BAU Olivier

Rapport Magistere2
2004/05
Partie 1/2

Stage I3D-GRAVIR,INRIALPES
Tuteur : Sabine Coquillart

Une étude de conflit entre *vision* et *proprioception*

Sommaire

0. Présentation de l'équipe

Ce stage a été effectué au sein de l'équipe I3D, sous la tutelle de Sabine Coquillart. Les trois principaux axes de recherche de cette équipe sont :

0.1.Techniques d'interactions

L'objectif de ce thème est la conception, l'étude et l'évaluation de paradigmes et métaphores d'interactions 3 Dimensions.

0.2.Etude de retours haptiques

Haptique : En rapport avec le sens du touché. Fait référence à tout les senseurs physiques impliqués dans le sens tactile, au niveau de la peau, des articulations, et impliqués dans les informations en rapport avec le retour de force fournis par les muscles et articulations.
Il existe plusieurs approches pour restituer une sensation de retour haptique : l'utilisation de périphériques à retour haptique (périphérique qui produit un retour de force de façon active), le retour pseudo–haptique (cf. Section II.3.2.Etude Bibliographique), retour haptique passif, ... L'objectif de ce thème est l'étude de ces différentes approches afin de permettre une meilleure caractérisation du retour haptique en fonction de la tâche à accomplir.

0.3.Etude de facteurs humains

Cet axe traite de l'aspect physiologique et psychologique des différents sens de l'homme ainsi que de l'ergonomie de l'interaction à des fins de choix et d'évaluation des solution proposées dans les autres axes de recherche. Des expérimentations, sont conduites soit en amont des recherches menées dans les deux axes précédents (tests psychophysiques sur la perception, évaluation de périphériques ou d'interfaces existantes), soit en aval de ces recherches (évaluation des approches mises au point par le groupe).

I. Introduction

Le sujet de ce stage fait intervenir les 3 axes de recherche de l'équipe, c'est une étude de facteurs humains, qui pourrait avoir des applications au niveau des retours haptiques, des techniques d'interactions.
Ce stage comporte deux aspects : un premier aspect réalité virtuelle/augmentée, techniques d'interaction 3D ; le deuxième aspect est en rapport avec les sciences cognitives. Nous allons tout d'abord exposer différents périphériques de visualisations , puis nous allons faire un état de l'art des paradigmes et métaphores d'interaction 3D (premier aspect). Nous allons ensuite donner quelques définitions de termes importants, et présenter quelques études mettant en jeu conflits et incohérences sensorielles, retours haptiques (aspect cognitif).
Enfin nous allons présenter la phase de développement.
Nous reviendrons en détails sur le sujet dans la partie « III. Problématique» .

II. État de l'art

II.0.Introduction

Réalité virtuelle : Technologie permettant une simulation interactive et en temps réel de la réalité. Elle est réalisée à l'aide d'images de synthèse, d'un environnement virtuel en 3D dans lequel on peut évoluer, interagir.

Réalité augmentée : Environnement réel (éventuellement modifié) dans lequel sont ajoutés des objets virtuels.

Nous noterons, dans la suite RA pour réalité augmentée et RV pour réalité virtuelle.

II.1. Périphériques de vision en réalité augmentée - réalité virtuelle.

Dans cette partie nous allons présenter différentes configurations, différents périphériques de réalité augmentée et virtuelle. Il existe deux grand types de systèmes : les systèmes mobiles (casques) et les systèmes à base de projections sur écrans.

II.1.1. Systèmes par projections :

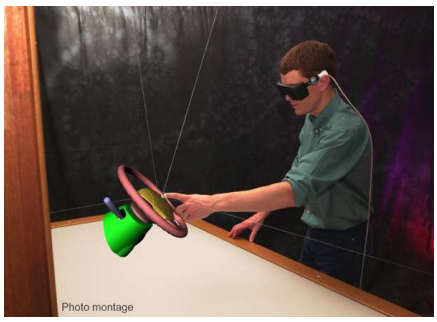
Il existe une multitude de périphériques, ayant globalement le même principe de fonctionnement.
On projette chaque point de vue d'une scène (un point de vue différent pour chaque oeil) sur un ou plusieurs écrans. L'effet 3D est obtenu grâce à des lunettes pour vision stéréoscopique (ex : lunettes à cristaux liquides) qui permettent de filtrer les images projetées, de sorte que chacun des yeux reçoivent l'image correspondant à *son* point de vue.

Nous présentons ici quelques exemples de configurations.

++Cave

Cube dont des faces sont des écrans, on peut alors projeter sur ces écrans les scènes 3D. Les utilisateurs évoluant dans le cube seront alors immergés dans l'environnement 3D.

++Plans de travails virtuels : Workbench



Cette configuration est un bureau virtuel, où les images sont projetées sur un plan horizontal et un plan vertical face à l'utilisateur.

-> Ces dispositifs ne permettent pas la modification de réel. La mobilité de l'utilisateur est restreinte.
On peut visualiser, interagir dans/face à un environnement 3D. Les lunettes ne restreignent le champ de vision que de façon négligeable.

II.1.2.Casques

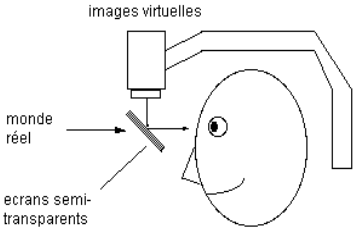
Les casques sont donc des périphériques mobiles, il en existe 4 sortes : les casques opaques, optiques, à projection rétinienne et enfin les casques vidéo.

++Casques Opaques

Glastron de Sony

Les casques opaques disposent d'un écran pour chaque oeil (pour créer l' effet stéréoscopique, i.e. la vision en 3D). Sur ces écrans sont projetés les points de vue respectifs de chaque oeil. Ce matériel est utilisé en réalité virtuelle seulement puisque on ne peut faire intervenir le réel. Ce casque permet donc une forte immersion, le champ de vision est cependant contraint par la -petite- taille des écrans.

++Casques optiques semi-transparents



ProView XL40/50 STm Kaiser Electro-Optics Inc.

L'utilisateur a devant chaque oeil un écran translucide. Il voit donc le réel par transparence. Des images virtuelles peuvent être ajoutées en étant projetées sur ces petits écrans. Les objets sont alors *superposés* au réel (le virtuel « *cache* le réel »). La définition du réel est parfaite, celle du réel est plus mauvaise. La calibration de ce type de matériel est relativement difficile. Les objets ne peuvent apparaître que dans un champs de vision relativement restreint.

++Système par projection rétinienne

Nomad Personal Display System de MICROVISION

Les images sont directement projetées sur la rétine de l'oeil. Le principal inconvenant se trouve dans le fait que la projection est difficile lorsque la fréquence des mouvements des yeux est trop importante, et les mouvements de trop grande amplitude.

++Casque Vidéo semi-transparent

Arvision-3D de Trivisio

Ce casque comporte 2 caméras vidéo et 2 écrans, un pour chaque oeil. Les écrans et les caméras sont 4 éléments indépendant. La vidéo peut donc être traitée, autrement dit le réel peut être modifié. Toutes sortes de données peuvent être affichées sur les 2 écrans : de la vidéo pure (réel), un environnement virtuel, ou les deux. Le principale avantage est que le réel peut être modifié. Il n'y a également pas de problèmes d' occlusions entre réel et virtuel. Les données (réelles et virtuelles) peuvent en effet être traitées avant d'être affichées. Le champ de vision est toute fois limité par l'angle de vue des caméras, et la taille des écrans. La définition des images fournies par ces mêmes cameras n'est pas très élevée.

II.2.Techniques d'interaction

Les différentes techniques exposées ici seront regroupées selon les tâches de base de sélection/manipulation, navigation et contrôle d'application utilisées dans les applications 3D.

Sélection/Manipulation:

Une classification suivant le référentiel d'utilisation sera employée [REF3]:
exocentrique : l'utilisateur porte un regard extérieur à la scène.
égocentrique : l'utilisateur est au centre de la scène. Le point de vue est donc défini par la position de son propre corps.

Techniques égocentriques:

Métaphore de "main virtuelle":

La sélection/Manipulation se fait avec une image virtuelle de la main,dans une espace local("portée de main") :
Elles sont plus souvent utilisées dans des applications avec casques.

++Curseur 3D : Ce curseur est "attaché" à la main de l'utilisateur. Le curseur peut être une image virtuelle de la main ("virtual hand"-RV-), ou alors un pointeur réel dont la position est suivie au cours du temps pour permettre l'interaction avec des objets virtuels (utilisation dans Studierstube [REF15]). Avec un mouvement naturel de la main, on peut placer le curseur sur l'objet à sélectionner (en le touchant). Le retour peut être aussi bien graphique, sonore, ou haptique, pour signaler à l'utilisateur que l'objet est candidat. Pour valider la sélection, on peut effectuer une pression sur un bouton d'un périphérique d'entrée, ou alors effectuer un mouvement "prédéfini", ou même par commande vocale.

++Sélection par occlusion : La sélection se fait en plaçant un objet ou l'index par exemple, entre l'oeil de l'utilisateur, et l'objet à sélectionner. L'objet, dont une partie est couverte, du point de vue de l'utilisateur, est l'objet sélectionné. En réalité augmentée, une interaction par occlusion a été développée dans le cadre d'une interface tangible[REF6], l'objet sélectionné est alors l'objet qui se trouve sur la marque couverte par l'index de l'utilisateur. La sélection est alors centrée sur l'objet à sélectionner, et non sur l'outil de sélection.

++Manipulation gestuelle réelle : En RA par exemple, l'interaction peut se faire de manière "directe" , sans avatar virtuel, comme par exemple dans [REF23], [REF24] ou encore [REF19], les objets virtuels se manipulent directement grâce à un gant ou la main, équipée de marqueurs, permettant de connaître sa position par rapport au objets virtuels. La position de la main peut être également connue par vision,la manipulation se fait alors à "main nue"(exemple en RA2D[REF25]).

La Sélection/Manipulation se fait avec une représentation virtuelle de la main. Cependant l'"espace d'action" réel est étendu :

++Homer technique[REF4] : Cette Technique est un mélange entre la technique de Virtual hand et celle de ray-casting. L'utilisateur sélectionne l'objet à l'aide d'un rayon virtuel. Après la "validation" de la sélection, la main virtuelle se place alors sur l'objet. Lorsque l'objet est relaché, la main revient dans sa position initiale.

