*, et de la Réalité Augmentée (RA), par exemple avec les applications mobiles pour téléphones intelligents, la *3DS* de Nintendo et

le *HoloLens* de Microsoft. Ce dernier en particulier, et tout autre appareil qui l'a précédé, a la particularité d'être un visiocasque. Un visiocasque est un appareil porté sur tête qui, à travers des écrans placés devant les yeux ou des écrans transparents, augmente la vision de l'utilisateur avec du contenu virtuel. Souvent couplés à un système de vision stéréoscopique et un système de suivi de la tête, ces appareils permettent une compréhension plus facile de scènes 3D (Ware et Franck, 1996) et, avec un système de vision par ordinateur pour le suivi des mains, ils peuvent offrir à un individu des méthodes d'interaction avec le monde virtuel. L'insertion de l'utilisateur dans la scène 3D a plusieurs avantages. Tout d'abord, il peut être préférable, pour certaines tâches, de manipuler 3 degrés de liberté ou plus simultanément au lieu d'être limité aux 2 degrés de la souris (Fuge et al., 2012; Hinckley et al., 1997; Toma, Gîrbacia et Antonya, 2012). Ensuite, le suivi des mains avec des caméras couleurs (RGB) et profondeur permet de sélectionner (Sharp et al., 2015) et déplacer approximativement avec ses doigts des objets selon 3 à 6 degrés de liberté, accélérant les tâches d'assemblage (McMahan et al., 2006). Finalement, des objets réels peuvent servir d'accessoires pour contrôler des objets virtuels, par contrôle direct ou indirect (Hinckley et al., 1994). Mais bien sûr, l'usage de ces technologies n'est pas sans problème. L'utilisation prolongée de visiocasques peut entraîner des maux de tête ou de cou, et le contrôle par les mains dans les airs provoque une fatigue musculaire après un temps. Les contrôles 3D sont moins précis que ceux 2D et sont très dépendants de la perspective. De plus, les algorithmes de vision sont limités par le champ de vision et ne peuvent rien face à l'occlusion, où un objet se trouve caché par un autre, bien que cette limitation puisse être surpassée par l'usage de dispositifs suivis par capteur.

Sachant les avantages et les limitations de chaque mode possible (interfaces classiques de bureau versus visiocasque à RA), il serait intéressant de se demander si la meilleure façon d'augmenter le rendement des professionnels dans un domaine comme la CAO ne serait pas de combiner ces deux modes, au lieu de tenter de les améliorer séparément. Par le passé, plusieurs auteurs ont noté que les systèmes faisant usage de la RA avaient des limitations importantes et qu'il était important de permettre à l'utilisateur de pouvoir transférer son travail vers un logiciel traditionnel. Dans leur article, Fuge et al. (Fuge et al., 2012)

mentionnent que les surfaces 3D produites à l'aide de leur prototype « can then be exported to third-party CAD software for further refinement and trimming operations, which are beyond the scope of the current work ». Ils admettent l'importance de pouvoir exporter les modèles, mais tout comme les autres chercheurs, ne se penchent pas sur l'implémentation réelle d'un système facilitant cette tâche. Voici donc la problématique approchée dans ce mémoire :

> Peut-on faire un système multimodal joignant un logiciel traditionnel de CAO et un système à RA de CAO et permettant de basculer aisément entre les deux afin d'augmenter la productivité ?
>
> Et quelles seraient les techniques d'interaction à utiliser pour faire usage des deux modes en s'appuyant sur les avantages mentionnés plus tôt tout en amortissant les limitations ?

Ici on utilise la définition de « multimodal » telle que donnée par Oviatt (Oviatt, 2007) : « Multimodal systems process two or more combined user input modes—such as speech, pen, touch, manual gestures, gaze, and head and body movements—in a coordinated manner with multimedia system output ».

Les hypothèses tirées de la problématique énoncée précédemment sont donc les suivantes :

1. Un système de CAO multimodal intégrant les interfaces classiques de bureau et celles de RA augmenterait la productivité des tâches de modélisation et d'assemblage face à chaque mode seul;
2. Des techniques d'interaction peuvent faire usage du contexte particulier du système multimodal et faciliter les tâches face aux techniques employées par d'autres systèmes utilisant un seul mode.

Pour vérifier ces hypothèses, une méthodologie a été prévue. Un prototype de ce système multimodal sera conçu, implémenté et analysé de façon qualitative. Le nom de code

**DualCAD** a été retenu car il s'agit d'un nom anglais propice à la publication future et qui représente les deux modes intégrés au système, soit **DesktopCAD** pour le mode logiciel traditionnel et **ARCAD** pour le mode en RA. Le tout fonctionnera sur un unique laptop et utilisera l'engin Unity3D pour effectuer le rendu, tant pour le mode traditionnel que le mode RA. Le module ARCAD utilisera les lunettes META (connues aussi sous le nom Spaceglasses) en tant que visiocasque suivi avec 6 degrés de liberté par des capteurs Polhemus Patriot. La Figure 1.1 représente graphiquement la structure à haut niveau du prototype. Cette structure sera expliquée plus en détails dans le CHAPITRE 2 - MÉTHODOLOGIE.

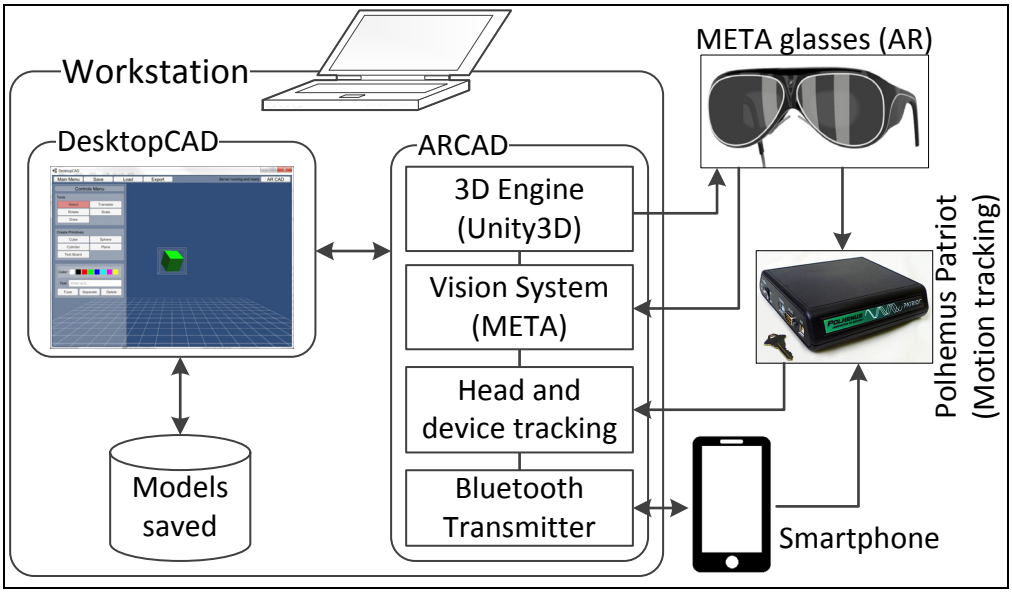


Figure 1.1 Structure du prototype DualCAD

La figure précédente inclut aussi un téléphone intelligent dans le système. En effet, nous avons suggéré l'utilisation d'un tel dispositif en lien avec un module de RA pour aller au-delà des limitations classiques des algorithmes de vision, telles que l'occlusion et le champ de vision. Étant un outil à la portée de presque tout le monde, son utilisation n'inclurait pas de coûts supplémentaires pour les utilisateurs et, bien que des capteurs de mouvements externes ont été utilisé pour faire le suivi en 3D pour cette recherche, des poussées récentes quant aux senseurs dans le téléphone et aux algorithmes de vision appliqués à la caméra du

dispositif permettrait éventuellement de remplacer ces capteurs externes (Google, 2015). En plus de permettre à l'utilisateur d'effectuer des mouvements précis avec son téléphone intelligent, ce dernier permet aussi d'afficher des informations critiques sur l'écran du téléphone, offre un écran multi-tactile précis et peut même être utilisé avec un stylet. Ce genre d'input est bien moins sensible aux faux-positifs et faux-négatifs, par opposition aux contrôles liés au suivi par vision de la main et des doigts, en plus d'offrir une surface précise à utiliser pour l'utilisateur. Finalement, les dispositifs permettant un contrôle indirect sont généralement plus stables que la main humaine et sont moins fatiguant. L'intégration d'un téléphone intelligent, connecté au système par Bluetooth, permettra donc de concevoir et implémenter des nouvelles techniques d'interaction qui pourront soutenir l'hypothèse 2.

Finalement, pour analyser le prototype conçu, une démonstration sera faite auprès d'utilisateurs experts en modélisation 3D. Ceux-ci, ayant beaucoup d'expérience avec des logiciels de CAO, pourront tester le système et le commenter. Ces impressions seront compilées et serviront à produire des conclusions et recommandations pour des projets futurs. Il est à noter que cette étude restera relativement informelle, se limitant à un petit groupe de testeurs, et faisant une analyse principalement qualitative. Si des tests quantitatifs avaient été prévus et seront même abordés dans ce mémoire, ils auront été mis de côté dû aux limitations matérielles des lunettes à RA. En effet, celles-ci offrent un champ de vision trop restreint et ne sont pas très confortable à utiliser sur une longue période. Les résultats en auraient donc été affectés et il nous a semblé plus sensé de se limiter à des analyses qualitatives pour la portée de ce mémoire.

En résumé, voici les objectifs de ce mémoire, expliqués au chapitre 1.6 :

- Concevoir et développer un système multimodal intégrant un logiciel traditionnel de CAO et un logiciel de CAO faisant usage de RA;
- Concevoir et implémenter des techniques d'interaction faisant usage du contexte introduit par le système mentionné précédemment;
- Analyser le prototype avec une étude informelle et effectuer des recommandations quant aux travaux futurs.

# CHAPITRE 1

# REVUE DE LA LITTÉRATURE

## 1.1 Souris versus contrôles 3D

L'arrivée de la souris d'ordinateurs, souvent attribuée à Douglas Engelbart et son équipe (Engelbart, 1970) a permis de façonner la manière dont on interagissait avec les ordinateurs pendant des décennies. Accompagnant le clavier, la souris facilite l'accès aux interfaces pour les utilisateurs novices tout en offrant un large éventail de possibilités pour un utilisateur expert. On pourrait aller jusqu'à dire que la souris a fait passer l'informatique d'un mode axé sur le texte vers un monde axé sur les interactions humain-machine. Et malgré la simplicité apparente de l'outil, la souris d'ordinateur présente des avantages significatifs quant à ses fonctions. Tout d'abord, comme les interfaces utilisateurs sont quasi-totalement toutes conçues sur un plan 2D, n'utilisant l'occlusion que pour cacher des items n'ayant pas le focus d'attention, un outil se déplaçant sur un plan 2D est plus que suffisant pour la majorité des interfaces. Ensuite, l'appareil est constamment posé sur une surface solide, ce qui signifie qu'il n'entraîne pas de poids supplémentaire à son utilisation, en plus de laisser le curseur au même endroit si on cesse de contrôler la souris. Du même fait, la main humaine, qui a tendance à trembler et qui pourrait causer des imprécisions dans une manipulation, est alors posée et le faible poids de la souris suffit à filtrer ces tremblements pour ne laisser que les mouvements désirés. Et comme l'ont décrit Balakrishnan et al. dans leurs travaux sur une nouvelle itération de souris (Balakrishnan et al., 1997), le fait d'avoir des boutons perpendiculaires au plan de déplacement a pour effet d'augmenter la stabilité de l'outil lors de clics. Finalement, le contrôle 2D peut être encore amélioré au niveau logiciel. Les déplacements de la souris peuvent être accélérés pour permettre d'accéder plus rapidement à des objets lointains, en accord avec la loi de Fitts (Fitts, 1954), sans pour autant impacter la précision du curseur lorsque celui se déplace à basse vitesse (Blanch, Guiard et Beaudouin-Lafon, 2004). On peut voir que la souris est donc tout aussi adaptée aux larges interfaces. Cette caractéristique fonctionne aussi avec les interfaces 3D, tel que l'ont prouvé Elmqvist et al. (Elmqvist et Fekete, 2008).

D'autres chercheurs se sont aussi intéressés aux performances de la souris en lien avec des interfaces 3D : Bérard et al. (Bérard et al., 2009) ont voulu comparer la souris à plusieurs autres outils offrant 3 ou plus degrés de liberté pour plusieurs tâches classiques dans des scènes 3D. La sélection d'objets est pour eux triviale puisqu'il est plus naturel pour un humain de pointer l'objet qu'il souhaite sélectionner, une technique qui est déjà bien représentée par la méthode de « picking », c'est-à-dire de passer le curseur d'une souris 3D devant un objet et de cliquer pour en effectuer la sélection. Ils se sont davantage penchés sur la tâche de déplacements 3D, qui a un besoin inhérent de manipuler 3 degrés de liberté. On aurait pu croire que des outils offrant 3 degrés de liberté seraient donc beaucoup plus efficaces pour cette tâche. Or, tel qu'ils l'ont prouvé à travers des tests utilisateurs, le fait de pouvoir éditer l'objet 2 degrés de liberté à la fois avec la souris a permis d'obtenir des opérations avec sensiblement la même rapidité qu'avec un outil 3D. De plus, les avantages de la souris énumérés plus tôt (précision, filtre des tremblements de la main) ne sont tout simplement pas égalés par les autres outils et donc les tâches de déplacement 3D étaient accomplies avec beaucoup de précision grâce à la souris. Ces résultats démontrent donc que la souris peut s'avérer très efficace lors de contrôles en 3D, surtout lorsqu'on l'enrichit de contraintes souvent présentes en manipulation : les objets doivent glisser l'un sur l'autre (2 degrés de liberté), les objets sont attirés l'un par l'autre (réduction des problèmes engendrés par l'absence de perspective), etc. Toutefois, se basant sur les résultats précédents, Wang et al. (Wang et al., 2011) sont parvenus à améliorer les outils 3D afin de les rendre plus efficaces (en terme de rapidité). En effet, l'utilisation de « vues de profondeur » dans l'environnement 3D permet à l'utilisateur d'interpréter plus aisément la position réelle de l'objet à déplacer et donc permet d'effectuer la tâche plus rapidement, au prix d'un taux d'erreur légèrement plus élevé. De ces études on en dégage que la souris a bel et bien sa place dans la scène 3D, mais qu'elle n'est peut-être pas la plus adaptée à tous les contextes.

Si la souris s'est imposée comme un outil de choix depuis longtemps, d'autres chercheurs ont proposé des outils offrant 3 ou plus degrés de liberté dans l'espoir d'obtenir de meilleures performances, du moins dans certains contextes. Dès 1994, Hinckley et al. (Hinckley et al.,

1994) proposaient d'utiliser des *accessoires passifs d'interface* (« real-world passive interface props ») pour la visualisation de modèles 3D dans le domaine médical. Les chercheurs notaient que les chirurgiens nécessitaient souvent des outils pour visionner des radiographies, mais que ceux-ci n'étaient pas intéressés à utiliser des interfaces complexes ou qui demandent trop de temps. Un problème notable était la visualisation d'un cerveau humain : comment l'utilisateur peut-il manipuler le modèle affiché à l'écran de manière à réduire le nombre d'opérations faites tout en rendant la transformation la plus naturelle possible. Il a donc été proposé d'utiliser un accessoire de tête (« head prop »), qui est un objet réel tenu en main et positionné en 3D par des senseurs, pour offrir à l'utilisateur une interface réelle pour manipuler le cerveau virtuel affiché à l'écran. Hinckley et son équipe ont aussi proposé l'utilisation d'une planche de découpage (« cutting-plane prop ») pour manipuler un plan traversant le cerveau manipulé par le premier accessoire et voir à l'écran sa représentation transversale. Ces deux dispositifs furent un succès parmi les chirurgiens consultés, notant qu'ils comprenaient immédiatement comment utiliser ces outils et qu'ils pouvaient trouver une information avec plus de facilité qu'avec une simple souris. C'était le début des outils à 3 degrés de liberté permettant de faire le lien entre les transformations réelles et virtuelles.

Hinckley poursuivit son analyse en 1997 (Hinckley et al., 1997) avec une étude d'utilisabilité comparant des techniques de rotation d'objets 3D. Ils utilisèrent une souris pour deux méthodes de rotation et un senseur enrobé d'une balle ou dénudé pour les deux autres méthodes. Ils remarquèrent que, pour des tâches impliquant une rotation en 3 degrés de liberté, les interfaces réelles permettaient d'accomplir plus rapidement la transformation que les techniques liées à la souris, sans toutefois sacrifier la précision. On remarque donc que, si la souris pouvait être supérieure aux outils offrant 3 ou plus degrés de liberté quant aux opérations de translation (Bérard et al., 2009), pour les tâches de rotation où il est difficile de transformer un déplacement linéaire de la souris en rotation (ou du moins difficile de l'interpréter pour un humain), ces outils s'avèrent plus efficaces que la souris. Cela soutient l'hypothèse que le meilleur outil dépendra toujours du contexte et que chacun aura ses avantages et ses lacunes.

D'autres chercheurs ont proposé ou évalué des outils conçus pour la CAO en particulier. Fuge et al. (Fuge et al., 2012) développèrent une méthode pour créer et modifier des surfaces 3D qui utilise des gestes de la main en tant qu'entrée à la place d'inputs dits conventionnels (souris, clavier). À l'aide d'un gant faisant le suivi des doigts avec 3 degrés de liberté et d'un visiocasque pour projeter la scène en RA, leur système permet de placer ou déplacer des points qui font partie d'une surface à 3 dimensions. Ils soutiennent qu'une telle méthode d'interaction rendent relativement aisé la tâche de création de surface, celle-ci étant particulièrement complexe et demandant en temps avec les logiciels traditionnels de CAO. De plus, l'intégration de tels outils à un environnement de RA faciliterait la modélisation en enveloppant l'utilisateur dans le monde à éditer et lui permettant donc un contact direct avec les surfaces. Toma et al. (Toma, Gîrbacia et Antonya, 2012) développèrent aussi un système pour effectuer des tâches de CAO. Ils furent plus ambitieux en concevant un outil utilisant des « interfaces multimodales », c'est-à-dire faisant usage d'au moins deux modes d'inputs combinés dans un système multimédia (Oviatt, 2007). Leur système est composé d'un environnement virtuel sous la forme d'une cave où les modèles sont projetés sur les murs et d'une multitude de méthodes d'interaction pour modifier ces modèles : reconnaissance de la voix, gestes des mains et positionnement des mains et de la tête. Il s'agit techniquement d'un environnement de RV car tout est projeté sur des parois statiques et qu'il n'y a pas de contacts directs entre les inputs et les modèles virtuels, mais ce projet est très intéressant car il comporte une étude le comparant à un logiciel traditionnel (clavier & souris) de CAO pour des tâches de modélisation et d'assemblage. Ils notèrent que leur système avait comme avantages une meilleure perception des profondeurs et des contrôles plus intuitifs et naturels, au prix d'exiger plus d'efforts de la part de l'utilisateur. Il fut aussi découvert que, si le système 3D ne semblait pas plus efficace que celui traditionnel pour les tâches de modélisation, ce premier dominait largement quant aux tâches d'assemblage en termes de rapidité.

Tous ces résultats semblent indiquer que certaines tâches de CAO impliquant de la précision, notamment la modélisation de volumes et les translations, ont avantage à être faits avec une

simple souris dans un logiciel traditionnel, tandis que les tâches impliquant une bonne perception de l'environnement ou bien des commandes difficilement traduisibles depuis le 2D, telles que la création de surfaces et les rotations, seraient mieux faites avec des outils 3D et des technologies à RA.

## 1.2 Au-delà de l'écran monoscopique

Si la combinaison de la souris, du clavier et de l'écran 2D s'est imposé comme étant la référence en ce qui a trait aux interfaces utilisateurs dans les 30 dernières années, cela n'a pas empêché un grand nombre de chercheurs et d'entreprises de tenter d'innover en proposant des outils différents. On dénote un succès flagrant des technologies mobiles, depuis les 10 dernières années. Cet aspect est critique à ce rapport de mémoire et sera donc abordé dans une sous-section distincte de cette revue de la littérature : 1.4 L'usage .

### 1.2.1 Écrans 3D

Une des innovations les plus prometteuses par rapport aux interfaces traditionnelles fut de convertir les écrans 2D en écrans 3D. Dans le monde du cinéma ou du jeu vidéo, la possibilité de mieux pouvoir déceler les indices de profondeur a pour effet d'augmenter le degré d'immersion et de généralement obtenir un résultat plus impressionnant. Dans un environnement plus professionnel, le 3D peut servir comme outils aux modélisateurs 3D, par exemple, pour mieux pouvoir constater le résultat de leur travail. L'intérêt pour cette technologie est certain, mais la technique à utiliser demeure encore un chaud sujet de discussion. L'idée principale derrière chaque méthode est la même : on désire émuler la stéréoscopie, c'est-à-dire la disparité entre l'image vue par l'œil droit et l'œil gauche, qui est un des meilleurs indices de profondeur dont disposent les êtres humains. Par exemple, un observateur regardant un objet éloigné aura une image quasi identique de l'objet sur chaque œil, tandis qu'un objet rapproché entraînera deux images avec des disparités notables. Le cerveau humain est habitué à interpréter ces disparités pour déterminer la distance d'un objet observé. Les méthodes suivantes représentent quelques façons d'obtenir un tel résultat.

Le 3D par **anaglyphe** (Beiser, 1981) s'obtient en projetant deux images surimposées vers un observateur muni de lunettes avec un filtre pour chaque œil : un rouge et un cyan. Les filtres servent à éliminer, pour chaque œil, l'image réservée à l'autre œil et a donc pour effet de montrer deux images distinctes à l'observateur, permettant donc la stéréoscopie. Cette méthode est peu coûteuse et relativement simple à implémenter, mais a pour effet de produire des images moins lumineuses et ne respectant pas tout à fait la coloration originale. De plus, les gens ayant un œil dominant ont de la difficulté à constater l'effet 3D. Les **écrans polarisés** (Phillips et Marez, 1987) fonctionnent quasiment de la même façon, en projetant 2 images vers des lunettes munies de filtres. La différence est, qu'au lieu d'utiliser une plage de couleur pour chaque image, celles-ci sont polarisées dans une direction précise. Originalement, les images étaient polarisées linéairement avec un écart de 90 degrés entre elles (par exemple à 45 et 135 degrés), mais cela obligeait l'utilisateur à visionner l'écran avec la tête à la verticale. Cette limitation fut résolue avec l'utilisation de polarisation en sens horaire et antihoraire. Cette méthode a pour avantage d'être accessible à un plus grand nombre de personnes et de ne pas affecter les couleurs, mais requiert un écran particulier et réduit une fois de plus la luminosité des images. Cette dernière limitation peut être solutionnée en faisant usage de **lunettes à obturation (active shutter)** (Lazzaro et al., 1998). Les lunettes bloquent alors l'image un œil à la fois, se synchronisant avec l'écran qui alterne en envoyant à tour de rôle l'image pour chaque œil non obstrué. Le concept original fait donc usage de chaque image non modifiée pour la présenter à l'utilisateur, ce qui entraîne une image de meilleure qualité que pour les techniques précédentes. Le prix à payer pour cette méthode est qu'il est nécessaire d'utiliser des lunettes plus lourdes (car munies d'un dispositif électronique) qui doivent être alimentées électriquement, soit par piles ou en étant connecté au téléviseur. De plus, le fait de faire alterner les images a pour effet de couper de moitié le nombre d'images réelles par seconde, à la manière de l'entrelacement. Finalement, il existe des techniques ne nécessitant pas l'usage de lunettes. **L'auto-stéréoscopie** (Grossmann, 2001), par exemple, fait usage d'un écran composé de fines bandes qui projettent chacune une partie de l'image dans une direction précise. Un observateur placé dans un angle étroit, perpendiculaire à l'écran, pourra observer, pour chaque œil, uniquement des bandes affichant l'image réservée à cet œil. On obtient donc un effet stéréoscopique car il

devient possible d'afficher une image différente pour chaque œil. Les problèmes les plus majeurs sont la nécessité d'utiliser un écran spécifique pour le visionnement, ainsi que le fait de devoir se placer dans un angle prédéfini. Certaines avancées tentent de contourner ce dernier problème, par exemple en offrant plusieurs angles possibles ou bien en choisissant l'angle de projection en fonction d'un système de détection d'observateur. Dans tous les cas, le nombre de personnes pouvant profiter de l'effet 3D est limité et ne peut pas être compensé par un plus grand nombre de lunettes.

### 1.2.2 Écrans interactifs et projection spatiale

Si les écrans stéréoscopiques augmentent l'immersion, ils ne sont pas suffisants pour amplifier les techniques d'interaction. Des extensions à ces écrans, ou bien des outils complètement nouveaux sont alors conçus pour permettre à un utilisateur d'interagir avec le monde virtuel sans devoir se limiter à une souris et avec le minimum de matériel requis, en dehors du dispositif d'affichage.

Il est possible de classifier certains de ces systèmes. D'abord, on retrouve **les dispositifs où les mains sont placées devant la scène**, c'est-à-dire entre l'interface d'affichage et les yeux de l'utilisateur. Grossman et al. développèrent ce qu'ils nomment des « écrans volumétriques 3D » (3D Volumetric Displays) (Grossman, Wigdor et Balakrishnan, 2004). Leur système est en quelque sorte une large sphère à l'intérieur de laquelle des hologrammes représentent une scène 3D. Un utilisateur peut se placer à sa convenance près de cette sphère et interagir avec la scène à l'aide de ses mains. Ses doigts sont suivis grâce à des caméras placées tout autour du dispositif. Les chercheurs simulèrent donc une surface multi-tactile pour laquelle il est possible de produire des techniques d'interaction faisant usage des doigts survolant la surface de la sphère. Le fait que l'écran volumétrique peut être utilisé de n'importe quel côté et ne nécessite pas de lunettes ou bien d'autres dispositifs portés sur soi pour visionner le contenu semble indiquer que ces systèmes seraient adaptés au travail collaboratif. En effet, plusieurs personnes pourraient interagir simultanément avec une scène 3D. Un autre système d'écran interactif pour lequel les mains sont placées devant l'interface a été conçu par Bogdan et al. :

HybridSpace (Bogdan, Grossman et Fitzmaurice, 2014). Celui-ci fait aussi usage de vision pour détecter les mains et la gestuelle, mais fonctionne cette fois avec un écran plat permettant le 3D avec des lunettes polarisées. L'innovation de ce dispositif, au-delà des techniques d'interaction intéressantes proposées pour interagir avec une scène 3D avec ses mains, est que l'utilisateur peut faire basculer l'affichage de 2D à 3D. En effet, les auteurs soutiennent que, même pour une scène 3D, certaines tâches s'appliquent mieux lorsqu'elles sont réalisées avec un affichage 2D traditionnel, alors que d'autres ont intérêt à être vues en stéréoscopie. L'usager peut donc basculer entre les deux modes, mais le système peut de lui-même choisir le mode adapté, utilisant des transitions pour faciliter le transfert cognitif pour l'humain.

Une autre façon de permettre l'interaction entre les mains et l'écran est de **placer les mains derrière le dispositif d'affichage**. Le système Toucheo, conçu par Hachet et al. (Hachet et al., 2011), en est un exemple. Une paroi transparente servant d'écran est placée à l'horizontale devant l'usager et celui-ci place ses mains sur une paroi tactile placée aussi à l'horizontale, sous l'écran. Une scène 3D est alors projetée sur l'écran et semble apparaître sur le même plan où se trouvent les mains de l'utilisateur. Ce dernier peut alors faire usage de la paroi tactile (qui est elle-même un écran 2D pouvant afficher des menus) pour interagir avec les objets de la scène. En plaçant les mains derrière l'écran, celles-ci se retrouvent en « contact » avec le monde 3D et il peut donc y avoir des manipulations directes, contrairement aux dispositifs présentés précédemment qui fonctionnaient par manipulations indirectes. Il s'en trouve aussi que les mains n'obstruent pas la vision de la scène 3D, permettant des opérations complexes sans jamais cacher le moindre détail. Bien sûr cela peut être quelque peu troublant, puisque l'occlusion n'est pas respectée : une main placée dans le monde virtuel devrait pouvoir cacher une partie des objets qui s'y trouvent lorsqu'elle entre en contact apparent avec ceux-ci. Un système similaire, SpaceTop, produit par Lee et al. (Lee et al., 2013), fait usage d'un écran ressemblant un peu plus aux écrans traditionnels. Au lieu d'utiliser une série d'écrans placés à l'horizontale, SpaceTop est constitué d'un écran transparent placé en angle devant une surface horizontale (incluant un clavier) où placer ses mains. Une caméra placée au-dessus de cet assemblage permet de faire le suivi des mains. Le

combo écran-clavier permet donc à un utilisateur de travailler comme à son habitude, mais aussi de faire usage de ses mains pour contrôler des éléments 2D. Et lorsque l'écran affiche des éléments d'une scène 3D, l'usager peut les contrôler « directement » avec ses mains. La combinaison de technologies 2D et 3D pourrait permettre le meilleur des deux mondes selon les auteurs. Hürst et al. ont aussi pu prouver qu'il n'était pas nécessaire d'avoir accès à des dispositifs aussi complexes pour avoir ses mains derrière le rendu 3D (Hürst et Van Wezel, 2013). En effet, afin de comparer plusieurs techniques d'interaction, ils développèrent un prototype de « sélection d'objets en l'air » (Midair finger selection). Dans leur système, un téléphone intelligent est tenu d'une main et affiche sur son écran ce que voit sa caméra, augmenté d'une scène 3D. L'autre main est libre d'interagir avec cette scène. Il y a donc un contact direct entre la main et les objets à manipuler. Toutefois, l'utilisation d'une simple caméra RGB de téléphone, tenue en main, limite les algorithmes de vision et donc leur système se contente de permettre une interaction sur un plan 2D. Des développements futurs pourraient toutefois retirer cette limitation et obtenir un résultat aussi précis que celui des dispositifs précédents.

Finalement, on peut identifier des dispositifs pour lesquels **un écran est augmenté par l'utilisation de tablettes**. Pour leur projet de recherche, Spindler et al. ont fait usage de ce qu'ils nomment des « écrans tangibles » (tangible displays) (Spindler et al., 2014), c'est-à-dire des dispositifs conscients de leur emplacement dans l'espace et tenus en main, pour visualiser et manipuler des données. Ils utilisent un large écran placé à l'horizontal sous forme de table pour afficher l'ensemble d'une scène virtuelle. Avec des tablettes, plusieurs utilisateurs peuvent avoir leur propre fenêtre vers cette scène pour obtenir plus d'informations ou effectuer des modifications. L'orientation et la position de la tablette sont utilisées pour déterminer quelle section de la scène est d'intérêt pour l'usager et celui-ci peut utiliser les contrôles tactiles de sa tablette pour éditer le contenu. L'avantage majeur d'un tel système est au niveau de la collaboration possible entre un grand nombre d'utilisateurs. Comme chacun peut avoir ses propres contrôles liés à une unique scène virtuelle, cette dernière peut être éditée de plusieurs façons en même temps. Dorta et al. eurent une idée similaire avec Hyve-3D (Dorta, Kinayoglu et Hoffmann, 2015). Ils poussèrent un peu plus

l'utilité des tablettes en les liant à des curseurs 3D dans une scène virtuelle dans un contexte de CAO. Au lieu de présenter cette scène sur une table, ils ont implémenté un environnement virtuel de type CAVE. Chaque utilisateur peut ainsi s'immerger dans le monde à éditer et apporter ses modifications à l'aide de sa tablette. Une seconde version de leur système, nommé « non immersif », permet d'utiliser une tablette avec un simple écran de laptop au lieu d'un environnement de RV. Dans les deux projets précédents, les auteurs notent l'intérêt majeur d'utiliser des dispositifs réels pour manipuler les objets virtuels : cette interface est intuitive, précise et permet d'utiliser l'écran tactile pour afficher des informations contextuelles ou entrer des commandes.

### 1.2.3 Le Spectrum de la réalité mixte

Pour bien saisir la différence entre Réalité Virtuelle (RV), Réalité Augmentée (RA) et autres termes de ce genre, il faut définir la Réalité Mixte (RM). La RM est tout système faisant un lien, à travers des interfaces d'input et d'output, entre le monde réel et un monde virtuel. Bien sûr chaque système peut décider d'utiliser plus ou moins de chaque monde et les interfaces varient beaucoup. C'est pour cela qu'on parle d'un Spectrum de la RM, représenté à la Figure 1.1 (Electricbadger, 2007).

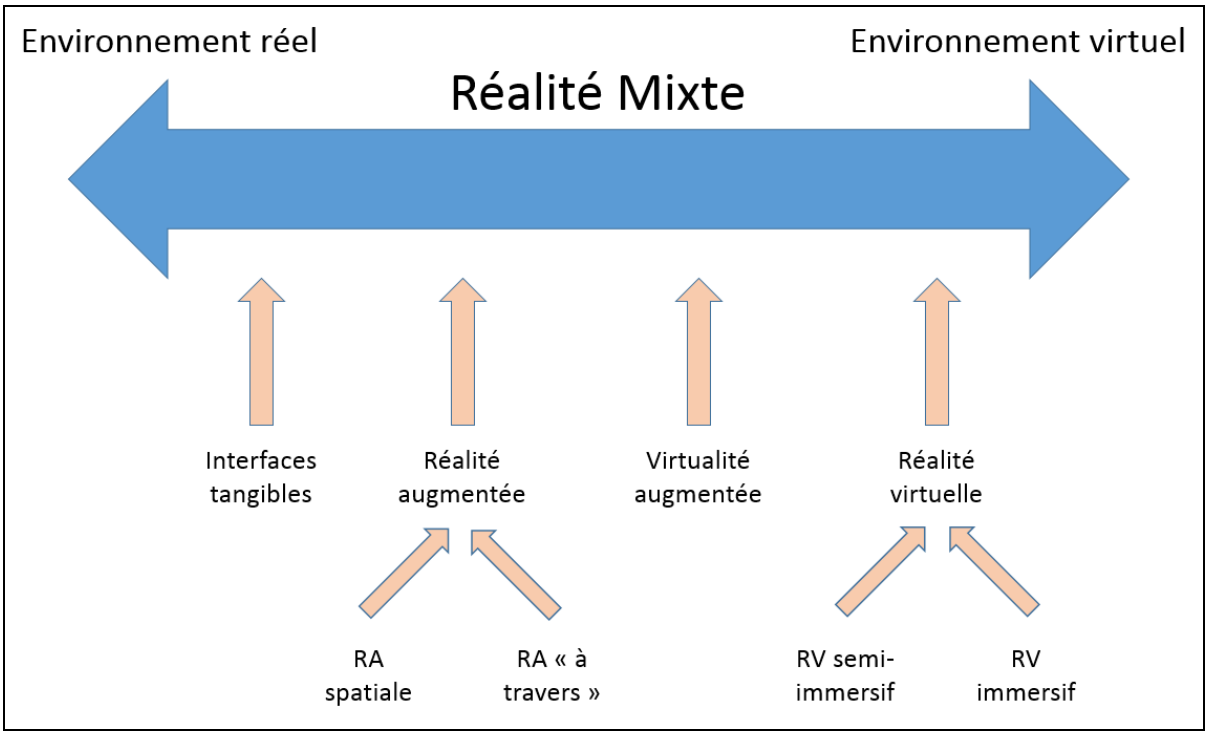


Figure 1.1 Spectrum de la Réalité Mixte, inspiré de (Electricbadger, 2007)

Du côté de l'environnement réel, on parle d'**interfaces tangibles** lorsqu'on utilise des objets physiques pour représenter et interagir avec des données virtuelles. Les travaux de Hinckley (Hinckley et al., 1994) entrent dans cette catégorie car il utilisait des têtes de poupées (objet physique) pour contrôler un modèle de cerveau (données virtuelles) représenté à l'écran. Cet espace du spectre est celui qui représente généralement les techniques d'interaction les plus naturelles, car il s'agit de directement manipuler des objets réels, mais qui offre le moins d'immersion, car le monde virtuel n'est vu qu'à travers l'écran d'ordinateur et les contrôles sont indirects. De plus, les applications sont souvent limitées car il ne peut y avoir de contrôles physiques parfaitement adaptés à tous les cas. Par exemple, la tête de poupée ne conviendrait peut-être pas pour manipuler un modèle d'avion. Il faut parfois utiliser des objets plus génériques.

Utilisant un peu plus du monde virtuel pour le combiner au monde réel, on retrouve la **réalité augmentée**. Par définition, la RA consiste à prendre le monde réel (par exemple une pièce filmée par une caméra) et à l'augmenter de détails virtuels (par exemple en ajoutant à la pièce

un meuble). La RA se divise elle-même en deux sous-catégories : la RA spatiale et la RA « à travers » (See-through). Cette première projette directement les objets virtuels dans le monde réel, souvent sous forme d'hologrammes. Les possibilités sont immenses, mais la technologie fait actuellement défaut à ce sujet. L'autre type, la RA « à travers » consiste à additionner une image à une projection d'une vraie scène, soit à travers une surface transparente (optique) ou à travers un traitement vidéo (vidéo). L'utilisation du système est alors réservée aux utilisateurs se trouvant dans l'orientation de l'écran, ou ayant sur eux le dispositif nécessaire. Cela peut paraître plus limitatif que la RA spatiale, mais les technologies actuelles sont beaucoup plus étendues sur celle-ci. Il existe plusieurs façons d'obtenir une RA « à travers » : avec un écran et une caméra (vidéo), avec un dispositif tenu en main muni d'une caméra (vidéo), avec des lunettes (optique) ou avec des visiocasques (optique ou vidéo). Des exemples de lunettes et de visiocasques seront présentés au sous-chapitre 1.2.5.

La **virtualité augmentée** se caractérise par le fait d'injecter des éléments réels dans une scène virtuelle, soit l'opposé de la RA. Regenbrecht et al. présentent un exemple intéressant de cette idée en prototypant un système de vidéoconférence (Regenbrecht et al., 2003). La conférence se donne dans une salle complètement virtuelle, mais les participants sont représentés par leur visage filmé par leur ordinateur personnel, donnant l'impression que les personnes sont toutes dans la même pièce. Un autre exemple serait d'utiliser une caméra pour balayer un objet réel et ainsi en obtenir un modèle virtuel pouvant être placé dans une scène 3D.

Pour finir, la **réalité virtuelle** consiste à faire interagir l'utilisateur avec un environnement complètement virtuel. On dit que c'est une RV semi-immersive lorsque le contact avec le monde virtuel se limite à un point d'entrée précis. Par exemple, un jeu vidéo traditionnel serait semi-immersif car le joueur accède au monde du jeu à travers un écran, mais rien d'autre. Il n'a pas l'impression d'être au cœur du jeu. La RV immersive, au contraire, est ce à quoi on pense généralement lorsqu'on parle de RV. En utilisant un visiocasque ou un système de projection remplissant le champ de vue de l'utilisateur, on plonge celui-ci dans l'univers virtuel. Il y a alors un détachement complet de la réalité, à l'exception des

interactions de l'usager. Le sous-chapitre 1.2.4 portera sur des exemples de dispositifs de RV.

### 1.2.4 Réalité virtuelle

Cette section portera plus particulièrement sur des exemples de visiocasques permettant la RV immersive, mais il demeure intéressant de présenter quelques exemples de technologies de RV se basant sur d'autres concepts. Les **dispositifs de RV non-immersifs** ont comme objectif de fournir un point de contact vers un monde complètement virtuel à un utilisateur, sans pour autant tenter de modifier l'environnement réel ou de le masquer. De cette définition, n'importe quel écran affichant une scène 3D virtuelle, par exemple un jeu vidéo, peut être considéré comme un dispositif de RV non-immersif. Mais pour citer un exemple de la littérature, Drettakis et al. ont fait usage d'un *Barco BARON workbench* pour offrir le point de contact vers le monde virtuel (Drettakis et al., 2004). Un utilisateur placé debout devant un large écran en angle peut visualiser et naviguer à travers des ruines ou des plans architecturaux précédemment modélisés à partir de données réelles, en plus de pouvoir y apporter certaines modifications. Dans ce cas on voit bien que seul l'angle de vision englobant l'écran permet d'interfacer avec le monde virtuel et que si l'utilisateur devait tourner la tête, il ne verrait plus que la salle réelle où il se trouve, faisant bel et bien de ce système un dispositif non-immersif.

Passant ensuite au monde des **dispositifs à RV immersifs ne faisant pas usage de visiocasques**, on dénote une technique qui a fait ses preuves et qui est désormais une référence du domaine : la CAVE de Cruz-Neira et al. (Cruz-Neira, Sandin et DeFanti, 1993). Signifiant « CAVE Automatic Virtual Environment », la CAVE est, comme son acronyme le suggère un dispositif fermé où se situe un ou plusieurs utilisateurs. Ceux-ci sont alors au centre d'une scène virtuelle projetée sur les parois de la salle. Il s'agit donc d'un environnement de RV complètement immersif et ne nécessitant pas de visiocasques, bien qu'on puisse l'augmenter de lunettes 3D. Les avantages d'un tel système sont significatifs : les projections d'image est une technologie mature et il est donc possible d'obtenir une image

relativement fidèle, le champ de vision n'est aucunement limité par un dispositif, il n'est pas nécessaire de porter de lourds appareils et, si la CAVE est suffisamment grande, il est possible d'avoir plusieurs utilisateurs en simultané. Toutefois, un tel système n'est pas sans limitations : le dispositif est aussi large qu'une pièce et son coût significatif en font un appareil inaccessible aux particuliers. Il n'en demeure pas moins que la CAVE fut employée à maintes reprises depuis sa conception afin de développer et tester des techniques d'interaction ou systèmes plus complexes, sans les limitations d'appareils comme les visiocasques (Toma, Gîrbacia et Antonya, 2012).

Passant ensuite au monde des **visiocasques à RV (immersifs)**, il serait inapproprié d'omettre de parler du Oculus Rift (Oculus, 2012). Ce dernier, qui a la particularité d'avoir été initialement financé à travers une campagne publique sur le site web *Kickstarter* (PBC, 2015), est considéré comme étant le « premier visiocasque à RV professionnel [grand marché] » (Gizmag, 2015). Le projet se démarqua en étant supporté par John Carmack (Wikipedia, 2015), un géant de l'infographie mieux connu pour son rôle dans la compagnie *Id Software,* qui devint en 2013 le CTO de la compagnie *Oculus VR*. Le dispositif lui-même a une résolution de 1080x1200 par œil, un taux de rafraichissement de 90 Hz et un champ de vision de 110 degrés. Le visiocasque est suivi avec 6 degrés de liberté et se branche directement dans un ordinateur Windows en tant que second écran. Un des objectifs principaux du Oculus Rift était d'en faire un accessoire de jeu vidéo et sa grande popularité a entraîné le développement d'une multitude de jeux et même la naissance d'un marché. Il n'en demeure pas moins qu'un dispositif d'une telle qualité est aussi intéressant pour des systèmes autres que les jeux vidéo. Des développeurs se sont déjà penchés sur l'idée d'utiliser un Oculus Rift pour augmenter un simulateur de vol, pour aider des patients avec des phobies à les affronter dans un environnement virtuel, et même pour la modélisation 3D (Krs, 2014).

La popularisation du Oculus Rift en 2012 tourna une page de l'histoire pour la RV. Celle-ci devenait finalement, après des décennies de développement et de suppositions, une technologie viable pour laquelle existait un marché. Il n'est donc pas surprenant que plusieurs géants de l'électronique et du jeu vidéo se penchent à leur tour sur le sujet. *Sony*

annonça en 2014 le Project Morpheus (Sony, 2014a), un visiocasque conçu pour la *PlayStation*. Avec une résolution de 960x1080 par œil, un taux de rafraichissement de 120 Hz, un champ de vision de 100 degrés et un suivi en 6 degrés de liberté, le dispositif s'annonce compétitif au Oculus Rift, bien qu'ils sont conçus pour des plateformes distinctes. Sur le marché du PC, le HTC Vive fut annoncé à la suite d'un partenariat entre *HTC* et *Valve*, le géant du jeu vidéo responsable de la populaire plateforme de jeux vidéo *Steam* (HTC, 2015). D'une résolution de 1080x1200 par œil, un rafraichissement de 90 Hz, un champ de vision de 110 degrés et un suivi en 6 degrés de liberté dans un espace de 15 pieds par 15 pieds, le dispositif sera appuyé par ses compagnies mères qui ont déjà une part de marché importante dans le monde du jeu vidéo et pourrait donc surpasser le Oculus Rift. Dans le marché du téléphone intelligent, *Samsung* développe actuellement le Samsung Gear VR, un dispositif utilisant un *Galaxy Note 4* ou *Galaxy S6/S6 Edge* pour l'affichage et le traitement (Samsung, 2015). Les lunettes elle-même contiennent les lentilles augmentant le champ de vision ainsi que des capteurs permettant le suivi de rotation en 3 degrés de liberté. Un tel dispositif n'est peut-être pas comparable en terme de performances aux appareils présentés précédemment, mais offre tout de même une option aux amateurs de jeux vidéo dits plus « casuels », d'une part avec sa fusion avec un téléphone intelligent, et d'une autre avec son prix d'environ 100$US. À noter que tous ces dispositifs sont prévus d'être sur le marché avant avril 2016 : il s'agit d'une époque excitante pour ceux qui s'intéressent à la RV.

### 1.2.5 Réalité augmentée

Tel que mentionné précédemment, il existe plusieurs façons d'obtenir une réalité augmentée : des affichages spatial, des affichages « à travers » optiques (visiocasque ou lunettes), des affichages « à travers » vidéo (visiocasque ou appareils mobiles). Ici nous nous pencherons uniquement sur une sélection de dispositifs marquants de type visiocasques ou lunettes, car il s'agit d'une part de ceux qui ont le plus évolués récemment, mais aussi car ils ont un lien plus évident avec le sujet de ce mémoire. Il est aussi important de noter que, bien que la littérature fasse actuellement une distinction entre les lunettes à RA et les visiocasques, celle-ci est plutôt arbitraire. Effectivement, les lunettes auront tendance à être moins lourdes et

encombrantes, à moins limiter la vision périphérique et à être plus socialement acceptable. Les visiocasques, quant à eux, sont généralement plus immersifs, offrent jusqu'à 6 degrés de liberté, disposent d'un espace d'affichage plus grand et profitent d'une puissance de traitement plus élevée. En somme il s'agit surtout de différence d'ampleur et des avancements quant à la miniaturisation des technologies impliquées en viendront à fusionner ces deux catégories.

Débutons donc par deux exemples de lunettes. Les Google Glass (Google, 2013) (Figure 1.2) furent la tentative du géant Google de percer dans le monde de la RA. Offrant plusieurs des outils Google (cartes, recherche, traduction, etc.) dans un appareil léger (43g) et compact, il était espéré que cet outil devienne monnaie courante dans la vie de tous les jours. Fait notable, il était possible d'interagir avec les lunettes à partir d'une surface tactile sur le côté, permettant donc des actions sans ambiguïté. Le projet est présentement arrêté, mais des rumeurs soutiennent que Google travaillerait sur une version plus avancée. *Sony* répondit à cet appareil avec leur Sony SmartEyeglass (Sony, 2014b), des lunettes avec une plus grande surface d'affichage au profit d'un poids plus grand (77g). Faits notables, ces lunettes disposent d'un micro et de reconnaissance de la voix, et l'ordinateur qui les contrôle est connecté par un câble et se tient dans une poche, offrant potentiellement plus de puissance que les Google Glass, au coût d'être un peu plus encombrant.

Figure 1.2 Les lunettes à RA Google Glass

S'approchant désormais plus des visiocasques à proprement parler, penchons-nous sur un exemple d'appareil effectuant un rendu vidéo pour la RA. Les Vuzix Wrap 1200DXAR (Vuzix, 2014) sont le plus récent effort de la compagnie *Vuzix* dans le monde de la RA. L'avantage premier de faire la RA à travers un rendu vidéo pour ensuite l'afficher devant les yeux de l'utilisateur est qu'on obtient généralement une meilleure qualité d'image ainsi qu'une fidélité des couleurs. En effet, les technologies optiques font usage de lentilles qui peuvent distordre la lumière et les couleurs, et les parois dites transparentes ne le sont jamais entièrement : une part non négligeable de la lumière est perdue. Les désavantages de faire un rendu vidéo sont que la scène réelle augmentée sera vue uniquement à travers la caméra, occasionnant donc une perte de qualité due aux limitations en résolution, en plus d'obstruer le champ de vision périphérique. Dans le cas des Vuzix Wrap 1200DXAR, une partie du champ de vision périphérique est accessible avec des parois transparentes, mais il y a plusieurs angles morts entre la zone affichée par les écrans et celle pouvant être vue par ces parois. En somme l'outil, qui se branche à un ordinateur pour profiter de sa puissance, est un compromis intéressant entre les lunettes légères pour usage de tous les jours et les gros visiocasques offrant plus de possibilités et de performance.

Un appareil des plus intéressants fut annoncé au cours de ce projet de mémoire : le HoloLens de Microsoft (Microsoft, 2015). Il s'agit d'un visiocasque muni d'un ordinateur intégré faisant la projection du contenu virtuel sur des lentilles transparentes. Là où l'outil devient innovateur est lorsqu'on se penche sur les contrôles disponibles : en plus de la détection des mains par une caméra, qu'on retrouve sur quasiment tous les visiocasques, et du contrôle par la voix qui, bien que pertinent, n'est pas nouveau, les HoloLens sont munies de caméras pointées vers l'intérieur du casque, suivant les yeux de l'utilisateur. Il est ainsi possible d'effectuer des commandes avec uniquement les yeux, dédoublant les possibilités d'interaction avec le système. Le champ de vision, actuellement estimé à 35 degrés, est parmi les plus grands qu'on retrouve dans les technologies de RA et le visiocasque permet aussi le 3D. Son plus grand défaut est probablement son poids (400g), mais celui-ci est plus petit que celui du Oculus Rift. Mais l'avantage principal des HoloLens ne vient pas de ses composants,

mais plutôt de son développeur. Microsoft est connu pour son système d'exploitation Windows et il a été mentionné plusieurs fois que la dernière itération, Windows 10, inclura des nouvelles méthodes d'interaction, notamment avec les HoloLens. Plusieurs exemples ont déjà été prototypés et présentés par la compagnie : visualisation d'environnements réels, immersion dans le jeu 3D Minecraft et même modélisation 3D. Il est envisageable qu'un assez grand investissement en technologies et support de la part de Microsoft suffise à ouvrir le marché de la RA tout comme l'a fait *Oculus VR* avec la RV. Somme toute, il s'agit probablement de l'appareil le plus prometteur actuellement, mais il est toujours en développement et donc on ne peut se baser que sur des démonstrations et des projections.

Finalement, on peut approcher le visiocasque META, de la compagnie du même nom (META, 2014). S'appelant originalement Spaceglasses, l'appareil offre des spécifications intéressantes par rapport à la compétition : projection sur lentilles transparentes, résolution de 960 par 540, champ de vision allant jusqu'à 40 degrés pour la version professionnelle (non disponible actuellement), un poids de 330g. Ce qui démarque surtout ce visiocasque des autres est qu'il est conçu pour être branché à un ordinateur, servant d'outil à celui-ci et non cherchant à le remplacer. Un ensemble de caméra RGB et infrarouge pour la profondeur placé sur le dessus du casque permet d'effectuer un suivi relativement fidèle des mains et l'utilisateur peut profiter d'un affichage 3D. Les limitations de l'appareil sont surtout au niveau du champ de vision : si la compagnie annonce un champ de vision de 40 degrés, celui-ci ne serait que pour la version professionnelle qui n'est pas encore sur le marché. La version développeur se limite à 35 degrés, et ce avec des lentilles masquant entièrement la vision périphérique. Sans ces lentilles, le champ tombe à 23 degrés, ce qui est très peu. Tout de même, les lunettes META demeurent un appareil intéressant du simple fait qu'il est disponible et qu'il est plutôt ambivalent dans un contexte d'interaction avec l'ordinateur.

Le Tableau 1.1 présenté plus bas compare en détails les spécifications des dispositifs à RA énumérés précédemment. À noter que, comme plusieurs de ces appareils ne sont pas encore sur le marché, certains détails sont pour le moment estimés ou carrément inconnus.

Tableau 1.1 Comparaison de plusieurs dispositifs de réalité augmentée

| | **Google Glass** | **Sony Smart-Eyeglass** | **Vuzix Wrap 1200DXAR** | **Microsoft HoloLens** | **META (dev)** | **META (pro)** |
|---|---|---|---|---|---|---|
| **Ordinateur** | Embarqué | Attaché | Externe | Embarqué | Externe | Externe |
| **Affichage** | Projection sur un œil | Affichage monochrome sur surface à 85% transparent | Affichage vidéo | Projection sur lentilles transparentes | Projection sur lentilles transparentes | Projection sur lentilles transparentes |
| **Résolution** | 640x360 | 419x138 | 852x480 par œil | Inconnu | 960x540 | 1280x720 |
| **Contrôles** | Surface tactile sur le côté, détection des mains par vision | Contrôle par voix, détection des mains par vision | Détection des mains par vision | Suivi des yeux, contrôle par voix, détection des mains par vision | Ordinateur, détection des mains par vision | Ordinateur, détection des mains par vision |
| **Champ de vision** | 14° | 20° | 35° | 35° | 23 à 35° | 40° |
| **Poids** | 43g | 77g | Inconnu | 400g | 330g | 180g |
| **Degrés de liberté** | 3 (limités) | 3 | 3 | 3 | 3 | 3 |
| **3D ?** | Non | Non | Oui | Oui | Oui | Oui |

## 1.3 La CAO et la Réalité Augmentée/Virtuelle

Pour comprendre le lien qu'on voit de plus en plus entre la CAO et les technologies à RA ou RV, il faut revenir aux avantages inhérents de ces systèmes. Le fait d'avoir une perspective associée à la tête (c'est-à-dire qui répond aux mouvements de celle-ci) et stéréoscopique permet d'interpréter plus rapidement des scènes 3D (Ware et Franck, 1996). Les technologies à RA sont souvent couplées à des systèmes de vision faisant le suivi des mains ou bien des capteurs de mouvements. Ce suivi permet d'obtenir des techniques d'interaction à 3 ou plus degrés de liberté (Sharp et al., 2015) et il a été démontré que des tâches d'assemblage pouvaient bénéficier de contrôles à 6 degrés de liberté (McMahan et al., 2006). Finalement, les techniques d'interaction de ces systèmes ont tendance à être plus intuitives et naturelles pour un novice.

Or, la CAO exige souvent un minimum d'immersion dans le système pour bien saisir l'environnement et les modèles qui le composent, et les logiciels traditionnels manquent cruellement de façons intuitives pour explorer ce monde (Toma, Gîrbacia et Antonya, 2012). Fuge explique que, si la précision atteinte par les logiciels traditionnels de CAO est utile pour la finalisation d'un projet, ceux-ci ont tendance à limiter la phase de conception avec des interfaces complexes qui bloquent la créativité (Fuge et al., 2012). Finalement, il faut se souvenir que les deux tâches les plus communes dans un logiciel de CAO sont la modélisation et l'assemblage et que celles-ci peuvent occuper chacune la majorité du travail à faire selon le contexte.

Le lien devient alors évident : l'immersion requise par la CAO est offerte par les systèmes à RA, ceux-ci se composent souvent de contrôles intuitifs qui pourraient faciliter la phase initiale de conception d'un travail de CAO, et les tâches d'assemblage, qui sont un point important de la CAO, se font mieux dans un environnement de RA. De ces faits, les tâches de CAO sont tout à fait appropriées pour tester une technique d'interaction faisant usage de la RA et il est normal de bâtir un système entier uniquement sur la CAO, puisqu'il semble y avoir tant à gagner. Des lecteurs attentifs remarqueront toutefois que ces technologies, autant utiles paraissent-elles, ne répondent pas à toutes les exigences d'un système de CAO, notamment la précision et le contrôle lors de la modélisation. Plusieurs auteurs suggèrent donc d'exporter le travail effectué dans leur système de RA vers un logiciel traditionnel de CAO pour profiter des avantages de la souris et du clavier (Fuge et al., 2012).

Ainsi, tel qu'abordé brièvement dans les sous-chapitres précédents, plusieurs auteurs se sont penchés sur la conception et le développement de systèmes à RA ou RV ayant pour but d'accomplir des tâches normalement réservées à des logiciels de CAO. Butterworth et al. (Butterworth et al., 1992) furent parmi les premiers, en 1992, à proposer un logiciel permettant l'édition de modèles 3D avec un visiocasque à RA. Leur produit, nommé 3DM, permet, à l'aide d'un visiocasque suivi par capteur et d'un contrôleur tenu en main lui aussi suivi avec 6 degrés de liberté, de contrôler « directement » les objets du monde virtuel. On entend, par contact « direct », que l'utilisateur peut faire une correspondance un-pour-un

entre le monde virtuel, incluant les modèles, et le monde réel, incluant sa main. En d'autres termes, l'usager a l'impression de pouvoir saisir les objets. Les auteurs introduisirent plusieurs concepts intéressants en lien avec la CAO par RA, tels que la nécessité de pouvoir manipuler la scène pour atteindre un objet éloigné, de pouvoir ajuster la taille de la scène et l'usage de marqueurs virtuels pour indiquer d'avance le résultat d'une commande. Ils conclurent que 3DM était un outil adapté à l'élaboration d'ébauches, permettant un travail naturel et rapide, mais qu'il était limité en ce qui a trait au travail de précision. Le système ne remplaçait donc pas entièrement les logiciels traditionnels de CAO.

Fuge et al. (Fuge et al., 2012) se démarquèrent en se concentrant sur la modélisation de surfaces. Cette tâche est selon eux très complexe et aurait avantage à être accomplie à l'aide de gestes naturels. Ils utilisèrent donc un gant muni d'un senseur de force permettant de déterminer la contraction de la main. Chaque bout de doigt fut aussi colorié d'une couleur différente afin de pouvoir être suivi dans l'espace par vision. Les auteurs précisent que leur objectif était de concevoir et de prototyper des méthodes d'interaction avec la gestuelle, et que l'utilisation d'un visiocasque à RA n'était qu'un artéfact nécessaire au système de vision. Toutefois, ils constatent que l'immersion joua un rôle crucial dans leur projet : le contrôle de surfaces « direct » avec la main s'avéra très utile et peut être un des plus grands facteurs de succès du système en entier. De plus, leur prototype introduisit la capacité de sauter directement d'un mode point à point (c'est-à-dire d'un mode où les surfaces sont représentées sous forme de points) à un mode surface (c'est-à-dire d'un mode où les surfaces ont une texture). Ce premier mode permet une plus grande liberté lors de l'édition de la surface alors que le second mode permet à l'utilisateur de constater plus aisément l'apparence réelle de la surface et d'en faire des modifications plus détaillées. Les conclusions de ces travaux sont que l'environnement de RA permet une immersion plus grande dans l'univers à éditer, que la gestuelle peut être appropriée à certaines tâches de CAO et que le fait d'offrir à l'utilisateur plusieurs méthodes d'interaction ou de représentation de l'information permet plus de flexibilité et, au final, de meilleures performances.

Au même moment, Toma et al. (Toma, Gîrbacia et Antonya, 2012) développaient ce qu'ils nomment un système multimodal immersif de CAO. Le terme « immersif » provient du fait qu'ils font usage d'une CAVE, c'est-à-dire d'un environnement de RV où l'utilisateur est entouré d'écrans où est projetée la scène virtuelle (Cruz-Neira, Sandin et DeFanti, 1993), tandis que le terme « multimodal » (Oviatt, 2007) est employé car leur prototype emploie plusieurs méthodes d'input différentes : reconnaissance de la voix, gestuelle, suivi de la tête et des mains. Les motivations des auteurs étaient que la majorité des applications actuelles de CAO avec des outils de VR se contentaient de permettre le visionnement de modèles 3D conçus par d'autres logiciels, sans permettre l'édition, et que même celles qui permettaient les modifications manquaient de rigueur et n'avaient pas eu d'impacts significatifs sur le monde de la CAO. Toma et son équipe espéraient donc pouvoir comparer leur système avec un logiciel traditionnel, impliquant souris, clavier et écran 2D. Leur problématique pouvait se résumer à la question suivante : « which interface of a VR CAD system is the most intuitive and natural interface in the design of 3D CAD parts? ». Un fait intéressant de leur système est qu'ils décidèrent d'utiliser un logiciel existant de CAO (SolidWorks) et de l'augmenter avec leurs interfaces multimodales. Ils passent très près de la possibilité de combiner dans le même système les interfaces traditionnelles et les interfaces de RA/RV, mais ne s'y attardent finalement pas. Leur article omet malheureusement d'expliciter en détail toutes les méthodes d'interaction, mais on peut comprendre que leur système permet à un utilisateur de créer des primitives, d'effectuer des extrusions pour modeler un objet et d'assembler plusieurs objets ensemble. Malgré la quantité impressionnante de méthodes d'input, le système semble se limiter à des opérations relativement simples, laissant de côté des tâches comme les rotations et la création de surfaces. Leurs conclusions sont qu'un tel système comporte des avantages quant aux opérations d'assemblage, mais que celui-ci est beaucoup plus demandant envers les utilisateurs. L'amplitude des gestes demandés à l'usager est un problème qu'il faudra résoudre avant de voir de telles méthodes dans un logiciel de CAO commercial.

Une équipe de recherche affiliée à *Autodesk* a conçu le système *HybridSpace* afin de prototyper l'idée de pouvoir basculer entre un affichage 2D et 3D et d'effectuer des tâches d'assemblage à l'aide de gestuelle. Ces travaux sortent quelque peu du cadre de la RA et de

la RV car tout est affiché sur un écran, mais ils demeurent intéressants pour ce mémoire car les auteurs combinent des interactions classiques de CAO (écran 2D) avec des interfaces souvent représentées dans des systèmes de RA (affichage 3D, contrôle par gestuelle). Contrairement aux travaux présentés précédemment, ce système permet à l'utilisateur d'interagir avec les objets virtuels avec ses mains, mais de façon indirecte. C'est-à-dire qu'au lieu d'avoir l'impression de toucher aux objets, l'utilisateur effectue des contrôles indirects avec sa main. Cela entraîne une moins grande immersion, mais pourrait toutefois résoudre la problématique des contrôles exigeants trop de mouvements à l'utilisateur. En effet, un petit déplacement de la main pourrait être converti en plus grand déplacement du curseur dans le monde virtuel et alors les mêmes techniques que celles utilisées pour la souris (accélération, élargissement des zones d'intérêt en fonction de la distance à parcourir) seraient intéressantes. Les chercheurs examinèrent l'usage de leur système avec plusieurs tâches de CAO et notèrent que le fait de mixer des interfaces 2D et 3D dans une même tâche pouvait avoir des bénéfices significatifs. Ils explorèrent aussi plusieurs techniques pour effectuer le passage entre les interfaces et réalisèrent qu'il était possible d'utiliser des « triggers » automatiques, qui font passer le système d'un mode à l'autre en fonction du contexte, sans pour autant déranger l'utilisateur, à condition d'effectuer des transitions doucement. Finalement, les auteurs déplorent le fait que leur système ne supporte pas un suivi de la tête, souvent présent dans les applications de RA, qui aurait pu grandement accroire les performances lors des tâches.