++Go-Go technique[REF5] : On applique un "mapping" non linéaire à l'extension de la main/bras de l'utilisateur. A partir d'une certaine limite d'extension, le bras se comporte comme une antenne télescopique. Ceci permet donc une sélection/manipulation dans un espace plus grand que la technique de virtual hand, mais est relativement imprécise "à longue distance". La distance de sélection est augmentée,mais finie.

++Silk Cursor[REF13] : La main de l'utilisateur guide à distance un curseur 3D transparent pouvant englober les objets à sélectionner (par exemple un cube).La sélection d'un objet est possible lorsqu'il se trouve "à l'intérieur" de ce curseur. Le choix de la taille du cube est alors primordial pour avoir le moins d'ambiguïtés possible en ayant un volume tout de même assez gros pour contenir les objets.

Techniques d'interaction "à distance" :
La Sélection/Manipulation se fait par "extension" du bras de l'utilisateur, permettant des actions a distance.

++Rayon virtuel (Ray casting)[REF2] : Un rayon virtuel peut être émis par la main de l'utilisateur ,un pointeur,...On test alors l'intersection de ce rayon avec les objets de la scène. Si intersection il y a, l'objet est marqué comme sélectionné. Lorsqu'il y a intersection avec plusieurs objets, l'objet le plus proche de l'utilisateur est alors marqué comme sélectionné. La sélection/ manipulation de petits objets ou d'objets lointains pose problème, car un petit mouvement de la main peut engendrer un grand déplacement angulaire. Cette technique a été reprise en RA, par exemple dans [REF18] dans le cadre d'une interface tangible. A l'aide de cartes munies de marqueurs, l'utilisateur manipule le marqueur portant le rayon et celui portant l'objet.

++Fishing reel[REF4] (canne a peche): On ajoute à la technique de "ray-casting" la possibilité de tirer l'objet sélectionné vers soit ou de l'éloigner, ceci grâce à un périphérique d'entrée avec boutons par exemple.

++Flashlight technique[REF7] : Le rayon est remplacé par un objet virtuel conique, ce qui facilite la sélection d'objets plus petits, ou lointains, car la forme de l'objet atténue le désavantage que présentait la technique de ray-casting pour les objets lointains. Cependant, le diamètre du rayon étant assez conséquent à une certaine distance de l'utilisateur, la sélection peut devenir ambiguë ce qui oblige à établir des conventions pour lever ces possibles ambiguïtés. Utilisation de cette technique en RA par exemple dans SenseShapes [REF26] une interface multimodale.

++Aperture technique[REF11] : Modification de la technique précédente. Le sommet du cône se situe au niveau d'un oeil de l'utilisateur (l'oeil dominant). La main est équipée d'un curseur dont l'extrémité représente le centre d'un cercle de rayon fixe dans le cône. Il peut, en éloignant (respectivement en approchant) diminuer la "largeur", autrement dit le rayon de la base de ce cône (respectivement l'augmenter).

++Gaze-directed sélection : La sélection d'un objet peut se faire en regardant l'objet,ce qui implique le suivi de l'orientation de la tête, ou, plus précis, le suivi des mouvements de l'oeil dominant. La validation de la sélection peut se faire de différentes manières : à l'aide d'un périphérique d'entrée, par commande vocale, etc...

++Image plane techniques[REF12] : Dans les interfaces de bureau, les images 3D sont représentées par leur projection sur le plan image 2D (affichage sur un écran), les objets sont alors manipulés dans ce plan. Cet aspect a été porté en environnement immersif 3D. L'utilisateur peut par exemple sélectionner et manipuler un objet de la scène en positionnant son pouce et son index de part et d'autre de la "projection" 2D de l'objet 3D qu'il perçoit ("head crusher" technique), ou alors en formant une sorte de cadre autour de l'image qu'il perçoit de l'objet ("framing hands" technique), ou encore en mettant sa main sous cette projection ("lifting palm" technique) donnant l'illusion dans la projection perçu de la scène que l'objet est posé sur sa main. Enfin, un autre mode, proche de la méthode par occlusion, consiste à positionner son doigt sur l'image 2D que l'utilisateur perçoit de l'objet ("Sticky finger" technique).
Cette technique a été reprise en RA par exemple dans [REF22]. La projection 2D "manipulée" ici est l'image sur un petit écran portable qui permet la visualisation des objets 3D.

Techniques exocentriques:
Sélection par changement d'échelle de l'environnement:
Ces techniques peuvent permettre la sélection dans des scènes complexes, avec risque d'ambiguïté de sélection assez élevé pour certaines techniques où l'utilisateur dispose d'une vue d'ensemble de la scène (par conséquent vue moins précise).

++Réduction de l'échelle de l'environnement/scène : L'utilisateur peut ainsi avoir accès aux objets qui n'étaient pas à sa portée, à l'échelle initiale, ou alors plus difficilement manipulable. Dans [REF9] & (Pierce et al.,1997) -"Scaled World Grab Technique"- , il est proposé une mise à l'échelle automatique lors de la sélection, pour une manipulation plus facile. Une fois l'objet sélectionné dans la scène à l'échelle initiale, cette échelle est modifiée pour que l'objet soit à portée de main de l'utilisateur. Lorsque la manipulation est terminée, on revient à l'échelle initiale.

++Copie de l'environnement avec réduction de l'échelle : WIM (World In Miniature)[REF8]. Une copie Miniature de l'environnement est a disposition de l'utilisateur. Il peut donc manipuler indirectement les objets de la scène, même ceux qui ne sont pas visibles dans sa position, en manipulant leur représentation dans cet intermédiaire. Les modifications produites dans ce "clone" sont alors appliquées dans la scène. Cette technique a été portée en réalité augmentée sur un PIP[REF14].

Autres Techniques:

++Interfaces tangibles : manipulation par l'intermédiaire d'un support physique:
Dans les interfaces tangibles (ex "shared space" [REF16]), les objets virtuels ont pour support des objets réels. La manipulation se fait alors par d'intermédiaire de ces supports. On peut aussi trouver des outils réel pour la manipulation d'objets virtuels. Ces objets réels peuvent être des cartes[REF15] de papier munies d'un marqueur, un "tableau" de marqueurs, ou encore un cube avec un marqueur sur chaque face. L'utilisateur peut par exemple, en effectuant une rotation de la carte/marqueur, appliquer une rotation à l'objet qui lui est attaché.

++Virtual Tricorder[REF17]
Un périphérique d'entrée (ici sourie 3D) auquel on peut donner plusieurs fonctions pour diverses manipulation/sélection. Un seul périphérique d'entrée pour plusieurs techniques d'interactions. Ceci a été portée en réalité augmentée.
Par exemple, la "tinmith-hand" [REF19], un gant permettant l'utilisation de différentes techniques de sélection et de manipulation (comme par exemple "virtual hand", "ray-casting", "voodoo dolls") dans une application RA en environnement extérieur.

++Manipulation/sélection Multimodale (exemple : SenseShape[REF26]) : L'utilisateur peut manipuler/sélectionner des objets par gestes, ou par commande vocale, ce qui permet alors d'atteindre des objets de la scène invisibles pour l'utilisateur. La multimodalité permet en combinant les différentes informations, de rendre la sélection d'objets plus précise.[REF21]

++Manipulation avec objet virtuel de commande avec contrainte:
Slider object : Bouton "ascenseur"dont on permet donc que la translation (contrainte). Si a chaque position on associe une valeur, cet objet devient un objet de contrôle. En effet, cet objet est alors un intermédiaire "contraignant" entre le périphérique d'entrée (qui ,peut-être ,fournit "trop" d'informations pour la tâche à effectuer, trop de degrés de liberté), et la manipulation de l'objet. Le choix de l'action associé à l'outil (manipulation de rotation,...) peut se faire par pression d'un bouton, etc.... Pour faire varier la "finesse" de manipulation, on peut prendre en compte un degré de liberté de plus qui serai la distance entre le curseur, la main, ou le périphérique d'entrée, et le bouton à manipuler. Cet objet de contrôle peut par exemple avoir pour support un PIP[REF14] ce qui permet d'ajouter une contrainte(retour haptique de la palette physique).

Scaling box : Pour permettre aussi bien un changement d'échelle uniforme, qu'un changement d'échelle sur un nombre d'axes donné, on peut entourer l'objet à manipuler d'une boîte virtuelle "fil de fer" , ce qui permet de définir un repère de l'objet et de contraindre la manipulation. En manipulant un coin de la boite, le changement d'échelle est uniforme, en manipulant une face, le changement ne se fait que sur un axe donné, etc... Avec une configuration matérielle permettant l'interaction à deux mains, et un périphérique du genre pinch glove sur chacune d'elles, on peut effectuer le changement d'échelle en étirant ou compressant la boite.

Navigation:
Techniques permettant le changement de position et d'orientation du point de vue de l'utilisateur.

++Déplacement à l'échelle : L'utilisateur se déplace physiquement dans l'environnement, ceci grâce à un espace équipé de façon adéquate pour le repérage de l'utilisateur dans cet environnement. Cette technique permet donc le déplacement, le changement de point de vue sur une scène virtuelle de façon naturelle(réelle). Exemple en RA dans Studierstube[REF15].
++Utilisation de mouvements de l'utilisateur : En RV notamment, on peut effectuer un déplacement dans la scène avec des "périphériques " de déplacement tel un tapis roulant,ou alors un vélo d'appartement.

++Trough the lens[REF20] : L'utilisateur dispose d'un deuxième point de vue, qui lui permet de compléter le point de vue qu'il a de la scène. Il peut ainsi naviguer et zoomer sur certains éléments avec cet

intermédiaire, sans modifier son point de vue réel. Une application de cette technique en RA avec le PIP [REF14], ou encore grâce dans [REF22], à un écran portatif.

++Steering (définition de la direction du mouvement)(RV) : On peut par exemple définir cette direction par l'orientation du "regard" de l'utilisateur-gaze-directed steering (réellement le suivi de l'orientation de la tête, ou idéalement l'orientation de l'oeil dominant),ou alors avec un rayon partant de la main, un curseur. La vitesse de déplacement peut se contrôler avec un autre outil.

++Target based travel (RV) : L'utilisateur choisit sa destination, une fois cette position validée, cette technique s'apparente à la "téléportation", donc pas de "perte de temps" entre l'origine et le but du déplacement. Mais le système peut éventuellement produire une transition entre les 2 points. Le point but peut par exemple être choisit à partir d'une carte de l'environnement.

++Route planning (RV) : L'utilisateur trace le chemin qu'il veut parcourir sur une carte de l'environnement, ou alors trace le chemin dans l'espace, ou manipule des icônes, afin de planifier une route. Cette technique permet à l'utilisateur d'accomplir d'autres taches pendant le déplacement.

++Techniques d'aide à l'orientation : Les techniques permettant à l'utilisateur de se repérer dans la scène virtuelle sont souvent les techniques qui le permettent dans la vie réelle, c'est à dire les plans, les boussoles, etc...[REF1]

Contrôle d'applications:

Le contrôle d'application en environnement 3D reprend souvent les techniques 2D:

++Commande vocale
++Périphériques à écran (ex:PDA)
++Menus Graphique (Virtual Widgets)
++Menus virtuels avec support physique (ex : PIP)
++Commandes gestuelles

II.3.Etudes de conflit sensoriels

II.3.1.Definitions

Proprioception :
Sensibilisation du système nerveux aux informations sur les positions et mouvements des membres et articulations.
La proprioception peut ce scinder en 2 catégories : proprioception *consciente et inconsciente*.
La proprioception inconsciente est un phénomène d'adaptation rapide intervenant dans le contrôle de la contraction musculaire, la station debout et les ajustements posturaux (ordre du reflex).
Quand à la proprioception consciente elle joue un rôle dans la sensation de force et de lourdeur accompagnant la contraction musculaire, perception de l'orientation/position du corps et de ses différents segments, notion de schéma corporel. Par abus de langage, le terme proprioception sera employé pour la proprioception consciente.

Conflit sensoriel:
Un conflit sensoriel se produit lorsque le cerveau reçoit des informations incohérentes, contradictoires provenant de sources sensorielles différentes. Par exemple lors d'un conflit entre proprioception et vision : lorsque l'on veut atteindre un objet, la position de la main est donnée entre autre par la vue, et les données proprioceptives. Ces deux informations sont compilées dans le cerveau, il en ressort qu'une seule donnée : la position de la main. Si le champs de vision est modifié, décalé (par exemple en regardant sa main dans un miroir), les données venant des deux flux ne correspondent pas à la même position, il y a alors conflit sensoriel. Le cerveau est capable de gérer ce conflit et de fournir , tout de même, « en sortie », qu'une seule information. Si l'incohérence est trop importante, il y a *rupture perceptive*.

Rupture Perceptive:
Il y a rupture perceptive lorsque le cerveau ne sait plus gérer un conflit sensoriel, les données sensorielles correspondent à des informations « trop » incohérentes. On se rend alors compte de ce conflit.

Illusion:
Les illusions correspondent à une « mauvaise interprétation » d'un flux de données sensorielles (fig.1).

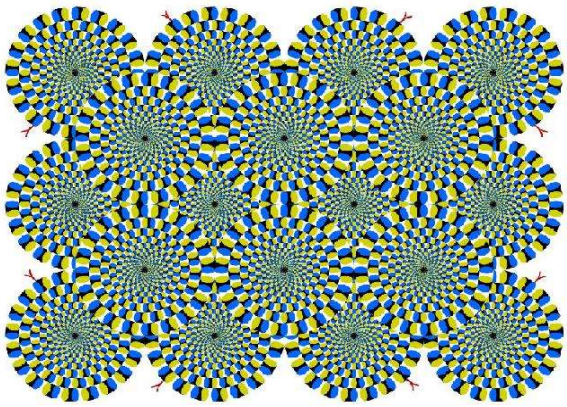


fig.1 : illusion d'optique.

II.3.2.Des études de Conflits sensoriels:

Nous allons ici exposer quelques travaux effectués sur des conflits sensoriels impliquant la vision et la proprioception.

Dans « Pseudo-haptique feedback : Can isométric input device simulate force feedback? » [Ref 32], il est question d'un conflit visio-haptique, c'est à dire un conflit entre les données sensorielles provenant de la vue, et celles impliquées dans la sensation de retour de forces.
Lors de tests, il est demandé à des sujets de comparer des ressort réels d'une part, et des ressorts virtuels d'autre part. Un conflit, une incohérence visio-haptique sera générée lors de l'évaluation du ressort virtuel.
Il est entendu par ressort virtuel le dispositif écran/périphérique isométrique (fig.2) :

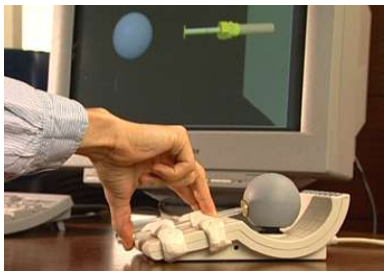


fig.2 : appréciation de la raideur du ressort virtuel : vue du ressort sur écran, appréciation de la raideur avec périphérique isométrique.

fig.3 : évaluation de la raideur d'un ressort réel

Le sujet peut *voir* le ressort à l'aide de représentation 3D sur un écran. La sensation de *retour de force* est procurée par un périphérique isométrique (il y a intensité de « retour de force » égale suivant toutes les directions), elle est donc *réellement* constante. On peut alors modifier la vision de l'utilisateur (ressort en 3D sur l'écran), on fait en sorte que l'un se rétracte plus rapidement que l'autre, mais toujours pour *un retour de force physique/réel constant,* d'où le conflit entre vision et retour haptique, puisque les deux sources sensorielles ne donnent pas la même information sur la raideur.
Il résulte de l'étude que la comparaison entre ressorts réels *est possible au même titre que* la comparaison entre ressorts virtuels. Aussi, on peut *comparer réel et virtuel*.
Nous pouvons émettre l'hypothèse de dominance de la vision sur les autres données sensorielles (impliquées dans la sensation de retour de force), dans ce contexte.

Concernant les conflits sensoriels entre vision et proprioception, des études ont permis une première hypothèse selon laquelle la vision était dominante sur la proprioception [Ref 28], c'est à dire que le cerveau, lors d'un conflit, donne un poids plus fort à la vision. Cependant, dans [Ref29], où le protocole expérimental prend en compte la précision des données sensorielles, il est émis que la « dominance » de la vision serait influencée par la précision des données sensorielles reçues par le cerveau. Dans [Ref 30], il est également montré que, lors d'une action d'adaptation (on donne un invariant au sujet, comme par exemple : l'index doit être sur cet objet à tout moment, puis on perturbe la position de l'objet) la tendance s'inverse lors d'un mouvement en « profondeur » : les données proprioceptives seraient alors dominantes sur les données visuelles. En effet il est plus facile d'apprécier une direction à suivre pour atteindre un but, qu'une distance, les données visuelles dans ce dernier cas sont alors moins précises. Pour les données proprioceptives, les données venant des articulations (ex : angles) sont moins précises lors d'un mouvement horizontal, que lors d'un mouvement « en profondeur » . Le fait que le mouvement du bras soit passif ou actif joue également un rôle, les données proprioceptives sont

moins précises dans le premier cas [Ref 31].
Le cerveau effectue une pondération des informations selon leur précision. Le résultat de la compilation des données sensorielles, la gestion par le cerveau du conflit sensoriel dépend donc du contexte de l'expérience, du protocole expérimental.
Différents périphériques de vision ont été utilisés pour générer un conflit : lunettes à prismes (translation horizontale du champ visuel), systèmes avec miroirs, ... : On peut difficilement modifier la nature de la transformation au cours du temps.
Pour une étude de conflit entre vision et proprioception, un périphérique avec lequel les données visuelles seraient *très malléables* au cours du temps est un élément important.

III. Problématique

III.1. Introduction : Hypothèses

L'hypothèse principale qui ressort des études et expérimentations exposées précédemment est que la vision peut être dominante lors d'un conflit entre vision et proprioception dans certaines conditions.
Dans ces même travaux, la génération du conflit ne se fait que de façon constante (pas de discontinuité dans l'espace, de la « politique de transformation ») . L'incohérence est produite par translation uniforme de données visuelles de positions. Hors ,avec la réalité augmentée, il est possible de *modifier* le réel de façon plus maniable, plus malléable, de manière un créer un conflit dans une zone donnée de l'espace par exemple.

III.2. Mon Travail

Problématique : Nous allons donc étudier un conflit entre vision et proprioception, généré par modification du réel avec un périphérique de réalité augmentée (modification de l'image dont dispose l'utilisateur de son bras lors d'une action de *manipulation*). Nous allons observer l'influence de ce conflit sur les gestes *réels* de l'utilisateur, ainsi que les éventuelles répercutions sur ces perceptions notamment au niveau haptique, et aussi les limites du conflit (cas de rupture perceptive), le but étant de porter ces résultats vers le domaine des techniques d'interaction, mais également de façon plus concrètes, à des applications de type médicales (en psychologie notamment).

Contexte :

Périphérique de vision:
Le périphérique de vision utilisé est le casque semi-transparent vidéo, on peut alors mêler virtuel et réel (fig.4) . De plus la video/le réel est totalement modifiable.

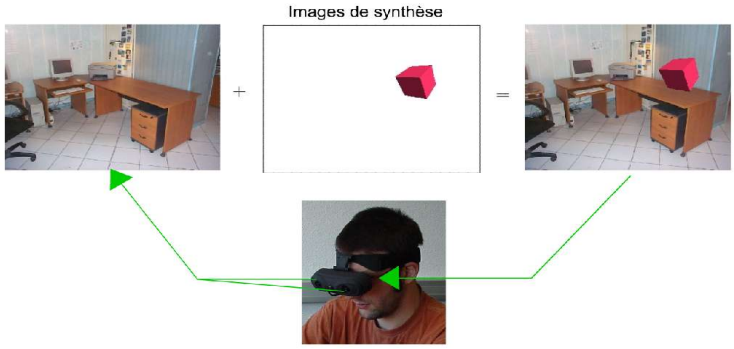


fig.4 : mélange réel virtuel grâce au casque vidéo.

Techniques d'interaction:
Les tests se feront dans un contexte de manipulation par pointeur (pointeur réel dans notre cas), le protocole expérimental est detaillé ci-après.