Ha et al. (Ha et Woo, 2010) n'ont pas bâti de système de CAO à proprement dit, mais ils comparèrent plusieurs types d'accessoires quant à des tâches de manipulation d'objets virtuels (qui sont des tâches importantes de la CAO) dans un environnement de RA. Ils désiraient déterminer quel accessoire réel, parmi une « tasse » (un cylindre), un « cube », une « pagaie » (un prisme) et une « pagaie étendue » (un prisme muni d'un bouton), serait le plus approprié pour effectuer des translations dans un plan 2D et dans un univers 3D. L'outil de pagaie étendu obtint les meilleurs résultats en termes de temps moyen par opération, mais les auteurs constatèrent certains détails intéressants. D'abord, si le bouton sur le prisme était très utile, les utilisateurs n'aimaient pas devoir le tenir enfoncé pour réaliser une translation.

L'option de pouvoir appuyer une fois pour débuter puis une seconde fois pour arrêter la translation aurait donc été utile. Ensuite, un bouton n'est pas nécessaire, mais aide grandement l'intuitivité des interfaces car il y a toujours une ambiguïté lorsqu'un contrôle est activé après un certain temps passé à un endroit ou dans une position précise. Finalement, les chercheurs suggèrent l'utilisation de marqueurs visuels pour aider la perception des profondeurs.

Une contribution notable fut faite non pas par un article scientifique, mais par une démonstration de l'industrie. *Microsoft*, qui développe actuellement le visiocasque à RA HoloLens, a développé quelques applications prototypes démontrant les capacités et l'usage de leur appareil. Dans une présentation d'un tel prototype, *Microsoft* annonça comment leur dispositif pouvait augmenter l'utilisation d'un logiciel traditionnel de CAO comme Maya, par *Autodesk* (HoloLens, 2015). Un utilisateur muni du HoloLens peut, en contact avec un ordinateur où s'exécute Maya, faire sortir de l'écran le modèle 3D en cours de modélisation et, à l'aide de la souris, le manipuler pour mieux l'observer. Il peut aussi faire afficher un modèle grandeur nature un peu plus loin et s'en approcher pour l'observer comme si l'objet était bien réel et se trouvait devant lui. Finalement, à l'aide de commandes vocales, l'usager peut effectuer quelques modifications simples au modèle pour voir en temps réel les changements. Bien qu'il s'agisse d'une démonstration publicitaire et non d'une analyse rigoureuse, le prototype sert de preuve de concept d'un système multimodal intégrant les logiciels classiques de CAO aux technologies émergentes de RA.

Pour conclure ce sous-chapitre, il est à noter que Dorta et al. (Dorta, Kinayoglu et Hoffmann, 2015) ont publié, vers la fin de ce projet de mémoire, un article portant sur un projet similaire à celui abordé dans ce rapport. Dénommé *Hyve-3D* pour « Hybrid Virtual Environment 3D », leur système permet d'effectuer des opérations de CAO dans un environnement de CAVE à RV à l'aide d'une tablette multi-tactile contrôlant un curseur 3D. La tablette est suivie avec 6 degrés de liberté et permet, en la déplaçant et en interagissant avec l'écran, de modéliser ou d'assembler des objets. Leur système inclue une version immersive (avec CAVE à RV) ainsi qu'une version non immersive où la tablette s'utilise tout simplement avec un écran de

laptop. Les différences importantes entre ce système et DualCAD sont les suivantes : DualCAD fait usage de visiocasque et non d'une CAVE, et DualCAD permet de sauter aisément vers le mode de bureau tandis que Hyve-3D n'est pas conçu pour basculer entre les modes immersif, non immersif et logiciel traditionnel de CAO. À tout le moindre, il s'agit d'une avancée impressionnante qui aurait pu davantage façonner les décisions prises lors de ce projet de mémoire, en plus de démontrer qu'il existe un intérêt certain pour l'utilisation de plateformes mobiles en lien avec les technologies de RV, RA et de CAO.

## 1.4 L'usage des appareils mobiles

Les technologies de RA dites mobiles (c'est-à-dire faisant usage de dispositifs portables pour augmenter une vision de la réalité) ont émergé principalement à partir de jeux vidéo. Dès 2002 on voyait l'arrivée du jeu ARQuake, un jeu de tir à la première personne faisant usage d'un visiocasque et d'un laptop tenu dans un sac à dos (Broll et al., 2008). Mais c'est en 2004 qu'apparut le premier jeu faisant usage de dispositifs tenus en main (RA « à travers » par vidéo) : AR Soccer se jouait uniquement avec un téléphone intelligent. Depuis, les applications faisant usage de ces appareils pour la RA se décuplèrent et l'arrivée des téléphones intelligents tels que le iPhone et les Android ne fit qu'accélérer le développement des jeux de RA. Certaines technologies ou librairies, telles que ARToolkit (ARToolworks, 2001), s'imposèrent rapidement en fournissant aux développeurs des outils pour le suivi d'environnement 3D par vision (à l'aide de marqueurs) et l'ajout d'objets virtuels à une scène réelle. ARToolkit est encore utilisé actuellement par beaucoup, vu son support pour de nombreuses plateformes et technologies. Mais certaines avancées récentes, notamment le Project Tango de *Google* (Google, 2015), ont prouvé qu'il était possible d'utiliser la caméra d'un appareil mobile et des algorithmes de vision pour effectuer le suivi en 3D de l'appareil sans avoir à utiliser des marqueurs. Il est envisageable que, dans un avenir rapproché, il soit possible d'obtenir un suivi fidèle avec 6 degrés de liberté (les 3 degrés de liberté de rotation étant déjà couverts par l'accéléromètre, le magnétomètre et le gyroscope) des téléphones intelligents sans avoir à les fusionner à des capteurs spécialisés.

Les interactions possibles avec les appareils mobiles faisant usage de RA sont nombreuses. Hürst et al. essayèrent de comparer trois types d'interactions possibles : un contrôle direct d'un objet virtuel tel que vu par la caméra d'un téléphone intelligent, un contrôle indirect en pointant l'objet virtuel, et un contrôle avec l'écran tactile (Hürst et Van Wezel, 2013). Les auteurs remarquent que les opérations complexes ont avantage à faire usage non seulement des mains suivies par vision, mais aussi de l'écran tactile, qui offre plus de précision et moins d'ambiguïté. Il s'agit d'une des fondations de ce projet de mémoire. Lakatos et al. améliorèrent encore plus cette idée avec leur système T(ether) (Lakatos et al., 2014). Celui-ci utilise une tablette en tant que fenêtre vers une réalité augmentée, ainsi qu'un suivi de la main pour le contrôle des objets virtuels. L'écran de la tablette offre toutefois plus de contenu : en fonction du contexte de sélection ou de transformation, la tablette affiche des menus appropriés qui permettent à l'utilisateur, en appuyant ou glissant ses doigts sur la surface tactile, d'effectuer des opérations demandant plus de précision, par exemple le choix de couleur. Le problème avec ce concept est que le menu contextuel cache alors la scène.

Si les appareils mobiles peuvent servir à eux-mêmes de dispositifs à RA, certains chercheurs les ont plutôt utilisés pour complémenter d'autres dispositifs de RA. Spindler et al. utilisèrent un large écran placé à l'horizontale pour afficher une scène virtuelle et plusieurs tablettes pour fournir aux observateurs une fenêtre vers le monde virtuel (Spindler et al., 2014). La tablette peut donc servir à obtenir des informations particulières ou des menus contextuels sur un objet sans pour autant cacher l'objet pour les autres utilisateurs. Dorta et al. allèrent encore plus loin en transformant ces tablettes en curseur 3D et en proposant de nouvelles techniques d'interaction faisant usage de ces curseurs dans un environnement de CAVE (Dorta, Kinayoglu et Hoffmann, 2015). S'approchant autant que possible d'un système complet de CAO, on voit l'utilité réelle des appareils mobiles pour ces tâches dans un environnement de RA.

## 1.5 Des secteurs à explorer

Suite à la revue de la littérature présentée précédemment, il peut être intéressant de mettre en contexte une grande quantité des travaux antérieurs et d'essayer d'identifier les régions où il reste des analyses à faire. Une taxonomie, sous la forme d'un tableau, rend plus facile la comparaison des projets de l'état de l'art et permet de constater, en un seul coup d'œil, les régions peu explorées. C'est donc cela qui a été fait afin de résumer cette revue de la littérature, mais aussi pour justifier les objectifs du projet. Deux taxonomies, l'une comparant les modes d'entrée et de sortie (input/output) et l'autre les techniques d'interaction, seront présentées dans les sous-sections qui suivent. Il est à noter que le prototype conçu lors de ce projet de mémoire (DualCAD et ses composantes DesktopCAD et ARCAD) a été inséré dans les taxonomies lorsqu'il s'appliquait. Ce choix a été fait afin d'éviter de dupliquer des informations et aussi de contenir à un seul endroit le travail fait en relation avec l'état de l'art. Il est donc normal que, lors d'une première lecture de ce mémoire, la présence du prototype dans les taxonomies ne soit pas évidente à comprendre. La suite de ce document référencera à plusieurs moments ces taxonomies, assurant ainsi une compréhension globale des tableaux suite à la lecture complète. Il est donc possible, pour compléter cette revue, de ne pas porter attention aux mentions de DesktopCAD et de ARCAD.

### 1.5.1 Taxonomie des inputs/outputs

Pour cette première taxonomie, il a été décidé de classer les systèmes antérieurs de CAO dans une matrice où chaque colonne représente un dispositif d'output et chaque ligne représente un dispositif d'input. Certains systèmes peuvent se retrouver dans plusieurs cellules s'ils font usage de plusieurs dispositifs. Cette taxonomie est présentée dans le Tableau 1.2.

Tableau 1.2 Taxonomie des inputs/outputs

| | | **OUTPUT** | | |
|---|---|---|---|---|
| | **Desktop & moniteur "tabletop"** | **Écran immersif (Taille d'un mur ou enveloppant)** | **Dispositif actif tenue en main** | **Visiocasque** |
| **INPUT** **Souris 2D, clavier** | (CAO traditionnel)<br>HybridSpace (Bogdan, Grossman et Fitzmaurice, 2014)<br>Kijima's Fused Environment (Kijima et Ojika, 1997)<br>SpaceTop (Lee et al., 2013)<br>DesktopCAD | | | HoloLens+Maya (HoloLens, 2015) |
| **Écran tactile** | Mockup builder (De Araùjo, Casiez et Jorge, 2012) | N/A | | |
| **PIP: Personal Interaction Panel** | Tangible displays (Spindler et al., 2014) | Studierstube (Szalavári et al., 1998) | Tangible displays | Personal Interaction Panel (Szalavári et Gervautz, 1997)<br>Studierstube<br>Construct3D (Kaufmann et Schmalstieg, 2003)<br>Virtual Gorilla Exhibit (Bowman et al., 1998) |
| **Dispositifs suivis en 3D** | Hyve-3D (Dorta, Kinayoglu et Hoffmann, 2015)<br>ARCAD | CAVE (Cruz-Neira, Sandin et DeFanti, 1993)<br>Hyve-3D<br>CAVE VR Interface (Mine, Yoganandan et Coffey, 2014) | Tangible displays | 3DM (Butterworth et al., 1992)<br>Personal Interaction Panel<br>Studierstube<br>Construct3D<br>Virtual Gorilla Exhibit<br>Kijima's Fused Environment<br>HVE Level Editor<br>ARCAD |
| **Écran tactile tenue en main (tél. intelligent)** | Hyve-3D<br>ARCAD | Hyve-3D<br>CAVE VR Interface | (applications mobiles)<br>Hyve-3D<br>CAVE VR Interface<br>HVE Level Editor (Wang et Lindeman, 2014)<br>ARCAD | HVE Level Editor<br>ARCAD |
| **Mains suivie en 3D** | HybridSpace<br>Mockup builder<br>MixFab (Weichel et al., 2014)<br>SpaceTop<br>6D Hands (Wang, Paris et Popović, 2011) | CAVE | Midair finger selection | HoloLens (Microsoft, 2015)<br>Virtual Gorilla Exhibit<br>(et plusieurs autres)<br>ARCAD |

D'abord quelques précisions quant au tableau précédent. Certaines cellules sont vides car il n'y avait pas de système de CAO faisant l'usage de cette combinaison d'input/output. De plus, il ne peut pas y avoir de systèmes utilisant des écrans de bureau tactiles pour un affichage autre que cet écran. Un système pourrait exister intégrant des techniques pour écran tactile et pour visiocasque par exemple, mais il ne s'agirait tout de même pas d'une combinaison « écran tactile » / « visiocasque » puisque la surface tactile de l'écran sert avant tout à afficher sur ce même écran : il faudrait alors introduire comme input les « surfaces tactiles ». De ce fait on remarque que cette matrice ne se veut pas être une liste exhaustive des dispositifs d'input et d'output, mais présente plutôt la liste des dispositifs utilisés par ce projet de mémoire ainsi qu'une sélection d'autres dispositifs revenant fréquemment dans la littérature pour ces logiciels. L'un des dispositifs d'input est le Personal Interaction Panel (PIP) (Szalavári et Gervautz, 1997), soit une surface non-active (par exemple une feuille de papier) pouvant être utilisée, à l'aide de caméras et de projecteurs ou visiocasques, pour présenter des informations ou un menu interactif à un utilisateur. Le PIP est en quelque sorte l'ancêtre des tablettes et téléphones intelligents présentement utilisés en lien avec la RA et de ce fait n'est plus tellement utilisé, mais a tout de même été inclus ici pour ne pas omettre les travaux réalisés avec cette technologie, ceux-ci étant très pertinents. Finalement, les cellules en bleu représentent les combinaisons d'input/output couvertes par le prototype DualCAD.

On remarque avec cette taxonomie qu'il y a des zones peu peuplées. Les combos ayant comme output **Desktop** et ayant comme input **Dispositifs suivis en 3D** ou **Écran tactile tenu en main** étaient en fait complètement vides lors du début de ce projet de mémoire et offraient donc des opportunités majeures d'innovation. Le projet qui s'ajouta par la suite, Hyve-3D (Dorta, Kinayoglu et Hoffmann, 2015), permet d'utiliser les techniques faites pour la RA dans un environnement de bureau, mais s'avère limité en termes de techniques d'interaction offertes par l'écran tactile, en plus d'être conçu à la base pour un environnement CAVE et non pour un visiocasque. La combinaison **Écran tactile tenu en main** / **Visiocasque** est aussi intéressante car la matrice semble indiquer que les travaux sur HVE Level Editor (Wang et Lindeman, 2014) ont exploré entièrement ce secteur. Or HVE Level Editor utilise une tablette pour modifier un monde virtuel non immersif et un

visiocasque pour observer, grâce à d'autres dispositifs, ce monde virtuel. Le lien n'est pas direct, c'est-à-dire que leur système ne permet pas d'utiliser la tablette pour manipuler le monde tel que vu par le visiocasque. Il y a donc intérêt à analyser l'usage d'un téléphone intelligent avec un visiocasque à RA dans une application de CAO.

### 1.5.2 Taxonomie des techniques d'interaction

Comme la dernière taxonomie représentait uniquement les combinaisons d'input/output et qu'il était difficile de constater si les systèmes utilisaient des techniques d'interaction similaires ou non, une seconde taxonomie a été produite : le Tableau 1.3 recense quasiment tous les systèmes présentés dans la taxonomie précédente (ceux perfectionnés par d'autres systèmes aussi présents n'ont pas été inclus) et indique quelles techniques d'interaction sont employées. Chaque technique est classée par dispositif ou par outil utilisé. Encore une fois, il ne s'agit pas d'une liste exhaustive des techniques d'interaction possible, mais plutôt d'une sélection de techniques populaires ou implémentées par le prototype réalisé.

Tableau 1.3 Taxonomie des techniques d'interaction

| Systèmes | Interaction | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Disp. de pointage 2D | | Position 3D des doigts | | | Dispositif de pointage 3D (stylet, accessoire) | | | Surface tenue en main ou "tabletop" | | | | | | | | |
| | | | | | | | | | Affi-chage | | Suivi en 3D | | | Multi-tactile | | Stylet | |
| | GUI 2D | Widgets 3D | Tracé de rayon | Manipulation directe | Manipulation indirecte | Tracé de rayon/wand | Manipulation directe | Manipulation indirecte | Menu 2D | Vue vers monde 3D | Tracé de rayon/wand | Manipulation directe | Manipulation indir. | Menu 2D | Manipulation d'objets | Écriture | Dessin géométrie |
| **DualCAD** | X | X | X | X | | | | | X | | X | X | X | X | X | X | X |
| HybridSpace | X | X | | | X | | | | | | | | | | | | |
| Virtual Gorilla Exhibit | | | | | X | X | | | X | X | | | | | | X | |
| 3DM | | | | | | | X | | | | | | | | | | |
| Mockup builder | | | | X | | | | | X | X | | | | X | X | | |
| Hyve-3D | | | | | | | | | X | X | | | X | X | X | | X |
| HoloLens+Maya | X | X | X | | X | | | | | | | | | | | | |
| Construct3D | | | | | | | | X | X | | | | | | | | |
| Kijima's Fused Environ. | X | X | | X | | | | | | | | | | | | | |
| SpaceTop | | | | X | | | | | X | X | | | | X | X | | |
| CAVE VR Interface | | | | | | | | | X | | | X | X | X | X | | |
| Tangible displays | | | | | | | | | X | X | | | X | X | X | | |
| Studierstube | | | | | | X | X | | X | X | | | | | | | X |
| HVE Level Editor (tablet) | | | | | | | | | X | X | | | | X | X | | |
| HVE L. E. (immersive) | | | | | | X | | X | | | | | | | | | |
| 6D Hands | | | | X | X | | | | | | | | | | | | |
| MixFab | | | | X | | | X | | X | X | | | | | | | |

On remarque instantanément qu'aucun système antérieur ne couvre une majorité des techniques d'interaction identifiées. Tout au plus, un système inclut 6 de ces techniques. Il y a donc un potentiel à investiguer la possibilité et les avantages de joindre ces techniques dans un seul système. On peut remarquer aussi certains choix possibles de conception. Plusieurs systèmes précédents utilisent des dispositifs de pointage en 3D, comme des stylets Polhemus. Comme il est possible d'utiliser un téléphone intelligent capable des mêmes fonctions et plus encore, les techniques faisant usage d'un dispositif de pointage ne sont pas nécessaires, tant qu'on développe un équivalant avec une surface tenue en main.

Les deux autres techniques d'interaction qui furent mises de côté pour ce projet de mémoire sont les manipulations indirectes par suivi de doigt et l'utilisation du téléphone comme

fenêtre vers le monde virtuel. Dans le premier cas, il nous semblait absurde d'avoir autre chose qu'un contrôle direct avec les mains puisque celui-ci est si naturel (et qui plus est le champ de vision de la caméra du visiocasque ne permettrait pas de manipuler un objet avec sa main placée sur ses genoux). Pour ce qui est d'obtenir une fenêtre vers le monde virtuel, nous considérons que le visiocasque offre amplement d'immersion pour observer la réalité augmentée et qu'il serait inutile d'utiliser le téléphone de cette façon en plus. Finalement, il semblerait que personne avant nous n'ait fait usage d'un téléphone intelligent en tant que baguette magique (wand) pour la sélection et les transformations.

## 1.6 Problématique et objectifs

Suite à cette revue de la littérature, il a été possible d'identifier des secteurs méritant une plus grande exploration et donc de définir une problématique en général. Celle-ci peut être résumée de la sorte :

> Peut-on réaliser un système multimodal joignant un logiciel traditionnel de CAO et un système à RA de CAO et permettant de basculer aisément entre les deux afin d'augmenter la productivité ? Quels seraient les avantages et limitations de chaque mode ?
>
> Et quelles seraient les techniques d'interaction à utiliser pour faire usage des deux modes en s'appuyant sur les avantages mentionnés plus tôt tout en amortissant les limitations ? Est-ce que l'utilisation d'un téléphone intelligent pourrait outrepasser les problèmes actuellement rencontrés par les techniques par gestuelle ?

Pour répondre à ces questions, ce projet de mémoire se concentrera sur 3 objectifs principaux, chacun pouvant être divisé en sous objectifs. Les sous-objectifs en gras sont ceux pouvant apporter une contribution à l'état de l'art, c'est-à-dire qui n'ont pas été réalisés auparavant.

1) Développer un système multimodal de CAO nommé DualCAD :
   a) Développer un logiciel traditionnel de CAO nommé DesktopCAD;
   b) Développer un logiciel de RA pour la CAO nommé ARCAD;
   **c) Permettre de sauter d'un mode à l'autre en un minimum d'actions.**
2) Développer des techniques d'interaction en RA faisant usage du visiocasque et d'un téléphone intelligent :
   a) Concevoir et implémenter des techniques en RA afin de répliquer les opérations classiques de CAO : translation, rotation, mise à l'échelle, etc.;
   **b) Concevoir et implémenter des nouvelles techniques en RA qui répondent à des besoins de CAO : édition de texture, création de volumes complexes, etc.**
3) Analyser le travail fait précédemment :
   **a) Présenter DualCAD à des experts du domaine et noter leurs commentaires et performances;**
   **b) Avec ces données, établir des recommandations pour les systèmes futurs;**
   **c) Expliquer les leçons apprises lors du projet pour faciliter le développement futur.**

On remarque donc que, comme il s'agit d'un projet exploratoire, les objectifs ne peuvent pas être très précis à l'étape actuelle. En effet, chaque objectif pourrait être rallongé indéfiniment dans l'espoir d'obtenir d'autres découvertes intéressantes. Somme toute, les résultats les plus intéressants de ce projet de mémoire devraient être sous la forme des recommandations. Celles-ci pourront guider les chercheurs et développeurs dans leurs travaux futurs.

# CHAPITRE 2

# MÉTHODOLOGIE

## 2.1 Procédure

Pour remplir les deux premiers objectifs présentés précédemment ainsi que les sous-objectifs qui y sont liés, il a fallu respecter une procédure bien définie. Premièrement, pour développer le prototype d'un système multimodal de CAO intégrant les interfaces traditionnelles aux interfaces de RA, nommé à partir de désormais **DualCAD**, il fallait développer une architecture permettant de lier deux sous-systèmes : **DesktopCAD** et **ARCAD**. Cette architecture fut bâtie à partir de projets précédents et utilisa une ébauche de chacun de sous-systèmes pour faciliter le développement et obtenir une preuve de concept. Une fois les deux ébauches étant en mesure de communiquer entre elles et de se transmettre des modèles complexes, il fut temps d'étendre les fonctionnalités du premier sous-système. Dénommé de la sorte pour indiquer qu'il s'agit d'un logiciel de CAO (CAD en anglais) utilisant les interfaces traditionnelles de bureau (Desktop en anglais), DesktopCAD se veut être un logiciel de CAO à part entière, bien qu'on se limitait aux fonctionnalités de base. Avec un clavier, une souris et un écran 2D, un utilisateur devait pouvoir créer des primitives, éditer leurs caractéristiques (couleur, texte, etc.) et les manipuler en 3D. À cela s'ajoutèrent des fonctionnalités de sauvegarde/chargement et, bien sûr, les fonctions de communication avec ARCAD. Il était prévu que les étapes d'architecture du système multimodal et le développement complet de DesktopCAD prendraient de 6 à 8 semaines.

Suite à la finalisation de DesktopCAD, la prochaine étape était de développer ARCAD. Il s'agit toutefois d'une opération bien plus longue et complexe et donc on peut la sous-diviser encore. D'abord, il fallait produire une preuve de concept de l'intégration d'un téléphone intelligent à ARCAD et, à partir de ce prototype, il fallait étendre l'application au niveau du téléphone pour qu'elle contienne toutes les fonctionnalités désirées. Ces deux étapes devaient prendre de 3 à 4 semaines. Ensuite, avec le téléphone fonctionnel, il fallait assurer son suivi avec 6 degrés de liberté. Bien que cela puisse sembler trivial, il était évident dès

l'amorcement du projet qu'il s'agissait d'un des aspects les plus complexes et donc il était prévu de passer de 4 à 6 semaines sur le suivi 3D et le système de calibration de celui-ci.

Finalement, avec l'architecture du système multimodal en place, DesktopCAD entièrement fonctionnel et toutes les composantes du téléphone prêtes à être utilisées, il était temps de passer à ARCAD. Le nom de ce sous-système combine la Réalité Augmentée (AR en anglais) à la CAO (CAD en anglais). L'étape consistait donc à implémenter les mêmes fonctionnalités que celles de DesktopCAD, mais dans un environnement de RA et utilisant le téléphone lorsque cela paraissait utile. Cette étape était la plus incertaine en termes de temps car toutes les fonctionnalités auraient pu être incluses, de la façon la plus simple possible, en seulement 3 semaines. Or, le second objectif majeur de ce mémoire était d'identifier le maximum de techniques d'interaction faisant usage de l'environnement offert par ARCAD. Il fut donc décidé de laisser 10 semaines fermes à cette étape pour permettre d'essayer un bon nombre d'idées.

Cela remplit donc les deux premiers objectifs poursuivis par le mémoire, mais il serait intéressant de comparer DualCAD aux travaux précédents (objectif 1) et d'analyser qualitativement les techniques d'interaction proposées par ARCAD (objectif 2). Pour le premier point, il fut choisi de produire une taxonomie, c'est-à-dire une matrice d'inputs et d'outputs possibles contenant plusieurs exemples de systèmes antérieurs, et d'y insérer DualCAD pour déterminer s'il y avait déjà eu un système qui touchait aux mêmes combinaisons d'input et d'outputs. Pour le second point, il s'agissait plus d'effectuer des tests informels, de comparer nos techniques entre elles et par rapport à celles de systèmes semblables. En tout, ces analyses devaient se faire en 3 à 5 semaines.

Il est à noter qu'il avait été prévu initialement d'effectuer des tests d'utilisabilité formels qui auraient permis d'évaluer quantitativement le système et les techniques d'interaction qu'il propose. Le projet avait d'ailleurs déjà obtenu l'accord du comité d'éthique de l'ÉTS quant à la réalisation des tests avec des participants humains et une étape de préparation de ces tests était prévue (et a même été réalisée : plus d'informations à la section 2.5 Conception).

Toutefois, il est vite devenu apparent que le matériel utilisé entraînait des limitations importantes. Le visiocasque utilisé possède un champ de vision significativement petit (23 degrés) et il y avait un risque trop important que cette caractéristique affecte négativement les tests d'utilisabilité. Ceux-ci furent donc remplacés par la démonstration de DualCAD à des utilisateurs experts (Objectif 3). Cette phase de tests devait durer de 1 à 2 semaines.

Le Tableau 2.1 résume les étapes de la procédure utilisée ainsi que le temps prévu initialement pour chacune. À noter que le travail concret (excluant la conception initiale du projet, la revue de la littérature, les projets tests pour se familiariser avec les technologies et la rédaction de ce rapport de mémoire) fut estimé durer de 26 à 35 semaines, soit 6 à 8 mois. Les estimations tiennent compte d'un travail à temps plein (30 heures par semaine) et donc ont pu être retardées par des tâches externes.

Tableau 2.1 Procédure de la méthodologie

| **Étape** | **Sous-étape** | **Durée estimée** |
|---|---|---|
| Ébauche du système multimodal de CAO | Architecture multimodale & ébauche des sous-systèmes | 2 à 3 semaines |
| | Implémentation de DesktopCAD | 4 à 5 semaines |
| Développement de l'outil téléphone intelligent | Architecture ARCAD vers téléphone intelligent | 1 à 2 semaines |
| | Application téléphone | 1 à 2 semaines |
| Suivi du téléphone intelligent avec 6 degrés de liberté | Suivi 3D du téléphone | 3 à 4 semaines |
| | Système de calibration du suivi 3D | 1 à 2 semaines |
| ARCAD | Fonctionnalités de ARCAD | 10 semaines |
| Tests | Démonstration auprès d'utilisateurs experts | 1 à 2 semaines |
| Analyse des résultats | Taxonomie & comparaison de DualCAD aux systèmes antérieurs | 1 à 2 semaines |
| | Analyse qualitative des techniques d'interaction de ARCAD | 2 à 3 semaines |

## 2.2 Matériel

Cette section aborde la liste du matériel qui fut requis pour l'accomplissement de ce projet de mémoire. Le laptop, un Dell Precision M4800 roulant Windows 7 et muni de 8 Go de RAM, d'un processeur i7-4600M @ 2.9 GHz et d'une carte graphique NVIDIA Quadro K2100M 4 Go, est l'outil principal du sous-système DesktopCAD, mais aussi le centre principal du traitement pour tout le système multimodal. Il dispose bien sûr d'un clavier et d'une carte Wifi, mais à cela s'ajoute une souris optique et une clé Bluetooth permettant de se connecter à un autre appareil à distance. Finalement, le laptop a aussi une sortie HDMI permettant d'ajouter un second écran. L'ordinateur ne fut pas sélectionné spécialement pour ce mémoire, mais il s'avéra suffisamment performant et ambivalent pour le projet.

Le téléphone intelligent qui fut retenu pour servir d'outil à ARCAD est le Samsung Galaxy Note 4. Il s'agissait de la version la plus récente de la série Note qui, en plus de disposer d'une puissance de traitement agréable et d'un émetteur Wifi/Bluetooth, a un écran relativement large (5,7 pouces) et est muni d'un stylet « S Pen ». Ces deux derniers critères distinguent l'appareil de ses concurrents et étaient essentiels pour nos besoins. En effet, nous ne voulions pas risquer de manquer d'espace d'affichage (ce qui exclue les plus petits téléphones intelligents) tout en pouvant prendre le dispositif dans une seule main (ce qui exclue les tablettes) et nous avions plusieurs idées de techniques d'interaction faisant usage d'un stylet. Ce modèle s'avéra donc très pertinent au projet, bien qu'un téléphone plus petit aurait aussi fait l'affaire finalement. À noter que cet appareil utilise le système d'opération Android, ce qui fut un plus vu notre expérience passée avec le développement pour cette plateforme.

Pour effectuer le suivi avec 6 degrés de liberté des différents appareils inclus dans le sous-système ARCAD, un Polhemus Patriot fut utilisé (Figure 2.1). Celui-ci est constitué de deux capteurs, pouvant être placés sur des appareils, d'un capteur source et d'une borne pour brancher le tout, qui se connecte au laptop par une prise USB. Chaque capteur est suivi avec 6 degrés de liberté, mais est sensible aux courants magnétiques et ne peut être utilisé que

d'un seul côté de la borne source à la fois (dû aux limitations d'identifier une position par un champ magnétique, il existe toujours deux réponses possibles et seule celle placée du côté dit actif de la borne source sera utilisée). L'appareil est normalement stable et assez fiable dans un rayon de quelques mètres (la précision se mesurant en centimètres), mais le modèle auquel nous avions accès perdait sa précision à partir d'un mètre. Cela ne fut pas trop limitatif pour les besoins du prototype toutefois.