IV. Protocoles de Tests

IV.1.Introduction

Dans cette partie, nous distinguerons main réelle et main virtuelle. La main réelle de l'utilisateur sera la *source des données proprioceptives*, alors que la main dans le monde virtuel (la main *vue* à travers le casque) sera la *source de données visuelles*. Les informations de position renvoyées par les deux flux peuvent coïncider, ou non. Si non, il y a conflit sensoriel.
Il ne faudra pas confondre main réelle au sens exposé ci dessus et main réelle « dans le monde de l'expérience » . Nous emploierons ce dernier terme pour parler du fait que dans l'environnement vu par l'utilisateur, au travers des lunettes, il peut y avoir des objet virtuels (en 3D) et des objets *en provenance* du réel (provenant des cameras vidéo).
Tout les tests s'effectuent par manipulation dans un espace virtuel à l'aide -de l'image vidéo- de la main réelle, dont on pourra modifier la position (position de la main vue par le sujet). Le mélange réel/virtuel (données vidéo/données virtuelles 3D programmées), ainsi que la modification de l'image réelle, du point de vue logiciel, sera traité dans la partie « V. Mise en oeuvre logicielle » .
Dans cette mise en oeuvre, les manipulations se feront par l'intermédiaire d'un pointeur provenant du réel, car il est plus facile de gérer la position dans l'espace d'un objet rigide, cylindrique, qu'un objet complexe comme la main (c'est pourquoi elle n'intervient pas directement dans l'action de manipulation).

Dans un premier temps, nous effectuerons des tests de façon informelle, ce qui va nous permettre de spécialiser les outils logiciels et de fixer les détails de tests plus formels, dont une base (scénarii généraux) est présentée dans cette partie.

IV.2.Tests informels

L'utilisateur se trouve face à un cube virtuel dans un espace totalement virtuel, hormis l'image de sa main qui elle, proviendra du réel . A l'aide d'un pointeur, provenant également du réel, il devra rentrer en contact avec les surfaces du cube (« toucher » le cube).
Le conflit sera alors généré en faisant en sorte que le pointeur ne traverse jamais le cube. On modifiera l'image que l'utilisateur aura du pointeur, de sa main et de son bras (partie réelle du monde de l'expérience), pour que le pointeur paraisse être « bloqué » au contact d'une face. Les données proprioceptives, et les données visuelles modifiées ne seront alors pas cohérentes (ne « renvoient » pas les même indications de positions) puisque le bras réel sera dans le cube, alors que le sujet verra l'extrémité du pointeur à sa surface.

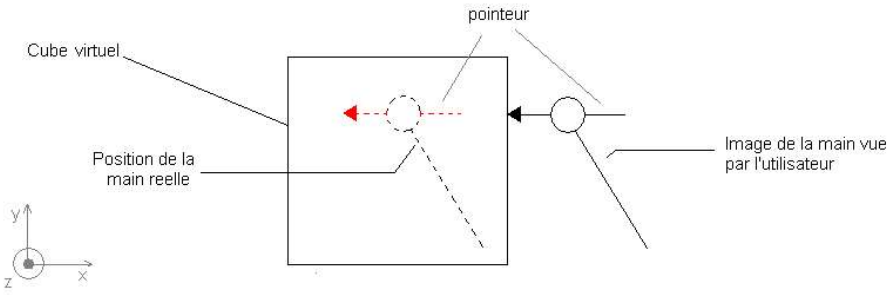


fig.0 : vue schématique (« de face ») d'une situation de conflit

Questions : Que ressentira l'utilisateur?
Quel sera l'influence de la modification sur les gestes réels de la main?
Est-ce que l'utilisateur va freiner son mouvement vers l'intérieur du cube en voyant qu'il ne peut virtuellement aller plus loin?
Y-aura-t-il rupture perceptive?
(...)

IV.3.Test de trajectoire

L'utilisateur se trouve face à un plan virtuel (toujours dans un espace virtuel) . Il devra, en suivant un guide (trace sur le plan virtuel) avec un pointeur, apprendre un mouvement prédéfini (fig.5). Une fois la trajectoire assimilée, on retire sa représentation du plan, les tests débuterons alors. Le sujet devra effectuer le mouvement plusieurs fois. Dans certaines phases nous modifierons la position de l'image de sa main durant son mouvement (fig.6), selon différents axes, « d'intensité » différente. Il y aura alors conflit sensoriel en ces périodes de modifications.

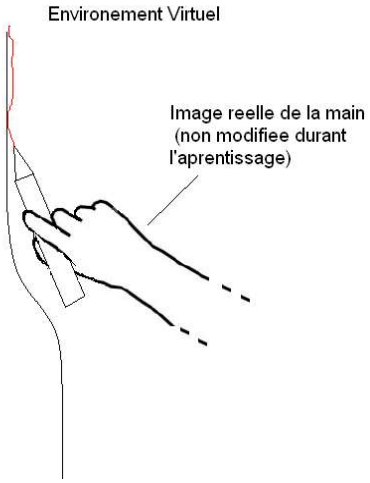


fig.6 : apprentissage

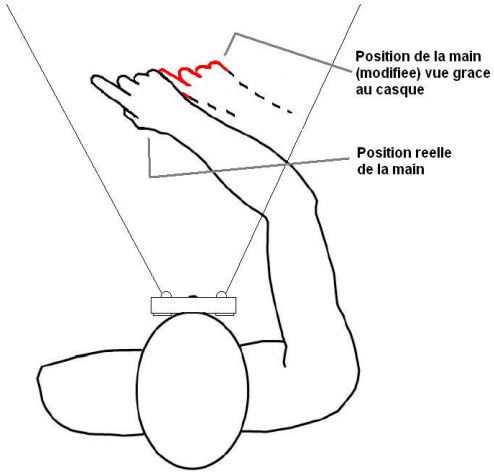


fig.5 : modification de la vision
- interaction avec pointeur, bien qu'il ne soit pas representé ici -

Questions : Lors d'une phase de mouvement avec conflit, quelle sera la trajectoire du geste réel par rapport à la trajectoire virtuelle (trajectoire initiale) apprise par le sujet?
Rupture perceptive dans quelles conditions?
(...)

IV.4.Test de Vitesse

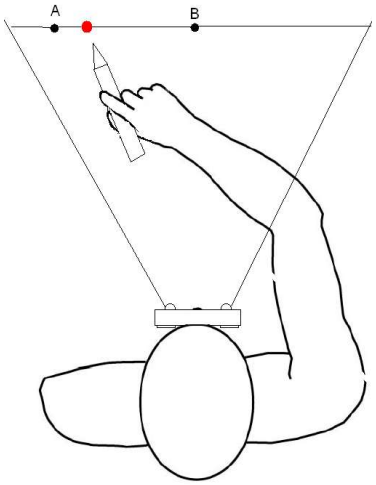


fig. 7 : apprentissage

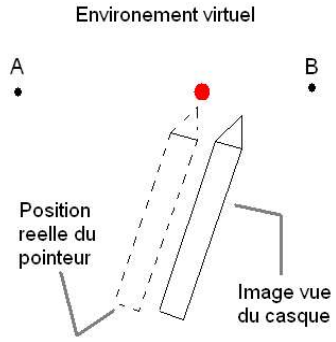


fig. 8 : modification de la vision

L'utilisateur suit, sur un plan virtuel, à l'aide d'un pointeur un point qui parcours un segment à vitesse constante (fig. 7). Durant des phases de l'expérience, on modifiera la position de sa main, ou/et sa vitesse (fig.8) suivant diverses directions.

Questions : Quel sera le segment réel parcouru et à quelle vitesse (en comparaison avec la vitesse du point à suivre) ?
Rupture perceptive dans quelles conditions?
(...)

-> Nous présentons dans la suite l'implémentation de la partie logicielle qui permet de générer le conflit sensoriel. Nous débuterons par la phase de tests préliminaires exposée ci-dessus, nous exposerons dans la suite le développement de l'application pour ces premiers tests.

V. Mise en oeuvre logicielle

Dans un premier temps, nous présenterons une plateforme de base (application repartie) . Nous donnerons quelques détails sur deux modules très importants dans le mélange réel et virtuel, ainsi que dans la génération du conflit sensoriel : module serveur vidéo et module serveur pour le suivi d'objets.
Dans un deuxième temps, nous présenterons la partie implémentation de ce qui va nous permettre de générer un conflit sensoriel, et d'effectuer les premiers tests informels.

V.0.Introduction

Le langage utilisé pour le développement est le C++, et la librairie graphique OpenSG (sur-couche de OpenGL) [Ref 27]. Cette librairie permet des rendus rapides, et fournit des outils pour la programmation d'applications reparties (cluster).
Elle est très utilisée pour la création d'applications en environnements virtuels. Nous parlerons de monde/camera/primitives *OpenSG* pour parler des éléments du monde virtuel programmé.
La spécification de point de vue dans la programmation OpenSG est un élément très important. Le but d'une telle application est d'immerger l'utilisateur dans un monde ou' réel et virtuel se mélangent. Il nous faut donc un lien entre le *point de vue physique* de l'utilisateur (son orientation dans le monde réel) et le *point de vue dans le monde virtuel* (monde programmé en OpenSG). Ce point de vue virtuel est spécifié lors de la création de cameras qui définissent un cône de vision (ou frustum, i.e. « «champs visuel » du point de vue programmation 3D). Les objets 3D présents dans le frustum (qui peut évoluer au cours du temps, puisque les cameras bougent selon le point de vue physique de l'utilisateur) vont être projetés sur le « plan image » . Cette projection correspond à l'image 2D que l'on voit sur les écrans du casque.
Ce lien entre les 2 points de vue va être le suivi de la tête de l'utilisateur. On appliquera alors des transformations de positions sur les cameras qui définissent le point de vue virtuel, à partir des données du suivi. Ceci permet alors de mélanger les mondes réel et virtuel, et de faire en sorte qu'ils soient dans un seul et même système de coordonnées (repère). Nous reviendrons sur le suivi d'objets dans la partie « V.1.3.Suivi de l'utilisateur et des outils de manipulation » .

V.1.Plateforme

V.1.1.Plateforme de base : miniosg

Une plateforme relativement génériques a été développée au sein de l'équipe pour des applications en réalité augmenté/virtuelle. Elle est utilisée pour des environnements sur « Worckbench » (cf. II.2.1. Systèmes par projections).
C'est une application repartie (fig. 9).