Figure 2.1 Le Polhemus Patriot, incluant un senseur

L'item le plus important pour ce projet est sans aucun doute le visiocasque. Celui-ci fut sélectionné parmi ceux disponibles en 2014 avec les critères suivants : il doit se brancher à un ordinateur pour que celui-ci effectue le traitement, il ne doit pas bloquer la vision périphérique et il doit disposer de caméras (incluant une caméra de profondeur si possible). Les autres caractéristiques, résolution, poids, champ de vision et plateformes supportées, ont été considérées, mais n'étaient pas vues comme étant critiques. C'est avec cet objectif qu'il a été décidé de commander un visiocasque META, version développeur (Figure 2.2) (META, 2014). Celui-ci, connu originalement sous le nom Spaceglasses, est un visiocasque optique projetant sur chaque lentille une image stéréoscopique de 480x540 (960x540 en tout) et disposant d'un champ de vision de 23 degrés. Une caméra RGB et une caméra infrarouge de profondeur sont placés dur le dessus, pointés vers l'avant. Des senseurs internent permettent

un suivi des rotations avec 3 degrés de liberté, mais ceux-ci ne furent pas utilisés pour le projet, qui nécessitait 6 degrés de liberté (d'où l'usage de capteurs de mouvement indépendants). Les lunettes se branchent dans une boîte alimentée électriquement, qui elle-même communique avec le laptop par USB en plus d'obtenir l'image à afficher par un câble HDMI. Les lunettes sont vues, pour l'ordinateur, comme un écran secondaire. Le visiocasque interagit avec l'ordinateur à travers un framework conçu pour Unity3D, un moteur 3D disposant de son propre éditeur, et fonctionnant en C#. Les caractéristiques de ce framework en firent un attrait puisque nous avions une expérience notable avec ces technologies.

Figure 2.2 Le visiocasque META (version développeur)

D'autres dispositifs, non essentiels au prototype DualCAD mais aidant grandement au développement, aux tests ou à la capture d'images, furent utilisés. Un séparateur HDMI actif fut utilisé pour copier le flux vidéo envoyé aux lunettes et le transmettre à un écran externe, permettant à un observateur de voir la même chose que la personne utilisant le visiocasque. Un autre appareil, un enregistreur HDMI, fut placé entre le séparateur et l'écran pour enregistrer ce que voit l'utilisateur. Finalement, un trépied, une caméra GoPro et une webcam furent utilisés pour filmer l'utilisateur et ses interactions avec DesktopCAD, ARCAD et même avec le téléphone intelligent.

## 2.3 Défis techniques

Cette section porte sur tous les problèmes rencontrés lors de ce projet de mémoire. Certains furent identifiés dès le début tandis que d'autres ne se présentèrent qu'en cours de route. Certains de ces défis incluront la solution trouvée ici-même, mais pour la plupart, la réponse au problème ne sera pas présentée avant les sous-chapitres Architecture, Conception et Implémentation, selon comment s'est faite la résolution.

Le premier défi à être identifier était de devoir effectuer un lien entre les sous-systèmes DesktopCAD et ARCAD. Ce lien n'est pas trivial puisqu'il doit y avoir des échanges complexes (des modèles 3D complets incluant des textures particulières). L'architecture complète de DualCAD devait donc être faite dans l'optique de ces échanges d'information entre deux logiciels. Heureusement, une preuve de concept d'échange d'informations avait été faite au préalable dans un projet test : pour afficher avec le visiocasque un graphique 3D produit avec Excel, un plug-in Excel fut fait en C# et celui-ci agit comme serveur TCP, envoyant les données à un client, l'engin 3D pour le visiocasque.

Ce qui nous amène à un autre problème : quelles technologies utiliser ? Comme on veut réaliser des opérations complexes, il faut un langage de haut niveau, mais qui peut tout de même effectuer des opérations au niveau de la carte graphique. Le fait que le framework META soit en C# pour Unity3D est une suggestion intéressante. Nous avions déjà beaucoup d'expérience avec ces technologies et elles semblent assez bien adaptées à nos besoins. Toutefois, il faut préciser que la version de C# qui est utilisée par Unity3D n'est pas celle développée par Microsoft, mais bien un sous-ensemble libre de droits, appelée Mono C#. Celui-ci est très proche de C# .NET 3.5, mais omet certaines fonctionnalités qui, nous le verrons, auraient été très utiles. Il y a aussi le problème que certaines composantes du projet imposent quelque peu leur propre technologie : le Polhemus Patriot offre son propre framework en C et le téléphone intelligent Android, bien que pouvant exécuter une application écrite en C#, a besoin de Java pour accéder à toutes les fonctionnalités désirées. Somme toute, il fut décidé d'utiliser le langage C# avec l'environnement Unity3D, puisqu'il

s'agissait d'un point central de toutes les technologies impliquées. Les problèmes engendrés par cette décision seront présentés plus bas, mais tous purent être résolus de façon satisfaisante au final.

En plus de la difficulté à joindre les sous-systèmes de DualCAD, il fallait aussi trouver une façon de permettre la communication entre ARCAD et le téléphone intelligent. Ce premier fonctionne avec Mono C# et le second exécute une application codée en Mono C#, Java ou un mélange des deux. Utiliser un câble n'est pas la meilleure des options puisque cela pourrait être encombrant pour l'utilisateur. Utiliser le Wifi est une option plus intéressante, mais une preuve de concept a démontré une faille importante de cette solution : le fait de communiquer avec un laptop placé sur le même réseau, qu'on accède par Wifi, nous rend dépendant du dit réseau et de la borne Wifi. Le système ne serait pas fonctionnel dans un environnement sans réseau ou pour lequel on n'a pas d'accès. De plus, le Wifi n'est pas très fiable et beaucoup d'informations étaient perdues par moment, ce qui n'est pas très pratique pour un outil de type télécommande pour lequel on espère une réaction rapide lorsqu'on entre une commande. Utiliser le laptop comme borne Wifi est une solution envisageable, mais la carte réseau du laptop utilisé n'est pas vraiment conçu pour cela et donne donc des résultats médiocres encore une fois. Une toute autre option est d'utiliser le Bluetooth, supporté de base par le téléphone et avec une clé pour le laptop. Il s'agit d'un protocole plus stable à courte distance et qui ne dépend pas d'une borne. Toutefois, la version de Mono C# de Unity3D ne supporte pas du tout les communications Bluetooth et donc on ne peut pas utiliser ce protocole directement.

Au-delà de la communication entre le téléphone intelligent et le système, un des usages de ce premier est de pouvoir en faire le suivi avec 6 degrés de liberté. Il en est de même du visiocasque qui, bien qu'offrant déjà 3 degrés de liberté, se doit d'être suivi dans l'espace 3D pour assurer une immersion complète de l'utilisateur. C'est pour cette raison que ce projet de mémoire a fait usage du Polhemus Patriot, un système complet de capteurs de mouvement permettant de faire le suivi de deux senseurs distincts. En plaçant un senseur sur le téléphone et un autre sur le visiocasque, il est donc possible d'obtenir, avec le framework de Polhemus,

la position de chaque capteur, en cm, par rapport à une borne référence, ainsi que leur orientation en angles d'Euler. Cela entraîne toutefois sa part de défis. En effet, le framework de Polhemus fonctionne en C et le système doit faire des requêtes et obtenir les positions en C#. Il faut donc s'assurer de convertir le framework en code interprétable ou bien produire un « wrapper » faisant le lien entre les deux couches. Ensuite, une preuve de concept décela un autre problème majeur : le champ magnétique des capteurs, utilisé pour situer ces capteurs, était affecté par les composants électroniques du téléphone et du visiocasque. Le suivi avec 6 degrés de liberté fonctionnait, mais devenait erroné dès qu'on attachait les capteurs directement sur les appareils. Il fallut donc déplacer les capteurs pour qu'ils soient à une certaine distance de l'électronique. Pour le téléphone cela impliqua d'utiliser une bande de plexiglass pour tenir le capteur à une dizaine de centimètres, et pour le visiocasque il suffit de placer le capteur sur le côté de l'appareil. Cela complique quelque peu les calculs puisque les capteurs ne se situent plus sur les centres de rotation des appareils, mais au moins le suivi peut se faire.

Une fois les données acquises quant au positionnement des appareils dans l'espace, celles-ci se doivent d'être converties en coordonnées utilisables par notre scène virtuelle. Le Polhemus Patriot utilise un système de type main droite pour lequel le vecteur unitaire Z est la direction UP. Les angles d'Euler sont interprétés dans cet ordre : yaw, pitch, roll. Par contre, pour Unity3D, qui utilise un système main gauche pour lequel le vecteur unitaire Y est UP, les angles d'Euler s'effectuent dans l'ordre suivant : roll, pitch, yaw. On voit tout de suite que le système d'axe diverge énormément entre les deux univers et qu'il faut une conversion à ce niveau. Ensuite, il faut ajuster les données pour que l'origine soit la position du visiocasque dans le monde virtuel (une contrainte du framework de META) et que les orientations correspondent à la réalité. Pour cela, et pour obtenir des informations sur l'utilisateur comme dans quelle main il tient le téléphone, il devient apparent qu'il faut une phase de calibration. Autre problème donc : comment effectuer cette calibration afin de la rendre aussi rapide et précise que possible.

Ensuite, il y a des défis portant sur l'interaction entre l'utilisateur et les sous-systèmes. Plus particulièrement, les manipulations 3D représentent les opérations les plus complexes et il faut se pencher sur les meilleurs techniques pour les accomplir. Dans le mode DesktopCAD, l'usager n'a qu'un clavier et une souris pour effectuer des translations, des rotations et des mises à l'échelle en 3 dimensions. Quelle est la meilleure façon de les réaliser tout en assurant la précision désirée ? Les mêmes opérations se retrouvent dans ARCAD, mais l'utilisateur a alors comme interfaces ses mains suivies par caméra et le téléphone intelligent. Il faut alors identifier d'autres techniques pour accomplir les tâches désirées.

Un autre défi, qui n'est pas lié au développement du prototype DualCAD en tant que tel, mais plutôt à la rédaction de ce rapport mémoire, à l'article lié et au vidéo démonstratif, est le fait de vouloir capturer en images et en film l'utilisation complète du système multimodal. Pour filmer l'utilisation de DesktopCAD, il est possible d'utiliser un logiciel d'enregistrement, tel que OBS (Open Broadcaster Software), pour enregistrer une séquence vidéo des opérations de l'utilisateur. À tout moment, l'utilisateur peut être filmé, avec un trépied et une caméra GoPro pour observer ses actions, mais cela ne convient pas pour montrer le contenu à l'écran du laptop et du téléphone (la luminosité des écrans rend la capture difficile), ou pour afficher les objets virtuels qui s'ajoutent à la scène réelle. L'enregistrement vidéo du téléphone peut se faire à partir de l'éditeur Unity3D : un émulateur de téléphone affiche l'application et un enregistreur logiciel sauvegarde la séquence vidéo. C'est vraiment lorsqu'on désire montrer ce que voit l'utilisateur qu'on frappe un mur. On pourrait vouloir montrer le point de vue de l'usager en enregistrant ce qui est envoyé au visiocasque, mais le sous-système ARCAD est demandant en performances et les enregistreurs logiciels sont eux aussi exigeants. Les performances du système seront grandement affectées et la vidéo serra de mauvaise qualité. On voudrait aussi pouvoir montrer la scène augmentée vue d'un observateur externe, mais comment augmenter la vision de la caméra avec le contenu virtuel ?

Finalement, un défi notable portait sur comment réaliser les tests utilisateurs. Ces tests furent finalement mis de côté, en raison des limitations du visiocasque, mais ils furent tout de même

planifiés et méritent donc d'être présentés dans ce rapport, ne serait-ce que pour guider des travaux futurs. Contrairement à des tests d'utilisabilité pour lesquels on veut comparer plusieurs techniques d'interaction pour une tâche précise, nous désirions comparer les sous-systèmes à travers une multitude de tâches et de techniques. Ces tâches sont d'ailleurs relativement complexes (positionnement avec 6 degrés de liberté de modèles 3D par exemple) et peuvent difficilement être confirmées par un ordinateur. Le mieux est encore de laisser un humain confirmer la réalisation de chaque tâche. Les tests seraient donc une série de tâches à effectuer avec DesktopCAD, avec ARCAD et finalement avec les deux en parallèle, le tout superviser par un chercheur et mesuré en temps. Le défi est donc d'identifier une façon efficace de donner les tâches à un participant, de lui laisser les accomplir tout en mesurant le temps requis, et d'imposer une confirmation d'un chercheur après chaque tâche.

## 2.4 Architecture

Par **architecture**, nous entendons le travail qui fut consacré à l'élaboration d'une structure matérielle et à la planification des liens de communications entre ces composantes matérielles. En d'autres termes, l'architecture représente le système à haut niveau et n'inclut que des pièces physiques, des sections majeures de logiciels et des protocoles de communication. Cela se distingue de la phase de **conception**, qui se penche principalement sur un de ces items précédents. Une sous-section d'un logiciel est découpée afin de planifier sa structure interne, souvent en prévision d'un défi quelconque. Finalement, **l'implémentation** représente le travail et les solutions apportées lors du développement du système. L'architecture et la conception se sont alors assurées d'offrir une structure apte à accomplir les tâches désirées, mais il peut rester des algorithmes notables à produire.

La Figure 2.3 présente l'architecture sommaire du système multimodal DualCAD. Celle-ci représente simplement les composantes du système tel qu'il a été imaginé suite à la revue de la littérature et donc il serait trivial d'en faire une analyse détaillée ici.

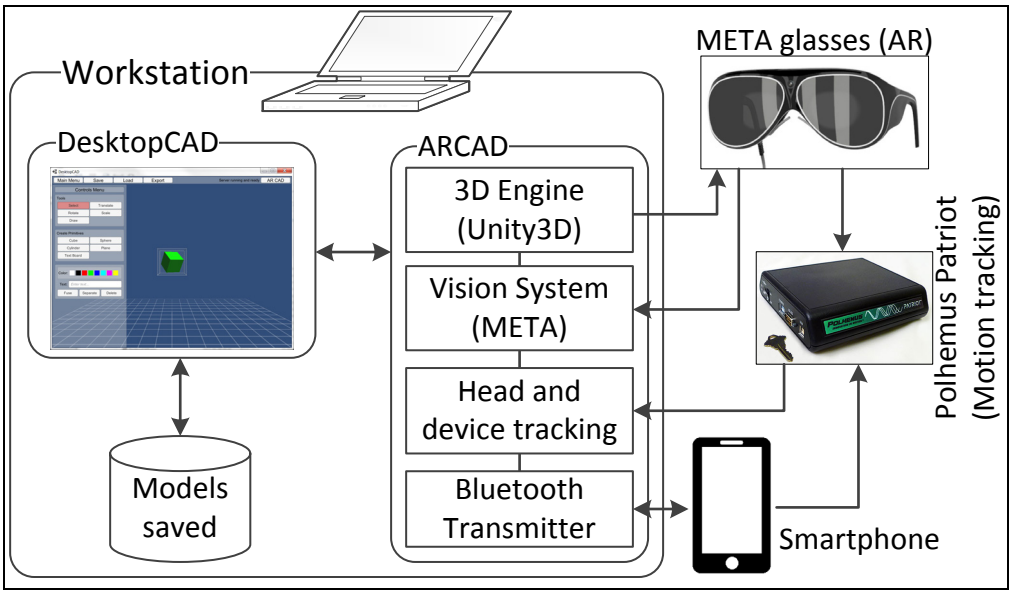


Figure 2.3 Architecture générale de DualCAD

### 2.4.1 Lien entre DesktopCAD et ARCAD

Pour comprendre les décisions qui furent prises quant à l'architecture du système de communication entre les sous-logiciels (DesktopCAD et ARCAD) de DualCAD, il faut comprendre le contexte. Les deux sous-systèmes sont des logiciels indépendants, codés en Mono C#, s'exécutant actuellement sur une même machine, mais pouvant éventuellement être sur des ordinateurs distincts. Il serait même envisageable que les deux logiciels soient codés dans des langages distincts. Il existe des librairies permettant d'établir une communication entre plusieurs langages de programmation, mais ils ne couvrent pas tous les langages possibles et ne solutionnent pas la possibilité qu'il faille une communication inter-machines.

L'idée consiste donc à utiliser un protocole de communication bien établi, dans ce cas TCP, pour transmettre des informations encodées en octets. C'est DesktopCAD qui prend le rôle du serveur TCP, puisque ce sous-système inclue les fonctions de sauvegarde et de chargement, et ARCAD est le client. Pour l'instant le client cherche simplement le serveur sur la machine locale (127.0.0.1) à un port précis, mais il serait trivial d'exécuter le client sur

une seconde machine, de la connecter à la première par un câble ethernet et de spécifier la bonne adresse IP. Un problème avec cette solution est qu'on perd l'abstraction des modèles 3D qu'on désire faire passer entre DesktopCAD et ARCAD. Ceux-ci sont représentés par des objets complexes, des textures et des positions/dimensions, mais il faut pouvoir les encoder et les décoder à l'autre extrémité. La technique utilisée pour accomplir cette tâche sera abordée au sous-chapitre Implémentation. La Figure 2.4 représente l'architecture conçue pour effectuer la communication entre les composantes majeures de DualCAD.

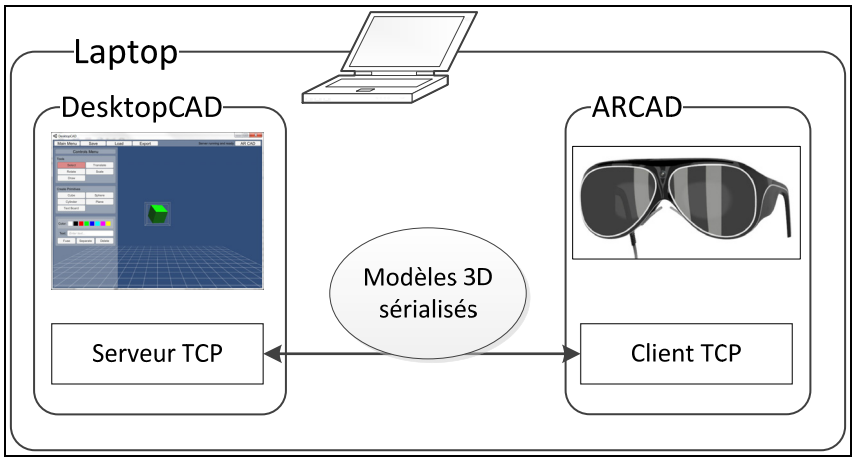


Figure 2.4 Architecture de la communication entre DesktopCAD et ARCAD

### 2.4.2 Lien entre ARCAD et le téléphone intelligent

Le lien entre le téléphone intelligent et le sous-système ARCAD représente un défi encore plus intéressant. En effet, l'idée originale était de simplement réimplémenter l'architecture présentée précédemment afin d'obtenir une communication TCP entre le téléphone et le laptop, à travers un réseau Wifi. C'est exactement ce qui fut fait et les résultats initiaux s'avérèrent prometteurs. Or, comme la section Défis techniques l'a présenté, cette solution rendait le système dépendant d'un réseau Wifi (et donc d'une borne), le rendant beaucoup moins portable. De plus, il s'avéra rapidement que le Wifi avait tendance à perdre des paquets par moment, ce qui était très frustrant pour l'utilisation du téléphone intelligent. L'architecture fut donc repensée pour se baser sur le Bluetooth, un protocole plus stable et qui a la caractéristique d'être point à point.

Pour pallier au fait que Mono C# (le langage utilisé pour le projet en lien avec Unity3D) ne supporte pas la communication par Bluetooth, tant pour le logiciel ARCAD que l'application sur le téléphone intelligent, il fut décidé d'utiliser un « man-in-the-middle », c'est-à-dire un tiers parti. On conserverait toute l'implémentation de communication par TCP qui a été faite pour ARCAD mais, au lieu d'interagir directement avec le téléphone par Wifi, c'est un autre logiciel, placé sur la même machine, qui servirait de client TCP. Appelé BluetoothToTCP pour des raisons évidentes, ce petit logiciel s'exécute en invisible dès qu'on lance le sous-système ARCAD (à moins d'être déjà en cours d'exécution) et n'offre une interface que si l'on clique sur son icône dans la barre des tâches (Figure 2.5). Cette interface (Figure 2.6) sert à effectuer le « debugging » de la communication, affichant des infos sur le serveur TCP et le relais Bluetooth connecté, en plus d'afficher les derniers messages reçus (et retransmis) de chacun. La fonction de BluetoothToTCP est simple : écouter les messages TCP du serveur (ARCAD), les convertir et les renvoyer au relais Bluetooth connecté (le téléphone intelligent), et vice versa. Le « man-in-the-middle » n'a aucune connaissance du sens des messages qu'il transmet : il reçoit simplement des bytes et les renvoie aussitôt. Il est codé en C# .NET, permettant donc de réutiliser le code de client TCP écrit précédemment tout en profitant d'un langage qui supporte pleinement la communication Bluetooth.

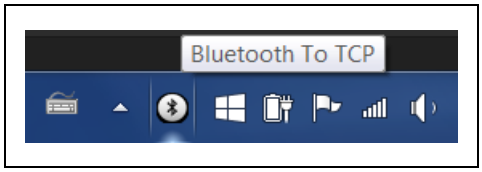


Figure 2.5 Icône de BluetoothToTCP dans la bâre de tâches

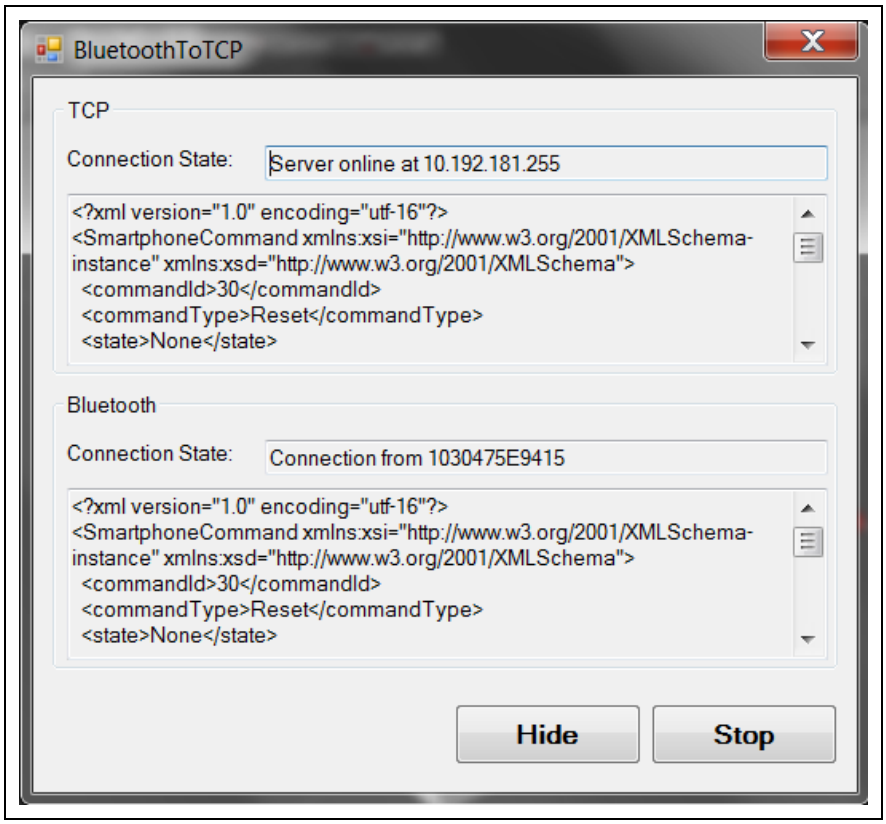


Figure 2.6 Fenêtre de debugging de BluetoothToTCP

Du côté du téléphone intelligent, il y a un enjeu similaire : il faut que le code Mono C# puisse envoyer, indirectement, des messages par Bluetooth et les recevoir de BluetoothToTCP. Ici la solution est plus simple encore : les applications pour Android supportent l'utilisation de code natif (Java) et celui-ci inclue des fonctions pour la communication Bluetooth. Il suffit donc d'inclure une section de code natif pour la communication Bluetooth dans l'application et de lier ce code au code Mono C# pour permettre l'envoi et la réception de messages. La Figure 2.7 démontre l'architecture complète de la communication entre ARCAD et le téléphone intelligent. Le diagramme inclue l'architecture basée sur le Wifi utilisée initialement (qui fonctionne encore dans le système si on désirait l'utiliser) ainsi que la nouvelle architecture axée sur le Bluetooth. Les langages utilisés pour chaque composante sont indiqués en rouge.

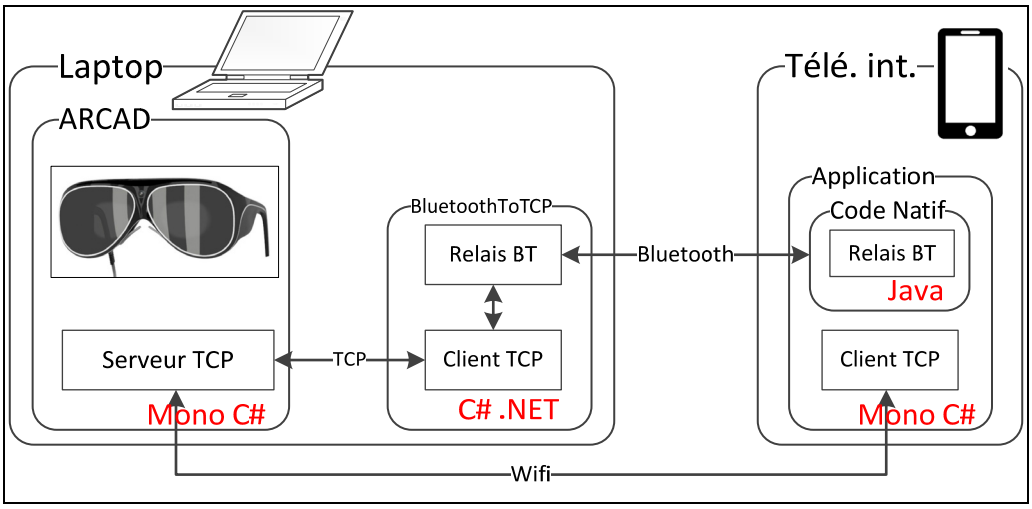


Figure 2.7 Architecture de la communication entre ARCAD et le téléphone intelligent

### 2.4.3 Filmer ARCAD selon plusieurs angles

Ensuite vint les préoccupations quant à la façon de filmer ARCAD et son utilisation. Certaines vues, telles que le point de vue d'un observateur externe, sont assez triviales à implémenter et on peut simplement utiliser un enregistreur logiciel pour DesktopCAD. Mais pour ARCAD, on ne peut pas se permettre d'allouer de temps de calcul à un enregistreur logiciel : il faut donc le sortir du système. Comme le système complet faisait déjà usage d'un séparateur HDMI pour prendre le flux vidéo envoyé aux lunettes et l'afficher sur un écran 2D (pour permettre à un observateur de voir ce que voit l'utilisateur), il suffit d'injecter une composante entre le séparateur et l'écran. Cette composante sera un enregistreur HDMI qui transmet le flux vidéo reçu tout en l'encodant en vidéo vers une clé USB, en utilisant son propre processeur et ne ralentissant donc pas le système ni l'affichage pour les lunettes. Une fois les enregistrements vidéo complétés, on peut retirer la clé USB, l'insérer dans un PC et en extraire les séquences. La Figure 2.8 présente l'architecture de ce système d'enregistrement, incluant la méthode avec laquelle on affiche sur un écran le flux vidéo vu par l'utilisateur. La Figure 2.9 est une image capturée par cette méthode. Notez que généralement, seul le contenu virtuel serait transmis aux lunettes pour l'afficher par-dessus la scène réelle vue à travers les lunettes. Mais pour la capture, la scène réelle, telle que vue par

la caméra RGB du visiocasque, est ajouté au flux vidéo afin d'obtenir un vidéo à réalité augmentée et ainsi simuler aussi près que possible la vision de l'utilisateur.

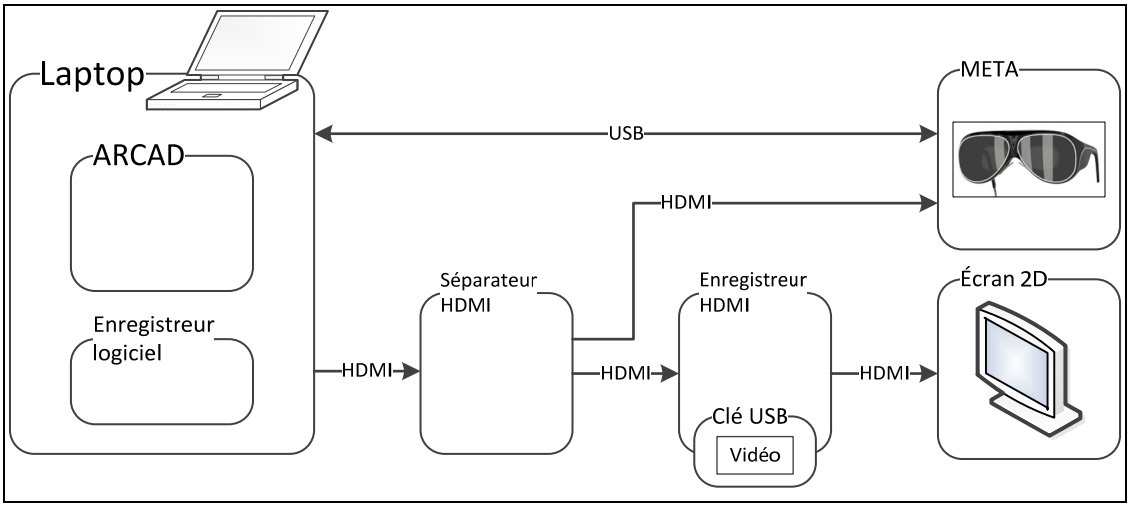


Figure 2.8 Architecture pour l'enregistrement vidéo de la vision utilisateur

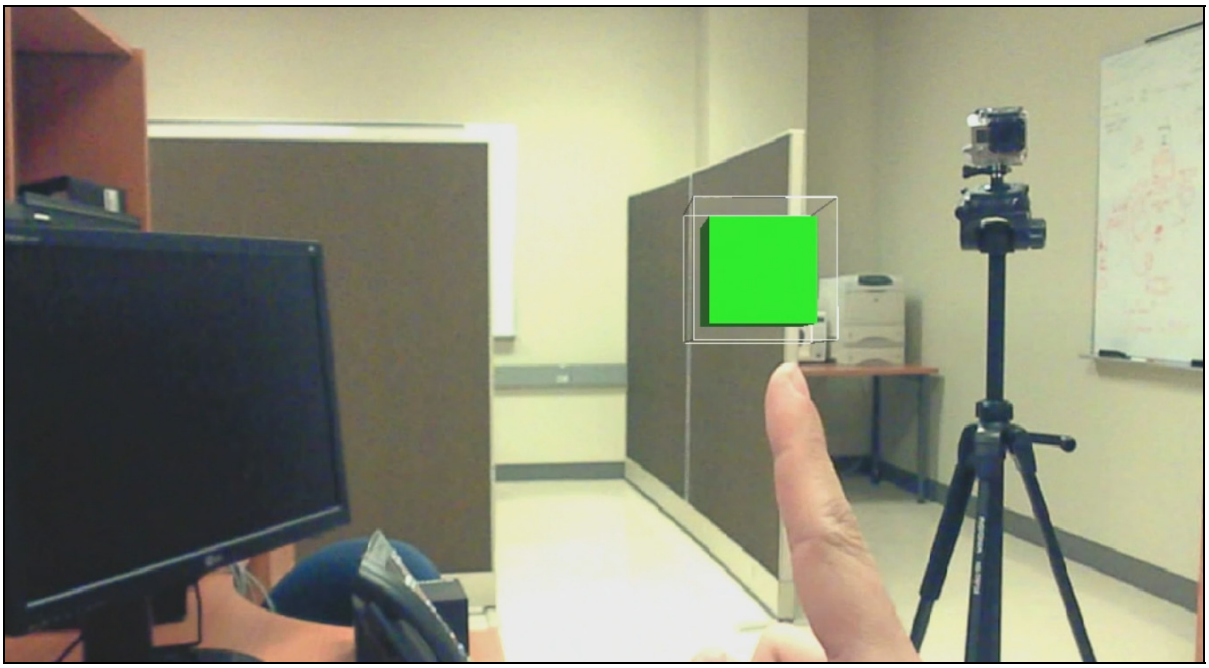

Figure 2.9 Capture de la vision utilisateur avec ARCAD

Le point de vue de l'utilisateur est intéressant pour saisir l'immersion offerte par le système à un individu, mais une vue à la première personne est plutôt limitée, surtout pour constater les profondeurs et les actions de l'usager. Une vue à la troisième personne qui inclue les objets virtuels complèterait bien les autres séquences vidéo, mais peut s'avérer complexe à réaliser. Les deux problèmes principaux sont les suivants : comment ajouter, en temps réel, les objets virtuels au flux vidéo capturé par une caméra, et comment placer ces objets au bon endroit afin de respecter la perspective ? Heureusement, ces deux problèmes sont en partie solutionnés par la librairie de vision ARToolKit. Une extension de cette librairie, conçue pour Unity3D, peut s'injecter dans le sous-système ARCAD et fusionner la capture vidéo d'une webcam au contenu virtuel de la scène 3D. Le tout est projeté sur un coin de l'écran et est capturé par l'enregistreur HDMI pour usage futur. Voilà donc le premier problème solutionné. Pour se situer dans l'espace, ARToolKit fait usage de marqueurs tels que le marqueur Hiro (Figure 2.10). En plaçant une feuille, sur laquelle a été imprimé le marqueur en question, dans la zone d'interaction de ARCAD, et en s'assurant que la webcam est bien calibrée et peut voir le marqueur, la librairie est capable de situer automatiquement la caméra dans l'espace 3D réel et donc de la placer correctement dans l'espace 3D virtuel à condition que le marqueur virtuel concorde avec celui réel pour l'utilisateur. Cette coordination peut être faite lors d'une calibration, en utilisant les touches du clavier pour positionner et angler le marqueur virtuel jusqu'à ce qu'il concorde avec celui réel. Les objets virtuels ajoutés au flux vidéo se retrouvent donc à la bonne position, dans le bon angle et à la bonne taille. La Figure 2.11 présente l'architecture de la capture vidéo augmentée à la troisième personne et la Figure 2.12 est un exemple de cette capture. On y voit l'objet virtuel et aussi le marqueur Hiro, recouvert d'un prisme rouge pour confirmer qu'il est vu par la webcam.