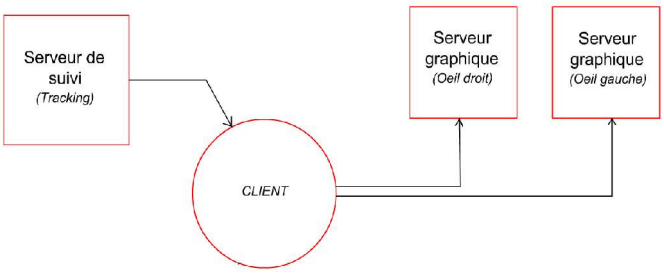


fig.9 : modules de miniosg

Le serveur de suivi (« Tracking ») fournit des positions et des rotations d'objets dans un repère prédéfini. Le client utilise ces données pour modifier le monde virtuel. Il envoi les mises a jours à effectuer aux serveurs graphiques. Ces serveurs graphiques (un pour chaque cameras OpenSG, c'est à dire un pour chaque oeil) se chargent ensuite des rendus des scènes 3D , ceci de façon très efficace (structure de graphe de scène OpenSG, structure présente dans la plupart des surcouches d'OpenGL ; ex : OpenInventor). Les autres communications sont assurées par le module ComLayer, qui utilise le protocole UDP: on privilégie la vitesse plutôt que la sécurité, pour plus d' efficacité.

V.1.2.Intégration module vidéo

Cette plateforme de base n'est pas totalement adaptée au casque semi-transparent vidéo. On doit intégrer un module (serveur) pour acquérir les donnée vidéo venant du casque (prise en compte du réel !).
Il récupère les données vidéo des 2 cameras, et les envoie au client pour être traitées.

Pour une raison d'éfficacité, le serveur envoi ces données bruts, sous un format « bas niveau » : codage bayer. Ce format correspond à la représentation des données récupérées par les capteurs des cameras. Il faudra donc effectuer une conversion de données (interpolations) , pour ramener chaque pixel à une représentation Rouge Vert Bleu (il faut compléter 2 composantes pour chaque pixel : fig.10).

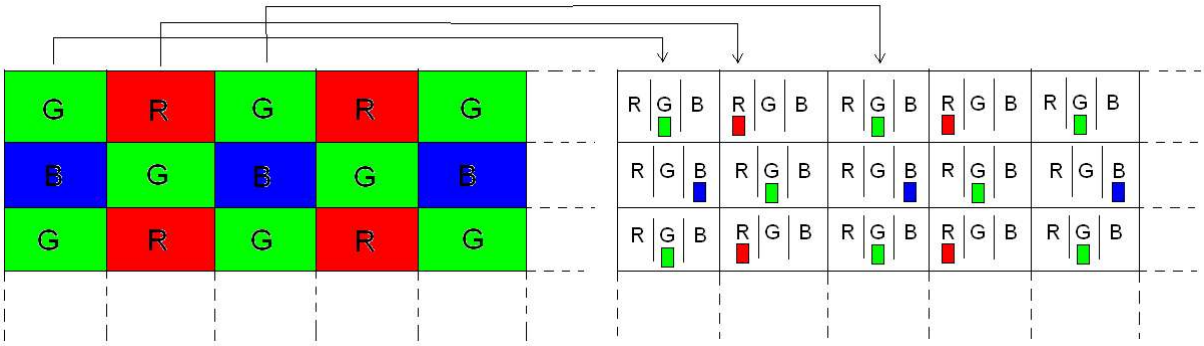


fig.10 : les données bayer sont incomplètes pour la représentation RGB

Ici l'organisation des capteurs des cameras est GRBG, mais il en existe des différentes (ex : RGBG, cela dépend des fabricants). Des composantes doivent être interpolées pour compléter le schéma RGB. Nous présentons en Annexe (Annexe A) de façon plus complète le principe d'interpolation pour la conversion Bayer->RGB (Algorithme NearestNeighbor).

V.1.3.Suivi de l'utilisateur et des outils de manipulation.

Le but du module de suivi est tout d'abord de faire coïncider le point de vue de l'utilisateur avec celui fourni par les cameras définies dans OpenSG (une camera pour chaque oeil). Le système de coordonnées utilisé dans OpenSG pour spécifier des positions de primitives 3D lors de leurs création, et les transformations, sera le même que celui du monde réel, c'est à dire celui spécifié lors de la calibration du système de suivi (même origine, etc...).
Les deux mondes doivent avoir le même repère pour que l'on puisse aussi avoir la position du sujet par rapport à des objets virtuels (programmés en OpenSG).
On récupère donc les coordonnées de positions renvoyées par le périphérique de suivi, et on applique la transformation adéquate aux cameras OpenSG.

On aura également besoin, notamment pour l'application de tests préliminaires, de connaître la position d'objets - accessoires- réels (il nous faudra connaître la position du pointeur par rapport au cube virtuel...)
Il existe différent périphériques pour le suivi d'objets dont les principaux sont:

Suivi Magnétique : Un bloc contenant 2 bobines émettent un champs magnétique, on place ce bloc à l'origine. Les récepteurs sont constitués de 3 petites bobines perpendiculaires entre elles, ce qui permet de déduire les 3 coordonnées du point dans l'espace, ainsi que 3 angles de rotations.

Suivi Optique : Des cameras infrarouges sont placées autour de la scène. Des boules reflechissantes sont placées sur les objets dont on veut suivre les mouvements. L'origine est spécifiée lors d'un processus de calibration du système.

Le système que nous utiliserons est le système optique (par infrarouges), car le système magnétique risque d'être interféré par les émissions du casque vidéo. Il est ,de plus, plus précis. Par contre, il y a des risques d'occlusions : si les cameras n'ont plus les marqueurs dans leurs champs de vision, la position de l'objet marqué ne pourra être connue...
Le module pour ce suivi était déjà présent dans la plateforme de base. On puise donc les informations de positions des objets réels souhaités dans ce module.

V.1.4.Synthèse : schema.

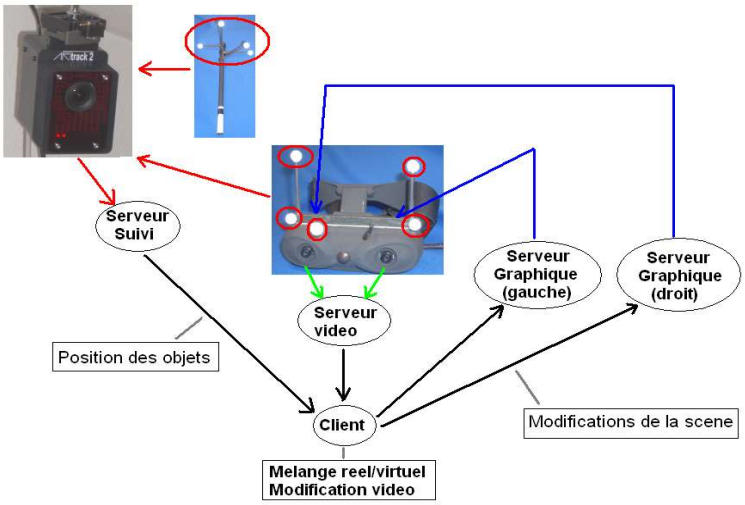


fig.11 : lien entre le matériel et la plateforme logicielle

V.2.Créer un conflit : Mélange entre réel (modifié ou non) et virtuel

La façon dont on va implémenter le mélange ente réel et virtuel, d'un point de vue programmation, est lié à l'implémentation pour la génération de conflit entre vision et proprioception. Nous exposerons donc une solution relativement générale, puis nous l'instancierons pour pouvoir effectuer les premiers tests(cube : IV.2.Tests informels).

V.2.1.Base de l'implémentation

Nous allons présenter le mélange entre réel et virtuel effectué dans le client, ayant accès aux services des modules présentés dans la section précédentes.

Nous spécifions donc 2 cameras OpenSG, ceci implique 2 cônes (frustum) qui correspondent au champ de vision de chaque oeil (i.e. à un point de vue sur la scène virtuelle) . La scène virtuelle n'est crée qu'une seule fois (graphe de scène).On pourra bien sur modifier la scene au cours du temps (modification de noeuds du graphe). Les position et paramètres de ces cameras doivent coïncider avec celles des cameras physiques (cf. Annexe B pour la calibration de la position des cameras virtuelles par rapport aux cameras vidéo). Si les caméras OpenSG n'ont pas le même écartement que les cameras physiques,ni les même paramètres intrinsèques (focale, etc...), l'effet stereo ne pourra pas fonctionner pour la main, le pointeur ET les objets virtuels.
Les cameras ont chacune un plan de projection chacune, ou est projeté la scène OpenSG. Nous créons 2 plans (Primitives OpenSG), parallèles à ces 2 plans de projections, en face de chaque camera. Ces plans suivront les mouvements de la camera : applications des même transformations que celles appliquées aux cameras sur base des données du serveur de suivi.
Ensuite nous appliquons une texture sur ces deux plans. Il s'agit des images vidéos des cameras du casque, récupérées en temps réel par l'intermédiaire du serveur vidéo. Nous utilisons des structures OpenSG relativement volumineuses au niveau du codage, mais très efficaces pour l'actualisation de textures en temps réel : Texture Chunks. Ainsi, la main réelle provenant de la vidéo sera integree dans l'environnement 3D OpenSG.
Pour les différents tests, il est question de manipulation en environnement virtuel avec image de la main réelle. Il nous faut donc, avant d'appliquer cette texture, effectuer un traitement sur les données vidéos afin de ne garder que la main . On pourra alors disposer de l'image de la main dans le monde virtuel OpenSG.
Pour cela, le sujet effectuera les manipulations devant un « fond bleu » .Nous convertirons les données vidéo RGB en RGBA (prise en compte de la transparence). On pourra alors « séparer » la main du reste du monde réel (cf. Annexe C).
Nous avons donc ses deux plans texturés avec la main seule qui évolue et manipule dans l'environnement virtuel (fig. 12).

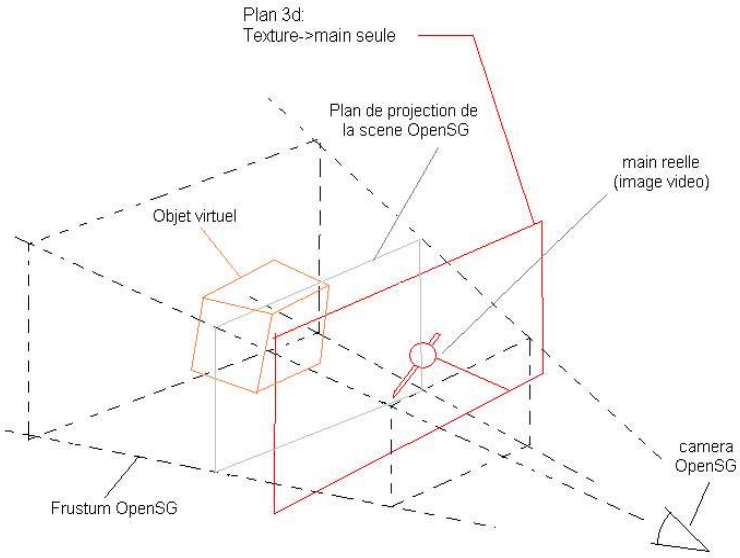


fig. 12 : schéma de l'organisation du monde virtuel (un oeil/camera virtuelle<-> un point de vue) -point de vue programmation 3D OpenSG-

Il est ensuite facile de créer un conflit entre vision et proprioception. Il suffit de déplacer ce(s) plan(s) OpenSG , de les transformer, etc... Ceci même titre que tout autres primitives 3D OpenSG:

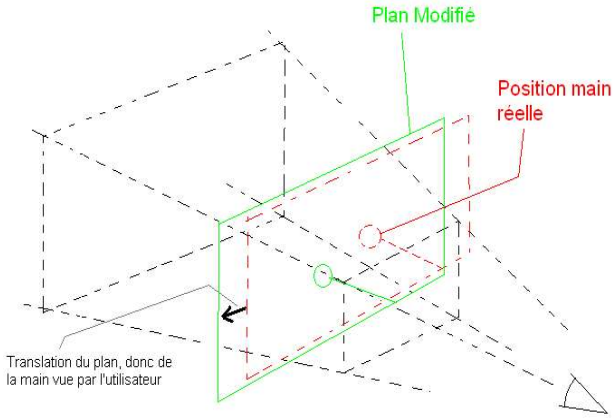


fig. 13 : création du conflit : on déplace le plan (texture = Vidéo/réel)

Nous avons placé les plans entre les cameras et le plan de projection, suffisamment proche des cameras, pour que lorsque l'on transforme la vue de la main, c'est à dire que l'on déplace les plans, il n'y ai pas le plan de l'oeil droit qui soit visible par l'oeil gauche, et inversement.

Avantages de l'implémentation : Avec cette méthode, l' image du réel peut être modifiée en tant que primitive de la scène 3D. Une autre solution serait de spécifier un premier plan fixe. Les modifications de positions de la main par projection (cf. V.2.2.Implémentation pour les tests préliminaires) auraient alors été effectuées sur l'image vidéo , avant d'être appliquée sur le premier plan. Mais dans ce cas, nous n'aurions pas pu profiter de l'efficacité de rendu des transformations que procure la librairie (Graphe de scène : seule les modifications sont envoyées au serveur graphiques). Un clone texturé du bras de l'utilisateur, par l'image réelle de la main pourrait être envisagé , mais on perdrait le caractère réel de la main/bras, puisque il est encore difficile de modéliser ce membre en 3D de façon réaliste.
Dans la partie suivante, nous allons préciser comment et quand effectuer ces transformations, *pour* l'application de tests informels.
Les limites de l'implémentation de façon générale fera l'objet d' un paragraphe (VI. Contraintes matérielles et logicielles, propositions de solution).

V.2.2.Implémentation pour les tests préliminaires

Nous disposons donc d'une application qui permet de déplacer la main, lors d'une action de manipulation dans l 'espace virtuel.
Les tests préliminaires consistent en un « contact » avec un cube virtuel à l'aide d'un pointeur reel. Nous plaçons donc ce cube dans la scène. Il nous faut maintenant implémenter une « politique » de transformation de position du pointeur/bras/main. On test la position du pointeur par rapport à la position du cube. Si le pointeur se trouve être à sa surface, ou à l'intérieur *alors* il faut appliquer une transformation à l'image du pointeur/main/bras, pour faire en sorte que le pointeur reste à sa surface.
Nous disposons de la position 3D du pointeur (suivi par infrarouges). On utilise la projection de ses coordonnées 3D sur le plan image. En effet, déplacer le bras en 3D, revient à déplacer sa projection (plan texture) de façon adéquate. Ceci revient à une translation d' une primitive OpenSG.

On récupère la matrice de projection des cameras OpenSG en temps réels, puisque leur position peuvent évoluer. On spécifie des coordonnées limites, qui correspondent aux points du cube. Lors du contact virtuel avec une face donnée (position modifiée du pointeur), on doit tout de même laisser à l'utilisateur la possibilité de « parcourir » la surface s'il le souhaite, il faudra donc ne restreindre que la composante du mouvement suivant une normale à la face (fig.14).

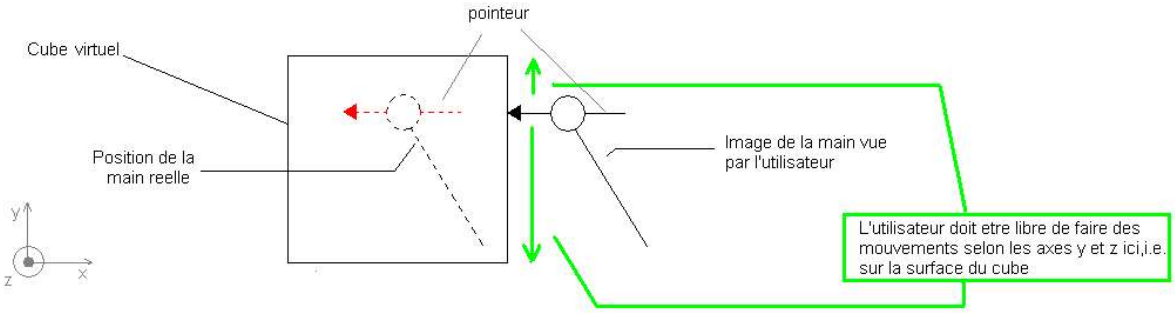


fig. 14 degrés de libertés a la surface du cube

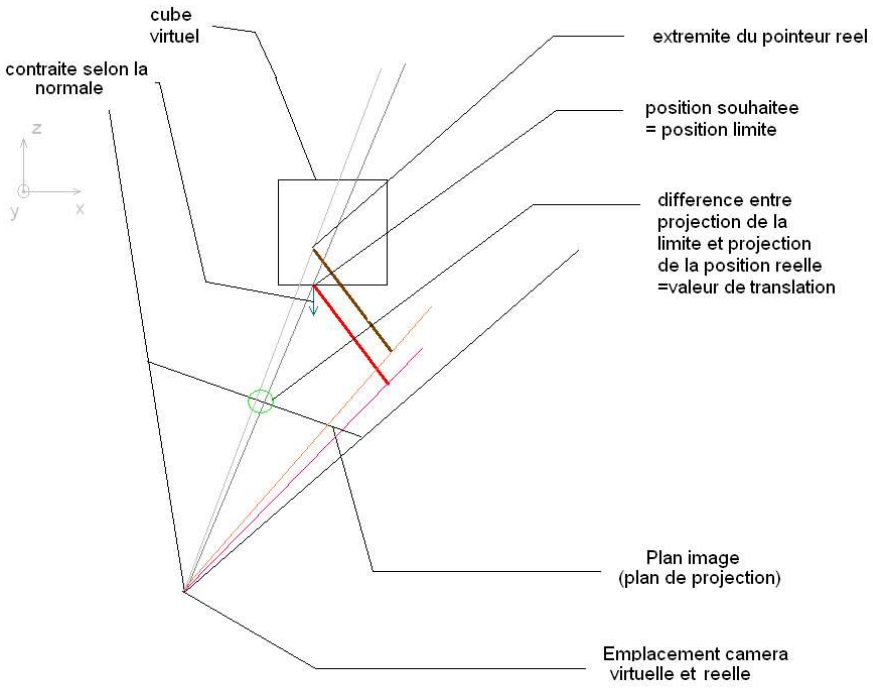


fig.15 représentation du calcul de la translation à effectuer

Les mouvements selon les autres composantes étant libres, on complète la matrice limite avec les coordonnées réelles du pointeur (cf. Annexe D pour un schéma du principe). On projette sur le plan image le point limite et le point correspondant à la position réelle du pointeur. Le point limite est le point le plus proche sur la surface «virtuellement touchée » du cube . On fait la différence entre les coordonnées limites projetées et les

coordonnées réelles projetées, ce qui nous donne la translation à effectuer sur les plans texturés par la Vidéo, pour contrer le mouvement.
Dans l'exemple de la fig.15, on se place dans le cas simple ou les normales des faces coïncident avec les axes du repère, au signe prêt. Le mouvement ne doit restreint selon l'axe z (direction de la normale). La matrice limite sera donc la matrice de position du pointeur, avec en z la valeur correspondant à la composante z des points de la face touchée. Le pointeur pourra alors continuer à évoluer selon les axes x et y(Cet exemple sera repris dans l' Annexe D).

*2 points de vue lors de l'utilisation de l'applicatio*n:

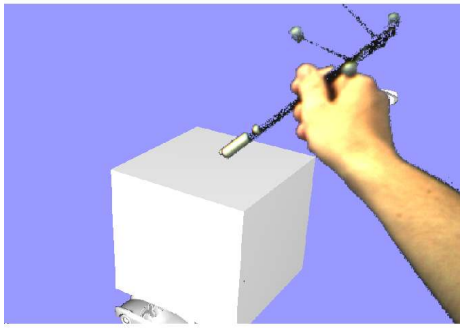


fig.16 : Vue du casque

fig.17 :Vue extérieure

Note : On remarque que des points de l'image du pointeur sont effacés (transparents). Ceci se produit lorsque le fond est de couleur trop sombre, que l'éclairage est trop faible. La couleur du fond se rapproche alors trop de la couleurs du pointeur . Le phénomène se produit typiquement lorsque des ombres apparaissent sur le fond uniforme.