Figure 2.10 Le marqueur Hiro utilisé pour situer la webcam avec ARToolKit

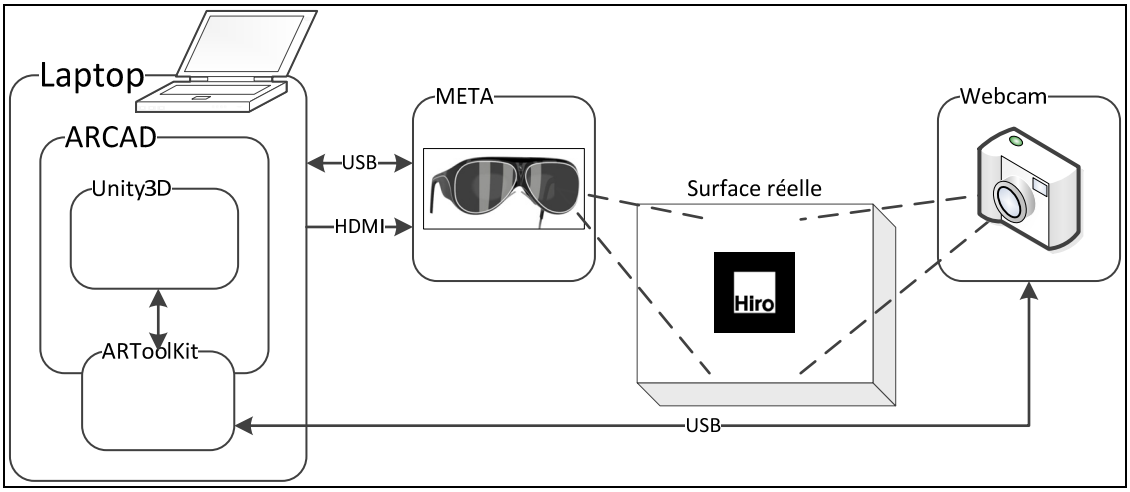


Figure 2.11 Architecture pour l'enregistrement vidéo à la troisième personne

Figure 2.12 Exemple de capture à la troisième personne

## 2.5 Conception

### 2.5.1 Calibrage du téléphone intelligent

Une des premières préoccupations de la conception de DualCAD a été sur comment effectuer le calibrage du sous-système ARCAD. Somme toute, voici les informations que nous espérions obtenir avec la calibration :

- La position et l'orientation du visiocasque au repos (origine);
- La différence de position et d'orientation entre la position réelle du téléphone et celle obtenue avec le suivi en 3D;
- La main qui tient le téléphone (c'est l'autre main qui sera suivie pour la gestuelle);

- La position et l'orientation de l'écran du laptop (pour animer le passage entre DesktopCAD et ARCAD);
- La distance inter-pupillaire (IPD) de l'utilisateur.

La procédure suivante a donc été planifiée pour obtenir toutes ces informations et les stocker. D'abord, lorsque l'utilisateur lance ARCAD, il est accueilli par l'interface du lanceur de ARCAD (Figure 2.13). Celui-ci lui offre la possibilité de choisir une calibration passée pour utiliser le sous-système ou bien d'en créer une nouvelle (ou d'en modifier une existante).

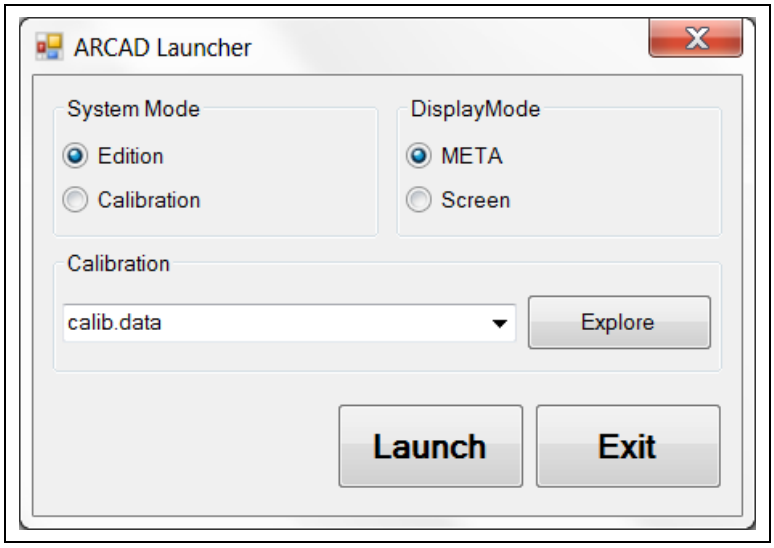


Figure 2.13 Le lanceur pour ARCAD

En lançant la calibration, le système affiche à l'utilisateur une scène vide, c'est-à-dire que le visiocasque n'affiche rien de virtuel et donc l'utilisateur ne voit que la scène réelle devant lui. Il doit alors placer sa tête confortablement en position de travail. C'est cette position et orientation qui sera considérée comme celle de repos et comme origine pour afficher des nouvelles données virtuelles lors de l'utilisation de ARCAD. L'utilisateur appuie sur la touche « Entrée » du clavier du laptop et le logiciel de calibration enregistre ces données pour plus tard et passe à l'étape suivante. S'affiche alors, à 35 centimètres en face de l'utilisateur, un prisme transparent rouge ayant les dimensions du téléphone intelligent (Figure 2.14). Cela représente où placer l'appareil pour calculer la différence entre sa position réelle

(supposément 35 cm devant le visiocasque) et sa position mesurée par les capteurs de mouvements. L'usager a alors du temps pour bien constater l'emplacement du prisme rouge et amener le téléphone à la bonne position et à la bonne orientation afin que ce dernier coïncide avec le prisme virtuel projeté dans sa vision. Il appuie sur « Entrée » et un compte à rebours de quelques secondes, indiqué par une barre de chargement radiale (Figure 2.15), se lance pour s'assurer que le téléphone reste stable. Un trop grand mouvement relancera le compte à rebours.

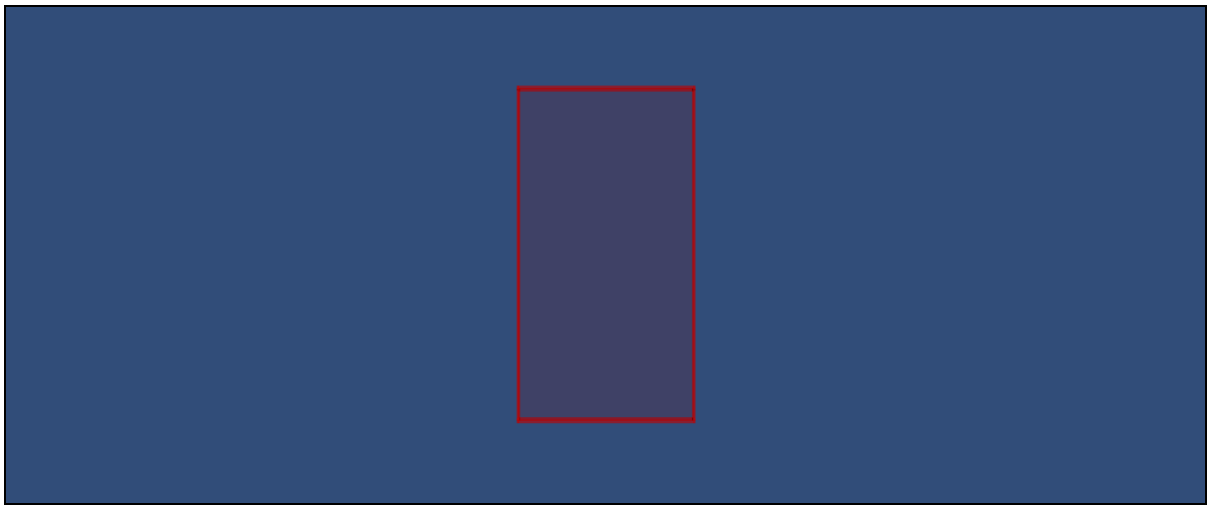

Figure 2.14 Calibration de ARCAD : établissement de l'origine pour le visiocasque

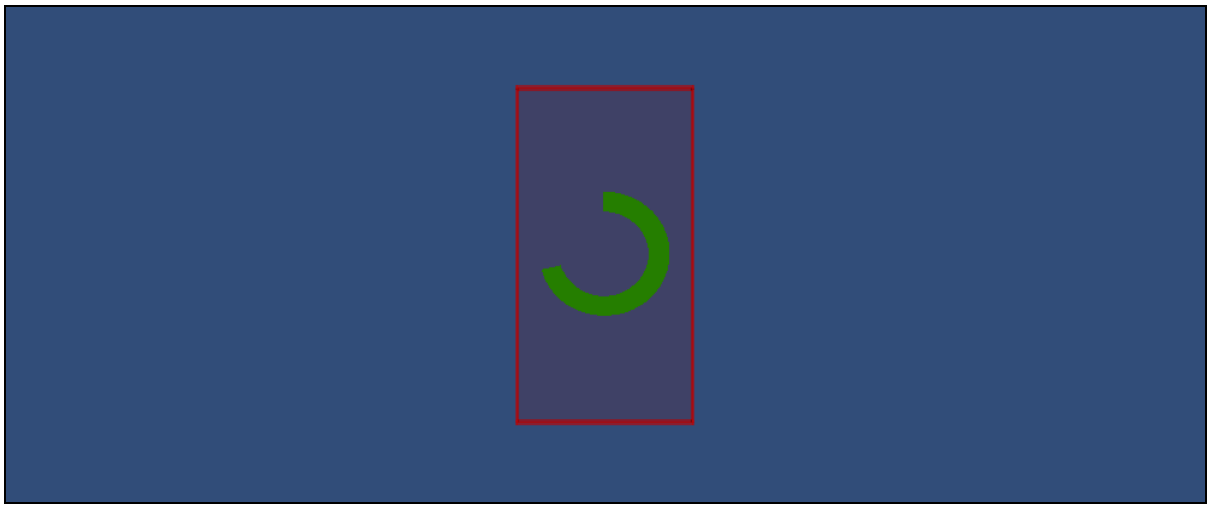

Figure 2.15 Calibration de ARCAD : validation de la position du téléphone

Une fois le compte à rebours terminé, la différence est notée et est tout de suite considérée pour afficher le téléphone virtuel, c'est-à-dire une sorte de fantôme permettant de confirmer visuellement que la position et l'angle virtuels du téléphone correspondent à présent à la réalité. Un autre menu s'affiche aussi, demandant à l'usager de placer l'appareil devant le libellé correspondant à la réalité : gauche pour s'il tient le téléphone dans la main gauche, droite si l'inverse (Figure 2.16). La main qui tient l'appareil est considérée comme la main non-dominante et c'est l'autre main, la main dominante, qui sera suivie par les algorithmes de vision. Un autre compte à rebours laisse le temps à l'utilisateur de bien choisir.

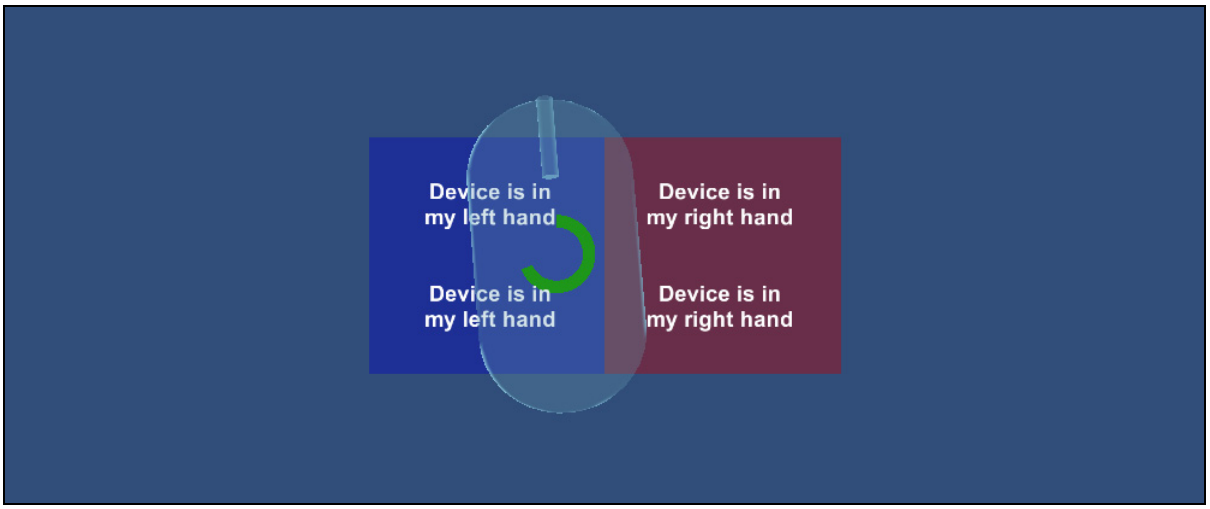


Figure 2.16 Calibration de ARCAD : sélection de la main non-dominante

Ensuite, le précédent menu disparait et un cadre vient s'attacher au fantôme du téléphone intelligent (Figure 2.17). Ce cadre a les dimensions de l'écran du laptop utilisé et doit être positionné de façon à coïncider avec l'écran. Avec la touche « Entrée », la position et l'orientation du cadre virtuel seront enregistrées comme étant celles de l'écran. À noter que, tout au long de la calibration, l'utilisateur peut utiliser les flèches du clavier pour ajuster la distance inter-pupillaire afin d'obtenir un meilleur effet de stéréoscopie.

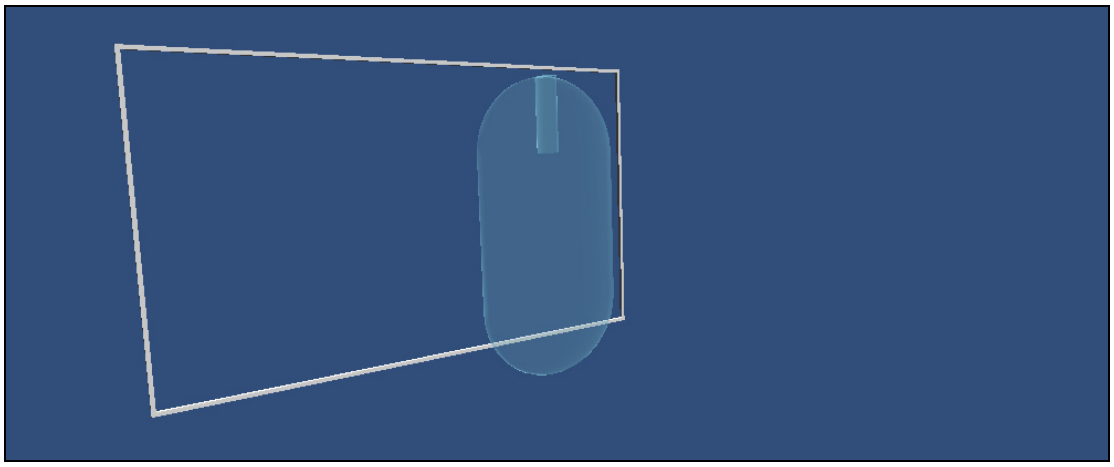

Figure 2.17 Calibration de ARCAD : positionnement de l'écran de laptop

Finalement, une fois toutes les opérations précédentes complétées, les données accumulées sont sauvegardées sous forme de fichier XML qui pourra servir de calibration pour un usage futur de ARCAD. Il n'est pas nécessaire de refaire la calibration, tant que l'utilisateur reste le même et que la borne source et le laptop ne bougent pas.

### 2.5.2 Opérations 3D dans ARCAD

Une autre étape qui nécessitait une tâche considérable de conception était la question des opérations 3D dans ARCAD. L'intérêt d'un système à RA pour la CAO est, comme on l'a vu précédemment, d'offrir des opérations naturelles et intuitives pour permettre des commandes allant jusqu'à 6 degrés de liberté, avec un contrôle direct ou indirect. Pour assister cette tâche, le téléphone intelligent, qui est suivi en 3D, offre une interface physique intéressante. La question est donc, comment peut-on utiliser les déplacements du téléphone pour effectuer des opérations sur des modèles virtuels ? Le déclanchement et la finalisation des commandes peuvent se faire à partir de l'écran tactile et peuvent être envoyés en tant que commandes simples à travers le système de communication établi précédemment, mais comment envoyer des informations plus complexes, par exemple un dessin pour la texture ?

Ainsi il fut décidé d'utiliser une structure spécifique pour tous les échanges d'informations entre le téléphone intelligent et ARCAD. Il s'agit d'une classe C# pouvant être encodée en

XML à partir d'une librairie quelconque avant d'être envoyée, pour être ensuite décodée à l'autre extrémité et interprétée nativement de la bonne façon. Cette classe, nommée SmartphoneCommand, est composée des éléments suivants :

- Un identifiant unique depuis l'exécution actuelle du système, pour permettre la confirmation de commandes critiques;
- Un type de commande, qui est un élément d'une liste et qui donne la nature de la commande;
- Un état, qui représente s'il s'agit d'une commande qui débute, qui se termine, ou qui donne une information sur la façon dont la commande a été saisie;
- Un mode, car une commande peut dépendre d'un contexte;
- Une valeur de retour, réservée pour l'identifiant unique de la commande à laquelle celle-ci répond;
- Une liste de lignes (point de départ en 2D, point d'arrivée en 2D, couleur, largeur) permettant de refaire un dessin ou un polygone;
- Une multitude de plages réservées à des formats distincts : un vecteur 3D, une chaîne de caractères, une couleur, une valeur à point flottant, une valeur booléenne.

La Figure 2.18 présente un exemple de cette structure sous la forme XML.

```
1  <?xml version="1.0" encoding="utf-16"?>
2  <SmartphoneCommand xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3      xmlns:xsd="http://www.w3.org/2001/XMLSchema">
4    <commandId>3</commandId>
5    <commandType>SwitchCAD</commandType>
6    <state>Released</state>
7    <mode>NoSelection</mode>
8    <text />
9    <coord>
10     <x>0</x>
11     <y>0</y>
12     <z>0</z>
13   </coord>
14   <color>1</color>
15   <lines />
16   <floatValue>0</floatValue>
17   <returnValue>0</returnValue>
18   <boolValue>false</boolValue>
19 </SmartphoneCommand>
```

Figure 2.18 Exemple de SmartphoneCommand : passage à DesktopCAD

Cette structure peut devenir relativement grande, surtout lorsqu'elle inclue une liste de lignes (Figure 2.19), mais elle a l'avantage d'être complète, solide et le système entier n'a besoin que de connaître cette classe pour interpréter n'importe quelle commande.

```
<?xml version="1.0" encoding="utf-16"?>
<SmartphoneCommand xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <commandId>9</commandId>
  <commandType>Drawing</commandType>
  <state>Released</state>
  <mode>Draw</mode>
  <text />
  <coord>
    <x>0</x>
    <y>0</y>
    <z>0</z>
  </coord>
  <color>1</color>
  <lines>
    <Line2D>
      <start>
        <x>0.116477273</x>
        <y>0.269602269</y>
      </start>
      <end>
        <x>0.116477273</x>
        <y>0.269602269</y>
      </end>
      <color>3</color>
      <width>3</width>
    </Line2D>
    <Line2D>
      <start>
        <x>0.116477273</x>
```

Figure 2.19 Exemple de SmartphoneCommand : extrait d'un envoi d'un dessin

Avec une telle structure, on peut utiliser quelques patrons pour faciliter son usage. Par exemple, au lieu de joindre manuellement toutes les options du menu radial, disponible sur l'écran tactile du téléphone, à une commande et un mode de SmartphoneCommand, il est possible d'utiliser directement la structure de cette classe pour peupler les menus et laisser la logique interne gérer les transitions et l'envoie de commandes. La Figure 2.20 représente un exemple de cette structure. On y voit que, pour chaque mode (mode où aucune sélection n'est

faite, mode pour la création de primitives, etc.), une série de tuples représentent les options possibles du menu à afficher : le titre de l'option, la commande à exécuter, le changement de mode à faire et l'état à considérer pour effectuer l'opération. Par exemple, certaines options doivent être déclenchées dès qu'on pose notre doigt au-dessus (Touched) alors que d'autres attendent qu'on relâche le doigt (Released). Une telle structure centralise toutes les opérations possibles du téléphone intelligent.

```
// ListOfCommands
// ************************************************************
ListOfCommands = new List<Tuple<string, Commands, Modes, States>>[(int)Modes.Count];
// NoSelection
ListOfCommands[(int)Modes.NoSelection] = new List<Tuple<string, Commands, Modes, States>>
{
    new Tuple<string, Commands, Modes, States>("DesktopCAD",        Commands.SwitchCAD,         Modes.NoChange,         States.Released),
    new Tuple<string, Commands, Modes, States>("Options",           Commands.None,              Modes.Options,          States.Released),
    new Tuple<string, Commands, Modes, States>("Draw Mesh",         Commands.None,              Modes.DrawMesh,         States.Released),
    new Tuple<string, Commands, Modes, States>("Create Primitive >",Commands.CreatePrimitive,   Modes.CreatePrimitive,  States.Touched),
    new Tuple<string, Commands, Modes, States>("Move Scene",        Commands.MoveScene,         Modes.SceneTrans,       States.Released),
    new Tuple<string, Commands, Modes, States>("Direct Transform >",Commands.Transform,         Modes.DirectTransform,  States.Touched)
};
// CreatePrimitive
ListOfCommands[(int)Modes.CreatePrimitive] = new List<Tuple<string, Commands, Modes, States>>
{
    new Tuple<string, Commands, Modes, States>("Cancel",            Commands.Cancel,            Modes.NoSelection,      States.Released),
    new Tuple<string, Commands, Modes, States>("Cube",              Commands.CreateCube,        Modes.NoChange,         States.Released),
    new Tuple<string, Commands, Modes, States>("Sphere",            Commands.CreateSphere,      Modes.NoChange,         States.Released),
    new Tuple<string, Commands, Modes, States>("Cylinder",          Commands.CreateCylinder,    Modes.NoChange,         States.Released),
    new Tuple<string, Commands, Modes, States>("Plane",             Commands.CreatePlane,       Modes.NoChange,         States.Released),
    new Tuple<string, Commands, Modes, States>("Text Board",        Commands.CreateTextBoard,   Modes.NoChange,         States.Released),
    new Tuple<string, Commands, Modes, States>("Build Prism",       Commands.None,              Modes.BuildPrism,       States.Released)
};
// ModelSelected
ListOfCommands[(int)Modes.ModelSelected] = new List<Tuple<string, Commands, Modes, States>>
```

Figure 2.20 Utilisation de SmartphoneCommand pour le menu radial

Au-delà de ces possibilités, on peut aussi utiliser SmartphoneCommand pour enregistrer un *callback* dans le système. Au lieu d'avoir chaque classe de transformations qui doit écouter les commandes entrantes, les décoder et les interpréter, les classes peuvent s'abonner, auprès d'un outil, à un ou plusieurs types de commandes et elles en seront informées lorsqu'une commande respectant leur critère sera reçue. L'envoie de commande passe aussi par cet outil : n'importe quelle classe peut bâtir sa propre commande et la transmettre à l'outil, qui s'arrange pour la communiquer.

Donc, avec la communication prise en charge, il devient facile pour une classe devant se charger d'une transformation, d'obtenir le déclanchement de la commande ainsi que les informations relatives au positionnement du téléphone intelligent (voir section

Implémentation pour plus de détails) pour accomplir la transformation désirée par l'utilisateur. Normalement le traitement est relativement simple : constater les coordonnées et orientation au début de l'opération, mesurer la différence avec les coordonnées et orientation actuelles, et appliquer cette différence sur l'objet à éditer. Une rotation de 90 degrés sur un axe du téléphone peut donc être convertie à une rotation de 90 degrés sur le même axe pour le modèle.

### 2.5.3 Tests utilisateurs

Comme il était initialement prévu d'accomplir des tests utilisateurs complets (rappel : ceux-ci ont été remplacés par une démonstration auprès d'utilisateurs experts), une démarche avait été prévue et donc le système avait été conçu en conséquent. L'enjeu était de déterminer comment fournir des tâches à un utilisateur pour ensuite mesurer le temps qu'il prendrait pour chaque tâche. ARCAD contient donc un menu de tâches. Lorsqu'un individu lance les tâches associées à la modélisation ou à l'assemblage (les deux modes que nous voulions observer), le système modifie aléatoirement l'ordre d'une liste préétablie de tâches et commence par afficher la première à l'utilisateur (Figure 2.21). S'affiche d'abord une image du résultat escompté ainsi qu'une description (a). L'utilisateur prend son temps pour comprendre le travail à faire, minimise l'image d'un clic et constate la scène de départ (b). Lorsqu'il est prêt, il peut appuyer sur « Start Task » et un compteur se lance (c). Une fois qu'il croit avoir terminé la tâche, il appuie sur « Done » et le compteur se met sur pause. Le validateur observe alors le travail accompli et, s'il confirme que la tâche est complète, appuie sur « Next Task » (d) pour enregistrer le temps écoulé et passer à la tâche suivante. Si la tâche n'est pas réellement finie, l'utilisateur peut reprendre son travail jusqu'à ce que ce soit le cas. Une fois les tâches réalisées, un rapport est produit.

Les tâches elles-mêmes sont stockées sous forme d'un fichier XML (Figure 2.22) qui contient les informations pertinentes à lancer la tâche, mais aussi des mots clés quant au travail qu'on désire évaluer. Le rapport de tâches est aussi sous la forme d'un fichier XML (Figure 2.23) et, en plus de contenir le temps requis à l'accomplissement de chaque tâche,

contient la liste des évènements pour chaque tâche, avec un *timestamp*, pour qu'on puisse plus aisément déterminer les opérations ayant pris le plus de temps.

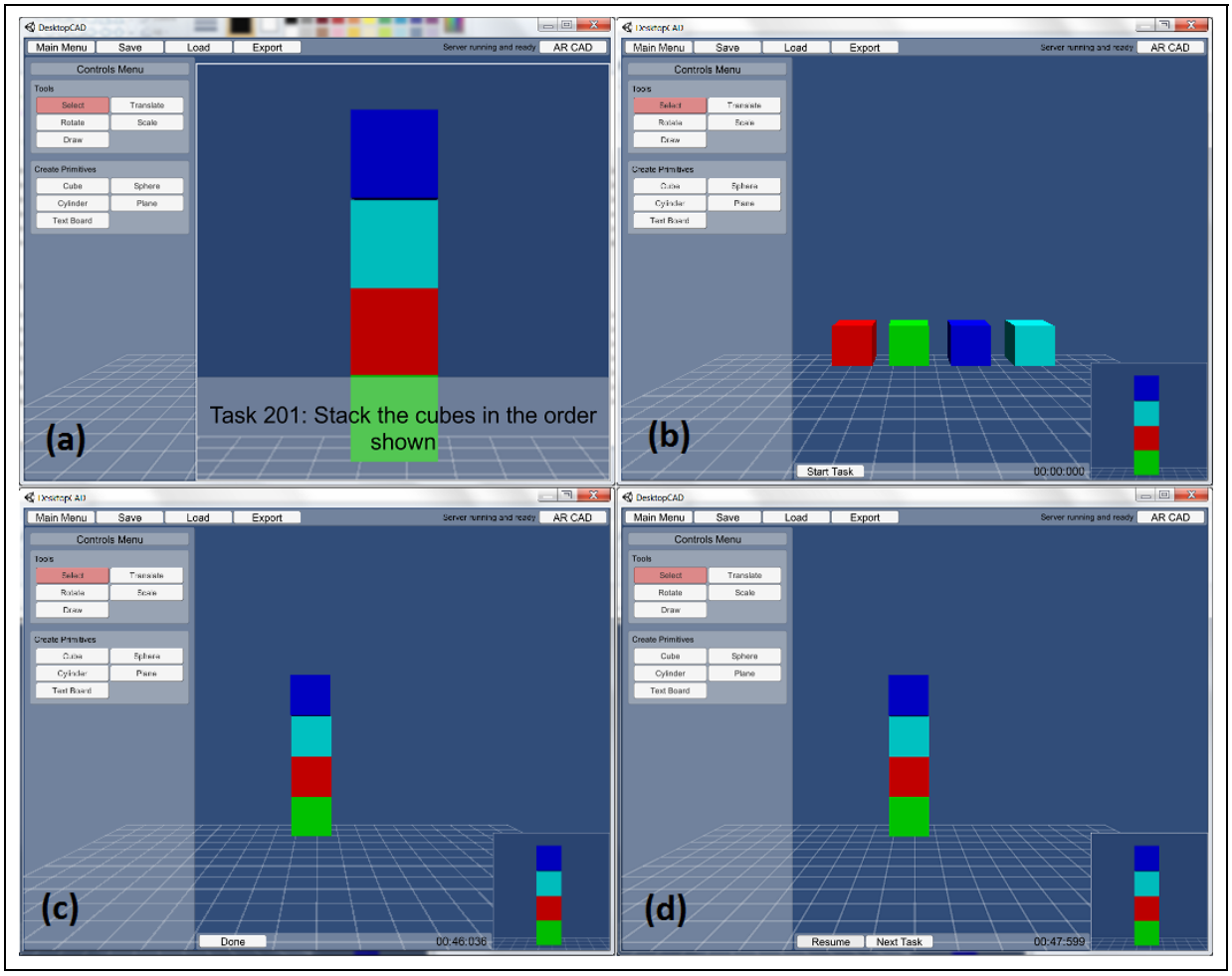


Figure 2.21 Une tâche pour les tests utilisateurs

```xml
<?xml version="1.0" encoding="utf-16"?>
<Tasks xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <tasksList>
    <ArrayOfTask>
      <Task>
        <ID>101</ID>
        <description>Black cube, twice as high as wide</description>
        <tags>create_cube;color;scale</tags>
        <imageName>task_image_101.png</imageName>
        <startFile></startFile>
        <endFile />
        <time>0</time>
        <eventsLog />
      </Task>
      <Task>
        <ID>102</ID>
        <description>Green sphere, twice as wide as high</description>
        <tags>create_sphere;color;scale</tags>
        <imageName>task_image_102.png</imageName>
        <startFile></startFile>
        <endFile />
        <time>0</time>
        <eventsLog />
      </Task>
      <Task>
        <ID>103</ID>
        <description>Cylinder, twice rotated</description>
```

Figure 2.22 Fichier XML des tâches pour les tests utilisateurs

```xml
    <Task>
      <ID>119</ID>
      <description>Recreate the flag of bulgaria with these 3 text boards</description>
      <tags>color;write</tags>
      <imageName>task_image_119.png</imageName>
      <startFile>task_119.data</startFile>
      <endFile />
      <time>21.91389</time>
      <eventsLog>
        <Logs>
          <side>DesktopCAD</side>
          <events>
            <string>time=003.63;command='ColorSelection';mode='DesktopCAD';state='1'</string>
            <string>time=007.69;command='TextChange';mode='DesktopCAD';state=' '</string>
            <string>time=009.70;command='ColorSelection';mode='DesktopCAD';state='3'</string>
            <string>time=012.37;command='TextChange';mode='DesktopCAD';state=' '</string>
            <string>time=013.61;command='ColorSelection';mode='DesktopCAD';state='4'</string>
            <string>time=019.71;command='TextChange';mode='DesktopCAD';state='Bulgaria'</string>
          </events>
        </Logs>
      </eventsLog>
    </Task>
```

Figure 2.23 Extrait d'un rapport de tests utilisateurs

## 2.6 Implémentation

### 2.6.1 Obtention des coordonnées du Polhemus en C#

Pour faire usage des données (coordonnées, rotations) captées par les senseurs du Polhemus Tracker, il fallait établir une communication entre la librairie de Polhemus, en C, et le cœur du système, en C#. L'idéal recherché était une solution permettant de faire des appels natifs en C# pour appeler des méthodes de la librairie Polhemus et recevoir les données désirées. Pour arriver à une solution convenable, deux caractéristiques des technologies présentes furent utilisées :

- Tous les langages .NET (et même Mono C#), lorsque compilés, sont exécutés par le *Common Language Runtime* (CLR) et, si deux programmes respectent certaines conditions, l'un d'eux peut faire des appels natifs à l'autre comme s'il s'agissait d'une librairie;
- Unity Pro permet l'importation de librairies DLL pour faire des appels natifs depuis le code vers celles-ci.