VI. Contraintes matérielles et logicielles, propositions de solution

Les principales limitations viennent du fait que, pour intégrer, modifier le réel dans le monde OpenSG, on ne dispose que des images données par les cameras vidéo. Les images étant de taille fixes, il est possible de modifier la position de l'image de la main, son apparence mais seulement dans la limite de l'image disponible. Voici quelques exemples qui illustre cette contrainte matérielle.

1. La main (réelle) sort du champ de vision des cameras. Par exemple, lors des premiers tests avec le cube, si l'on rentre en contact avec la face supérieure, et que l'on effectue un mouvement trop ample, que l'on exagère « l'appui » sur la surface du cube virtuel, la main réelle va se « baisser » jusqu'à sortir du champ des cameras physiques. Dans cette implémentation, on va translater l'image disponible pour maintenir l'extrémité du pointeur à la surface du cube. Il arrivera alors que l'on voit sa main coupée, puisqu'on effectuera une translation correcte du point de vue des coordonnées dans l'espace, mais l'image totale de la main n'est plus disponible. On aura une coupure, qui n'est autre que le bord inférieure de l'image Vidéo (fig.17).

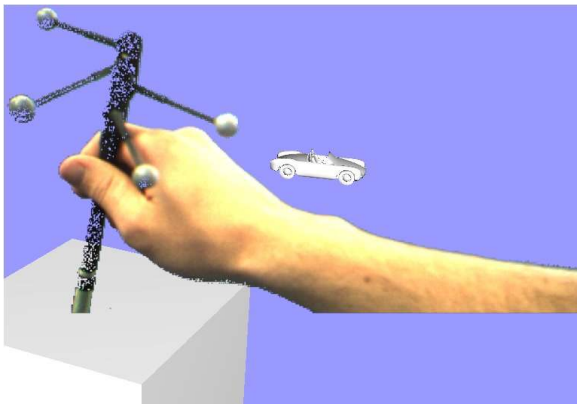


fig.18 : L'extrémité du pointeur est toujours à la surface du cube, mais la main réelle étant en dehors du champs des cameras, le bord inférieur est apparent.

Propositions de Solution : réduire le champs de vision de l'utilisateur, au niveau des écrans du casque, par exemple avec une bordure noire, ce qui donne un peu de marge (surplus) au niveau de l'image donnée par la camera.
Cela dépend également des tests, on pourrait mettre une contrainte physique aux mouvements du sujet, de façon à ce qu'il ne sorte jamais du champs de vision des cameras (exemples : coude sur la table, signal visuel quand le mouvement du sujet est trop ample, etc...)

2.Un autre problème venant de la vidéo est le cas de modification de la position de la main « en profondeur ». Si l'utilisateur « pousse » le cube qui se trouve face à lui, avec le pointeur, on va bien arrêter le mouvement, mais les cameras donneront une image de la main qui va être de plus en plus petite au fur et a mesure que la main (position réelle) va s'éloigner...
Propositions de Solution : On pourrait contrer le phénomène en changeant la taille de l'image. Cependant,dans le cas de l'agrandissement il y a aura une perte de qualité (pixelisation), ceci n'est pas acceptable pour l'étude de facteurs humains. Ce phénomène n'est notable que lorsque la position réelle de la main est très éloignée de la surface du cube touchée. Une solution peut être de limiter le geste réel de l'utilisateur.

Voici quelques limites de l'implémentation qui nous sert à générer le conflit sensoriel (modifier la position de la main) :

3.Ici, seule l'extrémité du pointeur est suivie, il y a donc un problème si l'on touche le cube avec une autre partie du pointeur . Il n'y aura pas de modification de position - repositionnement de l'image à la surface - dans certains cas puisque l'extrémité peut-être en dehors du cube.
Une solution serait de créer un double transparent du pointeur réel, dans le monde virtuel avec les primitives OpenSG (dont la position et l'orientation est bien sur cohérente avec la position du pointeur réel, ceci se fera grâce aux données de suivi du pointeur). On utilisera ensuite un algorithme de détection de collisions entre objets 3D pour connaître la position limite (de l'objet - et non plus extrémité du - pointeur au contact du cube?) lorsque l'on aura besoin de modifier la position du pointeur, pour qu'il reste à la surface du cube.

4.Il y a également une autre contrainte venant de l'implémentation. L'image de la main, dans le monde virtuel (monde « OpenSG ») a le statut de plan, ce qui implique qu'il ne peut être *que totalement* devant ou derrière d'autres objets virtuels. Ce qui a pour conséquence que le bras ne puisse pas « entourer » un objet. On ne peut donc pas avoir la main derrière le cube et une partie du bras au premier plan, cachant une portion de cube.
Une solution serait de découper le plan, l'image 2D du réel, en petites portions (dont la forme dépend de celles des objets de la scènes virtuelle). Grâce à un algorithme de culling, ou avec un Z-buffer (comparaison rapide de la profondeur des objets calculées de façon efficace par la carte graphique), on pourra comparer la profondeur des objets virtuels, donc comparer la profondeur des portions de la main avec les objets virtuels. Puis, en temps réel on place des portions derrière ou devant les objets virtuels, suivant la position du bras par rapport à ces même objets (il faudra que l'intégralité du bras soit suivi).

Certaines de ces contraintes peuvent ne pas être problématiques, cela dépend du protocole expérimental. Dans notre cas, pour les 3 expériences, il nous faudra par exemple trouver une solution pour le premier problème (« main coupée »), la main étant toujours *strictement* au premier plan, nous n'avons pas à traiter le problème précédent.

VII. Travaux futurs

VII.1. États courants des travaux

Une première implémentation (avec notamment une première version de l'algorithme pour la séparation de la main du reste de l'image) a permis des tests qui se sont révélés prometteurs. Le contact avec le cube procure des sensations intéressantes, dont l'étude se fera de façon rigoureuse par la suite, sur des sujets extérieurs au projet.
Cette première implémentation (on ne touchait qu'une face du cube) n'était pas encore adaptée en vue de tests sur des sujets « extérieurs » au problème. L'algorithme de « détour » de la main a été amélioré, car on pouvait voir la translation : des portions de réel autres que la main était visibles, on pouvait donc voir les bordures (coupures) de l'image vidéo lors de la translation des plans.
Les derniers essais ont soulevé un problème de latence important lors de l'affichage de la mise à jour de la vidéo (donc du réel) ce qui donne une impression de mollesse au contact du cube. Ce phénomène etait moins important lors de l'utilisation du premier algorithme, mais il couvrait un plus faible panel de teintes pour l'algorithme de séparation de la main. La latence est un critère essentiel dans l' étude de facteurs humains. Le travail courant est donc un travail d'optimisation de l'algorithme de séparation de la main.

VII.2.Travaux futurs

L' application developpée pour le touché du cube virtuel, sera utilisée pour les tests formels de trajectoire et vitesse. Un travail d'optimisation sera fait d'en un premier temps, puis dans un second temps un protocole experimental plus détaillé sera produit. En effet, il faudra en autre réfléchir au nombre et type de(s) sujets, le temps que durera l'experience, les conditions de lumière, la position du sujet par rapport aux éléments virtuels lors des mouvements, ...
Ce travail est repris cette année par un doctorant.

Références :

1 : R.P.Darken & HelsinCevik.
« Map Usage in Virtual Environments: Orientation Issues»

2 : A.Bolt.
« Put-that-there : Voice and gesture at the graphics interface ».1980.

3 : I.Poupyrev et al., 1998.
« Egocentric Object Manipulation in Virtual Environments:
Empirical Evaluation of Interaction Techniques. »

4 : D.Bowman et al.,1997.
« An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. »

5 : I. Poupyrev et al.
« The Go-Go interaction Technique:Non-linear Mapping for Direct Manipulation in VR. »

6 : G.A.Lee et al.
« Occlusion based Interaction Methods for Tangible Augmented Reality Environements. »

7: Liang et al.
« JDCAD:A highly interactive 3D modeling system ».1994

8 : Stoakley et al.
« Virtual reality on a WIM:interactive worlds in miniature ».1995

9 : M.R.Mine et al.
« Moving Objects In Space: Exploiting Proprioperception In Virtual-Environment Interaction »

10 : J.S.Pierce et al.
« Voodoo Dolls : Seamless Interaction at Multiple Scales in Virtual Environment ».

11 : A.Forsberg et al.
« Aperture Based Selection for Immersive Virtual Environments ».

12 : J.S. Pierce et al.
« Image Plane Interaction Techniques In 3D Immersive Environments ».

13 : S.Zhai et al.
« The "Silk Cursor":Investigating Transparency for 3D Target Acquisition ».

14 : Z.Szalavari & M.Gervautz.
« The Personal Interaction Panel-A Two-Handed Interface for Augmented Reality ».

15 : Z. Szalavàri et al.
« "Studierstube".An Environment for Collaboration in Augmented Reality»

16 : H.Kato et al.
« Virtual Object Manipulation on a Table-Top AR Environment »

17 : Wloka & GreenField.
« The virtual tricorder : a uniform interface for virtual reality.»1995.

18 : H.Slay et al.
« Interaction Modes for Augmented Reality Visualization. »

19 : B.H.Thomas,W.Piekarski.
« Glove Based User Interaction Techniques for Augmented Reality in an Outdoor Environment »

20 : S.L.Stoev et al.
« The Trough-The-Lens Metaphor:Taxonomy and Application .»

21 : E. Kaiser et al.
« Mutual Disambiguation of 3D Multimodal Interaction in Augmented and Virtual Reality. »

22 : J.Rekimoto.
« Transvision:a hand-held augmented reality system for collaborative design. »

23 : S. Veigl et al.
« Two-Handed Direct Interation with ARToolKit. »

24 : D.Scmalstieg et al.
« Finger tracking for interaction in augmented environments. »

25 : J.L.Crowley et al.
« Finger Tracking as an Input Device for Augmented Reality. »

27 : www.opensg.org

28 : Welch,R.B., and Warren,D.H.
« Intersensory interactions. »

29 : RJ van Beers et al. ,1999
« Integration of proprioceptive and visual position-information »

30 : RJ van Beers et al. ,2002
« When feeling is more important than seeing in sensorimotor adaptation »

31 : J.Paillard et al.
« Active and passive movements in the calibration of position sense. »

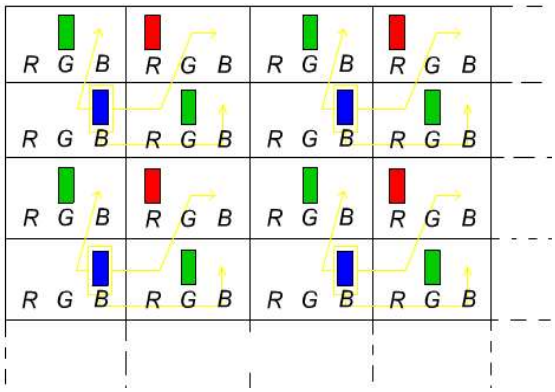
32 : Anatol Lecuyer et al.
« Pseudo-haptique feedback : Can isométric input device simulate force feedback? »

Annexe A : Conversion Bayer -> RGB : Principe de l'algorithme.