La solution devient alors évidente et il ne reste que des détails d'implémentation : en faisant un *wrapper* C/C++ pour la librairie Polhemus et en le compilant sous forme de librairie DLL, cette librairie peut être placée dans le projet ARCAD et être appelée pour obtenir les données pertinentes. Les méthodes du *wrapper* doivent être structurées de la bonne façon afin qu'elles puissent être appelées de n'importe quel code CLR et il est important alors de gérer les conversions de type entre les langages, autant dans le *wrapper* que dans le code C#. Une simple interface, en C#, permet de gérer ces détails techniques et d'offrir des méthodes équivalentes plus simples.

### 2.6.2 Conversion des coord. Polhemus aux coord. du monde virtuel

Tel que mentionné précédemment (Défis techniques), le système de coordonnées de Polhemus n'est pas le même que celui de Unity : main droite vs. main gauche, axes différents, ordre des angles d'Euler. La conversion se fait mathématiquement, mais n'est pas

triviale pour autant. D'abord les axes sont convertis d'un système à l'autre, en s'assurant d'inverser la valeur du 3ème axe (UP), passant effectivement d'un système main droite à un système main gauche. Les rotations sont corrigées en outrepassant toutes les méthodes et structures classiques pour gérer les orientations (les quaternions par exemple) et en effectuant les rotations, un axe à la fois, dans l'ordre défini par le système de coordonnées de Polhemus. L'orientation obtenue est alors lue par Unity selon son système de coordonnées et peut être récupéré pour avoir les angles d'Euler (ou un quaternion) convertis. Ces coordonnées et orientations peuvent ensuite être traitées avec les paramètres obtenus lors de la calibration afin de bien situer les objets dans l'espace virtuel.

### 2.6.3 Opérations 3D dans DesktopCAD

Contrairement aux opérations 3D avec ARCAD, qui sont des enjeux de conception, les techniques de manipulation 3D avec clavier & souris sont connues et ont déjà fait leur preuve. Il s'agit donc de détails techniques qui portent sur comment implémenter ces techniques dans le logiciel DesktopCAD.

La souris est utilisée pour les déplacements de la caméra dans l'espace : un clic droit maintenu permet de faire tourner la caméra pour suivre le curseur et, avec les touches WASD, de se déplacer dans la scène virtuelle. Les touches directionnelles permettent de faire tourner la caméra autour d'un objet sélectionné et la roulette rapproche ou éloigne la caméra de l'objet. La souris est aussi utilisée pour cliquer sur les contrôles des menus, sélectionner des couleurs, créer une primitive et dessiner sur une surface.

Le défi le plus notable porte sur les opérations possibles sur les primitives : translation, rotation et redimensionnement. Il a été décidé d'utiliser des widgets, c'est-à-dire des composantes s'ajoutant à une primitive et offrant des contrôles pour effectuer les transformations désirées. Dans chacun des trois cas, les widgets permettent d'effectuer une opération sur un seul axe à la fois, ralentissant légèrement les opérations complexes, mais

offrant beaucoup plus de précision. La Figure 2.24 présente le widget de rotation : 3 cercles, chacun perpendiculaire à un axe, permet d'effectuer une rotation selon cet axe.

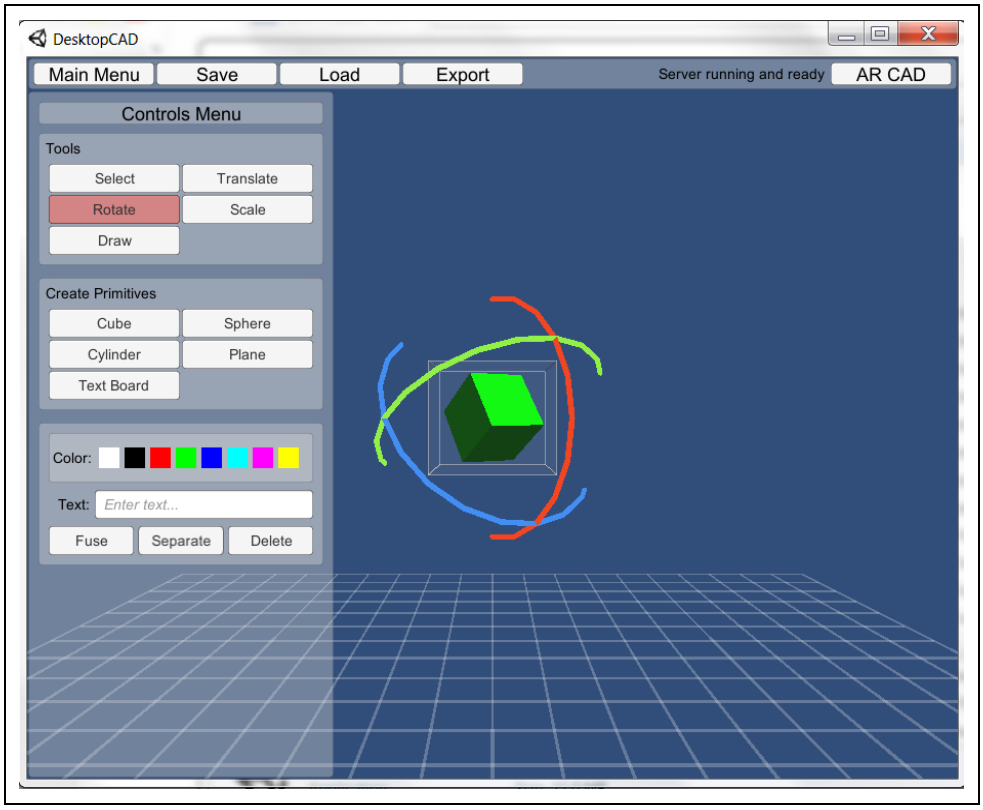


Figure 2.24 Widget de rotation dans DesktopCAD

Dans les trois transformations, la souris est utilisée pour cliquer sur un des éléments du widget et ensuite glisser dans un sens ou dans l'autre, entraînant une transformation naturelle et prévisible. La détection d'un clic se fait à partir d'un tracé de rayon et le sens et la magnitude du glissement se calculent à partir de la tangente 3D de l'élément du widget sélectionné. Les widgets s'affichent pour une seconde caméra qui effectue son rendu après la caméra principale et donc les widgets apparaissent toujours au-dessus des primitives (pas d'occlusion). Le widget de rotation ajoute une particularité : seule la moitié nous faisant face de la sphère du widget est affichée, afin de ne pas mélanger l'utilisateur. Cet effet est obtenu en plaçant une surface qui ne donne pas de rendu, mais qui est tout de même considérée dans le calcul d'occlusion du shader particulier du widget.

### 2.6.4 Encodage des modèles

Tous comme les commandes envoyées entre le téléphone intelligent et ARCAD, il est nécessaire de pouvoir encoder les modèles et leurs paramètres (position, couleur, texture, etc.) afin de transmettre les détails de la scène entre ARCAD et DesktopCAD ou bien d'enregistrer la dite scène pour la charger plus tard. Ce transfert se fait avec le protocole TCP et l'enregistrement n'est qu'une question de sauvegarder un fichier avec l'encodage, mais il est important de choisir un *parser* (codeur/décodeur) approprié. Si un codeur XML suffisait pour les commandes, il faut quelque chose de plus approprié aux caractéristiques complexes des modèles 3D pour ce problème. Le module *Unity Serializer* (Talbot, 2012) a donc été choisi pour son large éventail de possibilités. Bien que ce module soit fait à la base pour sauvegarder et charger une scène entière, il a été possible, après modifications, d'encoder un seul élément du graphe de scène ainsi que ses enfants. Le module ne sauvegarde malheureusement pas tous les détails, mais ceux-ci peuvent être stockés dans un objet sous la forme de données primitives pour ensuite être reconstruits avec un script lors du chargement. Le tout donne donc un encodeur relativement solide dans les contraintes de ce projet, mais qui ne pourrait vraisemblablement pas s'adapter directement à l'ajout de caractéristiques sur les modèles.

## 2.7 Démonstration de DualCAD à des utilisateurs experts

Suite à la finalisation du prototype DualCAD, celui-ci fut présenté à des utilisateurs experts. 5 individus affiliés à l'Université du Québec en Abitibi-Témiscamingue (UQAT), soit 3 enseignants et 2 étudiants, purent observer le système et le tester. Chaque testeur se considérait comme expert en logiciels de modélisation 3D et possédait de 3 à 10 ans d'expérience avec des logiciels comme *Maya*, *3DS Max* ou *ZBrush*, en plus d'avoir une expérience préalable avec un visiocasque à RV ou RA. Une période d'environ 45 minutes fut allouée à chaque testeur et se déroula comme suit : d'abord le prototype fut introduit en présentant les fonctionnalités offertes et leur utilisation, puis le testeur utilisa le système pour accomplir des tâches prévues. Les commentaires faits par les testeurs lors de la démonstration furent notés et certaines questions, portant sur l'expérience antérieure des

testeurs ainsi que sur leur opinion du système, furent posées. Tous les résultats furent recueillis et sont disponibles à la section 3.3 Résultats de la démonstration aux utilisateurs experts.

# CHAPITRE 3

# RÉSULTATS

Comme le but de ce mémoire était de (1) produire un prototype d'une application fusionnant les interfaces traditionnelles de bureau et celles de RA pour effectuer un travail de CAO, de (2) comparer et explorer des techniques d'interaction novatrices pour l'environnement de RA et (3) d'analyser le système à l'aide de tests informels, les résultats du projet peuvent être subdivisés en ces 3 catégories. D'abord une analyse qualitative de chacun des modes et de leurs interactions a été faite et est présentée ici. Ensuite se retrouve un bilan des techniques d'interaction pour ARCAD représentant un certain degré d'innovation ainsi qu'une analyse qualitative de celles-ci. La troisième section présente le résultat de la présentation du prototype aux utilisateurs experts. Il est aussi pertinent de revenir aux taxonomies des Tableau 1.2 et Tableau 1.3 pour constater la position qu'occupe DualCAD par rapport aux travaux antérieurs.

## 3.1 Les deux modes

Le passage entre DesktopCAD et ARCAD se fait avec un bouton placé en haut à droite de l'interface du logiciel de bureau et le passage de ARCAD à DesktopCAD se fait un glissement du doigt sur l'écran tactile du téléphone. L'objectif était d'exiger une seule opération non-ambiguë pour effectuer le transfert et donc l'utilisateur peut lancer l'opération à tout moment, sauf lors d'opérations complexes qui se doivent d'être achevées d'abord. Cette façon de faire se distingue d'autres techniques proposées par des travaux antérieurs, notamment celles suggérant d'effectuer le transfert automatiquement lorsque l'attention de l'utilisateur passe d'un environnement à l'autre. Par exemple, on pourrait penser que si l'usager tourne la tête pour ne plus faire face à son écran, c'est qu'il veut passer en mode RA. Cela peut toutefois causer de l'ambiguïté puisque les mouvements de la tête ne sont pas toujours reliés au travail en cours. De plus, cela entraîne un angle mort de RA dans la zone où se trouve l'écran d'ordinateur puisque l'utilisateur ne peut pas placer un objet virtuel à cet endroit sans passer automatiquement en mode Bureau. D'autres techniques intéressantes se

basent sur la gestuelle pour détecter la volonté de l'utilisateur de changer de mode. Par exemple celui-ci pourrait effectuer une geste de pincement devant son écran pour sélectionner un objet et le faire passer dans le monde de la RA. Cette approche est plus lente qu'un clic, mais pourrait être considérée plus naturelle et a l'avantage de placer directement l'utilisateur en position pour déplacer l'objet. Somme toute, un compromis de rapidité, facilité et fiabilité fut atteint par notre approche.

Une analyse qualitative et des tests préliminaires ont démontré que, comme on s'y attendait, le mode DesktopCAD offre beaucoup plus de précision grâce à la souris et est souvent plus rapide que ARCAD lorsqu'il s'agit d'opérations relativement simples (translations, rotations selon un seul axe, etc.). De plus, certaines opérations comme l'édition de texte est nettement plus rapide et agréable avec un clavier qu'avec un écran tactile ou un clavier virtuel. À l'autre extrémité, ARCAD domine pour les opérations ne pouvant pas être faites naturellement par la souris, comme les rotations sur plusieurs axes. Tel que décrit plus bas, certaines opérations avec contact directs n'ont tout simplement pas d'équivalent avec le combo souris-clavier et donc les transformations sont plus prévisibles avec ARCAD.

Tout cela montre bien que chaque mode a ses forces et faiblesses et qu'il y aurait une valeur à permettre de choisir le mode en fonction de la tâche à réaliser, peut-être en effectuant d'abord des opérations brutes avec les techniques de RA pour ensuite préciser le tout avec l'environnement de bureau. Là où les résultats sont incertains est lorsqu'il s'agit de prouver que le temps et l'effort requis pour passer d'un mode à l'autre est inférieur au temps et à l'effort gagné en faisant chaque tâche dans son environnement optimal. Il n'est pas certain que, bien qu'une rotation se fasse mieux avec ARCAD, il soit préférable pour un utilisateur de changer de mode pour cette seule tâche. Il s'agirait en effet d'un objectif intéressant pour un travail futur.

## 3.2 Techniques d'interaction

### 3.2.1 Écran tactile & stylet

Sans pour autant s'attarder sur les téléphones intelligents, le fait d'utiliser un dispositif muni d'un écran tactile et d'un stylet ouvre beaucoup de portes quant aux interactions possibles. Comme il a été démontré que les stylets étaient de meilleurs outils pour effectuer des dessins à main levée, des techniques d'interaction faisant usage de ces propriétés pouvaient être introduites dans le système ARCAD. Si DesktopCAD permet de dessiner sur une surface avec la souris, la fonctionnalité équivalente sur ARCAD est bien plus agréable, à l'aide du stylet. Le stylet est aussi utilisé pour dessiner des formes géométriques comme des polygones qui serviront de base à des techniques d'interaction explorées plus loin (Figure 3.1).

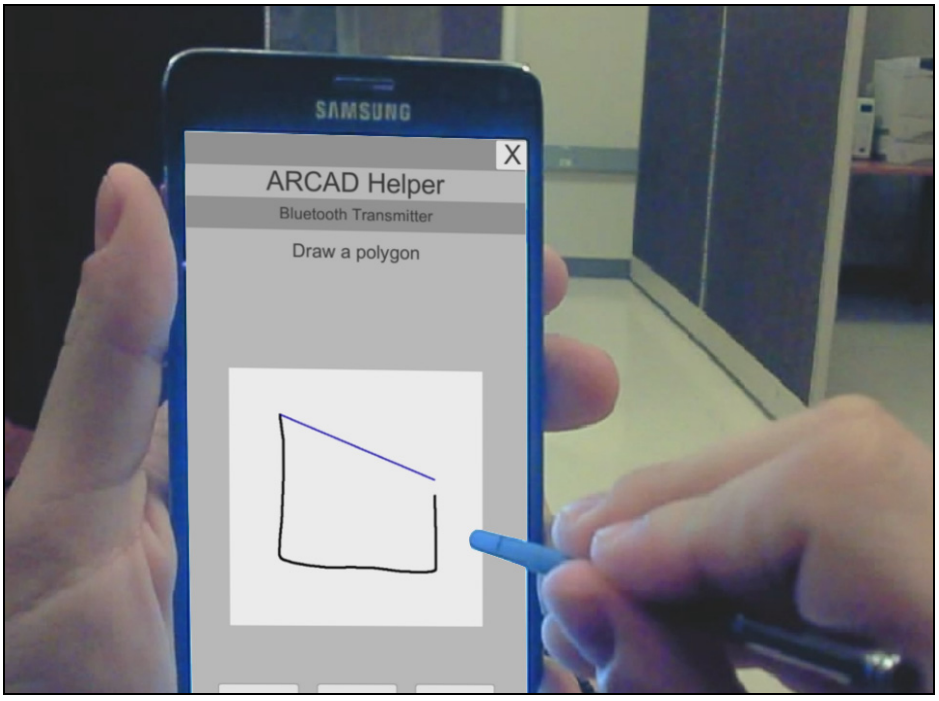


Figure 3.1 Dessiner un polygone avec le stylet

Mais un écran tactile n'a pas besoin de se limiter aux interactions avec un stylet et peut faire usage des doigts de l'utilisateur pour de nombreux contrôles. De simples boutons ont été ajoutés à l'interface du téléphone intelligent pour aller au-delà des boutons physique du

dispositif et l'application mobile fait usage du toucher à plusieurs moments pour que l'utilisateur puisse signaler le démarrage ou l'arrêt d'une transformation. Mais ce sont les opérations de glissement qui furent les plus utiles et intéressantes pour interagir avec ARCAD. Un menu radial (Figure 3.2), se modifiant en fonction du contexte, permet à un utilisateur expert de glisser son doigt sur la surface de son téléphone et d'effectuer des opérations dans même avoir à le regarder. Comme le centre d'attention se veut être la scène réelle augmentée des objets virtuels, il est donc utile d'avoir des contrôles qui ne nécessitent pas à l'utilisateur de baisser ou tourner la tête. Un utilisateur novice aura besoin de regarder l'écran pour apprendre les différents menus offerts, mais ne sera pas plus désavantagé que s'il s'agissait de simples boutons, en plus de graduellement devenir un utilisateur expert. D'autres fonctionnalités font aussi usage d'un glissement de doigt, par exemple pour définir avec précision une opération d'extrusion (à l'instar d'un *slider*) ou pour effectuer des rotations sur un objet sélectionné et placé devant le dispositif, comme si on poussait un objet réel.

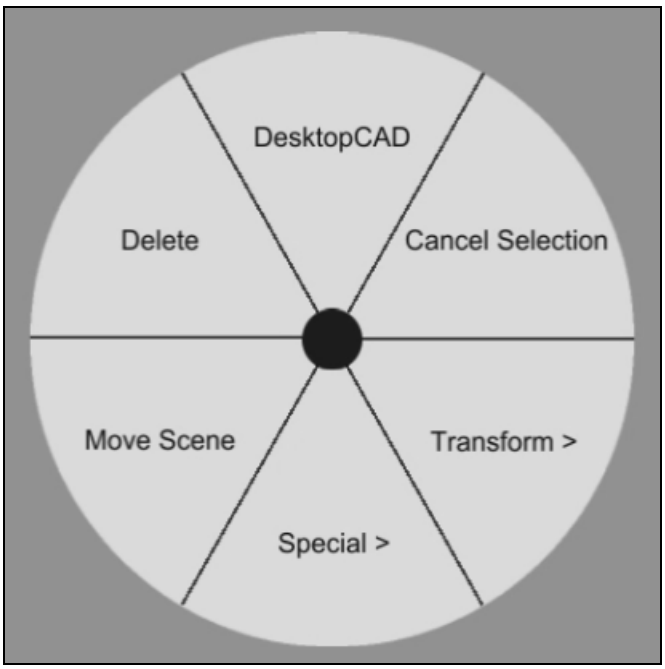


Figure 3.2 Menu radial sur le téléphone intelligent

### 3.2.2 Combinaison de la RA et d'un téléphone intelligent

L'utilisation d'une surface tactile et d'un stylet est intéressante dans certains contextes, mais rien n'empêche l'emploi de telles technologies dans un environnement de bureau. L'apport réel des fonctionnalités faisant usage du téléphone intelligent est plus en lien avec les limitations des casques à RA actuels. En effet, des tests préliminaires ont démontré que le matériel utilisé n'offrait pas une résolution suffisante pour permettre la lecture de petits caractères, n'était pas tout à fait stable et donc rendait difficile la lecture mais aussi l'entrée de commandes, et souffrait de problèmes d'occlusion et de champ de vision limité rendant certains contrôles inutilisables.

Le téléphone intelligent s'est avéré être un outil de choix pour combler les lacunes d'un dispositif à RA. Toute information détaillée peut être affichée sur l'écran du téléphone et la lecture se fait alors aisément par l'utilisateur. Un désavantage apparent est que l'usager ne regarde pas en permanence l'écran du téléphone et pourrait donc manquer des informations importantes, mais il est possible d'alléger la problématique en affichant en grands caractères dans la RA l'information ou bien une indication demandant à l'usager de porter attention au téléphone. Des tests initiaux ont confirmé qu'il était préférable d'afficher les messages sur le téléphone.

Si les boutons logiciels d'un écran tactiles sont généralement moins fiables que ceux physiques, ils demeurent néanmoins préférables à ceux de la RA. Du moins c'est ce que montrèrent nos expérimentations initiales. La tâche d'appuyer sur un bouton virtuel flottant dans l'espace exige un mouvement ample du bras et au moins une seconde, en plus d'être prône à l'erreur (les utilisateurs novices rataient fréquemment leur cible et pouvaient même toucher d'autres boutons). À l'opposé, les boutons du téléphone intelligent peuvent être appuyés avec un léger mouvement en une fraction de seconde et le taux d'erreur est faible, surtout lorsque l'utilisateur a le menu en vue. De plus, les menus glissants permettent d'enchaîner encore plus rapidement des commandes, surpassant davantage l'input par gestuelle pour le casque à RA.

Finalement, l'input avec le téléphone intelligent ne nécessite pas le *line-of-sight*, c'est-à-dire le fait que l'utilisateur regarde expressément le dispositif. Les techniques par gestuelle reconnus par vision ne fonctionnent que lorsque la caméra peut voir les mains et n'est donc pas approprié dans des cas d'occlusion (un objet cache la main) ou lorsque la caméra est pointée ailleurs. L'outil de téléphone intelligent développé pour ARCAD peut être employé dans n'importe quel cas et l'utilisateur n'a pas besoin de le regarder. Il a été remarqué que les usagers préféraient souvent tenir le téléphone près de leurs jambes, hors de leur champ de vision, pour ne pas entraîner de fatigue musculaire ou bloquer leur vue. Cela est donc possible par le seul fait d'avoir employé un dispositif additionnel.

Finalement, pour en revenir aux motivations, le téléphone intelligent a été choisi car il s'agit d'un outil accessible par une grande portion de la population et qu'il est assez léger et petit pour être tenu dans une seule main. Nos observations vont en accord avec le fait que la taille était suffisante pour les tâches à réaliser.

### 3.2.3 Suivi avec 6 degrés de liberté du téléphone intelligent

Si l'écran tactile du téléphone est des plus utile pour entrer des commandes simples ou démarrer/arrêter des transformations, il faut encore développer des techniques pour réaliser ces transformations en 3D. À la base, les trois opérations principales à réaliser sont la translation, la rotation et la mise à l'échelle (scaling). La translation seule peut se faire grâce au suivi des mains et sera abordé dans la prochaine sous-section. Les deux autres opérations sont plus complexes à réaliser.

Prenons la rotation en 3D d'abord. Avec DesktopCAD celle-ci est réalisée à l'aide de widgets, mais il a été démontré que ceux-ci, bien que précis, ne sont pas naturels pour un utilisateur et ARCAD se veut être le plus intuitif et rapide possible. Les algorithmes de vision ont de la difficulté à distinguer avec précision la rotation d'un objet (par exemple une main) et donc c'est généralement une technique d'interaction à deux mains qui est employée. Or,

cette technique est sensible à l'occlusion car une main peut cacher la seconde et manque généralement de précision. Hinckley (Hinckley et al., 1994) a démontré en 1993 l'intérêt d'utiliser des interfaces physiques pour réaliser des rotations sur un objet virtuel. Comme le téléphone est suivi avec 6 degrés de liberté, il peut servir d'interface physique : une fois l'objet sélectionné et la transformation de rotation lancée, une rotation quelconque du téléphone sera reproduite sur l'objet (Figure 3.3 c et d). On obtient donc une technique d'interaction que les utilisateurs décrivent d'intuitive, rapide et relativement précise. De plus, le téléphone peut manipuler l'objet à distance, réduisant la fatigue musculaire engendrée par la transformation.

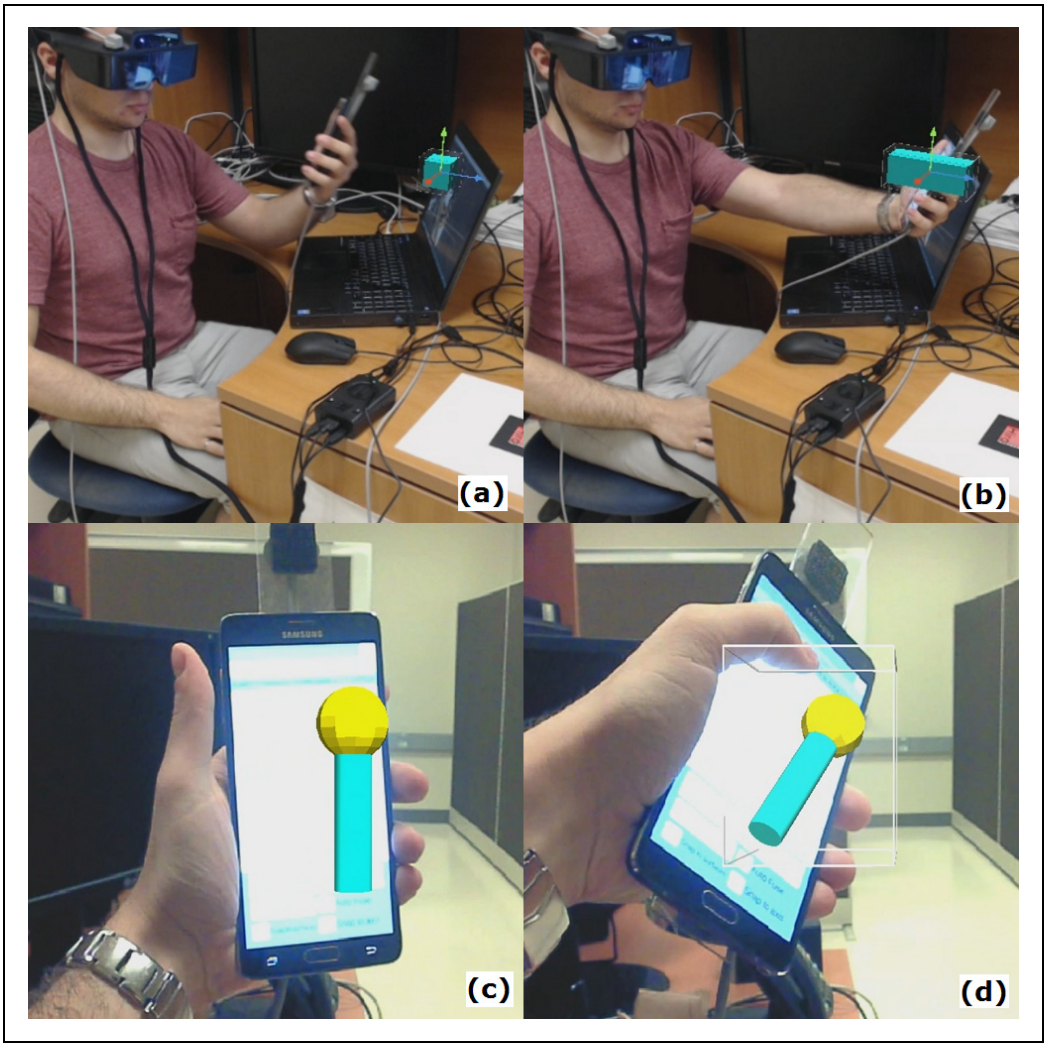


Figure 3.3 Usage du suivi en 3D du téléphone intelligent dans ARCAD

La transformation de mise à l'échelle se base quant à elle sur une technique similaire, mais fait aussi usage des widgets utilisés pas DesktopCAD. Tel qu'on peut l'observer dans la Figure 3.3 a et b, des axes s'affichent lorsque l'utilisateur lance la transformation. Il peut ensuite déplacer le dispositif dans l'espace et, en fonction de la direction et de la magnitude du déplacement, le modèle sera redimensionné selon un ou plusieurs axes. Cette technique est pratique pour redimensionner rapidement un objet en fonction de son contexte, mais manque malheureusement de la précision qu'on souhaite souvent avec la mise à l'échelle et qu'on retrouve dans DesktopCAD. Il s'agirait donc davantage d'un outil pour obtenir un modèle brouillon.

Jusqu'à présent, les techniques d'interaction n'ont utilisé que 3 degrés de liberté chacun : 3 degrés de position pour la mise à l'échelle et 3 degrés d'orientation pour la rotation. Mais comme le téléphone est suivi en 6 degrés de liberté, il serait dommage de ne pas en faire usage. Une quatrième transformation a donc été introduite, nommée « 6 DoF » car elle faisait usage des 6 degrés de liberté. Il s'agit en fait d'une transformation de translation et de rotation : l'usager peut, tout comme pour la transformation de rotation, angler le téléphone pour angler le modèle, mais il peut aussi déplacer le dispositif pour déplacer de la même façon le modèle virtuel. Cette opération est très appréciée par les utilisateurs plus expérimentés, qui ont tendance à primer sur la rapidité des opérations en séquence. Ceux-ci peuvent réaliser en un seul mouvement naturel une translation et une rotation en 3D alors qu'autrement il faudrait deux mouvement ou, pour DesktopCAD, jusqu'à 6 déplacements avec les widgets.

À noter aussi que toutes les opérations précédentes incluent des options permettant de verrouiller les axes ou de s'attacher aux surfaces voisines. La première option a comme effet d'appliquer la transformation selon un seul axe (celui pour lequel il y a le plus de différence) tandis que la seconde crée un effet *snap* qui aide à joindre deux objets ensemble. Les transformations deviennent alors plus simples et rapides.

### 3.2.4 Suivi des mains

Bien que le suivi des mains et de la gestuelle avec un système de vision n'est pas adéquat pour bien des interactions et des transformations, il serait faux de dire qu'il n'a pas sa place dans ARCAD. En effet, certaines fonctionnalités n'incluent pas de problèmes d'occlusion et se font quasi uniquement dans l'espace observé par l'utilisateur et la caméra. Deux de ces fonctionnalités ont donc été implémentées pour ARCAD.

Premièrement, même si la sélection peut se faire en approchant le téléphone intelligent d'un modèle, il a été remarqué que les utilisateurs préféraient garder le dispositif près d'eux et interagir avec leur main dominante. Une fonctionnalité pour sélectionner un objet virtuel en le pointant du doigt a donc été ajoutée. Un cercle de chargement s'affiche lorsque l'usager met son doigt près d'un item (Figure 3.4) et celui-ci devient sélectionné après un court instant. Cette technique est naturelle et permet de passer directement à la seconde technique.

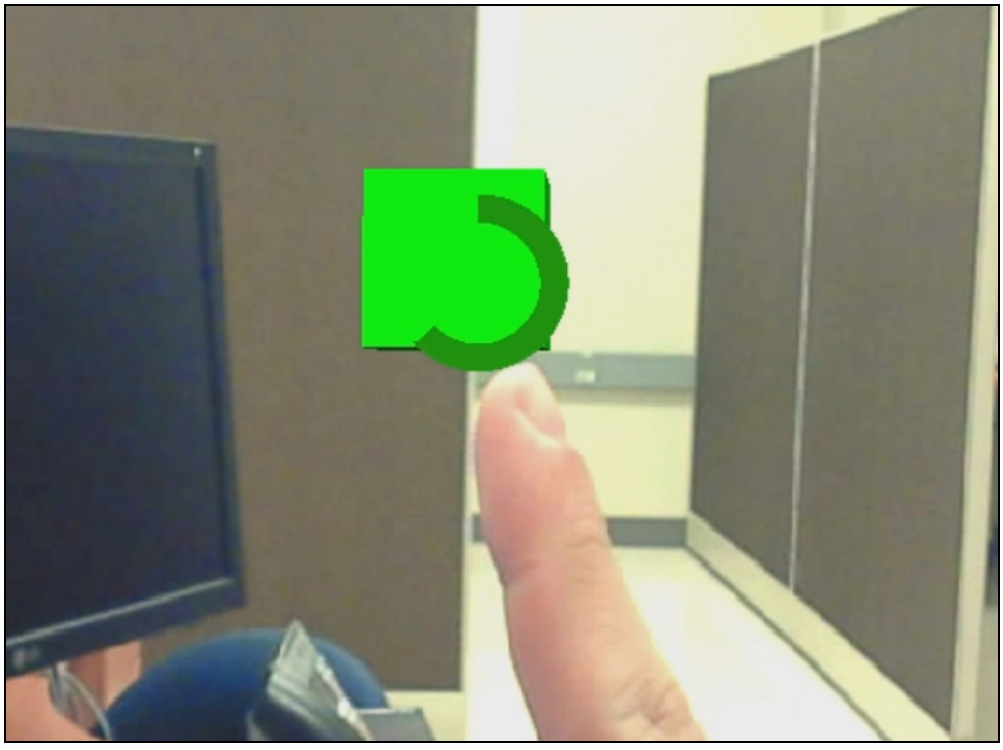

Figure 3.4 Sélection d'un modèle avec le doigt dans ARCAD

La seconde technique porte sur la transformation de translation. Comme le suivi en 3D d'un doigt est relativement précis et qu'une translation se fera presque toujours dans le champ de vision de l'utilisateur, il est possible d'utiliser la vision pour l'accomplir. Une fois la transformation lancée, le modèle sélectionné suivra le doigt de la main dominante de l'utilisateur jusqu'à ce que le casque perde de vue le doigt ou que la transformation soit arrêtée avec le téléphone. L'utilisateur peut donc sélectionner un objet virtuel avec sa main dominante, lancer une translation avec sa main dominante au repos sur ses genoux, effectuer la translation naturellement avec la main dominante et finalement terminer le tout avec la main non dominante, toujours au repos. Le résultat est une opération fluide et intuitive qui fait usage de la précision de la main humaine tout en exerçant le moins de possible d'effort.

### 3.2.5 Transformations directes & indirectes

La Figure 3.3 montre quelques techniques d'interaction en lien avec des transformations 3D, mais on pourrait décrire la rotation effectuée et (c) et en (d) comme étant une transformation directe. Par « transformation directe », on entend une opération où il semble y avoir superposition entre le contrôleur (dans ce cas le téléphone intelligent) et le contrôlé (le modèle virtuel). Si l'objet parait être à l'intérieur du téléphone et qu'on déplace ce dernier, alors l'objet suivra le téléphone et sera toujours à l'intérieur. Il y a contact « direct » entre le téléphone intelligent et le modèle. Cela peut sembler une interaction évidente, mais elle n'est pas possible avec un écran traditionnel et est propre aux interfaces immersives. L'avantage est que la transformation est d'autant plus naturelle et prévisible qu'il n'y a aucune disparité entre l'objet physique et celui virtuel. ARCAD permet aussi de sélectionner un mode de transformation et d'accomplir une transformation de ce type sur plusieurs objets, l'un à la suite des autres en déplaçant le téléphone sur chacun, tour à tour. Cela est avantageux lorsqu'on désire accomplir plusieurs opérations similaires sur plusieurs objets.

Toutefois, le fait de tenir le dispositif en l'air devant soi peut être quelque peu fatiguant à long terme et un usager pourrait vouloir réaliser plusieurs transformations de types différents

sur un seul objet. C'est pourquoi ARCAD supporte aussi un mode de transformations indirectes. Celles-ci sont réalisées en sélectionnant d'abord un modèle (avec le doigt par exemple) et en choisissant ensuite un type de transformation (rotation, mise à l'échelle, 6 DoF). Le téléphone intelligent pourrait techniquement être tenu au même endroit que le modèle et alors le résultat serait le même que pour une transformation directe, mais on peut généralement s'attendre à ce que le téléphone soit tenu à une position plus confortable, par exemple au-dessus d'un genou. L'utilisateur peut donc alors déplacer le dispositif ou changer son orientation (en fonction de la transformation choisie) pour provoquer un changement semblable au niveau de l'objet. On dit alors qu'il s'agit d'une transformation indirecte car il y a une distance entre le contrôleur et le contrôlé. La transformation est légèrement moins évidente, mais reste naturelle et prévisible car elle fait usage d'un dispositif physique. L'avantage réel et le fait d'éviter la fatigue musculaire que provoquerait le fait de devoir tenir le téléphone devant soi sur de longues périodes. De plus, cette technique est plus avantageuse lorsque l'utilisateur doit réaliser plusieurs opérations différentes sur un seul objet. Il n'a pas besoin de le sélectionner à chaque fois, changeant simplement le mode de transformation, pour une séquence plus rapide de modifications.

### 3.2.6 Draw-and-Drop

« Draw-and-Drop » est le nom donné à la première des deux fonctionnalités novatrices introduites par ARCAD, faisant pleinement usage du téléphone intelligent et de son suivi en 3D. Inspiré par *SketchUp*, un logiciel de CAO basé sur des opérations d'extrusion, Draw-and-Drop permet à un utilisateur de définir la base d'un prisme, la hauteur de celui-ci et sa position et orientation dans l'espace, le tout dans un processus simple, rapide et précis. La Figure 3.5 présente les différentes étapes de ce processus.

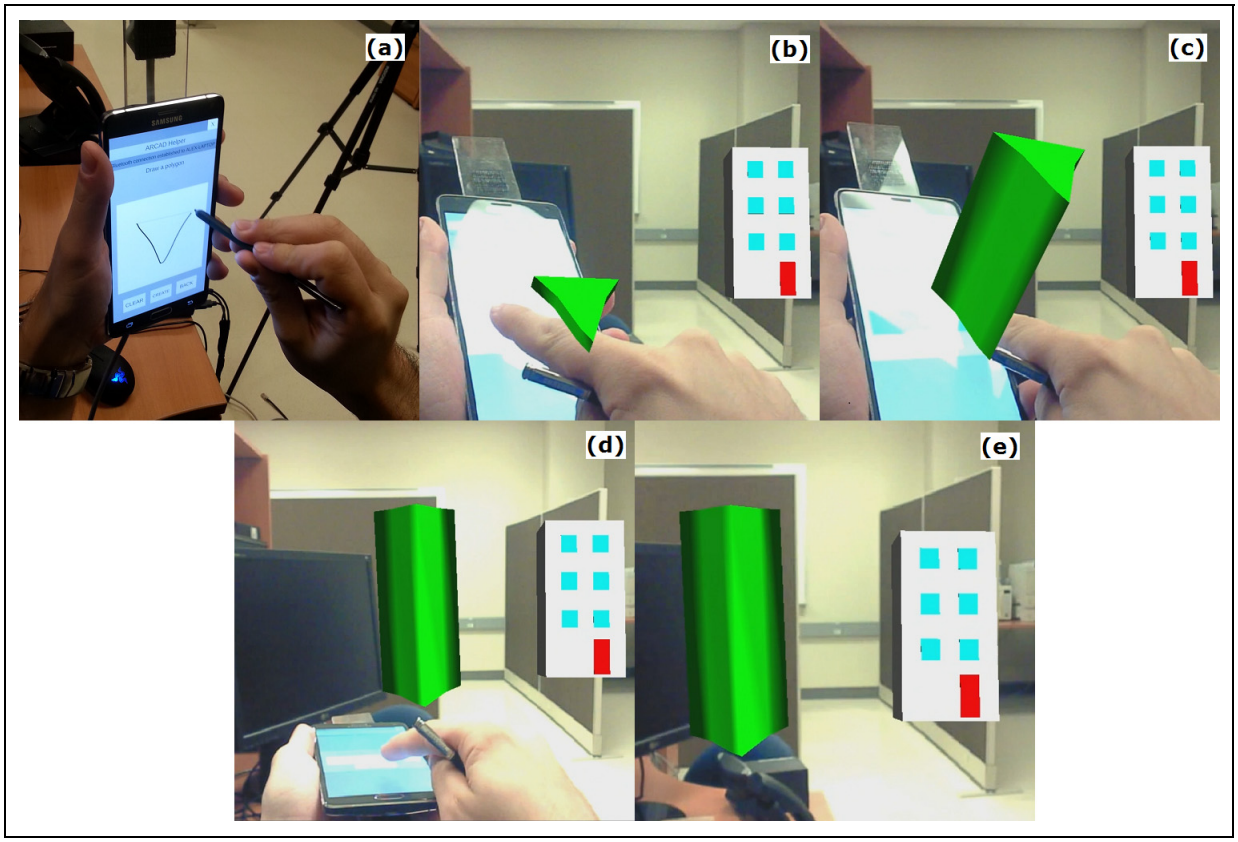


Figure 3.5 Étapes de la technique Draw-and-Drop

La technique d'interaction Draw-and-Drop se divise donc en les opérations suivantes, chacune se succédant :

(a) Avec le stylet, l'utilisateur dessine un polygone sur l'écran tactile du téléphone intelligent. L'application permet de compléter automatiquement la forme lorsque le stylet est relevé et le polygone est alors simplifié pour réduire le nombre d'arêtes sans pour autant trop diverger de la forme dessinée;

(b) Si l'utilisateur est satisfait du polygone dessiné, il appuie sur un bouton et un prisme virtuel, ayant comme base le polygone, apparait devant l'écran. Ce prisme est initialement petit en termes de hauteur et flotte à 2 ou 3 centimètres au-dessus de l'écran, suivant celui-ci lorsqu'on bouge le téléphone;

(c) L'utilisateur glisse son doigt ou le stylet sur la surface tactile du téléphone pour effectuer une extrusion du polygone, c'est-à-dire d'augmenter ou de diminuer la

hauteur du prisme selon ses besoins. L'extrusion peut être faite en plusieurs glissements de doigts;

(d) Une fois la bonne hauteur atteinte, l'utilisateur utilise le téléphone pour déplacer le prisme dans la scène afin de le positionner à l'endroit voulu. Il s'agit effectivement de la technique *6 DoF* vue plus tôt. La hauteur demeure ajustable à ce moment, si l'usager souhaite l'ajuster après le déplacement du prisme;

(e) Finalement, l'utilisateur appuie sur un bouton et le prisme est détaché de l'écran, demeurant à l'endroit où il était à cet instant. Le prisme devient donc un modèle comme un autre et peut être manipulé et édité comme bon nous semble.

Draw-and-Drop fait donc usage de plusieurs facteurs offerts par ARCAD : le téléphone intelligent tenu en main, l'écran tactile, le stylet, le suivi avec 6 degrés de liberté du téléphone et les lunettes à RA. Cette technique n'a pas vraiment d'équivalent dans DesktopCAD car il est généralement compliqué de définir un prisme à base complexe. Draw-and-Drop introduit donc une capacité additionnelle au système DualCAD. Les utilisateurs ayant testé cette fonction ont remarqué qu'elle était rapide et moins contraignante que de devoir utiliser des primitives. De plus, l'usage de la surface du téléphone permet d'effectuer les étapes de dessin et d'extrusion dans une position confortable pour seulement ensuite venir positionner le prisme dans la scène augmentée.

### 3.2.7 Touch-and-Draw

La seconde technique introduite par ARCAD, nommée « Touch-and-Draw » est semblable à la première sur plusieurs points. D'abord elle fait usage des mêmes facteurs (stylet, écran tactile, suivi en 6 degrés de liberté, etc.), mais elle permet aussi d'effectuer une partie du travail dans une position confortable avant de positionner le modèle édité. La différence majeure est que, si Draw-and-Drop permettait de créer des prismes complexes, Touch-and-Draw a pour but d'éditer et de manipuler des modèles existants.

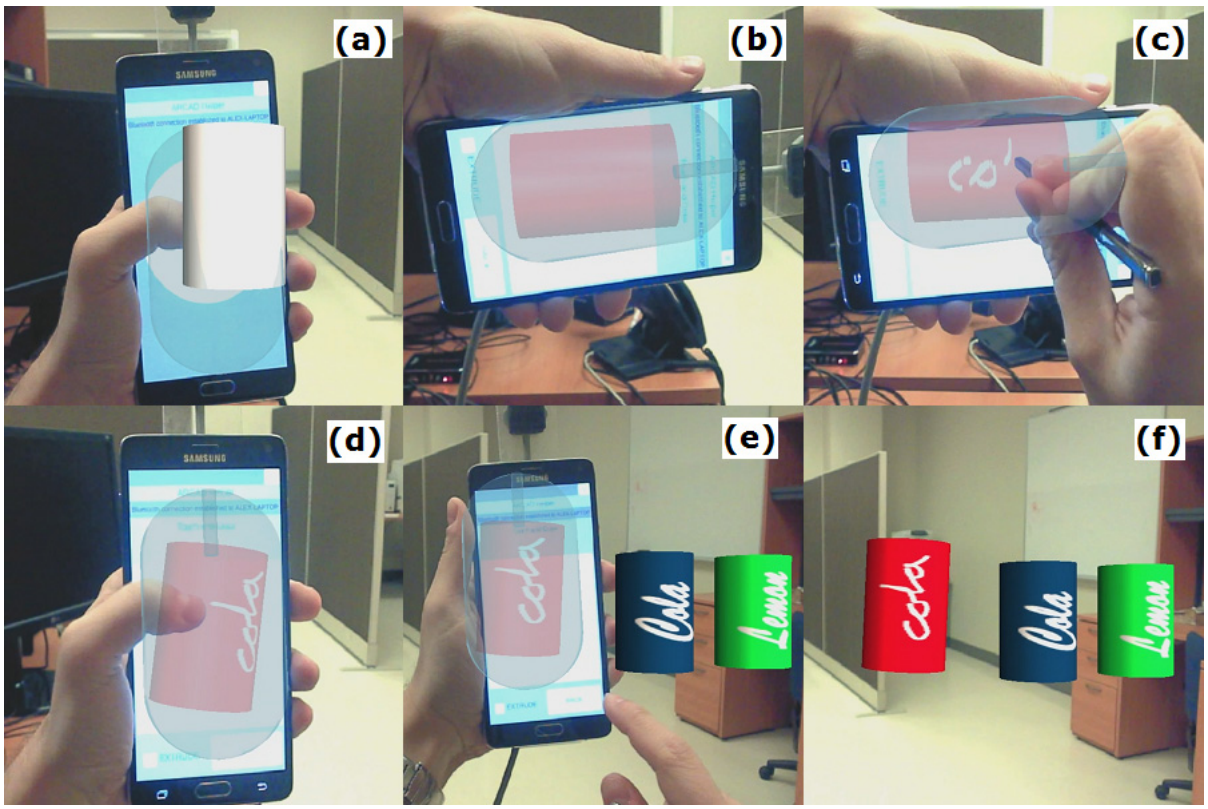


Figure 3.6 Étapes de la technique Touch-and-Draw

La Figure 3.6 démontre les étapes de Touch-and-Draw :

(a) D'abord l'utilisateur vient saisir un modèle avec le téléphone intelligent, comme pour lancer une transformation directe;

(b) Une fois le modèle sélectionné, celui-ci devient transparent. Cet aspect est critique car le modèle ne doit pas masquer la surface du téléphone. Le modèle devient aussi attaché au téléphone et suivra ses déplacements, tout comme avec la transformation *6 DoF*;

(c) L'utilisateur peut utiliser le stylet pour dessiner sur la surface du modèle sélectionné. ARCAD fait le lien un-pour-un entre les coordonnées de la surface tactile et la position dans l'espace, de manière à faire croire à l'usager qu'il dessine ou écrit réellement sur l'objet;

(d) L'utilisateur peut utiliser ses doigts pour effectuer des rotations au modèle, en glissant par exemple son pouce de gauche à droite. Cela lui permet de dessiner sur la

surface opposée ou bien de changer l'orientation du modèle avant la prochaine étape;

(e) Une fois le modèle édité à souhait, l'utilisateur peut faire usage du suivi avec 6 degrés de liberté pour replacer l'objet dans la scène;

(f) Une pression sur un bouton et le modèle est relâché, retrouvant son opacité.

S'il est possible avec DesktopCAD de dessiner sur des surfaces planes, il est plus complexe de dessiner avec une souris sur des surfaces complexes. Et même avec une interface à RA munie d'un stylet suivi en 3D, dessiner directement sur un modèle n'est pas à la portée de tous puisque l'absence de contact physique en déstabilise plus d'un. C'est pourquoi les systèmes d'édition de texture se limitent souvent à utiliser une surface tactile pour éditer à distance la surface. Mais comme ARCAD dispose d'une surface tactile suivie avec 6 degrés de liberté, il est possible d'utiliser cette surface pour simuler un contact physique (et surtout direct) avec la surface à éditer. L'édition de texture se fait donc plus intuitivement et les personnes ayant testé cette fonctionnalité ont remarqué qu'ils appréciaient la correspondance directe entre leurs tracés et ceux qui apparaissaient sur la texture. De plus, le fait de pouvoir effectuer une rotation sur le modèle sélectionné offre non seulement une façon de plus d'effectuer une transformation de rotation, mais permet de dessiner sur plusieurs surfaces d'un même modèle dans une seule opération, c'est-à-dire sans avoir à désélectionner et resélectionner le modèle pour chaque surface. Cela améliore vraisemblablement les performances lors de tâches complexes. Finalement, tel que mentionné précédemment, cette fonctionnalité permet à l'utilisateur de réaliser une majeure partie de la tâche dans une position confortable, réduisant donc la fatigue musculaire.

## 3.3 Résultats de la démonstration aux utilisateurs experts

Ce chapitre présentera les commentaires et réponses des utilisateurs experts lors de la démonstration du prototype DualCAD. La section n'inclura pas d'analyse détaillée des opinions émises par les testeurs, se contentant d'établir les faits enregistrés et de les mettre en

contexte au besoin. Pour une analyse plus poussée, référez-vous à la section 4.1 Recommandations.

Tout d'abord, avant même d'avoir présenté le prototype, plusieurs individus mentionnèrent leur engouement envers les technologies utilisées par le système. En effet, ayant vu le visiocasque, les testeurs affirmèrent avoir hâte d'essayer le produit. Un testeur en particulier expliqua qu'il avait toujours rêvé de pouvoir accomplir son travail de modélisation avec des technologies de RA et de RV. Plus précisément, les usagers espéraient pouvoir accomplir des tâches d'assemblage, d'architecture, d'animation et de visualisation de modèles existants. Cette dernière tâche fut soulignée en particulier car les individus se plaignaient de ne pas avoir d'outils adaptés à la démonstration de leur travail aux clients. Fait notable : il y eut désaccord de la part des usagers sur la décision d'utiliser la RA au lieu de la RV. Si une personne aurait préféré être dans son monde purement virtuel pour se concentrer sur son travail, une autre affirmait préférer la RA car cela lui permettait de s'orienter et entraînait moins de maux de tête.

Ensuite, lorsque les utilisateurs purent tester par eux-mêmes le prototype, plusieurs commentaires furent émis, en particulier sur ARCAD et ses fonctionnalités. Un testeur, après avoir essayé les fonctionnalités de translation et de rotation, suggéra de les combiner. Il fut donc agréablement surpris de constater que cette fonctionnalité existait déjà, sous le nom de « 6 DoF ». D'autres exclamèrent leur enthousiasme face à cette fonctionnalité, expliquant qu'ils aimaient avoir un plein contrôle. Les deux fonctionnalités nouvelles, Draw-and-Drop et Touch-and-Draw, furent aussi bien reçues, obtenant chacune des commentaires sur l'aspect nouveau et sur le grand nombre de possibilités offertes. Chaque fonctionnalité entraîna toutefois quelques problèmes pour au moins un utilisateur : la séquence d'actions requises s'avéra quelque peu complexe pour des utilisateurs débutants et certains durent s'y reprendre à plusieurs reprises avant d'obtenir le résultat espéré.

Un autre problème constaté fut la réticence qu'avaient certains testeurs à passer leur attention de la scène 3D devant eux au téléphone intelligent. En effet, en partie dû au faible champ de

vision du visiocasque, il n'était pas possible pour un utilisateur de voir simultanément un objet d'intérêt et le menu du téléphone intelligent. Il était donc nécessaire de bouger la tête, perdant de vue la scène ou le téléphone, ou bien de tenir le dispositif devant soi, près de la scène, et de changer son attention en bougeant les yeux. Ce problème fut le plus notable pour l'utilisateur qui avait mentionné souffrir de troubles d'orientation avec un visiocasque : comme le fait de bouger la tête rendait plus difficile l'orientation, le testeur se sentait obligé de rester la tête figée, tenant le téléphone à bout de bras devant lui. Plusieurs personnes suggérèrent d'afficher des informations ou du *feedback* à l'aide du visiocasque afin de limiter les situations où il est nécessaire de regarder le téléphone. D'autres mentionnèrent qu'il n'y aurait pas ce problème si le contrôle avec le dispositif pouvait être fait sans le regarder, comme avec une manette.

Il fut aussi constaté que les testeurs avaient tendance à essayer des actions non supportées par le système ou bien d'effectuer des opérations de la mauvaise façon. Dans le premier cas, nous avons remarqué que certains utilisateurs essayaient de saisir les objets virtuels avec leur main pour les manipuler directement. Une personne crut même y parvenir alors qu'elle effectuait en fait l'opération avec le téléphone dans l'autre main. Il est à noter que les rotations à une seule main n'étaient pas permises par le système car celles-ci sont très difficiles à interprétées par les algorithmes de vision. Cela n'a pas empêché les testeurs d'essayer les actions qui leur semblaient les plus naturelles d'abord. Sur un point de vue similaire, si certains gestes considérés comme naturels sont permis par DualCAD (par exemple sélectionner un objet en le pointant avec son doigt en I devant nous), ils ne furent pas tous bien effectués du premier coup. Par exemple, deux testeurs essayèrent de sélectionner un objet en le pointant à bout de bras (dans ce cas la main et le bras cachent le doigt, masquant donc l'action pour le système). Il y a donc aussi des problèmes liés aux utilisateurs débutants effectuant mal certains gestes, même si ceux-ci sont permis par le système.

Suite à l'essai du prototype, les testeurs ont pu ajouter certains commentaires. Un individu utilisa comme métaphore l'argile, expliquant qu'il voudrait pouvoir manipuler et façonner

des objets virtuels avec seulement ses mains, comme s'il s'agissait de pâte réelle. Selon lui un téléphone intelligent, ou tout autre dispositif, ne fait qu'ajouter un mur entre l'utilisateur et son but, même si le testeur reconnaissait que la technologie actuelle ne permettait probablement pas ce qu'il souhaitait. En d'autres mots, cette personne aurait voulu pouvoir se passer du téléphone intelligent, bien qu'elle comprenait sa nécessité actuelle. Cette opinion ne fut toutefois pas vraiment partagée par les autres testeurs : ceux-ci n'étaient pas d'accord sur le fait que tout devrait pouvoir être possible avec la gestuelle et aimaient généralement avoir un dispositif physique en main pour simuler le contact avec les objets virtuels et obtenir plus de précision.

Finalement, les utilisateurs experts apportèrent d'autres points importants en lien avec leurs besoins face à un système similaire. Tout d'abord, un logiciel de CAO faisant usage de la RA ou de la RV devrait s'ajouter à leur environnement de travail et non le remplacer. Les experts ont déjà beaucoup d'expérience avec leurs logiciels de CAO de bureau et ne veulent pas les voir complètement remplacés. Un nouveau système devrait donc pouvoir se greffer aux anciens tout en ajoutant une valeur supplémentaire au lieu de simplement refaire les fonctionnalités actuelles. Ensuite, les testeurs expliquèrent que la force des logiciels existants était qu'ils étaient adaptés aux utilisateurs experts. Ces personnes n'avaient pas d'objection à ce qu'un nouveau système demande une grande courbe d'apprentissage, tant qu'il apporte un avantage majeur en bout de ligne. Du même fait, aucun testeur n'était inquiet de devoir débourser de larges sommes pour un tel système. L'opinion générale était que la précision et les possibilités primaient par-dessus tout, quitte à nécessiter du matériel très dispendieux. Il serait intéressant de poursuivre l'analyse auprès d'autres utilisateurs (non-experts) afin de déterminer s'ils partagent ces vues.

# CHAPITRE 4

# DISCUSSION DES RÉSULTATS

## 4.1 Recommandations

Suite aux observations faites lors de la démonstration auprès d'utilisateurs experts et en fonction des commentaires recueillis, il est possible de faire ressortir les points intéressants et prometteurs du prototype DualCAD, mais aussi ses problèmes, limitations et points à améliorer. L'analyse de toutes ces données permet de produire une liste cohérente (mais non exhaustive) de recommandations qui, nous l'espérons, pourront aider le développement d'applications de CAO futures.

### 4.1.1 Le téléphone intelligent

Le téléphone intelligent fut une partie cruciale du prototype ARCAD. Offrant d'abord les avantages d'un accessoire réel tenu en main (suivi en 6 degrés de liberté, retour tactile), l'écran tactile permet aussi d'afficher clairement une grande quantité d'informations en plus d'accepter l'input des doigts et d'un stylet. Ces avantages ont été discutés préalablement et sont généralement connus. La première recommandation porte plutôt sur les limites qu'on devrait s'imposer quant à l'usage d'un tel dispositif dans une application de CAO. En effet, bien que l'écran du téléphone soit bien plus lisible que celui du visiocasque, il a été remarqué que les utilisateurs évitaient tant que possible de regarder ce premier, surtout lors de manipulation dans la scène virtuelle. Il faudrait donc tout d'abord afficher des informations de base, peu détaillées et non sensibles à la faible résolution du visiocasque, dans le champ de vision de l'usager. Un simple aperçu ou retour visuel, affiché en quasi permanence sur le visiocasque, rendrait moins fréquente la nécessité de quitter du regard la scène pour passer à l'écran du téléphone. Bien sûr, cet écran n'en devient pas pour autant inutile : le détail des informations sera toujours sur le téléphone, laissant toujours toutes les données accessibles à l'utilisateur. L'idée principale de cette recommandation est **d'avoir à regarder le téléphone le moins possible**. Cela s'applique pour l'affichage, mais aussi pour l'input. Un menu radial

a été implémenté dans ARCAD pour permettre aux utilisateurs experts d'entrer des commandes sans avoir à regarder le dispositif, mais les tests avec les usagers débutants avec DualCAD ont démontré que le menu ne les aidait pas, à leur niveau, à saisir des commandes en dehors de leur champ de vision. Encore une fois, il serait possible d'afficher un retour visuel sur le visiocasque pour qu'un usager voie en tout temps le menu radial et son état au fur et à mesure qu'il le parcourt. Cela devrait encore une fois limiter les déplacements de tête, laissant l'utilisateur saisir des commandes sans regarder le dispositif, même s'il n'est pas familier avec le menu. On peut aussi imaginer que ce menu, affiché avec le visiocasque, pourrait être retiré pour les utilisateurs experts, laissant plus d'espace pour la scène virtuelle.

Ensuite, comme il a été remarqué que les testeurs avaient tendance à d'abord essayer de manipuler les objets par gestuelle (par exemple essayer de faire une rotation en prenant un objet virtuel en main), il faudrait qu'un système futur **priorise les interactions par gestuelle plutôt que par téléphone, lorsque la technologie le permet**. En effet, même si certaines opérations par gestuelle (par exemple la rotation à deux mains) sont peu précises, il ne faut pas s'empresser de les remplacer complètement par un équivalent avec le téléphone. Ces techniques d'interaction sont généralement les plus naturelles pour un usager débutant et celui-ci s'attend à obtenir un résultat lorsqu'il les essaie. Il faut donc permettre ces opérations, tout en offrant la possibilité d'utiliser le téléphone pour une plus grande efficacité/précision. Seulement lorsqu'une opération est excessivement inefficace devrait-on faire seulement usage du téléphone. Cette politique assurera que le système reste accessible aux débutants, tout en offrant la précision recherchée par les experts.

### 4.1.2 Reconnaissance de la gestuelle

Il a été mentionné, dans les résultats de la démonstration aux utilisateurs experts, que les testeurs avaient tendance à essayer des opérations (souvent par gestuelle) non permises par le système ou bien d'en effectuer des permises, mais d'une manière erronée. Même avec la recommandation précédente d'inclure autant que possible toutes les opérations par gestuelle naturelle, il est certain que certaines vont au-delà des capacités de la technologie actuelle et

ne peuvent donc pas être supportées par le système. Il n'en demeure pas moins que le logiciel devrait **reconnaître l'intention derrière une opération non supportée afin d'indiquer la bonne technique**. Par exemple, si un utilisateur devait essayer d'effectuer une rotation à une seule main, l'application ne serait pas en mesure de bien traiter cette opération, dû aux limitations des algorithmes de vision. Voyant l'échec de sa tentative, l'utilisateur pourrait éprouver de la frustration, pensant qu'il effectue mal l'opération. Mais si le système est en mesure de détecter la tentative et, sans pour autant pouvoir la traiter, simplement pouvoir l'identifier, il peut guider l'utilisateur vers la technique appropriée. Dans cet exemple-ci, le logiciel réalise, après une ou deux tentatives, que l'utilisateur essaie de faire une rotation et informe donc ce dernier, à l'aide du visiocasque, que les rotations devraient être faites avec deux mains ou à l'aide du téléphone intelligent. Un mini tutoriel pourrait même s'afficher sur l'écran du téléphone.