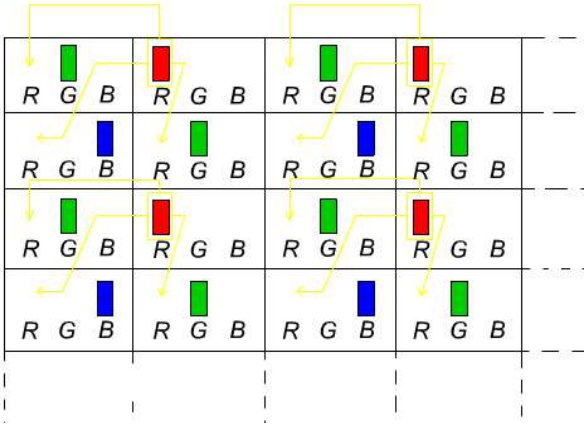
Algorithme Nearest Neighbor :
On recopie les données Bayer dans le pattern RGB, qui en ressort incomplet (fig.10,partie V.1.2.Integration module vidéo).

On comble ensuite le manque d'informations, par interpolation (on regarde la composante du plus proche voisin...)

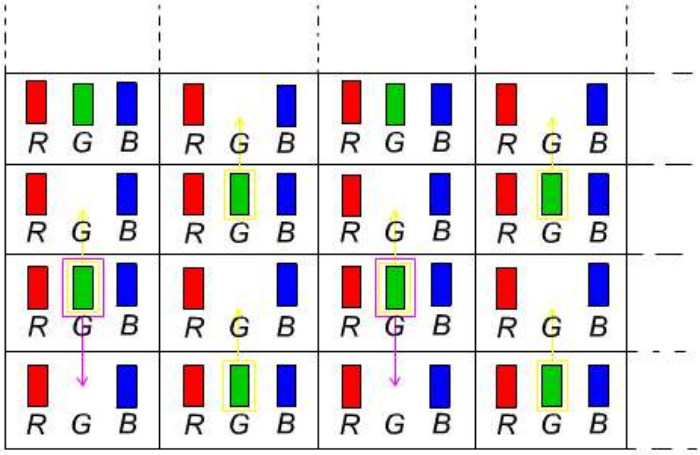
Puis les composante de bleu :



On rajoute les composantes de rouge :

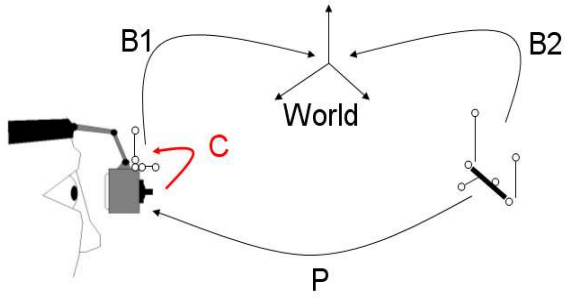


Enfin on rajoute le vert. On note que, la dernière ligne est interpolée 2 fois....



Annexe B : Calibration du casque

Coordonnées homogènes : systèmes de coordonnées (4 coordonnées en 3D, i.e. ajout de la coordonnées homogène w) facilitant les transformations géométriques dans l'espace. En effet, on peut représenter des matrice de transformations grâce à ce système (rotation, translation, changement d'échelle). Une succession de transformation, de changement de repères s'effectue alors avec des produits de ces matrices.



C = inv(P) x B2 x inv(B1)

fig.19 : Repères et matrices de changement de repères en jeu

On veut donc faire coïncider les 2 points de vue réels et virtuels. On veut alors connaître la position des cameras virtuelles par rapport a la position de la tête de l'utilisateur suivie par le périphérique à infrarouges. Les mouvements des cameras physiques seront alors appliquées au cameras virtuelles. D'un point de vue programmation, on recherche donc la matrice de position (matrice 4*4 contenant rotations et translations en coordonnées homogènes) des cameras dans le repère du casque suivi. On utilise pour cela un point intermédiaire (que nous noterons Point), de - matrice de - position B2 dans le repère commun mondes réels et virtuels, que l'on l'appellera «repère du monde » par abus de langage.

Donc, nous connaissons la position de Cam et Point dans le monde, respectivement les matrices de positions B1 et B2 de la fig.19 (repère commun au réel et au virtuel). Grâce a un logiciel de calibration, qui calcul la position d'un point dans le repère camera, on obtient la matrice P. On peut alors déduire des différentes matrices de changement de positions (matrices de changement de repère), la matrice C de position des cameras virtuelles par rapport aux données de suivi du casque.

Annexe C : Calcul de la valeur de translation du point de vue « matriciel »

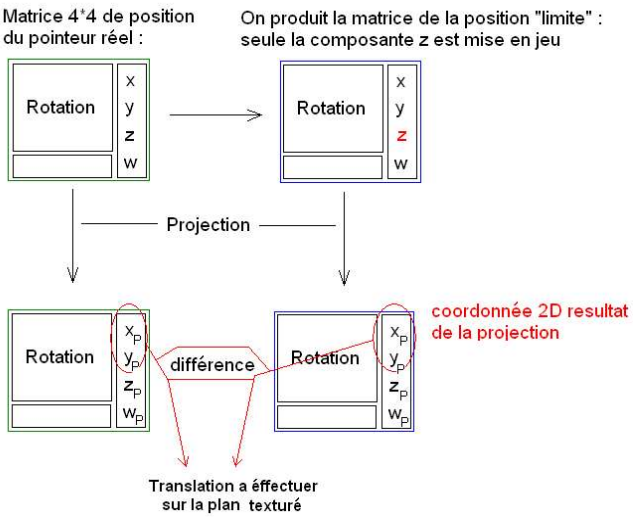


fig. 20 : calcul de la transformation 2D a partir de matrices 3D

Annexe D : Séparation de la main du reste d'une image vidéo

Principe général :
Le sujet est place devant un fond uniforme bleu. On doit donc enlever (rendre transparent, format RGBA) tout les pixels bleus de la vidéo capturée. Le problème est que la teinte du fond varie suivant l'éclairage. En effet, lorsque l'on filme un objet dont la couleur est uniforme il est extrêmement rare que tout les pixels aient tous le même code RGB. L'algorithme consiste donc a parcourir tout les pixels de l'image, et d'enlever les pixels compris dans un intervalle de teinte. La représentation utilisée est HSV (Hue=teinte,S=saturation,Value=intensité lumineuse).

Proposition d'optimisation :
L'optimisation en cours de développement consiste à ne parcourir qu'une aire plus restreinte de l'image vidéo (plutôt que chaque pixel, à chaque acquisition). Pour cela, on utilise les coordonnées des deux extrémités du pointeur. On dispose d'une extrémité et des angles de rotations, on peut donc en connaissant la taille du pointeur connaître les coordonnées de la deuxième extrémité (on utilise ici aussi les matrices de coordonnées homogènes). On projettera alors ces coordonnées sur le(s) plan(s) image -un pour chaque oeil- (même procédé que pour déterminer la valeurs de la translation à effectuer dans la partie V.2.2.Implementation pour les tests préliminaires). On supposera que, approximativement au milieu – du segment projeté - du pointeur se trouve la main. On sait alors que autour de ce point se trouve la main. On utilise également le fait que le bras coupe *au moins* un bord de l'image (bord droit pour les droitiers ou bord gauche pour les gauchers ou bords inférieur pour les 2), au plus 2 bords (bord droit pour les droitiers, bord gauche pour les gauchers ET bords inférieur pour les 2). On parcours donc la ligne -de pixels- du bas de l'image, la première colonne et la dernière colonne (de pixels représentation RGB). On peut alors trouver les premiers points « non bleus», c'est à dire appartenant au bras (on fera en sorte qu'il n'y ai pas d'autre pixels que ceux de l'image de la main, il faudra donc un interval de teintes ne laissant pas d'ambiguïtés). On « relie » les 4 points obtenus, on obtient alors une aire plus restreinte à parcourir :

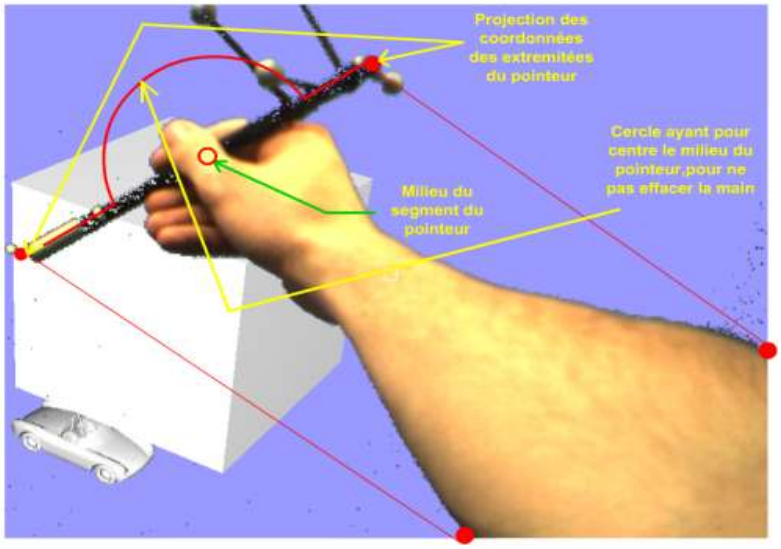


fig. 21 : optimisation

On utilisera ensuite l'algorithme de Bresenham (algorithme de base de tracé de cercle et segment) sur les indices de parcours de l'image, dans le but de ne parcourir que l'aire délimite en rouge sur la fig.