L'idée est semblable lorsqu'il s'agit d'une opération supportée, mais mal effectuée par l'utilisateur. Le système devrait **reconnaître l'intention derrière une opération supportée mais mal effectuée afin d'indiquer la bonne façon de faire**. Si l'utilisateur pointe un objet pour le sélectionner, mais que sa main cache son doigt, rendant ambiguë la sélection, l'application devrait encore une fois identifier l'action désirée et, à l'aide du visiocasque, informer l'individu de la bonne façon de faire. Ici aussi, des informations plus détaillées pourraient être mises sur l'écran du téléphone. Cette doctrine permettrait aux utilisateurs de passer de débutant à expert sans l'aide d'un tutoriel explicite ou d'un guide externe : le système lui-même apprend aux usagers comment réaliser les opérations.

### 4.1.3 La CAO et la réalité mixte

L'usage de la RA pour le prototype DualCAD a permis de constater certains problèmes liés à ces technologies. Plus particulièrement, et en lien avec les commentaires des testeurs, on retrouve le débat pour l'usage de la RV ou de la RA. Les testeurs ne s'entendaient pas sur le paradigme à privilégier, mentionnant des avantages pour chacun : la RV permet une meilleure immersion, retirant les distractions entourant l'utilisateur, alors que la RA aide les

utilisateurs à s'orienter, cause moins de maux de tête et permet l'usage de l'écran du téléphone. Actuellement les technologies de RV sont plus avancées que celles de RA, offrant des résolutions en champ de vision bien plus élevés. On peut donc affirmer que, dans l'état actuel, un logiciel de CAO à RV serait plus performant. Toutefois, en assumant que la RA rattrapera son concurrent technologique, on ne peut pas dire qu'une technologie est intrinsèquement supérieure à l'autre. La recommandation porte donc sur le fait de **faire usage du spectrum entre la RA et la RV**. Si le choix entre les deux dépend de l'utilisateur et des circonstances, alors le système devrait permettre non seulement les deux, mais aussi un compromis entre ceux-ci. Les visiocasques à RA par vidéo (c'est-à-dire les casques munis de caméras affichant le flux vidéo sur les écrans du casque) permettent d'avoir autant la RA, en ajoutant la scène virtuelle à ce que les caméras voient, la RV, en coupant le flux vidéo de la caméra, et un mélange des deux, en atténuant le flux vidéo. Le système lui-même pourrait sélectionner le ratio idéal entre la RA et la RV selon la situation, ou bien l'utilisateur pourrait le modifier selon ses désirs.

Finalement, il est aussi question de la portée exacte d'un logiciel de CAO en RA/RV. Les testeurs s'entendaient pour dire que ARCAD était approprié pour les tâches d'assemblage et de visualisation, mais que le logiciel laissait à désirer pour ce qui était des tâches de modélisation. Cela est en accord avec nos hypothèses initiales puisque l'étape de modélisation exige beaucoup de minutie et de précision et donc se réalise mieux avec un outil comme la souris. Toutefois, une étape « brouillon » de modélisation pourrait être faite avec un système en RA avant d'être complétée avec un système de bureau traditionnel. Somme toute, **la partie RA/RV d'un système multimodal devrait se concentrer sur les tâches de travail de modélisation brouillon, d'assemblage et de visualisation, laissant le travail de précision au mode bureau**. Ce mode de bureau ne devrait d'ailleurs pas être réinventé. Si, pour ce projet, DesktopCAD a été conçu pour le mettre en contexte avec ARCAD, un système commercial devrait absolument être bâti sur un logiciel de CAO existant. En effet, les utilisateurs ont déjà leurs habitudes et leur expérience avec un logiciel ne devrait pas être mise de côté. C'est donc pour cela **qu'un mode à RA/RV devrait bâti par-dessus un logiciel établi de CAO, et non à part ou en parallèle**.

## 4.2 Leçons apprises

### 4.2.1 Le matériel et le champ de vision

Du moment où ces technologies ont été imaginées, la réalité virtuelle et la réalité augmentée ont fait rêvé les gens. Le fait de pouvoir s'immerger dans un monde virtuel ou bien de fusionner des éléments de celui-ci avec la réalité pour ensuite manipuler les objets virtuels comme s'ils étaient physiques parait trop beau pour être vrai. Et comme bien souvent, si c'est trop beau, c'est qu'il y a une part d'exagération. Les développeurs des années 90 s'en sont rendus compte et les technologies ont été quelque peu mises de côté, du moins par l'industrie en général en attendant que la science rende ces idées réalisables. La réalité virtuelle (RV) trouva son essor dans les 5 dernières années, en lien avec la miniaturisation de composantes clés (telles que les gyroscopes, accéléromètres et magnétomètres) et la hausse en performance des ordinateurs, désormais capables de générer en temps réel des images à résolution suffisante pour être placée devant l'œil humain. Sauf que si la RV et, plus précisément, les visiocasques à affichage vidéo ont atteint leur singularité en termes de commercialisation, la RA et les visiocasque à affichage « à travers » (see-through) ont encore des défis majeurs. Présentement, l'usage de lentilles semi-transparentes pour la superposition d'images virtuelles par-dessus la réalité a pour effet de perdre une part de l'opacité des objets virtuels et de réduire la luminosité de la scène réelle. Les projecteurs utilisés sont limités en résolution et, bien que les performances des ordinateurs permettent des images bien plus grandes, le matériel physique limite celles-ci. Finalement, et représentant probablement le problème le plus critique, les lentilles limitent le champ de vision utilisable par les visiocasques. En effet, des lentilles trop grandes ont présentement pour effet de distordre la réalité et donc de créer une disparité entre le monde vu à travers la lentille et celui vu par la vision périphérique. Certains appareils soutiennent solutionner en partie ce problème en bloquant la vision périphérique, mais on obtient alors un dispositif ayant les mêmes inconvénients qu'un visiocasque vidéo (pas de vision périphérique par exemple) et sans les avantages de ceux-ci (grande résolution par exemple).

Somme toute, ce n'est actuellement pas un problème de logiciel ou d'interaction humain-machine qui touche la RA, mais plutôt un problème matériel (hardware). Des prototypes de systèmes et des tests de techniques d'interaction peuvent être faits, mais ceux-ci ne pourront pas atteindre leur plein potentiel tant que le matériel, et principalement le champ de vision, n'aura pas progressé. Ce projet de mémoire pourrait donc être repris quasiment tel quel, dans 10 à 20 ans, avec un visiocasque de cet époque pour réellement le tester et constater hors de tout doute l'apport des concepts présentés précédemment. Mais d'ici là, nous ne pouvons que projeter quant à l'intégration de ces techniques dans un système commercial.

### 4.2.2 Techniques d'interaction non efficaces

Dans le but de concevoir et développer des techniques d'interaction faisant usage des composantes de ARCAD, plusieurs idées furent proposées et certaines de ces idées furent implémentées dans le système. La plupart de ces fonctionnalités présentèrent un attrait quelconque au projet et furent présentés dans les sections précédentes. Toutefois, certains essais se finirent en échec, soit car la technologie utilisée n'était pas adaptée ou bien car la technique n'est tout simplement pas efficace pour réaliser le travail voulu. Certaines de ces techniques sont donc présentées ici, dans l'espoir que les leçons apprises de ces erreurs puissent être utiles à d'autres.

L'une des fonctionnalités imaginées portait sur la création de prisme en 3D. Inspirée par le court film World Builder (Branit, 2013) (Figure 4.1) où un humain immergé dans un monde virtuel débute l'édition du monde l'entourant en créant un cube devant lui avec ses deux mains. La technique parait simple et intuitive : l'utilisateur place d'abord ses deux main ensemble, ses doigts dans une gestuelle de pincement, et écarte ensuite ses mains comme s'il étirait un objet. Un cube apparait alors et s'élargit graduellement en fonction de l'écart entre les mains. On a l'impression que l'usager a pris initialement un minuscule cube et en a étiré les coins pour en faire un plus grand cube. Pour ARCAD, le suivi en 3D des deux mains paraissait risqué, dû au champ de vision limité des caméras des lunettes, mais il était possible de remplacer l'une des mains par le téléphone intelligent. L'opération est donc lancée en

appuyant de sa main dominante sur l'écran du téléphone, tenu dans la main non-dominante, et, écartant les mains, un cube était créé et agrandi. La fonctionnalité marchait tel que prévu, mais un résultat inattendu fut constaté : le fait de créer devant soi un cube ayant des dimensions de 50 centimètres à 1 mètre implique que le cube occupera une portion considérable (si ce n'est totale) du champ de vision de l'utilisateur. En effet, il a vite été constaté qu'il n'était pas pratique de manipuler des objets de cette taille si près de soi ou avec un contact direct. Cet effet peut même être observé dans le film : le cube créé devient rapidement trop gros pour que l'utilisateur puisse réellement constater sa taille. Une coupure de caméra et un changement d'angle cache en partie le problème. Il faut se rappeler que ce film est à caractère de divertissement et il est donc normal que les créateurs aient pris certaines libertés quant à la réalité pour mieux impressionner. Mais dans un projet réel comme ARCAD, cette création et manipulation de cubes n'est pas pratique et a donc été exclue du projet.

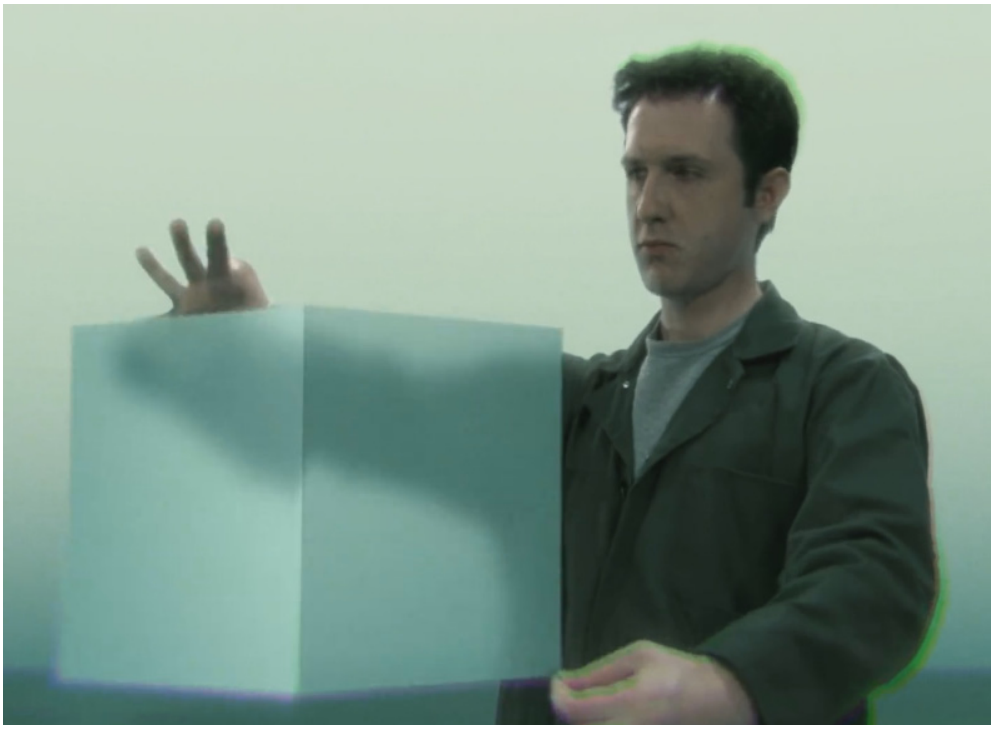

Figure 4.1 Création de prisme, tiré de World Builder (Branit, 2013)

D’autres fonctionnalités mineures furent rejetées d’emblée car, même si celles-ci étaient représentées dans la science-fiction, elles sont généralement difficiles à effectuer, dépendent de technologies pas encore au point ou bien ne sont tout simplement pas pratiques. On pense entre autres aux boutons de RA, qui sont difficiles à appuyer avec précision, ou carrément inatteignables lorsque placés près d’une surface, ou bien aux fonctionnalités permettant de dessiner avec un stylet en l’air. Cette méthode n’est pas naturelle car elle ne fait pas usage d’un contact physique et déstabilise trop souvent l’utilisateur. Somme toute, certaines techniques paraissent impressionnantes au cinéma, mais ne seront jamais réellement pertinentes aux interfaces usager.

# CONCLUSION

## Retour sur les objectifs

Tel que précisé au chapitre 1.6 Problématique et objectifs, les objectifs de ce projet de mémoire se voulaient surtout exploratoire. Le but était de (1) prototyper un système multimodal de CAO nommé DualCAD, de (2) développer des techniques d'interaction avec téléphone intelligent pour le mode à RA et de (3) tester le prototype auprès d'utilisateurs experts afin de faire des recommandations pour des systèmes futurs.

DualCAD a bien été développé, intégrant un mode de bureau (DesktopCAD) et un mode à RA (ARCAD) et permettant de sauter d'un mode à l'autre très rapidement et aisément. Toutes les opérations possibles dans DesktopCAD ont été reproduites dans ARCAD, faisant usage de la gestuelle et du téléphone intelligent. Une attention particulière fut donnée sur les fonctionnalités liées au téléphone, explorant les manipulations directes, indirectes, par « wand », avec stylet, etc. Deux techniques complètement nouvelles (Draw-and-Drop et Touch-and-Draw) furent aussi introduites et développée pour ARCAD, permettant respectivement de créer des volumes complexes à partir d'un dessin de polygone, et de saisir en main un modèle pour l'éditer avec le stylet. Finalement, le prototype fut montré à des utilisateurs experts en modélisation 3D et leurs commentaires et impressions furent recueillis et analysés. De ces tests et du développement du logiciel ressortirent des leçons apprises ainsi que des recommandations faites pour des développer futur.

Ainsi chacun des 3 objectifs majeurs, incluant leurs sous-objectifs, furent remplis par ce projet de mémoire. Bien sûr ce n'était pas tous les objectifs qui étaient quantifiables et donc il est difficile de prouver hors de tout doute que suffisamment d'effort a été mis pour chacun, mais nous sommes très satisfaits des résultats obtenus, tant pour l'impressionnante pièce d'ingénierie que représente DualCAD, que pour les recommandations faites qui, selon nous, pourraient considérablement aider la littérature.

**Contributions**

Par soucis de clarté, quitte à créer de la redondance avec les sections précédentes, voici une liste des contributions faites à l'état de l'art par ce projet de mémoire :

1. Le prototype de système multimodal DualCAD, intégrant les sous-systèmes DesktopCAD et ARCAD, permettant respectivement d'accomplir des tâches de CAO avec une interface clavier-souris-écran et d'accomplir ces mêmes tâches avec une interface basée sur un visiocasque à RA et un téléphone intelligent. Le prototype permet le passage rapide entre les modes afin de démontrer que certaines tâches se réalisent mieux dans un sous-système que dans l'autre;
2. La proposition, la conception et l'implémentation de techniques d'interaction pour ARCAD faisant usage d'un téléphone intelligent suivi avec 6 degrés de liberté. Certaines techniques connues furent repensées pour ce nouvel outil et d'autres, telles que Draw-and-Drop et Touch-and-Draw furent introduites;
3. Une taxonomie d'inputs et d'outputs ainsi qu'une taxonomie de techniques d'interaction permettant de comparer DualCAD aux systèmes produits par des travaux antérieurs;
4. Des recommandations faites à la suite de tests informels avec des utilisateurs experts en modélisation 3D.

**Limites du travail**

La limitation principale de ce projet de mémoire et l'absence de tests formels sur une large population et donc de résultats quantitatifs. Pour répondre directement à la question « un système multimodal fusionnant un logiciel CAO de bureau et un logiciel CAO de RA est-il plus performant que chaque sous-système pris individuellement ? », il faudrait ces résultats quantitatifs. En l'absence de ceux-ci, ce rapport ne peut que comparer qualitativement les composantes du prototype et les techniques d'interaction proposées.

DesktopCAD s'est limité aux interfaces les plus classiques de bureau, à savoir un clavier, une souris 2D et un écran 2D, mais il aurait pu être intéressant d'inclure des interfaces supplémentaires, telles qu'une surface tactile pour dessiner ou un écran 3D. Au même titre, ARCAD fit grandement usage des capacités des lunettes META et du téléphone intelligent, mais ne toucha pas aux possibilités apportées par d'autres outils de RA (gants électroniques, stylets suivis en 3D, etc.). L'idée était de ne pas encombrer l'utilisateur avec trop de dispositifs et, autant que possible, de reproduire les interactions de ces dispositifs avec le téléphone, mais une analyse plus poussée sur les différences entre les outils aurait pu être faite.

Finalement, les systèmes de CAO n'incluent qu'une sélection de contrôles possibles parmi les plus simples qu'on retrouve souvent dans les logiciels professionnels. Par exemple, DualCAD ne permet pas la création de surface complexe ou bien la construction d'un squelette reliant les modèles entre eux. Seules certaines techniques d'interaction, notamment Draw-and-Drop, furent implémentées en lien avec des opérations plus complexes.

**Directions futures**

Tel que mentionné plus tôt, des chercheurs désirant poursuivre le travail de mémoire pourrait se concentrer sur l'évaluation d'une ou plusieurs techniques d'interaction introduites par ce travail. Il serait aussi intéressant de reproduire le système avec du matériel plus avancé lorsque celui-ci sera disponible, par exemple les HoloLens de Microsoft. Dans ces deux cas, des tests utilisateurs formels aideraient beaucoup à pousser la contribution de ce mémoire. Une autre option serait de comparer DualCAD à d'autres systèmes faisant usage de dispositifs d'input et d'output différents. Par exemple est-ce qu'un écran 3D apporterait une valeur au système ? Ensuite, la suggestion d'utiliser un téléphone intelligent suivi en 3D est relativement nouvelle dans l'état de l'art et plusieurs techniques peuvent encore être conçues. C'est une toute nouvelle facette de la réalité augmentée qui s'ouvre et il serait selon nous très intéressant de s'y pencher.

## LISTE DE RÉFÉRENCES BIBLIOGRAPHIQUES


ARToolworks. 2001. « ARToolKit ». < http://artoolkit.org/ >. Consulté le 2015-11-12.

Balakrishnan, Ravin, Thomas Baudel, Gordon Kurtenbach et George Fitzmaurice. 1997. « The Rockin'Mouse: integral 3D manipulation on a plane ». In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*. p. 311-318. ACM.

Beiser, Leo. 1981. *Anaglyph stereoscopy*.

Bérard, François, Jessica Ip, Mitchel Benovoy, Dalia El-Shimy, Jeffrey R Blum et Jeremy R Cooperstock. 2009. « Did "Minority Report" get it wrong? Superiority of the mouse over 3D input devices in a 3D placement task ». In *Human-Computer Interaction–INTERACT 2009*. p. 400-414. Springer.

Blanch, Renaud, Yves Guiard et Michel Beaudouin-Lafon. 2004. « Semantic pointing: improving target acquisition with control-display ratio adaptation ». In *Proceedings of the SIGCHI conference on Human factors in computing systems*. p. 519-526. ACM.

Bogdan, Natalia, Tovi Grossman et George Fitzmaurice. 2014. « HybridSpace: Integrating 3D freehand input and stereo viewing into traditional desktop applications ». In *3D User Interfaces (3DUI), 2014 IEEE Symposium on*. p. 51-58. IEEE.

Bosomworth, Danyl. 2015. « Mobile Marketing Statistics 2015 ». < http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/ >. Consulté le 2015-10-29.

Bowman, Doug A, Jean Wineman, Larry F Hodges et Don Allison. 1998. « Designing animal habitats within an immersive VE ». *IEEE Computer Graphics and Applications*, n$^{o}$ 5, p. 9-13.

Branit, Bruce. 2013. « World Builder ». < https://www.youtube.com/watch?v=QP3YywgRx5A >. Consulté le 2016-01-20.

Broll, Wolfgang, Irma Lindt, Iris Herbst, Jan Ohlenburg, Anne-Kathrin Braun et Richard Wetzel. 2008. « Toward next-gen mobile AR games ». *IEEE Computer Graphics and Applications,* vol. 28, n$^{o}$ 4, p. 0040-48.

Butterworth, Jeff, Andrew Davidson, Stephen Hench et Marc T Olano. 1992. « 3DM: A three dimensional modeler using a head-mounted display ». In *Proceedings of the 1992 symposium on Interactive 3D graphics*. p. 135-138. ACM.

Cruz-Neira, Carolina, Daniel J Sandin et Thomas A DeFanti. 1993. « Surround-screen projection-based virtual reality: the design and implementation of the CAVE ». In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*. p. 135-142. ACM.

De Araùjo, Bruno R, Géry Casiez et Joaquim A Jorge. 2012. « Mockup builder: direct 3D modeling on and above the surface in a continuous interaction space ». In *Proceedings of Graphics Interface 2012*. p. 173-180. Canadian Information Processing Society.

Dorta, Tomas, Gokce Kinayoglu et Michael Hoffmann. 2015. « Hyve-3D and rethinking the 3D cursor: unfolding a natural interaction model for remote and local co-design in VR ». In *SIGGRAPH 2015: Studio*. p. 43. ACM.

Drettakis, George, Maria Roussou, Nicolas Tsingos, Alex Reche et Emmanuel Gallo. 2004. « Image-based techniques for the creation and display of photorealistic interactive virtual environments ». In *Proceedings of the Eurographics Symposium on Virtual Environments*. p. 9. The Eurographics Association.

Veuillez sélectionner un type de document autre que « Generic » afin de faire afficher la référence bibliographique.

Elmqvist, Niklas, et Jean-Daniel Fekete. 2008. « Semantic pointing for object picking in complex 3D environments ». In *Proceedings of Graphics Interface 2008*. p. 243-250. Canadian Information Processing Society.

Engelbart, Douglas C. 1970. *X-y position indicator for a display system*. < https://www.google.com/patents/US3541541 >.

Fitts, Paul M. 1954. « The information capacity of the human motor system in controlling the amplitude of movement ». *Journal of experimental psychology,* vol. 47, n° 6, p. 381.

Fuge, Mark, Mehmet Ersin Yumer, Gunay Orbay et Levent Burak Kara. 2012. « Conceptual design and modification of freeform surfaces using dual shape representations in augmented reality environments ». *Computer-Aided Design,* vol. 44, n° 10, p. 1020-1032.

Gizmag. 2015. « Extended interview: The Oculus Rift, in its creators' own words ». < http://www.gizmag.com/oculus-rift-interview/38002/ >. Consulté le 2015-11-10.

Google. 2013. « Google Glass ». < https://en.wikipedia.org/wiki/Google_Glass >. Consulté le 2015-11-05.

Google. 2015. « Project Tango ». < https://www.google.com/atap/project-tango/ >. Consulté le 2015-09-24.

Grossman, Tovi, Daniel Wigdor et Ravin Balakrishnan. 2004. « Multi-finger gestural interaction with 3D volumetric displays ». In *Proceedings of the 17th annual ACM symposium on User interface software and technology*. p. 61-70. ACM.

Grossmann, Christoph. 2001. *Method and device for autostereoscopy*.

Ha, Taejin, et Woontack Woo. 2010. « An empirical evaluation of virtual hand techniques for 3D object manipulation in a tangible augmented reality environment ». In *3D User Interfaces (3DUI), 2010 IEEE Symposium on*. p. 91-98. IEEE.

Hachet, Martin, Benoit Bossavit, Aurélie Cohé et Jean-Baptiste de la Rivière. 2011. « Toucheo: multitouch and stereo combined in a seamless workspace ». In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. p. 587-592. ACM.

Hinckley, Ken, Randy Pausch, John C Goble et Neal F Kassell. 1994. « Passive real-world interface props for neurosurgical visualization ». In *Proceedings of the SIGCHI conference on Human factors in computing systems*. p. 452-458. ACM.

Hinckley, Ken, Joe Tullio, Randy Pausch, Dennis Proffitt et Neal Kassell. 1997. « Usability analysis of 3D rotation techniques ». In *Proceedings of the 10th annual ACM symposium on User interface software and technology*. p. 1-10. ACM.

HoloLens, Microsoft. 2015. *Microsoft HoloLens: Partner Demo with Maya by Autodesk*. Microsoft HoloLens. < https://www.youtube.com/watch?v=q0K3n0Gf8mA >. Consulté le 2015-09-24.

HTC. 2015. « HTC Vive ». < http://www.htcvr.com/ >. Consulté le 2015-11-05.

Hürst, Wolfgang, et Casper Van Wezel. 2013. « Gesture-based interaction via finger tracking for mobile augmented reality ». *Multimedia Tools and Applications,* vol. 62, n$^{o}$ 1, p. 233-258.

Kaufmann, Hannes, et Dieter Schmalstieg. 2003. « Mathematics and geometry education with collaborative augmented reality ». *Computers & Graphics,* vol. 27, n$^{o}$ 3, p. 339-345.

Kijima, Ryugo, et Takeo Ojika. 1997. « Transition between virtual environment and workstation environment with projective head mounted display ». In *Virtual Reality Annual International Symposium, 1997., IEEE 1997*. p. 130-137. IEEE.

Krs, Vojtěch. 2014. « Sculpting in Virtual Reality - Oculus Rift DK2 + Razer Hydra ». < https://www.youtube.com/watch?v=jnqFdSa5p7w >. Consulté le 2015-11-10.

Lakatos, David, Matthew Blackshaw, Alex Olwal, Zachary Barryte, Ken Perlin et Hiroshi Ishii. 2014. « T(ether): Spatially-aware handhelds, gestures and proprioception for multi-user 3D modeling and animation ». In *Proceedings of the 2nd ACM symposium on Spatial user interaction*. p. 90-93. ACM.

Lazzaro, Gerard M, David C Swift, Gregory J Hamlin et Sadeg M Faris. 1998. *Stereoscopic 3-D viewing system and glasses having electrooptical shutters controlled by control signals produced using horizontal pulse detection within the vertical synchronization pulse period of computer generated video signals*.

Lee, Jinha, Alex Olwal, Hiroshi Ishii et Cati Boulanger. 2013. « SpaceTop: integrating 2D and spatial 3D interactions in a see-through desktop environment ». In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. p. 189-192. ACM.

McMahan, Ryan P, Doug Gorton, Joe Gresock, Will McConnell et Doug A Bowman. 2006. « Separating the effects of level of immersion and 3D interaction techniques ». In *Proceedings of the ACM symposium on Virtual reality software and technology*. p. 108-111. ACM.

META. 2014. « Products: META Company ». < https://www.getameta.com/products >. Consulté le 2015-09-24.

Microsoft. 2015. « Microsoft HoloLens ». < http://www.microsoft.com/microsoft-hololens/en-us >. Consulté le 2015-11-05.

Mine, Mark, Arun Yoganandan et Dane Coffey. 2014. « Making VR work: building a real-world immersive modeling application in the virtual world ». In *Proceedings of the 2nd ACM symposium on Spatial user interaction*. p. 80-89. ACM.

Oculus, VR. 2012. « Oculus rift-virtual reality headset for 3d gaming ». < http://www.oculusvr.com >. Consulté le 2015-11-05.

Oviatt, Sharon. 2007. « Multimodal interfaces ». In *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*, 2nd edition. p. 413-432.

PBC, Kickstarter. 2015. « Kickstarter ». < https://www.kickstarter.com/ >. Consulté le 2015-11-10.

Phillips, Thomas E, et John Marez. 1987. *Stereoscopic three dimensional large screen liquid crystal display*.

Regenbrecht, Holger, Claudia Ott, Michael Wagner, Tim Lum, Petra Kohler, Wilhelm Wilke et Erich Mueller. 2003. « An augmented virtuality approach to 3D videoconferencing

». In *Proceedings of the 2nd IEEE/ACM international Symposium on Mixed and Augmented Reality*. p. 290. IEEE Computer Society.

Samsung. 2015. « Samsung Gear VR ». < http://www.samsung.com/global/galaxy/wearables/gear-vr/ >. Consulté le 2015-11-05.

Sharp, Toby, Cem Keskin, Duncan Robertson, Jonathan Taylor et Jamie Shotton. 2015. « Accurate, Robust, and Flexible Real-time Hand Tracking ». In *Proc. CHI*. Vol. 8.

Sony. 2014a. « Project Morpheus ». < https://www.playstation.com/en-us/explore/project-morpheus/ >. Consulté le 2015-11-05.

Sony. 2014b. « SmartEyeglass ». < http://developer.sonymobile.com/products/smarteyeglass/ >. Consulté le 2015-11-05.

Spindler, Martin, Wolfgang Büschel, Charlotte Winkler et Raimund Dachselt. 2014. « Tangible displays for the masses: spatial interaction with handheld displays by using consumer depth cameras ». *Personal and Ubiquitous Computing,* vol. 18, n° 5, p. 1213-1225.

Sung, Eunmo, et Richard E Mayer. 2012. « Students' beliefs about mobile devices Vs. desktop computers in South Korea and the United States ». *Computers & Education,* vol. 59, n° 4, p. 1328-1338.

Szalavári, Zsolt, et Michael Gervautz. 1997. « The Personal Interaction Panel–a Two-Handed Interface for Augmented Reality ». In *Computer graphics forum*. Vol. 16, p. C335-C346. Wiley Online Library.

Szalavári, Zsolt, Dieter Schmalstieg, Anton Fuhrmann et Michael Gervautz. 1998. « "Studierstube": An environment for collaboration in augmented reality ». *Virtual Reality,* vol. 3, n° 1, p. 37-48.

Talbot, Mike. 2012. « Unity Serializer ».

Toma, Mădălina Ioana, Florin Gîrbacia et Csaba Antonya. 2012. « A comparative evaluation of human interaction for design and assembly of 3D CAD models in desktop and immersive environments ». *International Journal on Interactive Design and Manufacturing (IJIDeM),* vol. 6, n° 3, p. 179-193.

Vuzix. 2014. « Wrap 1200DXAR ». < https://www.vuzix.com/augmented-reality/products_wrap1200dxar/ >. Consulté le 2015-11-05.

Wang, Guangyu, Michael J McGuffin, François Bérard et Jeremy R Cooperstock. 2011. « Pop-up depth views for improving 3D target acquisition ». In *Proceedings of*

*Graphics Interface 2011*. p. 41-48. Canadian Human-Computer Communications Society.

Wang, Jia, et Robert Lindeman. 2014. « Coordinated 3D interaction in tablet-and HMD-based hybrid virtual environments ». In *Proceedings of the 2nd ACM symposium on Spatial user interaction*. p. 70-79. ACM.

Wang, Robert, Sylvain Paris et Jovan Popović. 2011. « 6D hands: markerless hand-tracking for computer aided design ». In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. p. 549-558. ACM.

Ware, Colin, et Glenn Franck. 1996. « Evaluating stereo and motion cues for visualizing information nets in three dimensions ». *ACM Transactions on Graphics (TOG),* vol. 15, n$^{o}$ 2, p. 121-140.

Weichel, Christian, Manfred Lau, David Kim, Nicolas Villar et Hans W Gellersen. 2014. « MixFab: a mixed-reality environment for personal fabrication ». In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*. p. 3855-3864. ACM.

Wikipedia. 2015. « John Carmack ». < https://en.wikipedia.org/wiki/John_Carmack >. Consulté le 2015-11-10.