GRASSET RAPHAËL
MST ESI 2ème Année

ENVIRONNEMENT DE REALITE AUGMENTEE

Tuteur : Jean-Dominique Gascuel
Laboratoire iMAGIS/GRAVIR (Saint Martin d'Hères).

TABLE DES MATIERES

I. Introduction

La réalité virtuelle est une façon pour l'humain de visualiser, manipuler, interagir avec des données complexes à l'aide d'un ordinateur. La réalité augmentée est un domaine dérivé de la réalité virtuelle où la notion d'immersion n'est plus la même qu'en réalité virtuelle : on choisit d'avoir une vision dans le monde virtuel superposé au monde réel, et non plus une immersion complète dans le monde virtuel. On met ce principe en œuvre en utilisant des casques de réalité virtuelle transparents. Cette nouvelle voie de la recherche correspond à un manque de possibilités de la réalité virtuelle : communication extérieure impossible, manque d'information sur l'environnement réel, besoin de visualiser les 2 mondes ...

Quelques projets en réalité augmentée (RA) permettant à plusieurs utilisateurs d'interagir à travers un monde de RA commun existent à ce jour. Mais les systèmes ont des temps de réponse trop longs et l'interaction est difficile.
Les applications de la réalité augmentée sont nombreuses et variées :

- Médecine et chirurgie : assistance pour les opérations.
- Physique : aide à la visualisation et modélisation de donnée.
- Aéronautique et militaire : programmes de simulation.
- Jeux informatiques : des jeux avec une interaction plus importante.

Peu de travaux ont été effectués dans ce domaine. Pourtant l'industrie du jeu est actuellement en pleine croissance atteignant un volume d'affaire mondial de 120 milliards de francs. Le développement des jeux multi-utilisateurs correspond à un besoin d'un environnement de jeu partagé et de nouveaux types d'interactions.

Dans ce stage, on s'intéresse donc au développement d'un environnement de réalité augmentée qu'on appliquera à un concept de jeu multi-utilisateurs temps réel.
Pour la navigation dans le monde virtuel, on utilisera un casque de visualisation muni d'un capteur magnétique. Tout mouvement de la tête de l'utilisateur impliquera une modification de la vision de l'utilisateur : on pourra donc se déplacer dans le monde virtuel par déplacement de la tête. Les utilisateurs pourront modifier et interagir avec le monde virtuel à l'aide d'un périphérique externe qui sera lui aussi muni d'un capteur magnétique.
On veut de plus pouvoir avoir plusieurs utilisateurs dans le même monde virtuel. Les utilisateurs pourront soit agir indépendamment les uns des autres, soit agir en coopération. Les utilisateurs devront donc pouvoir réaliser des actions sans modifier les actions d'un autre utilisateur. Une difficulté consiste à obtenir un rendu d'image rapide pour pouvoir avoir une impression de mouvement dans le monde virtuel. De plus l'image devra être de qualité satisfaisante pour susciter une impression de réalité de la part de l'utilisateur.

Le travail consiste donc à:

- Réaliser le pilotage du capteur magnétique (« tracker ») pour avoir sa position et son orientation dans le monde réel.
- Eliminer les problèmes de bruit et de manque de précision du capteur.
- Associer le capteur au casque et à une manette pour suivre le mouvement de ces objets.
- Définir et réaliser les possibilités d'interaction de la part de l'utilisateur dans le monde virtuel.
- Gérer le multi-utilisateurs et les problèmes qu'il pose (concurrence sur des objets, objets privés...)

- Développer une application de démonstration (de type jeux) basé sur l'environnement réalisé.

L'environnement de travail sera le suivant :

- Matériel:

- La plate-forme de destination sera principalement une ONYX2 de Silicon Graphics : 4 processeurs MIPS® R10000,8 Mo de mémoire cache, 64 Mo mémoire de texture. On travaillera aussi sur une architecture 02 (processeur MIPS® R5000).
- Un casque de visualisation I-GLASSES! ™ de VIRTUAL i.O™ : écran LCD, Résolution 180.000 pixels, vision stéréoscopique, champ visuel 30°.
- Un capteur de position magnétique FLOCK OF BIRD™ d'Ascension Technology Corporation : 144 mesures/s, rayon d'action de 0.90 m, sorties en coordonnées cartésiennes ou angles d'orientation.

- Logiciel :

- Système d'exploitation IRIX 6 (UNIX de Silicon Graphics).
- compilateur C++ d'IRIX.
- librairie OPEN INVENTOR (sur couche d'OPENGL)

II. Etat de l'art

1. Historique

a. Réalité virtuelle

La réalité virtuelle (RV) a commencer au début des années 60 avec la création de la première machine virtuelle par Heilig. La réalité virtuelle consiste à plonger un utilisateur dans un monde virtuel. Les 3 points clés sont :

- L'**immersion** : l'utilisateur est coupé du monde réel, il est immergé dans le monde virtuel.
- L'**interaction** : l'utilisateur peut interagir dans ce monde virtuel, se déplacer ou saisir des objets.
- L'**imagination** : les applications en réalité virtuelle reposent surtout sur la créativité des chercheurs, et donc les domaines d'application dépendent fortement de l'imagination du chercheur.

Durant les années 80 les premiers système de RV commencent à apparaître : les recherches sont principalement effectuées sur les périphériques car la technologie de l'époque ne permet pas d'avoir des systèmes de moteur 3D et d'affichage assez puissant. Les premiers casques (« head mounted display ») et les premiers gants (« data glove ») apparaissent. Les recherches sont principalement effectuées en coopération avec l'armée américaine qui s'intéresse de près à cette technologie : elle trouve son intérêt dans les simulateurs de vol, les entraînements virtuels pour ses soldats, et pour la recherche scientifique. Au début des années 90, les premiers véritables projets se manifestent et sont principalement dans le domaine scientifique.

C'est le cas du projet GROPE dirigé par Fred Brooks, utilisant un bras manipulateur et un casque stéréoscopique pour la visualisation et la manipulation de molécules chimiques [Brooks 1990]. Le projet est complété par un retour d'effort et une meilleure précision dans la manipulation [Taylor 1993].

Un des principaux projets du début des années 90 est le projet virtual WIND TUNNEL à NASA-AMES de Steve Bryson (**figure 1**). En utilisant un BOOM (bras articulé avec casque représenté **figure 2**) et un gant virtuel comme outils d'interaction, les scientifiques sont capables de voir et d'interagir avec des images stéréoscopiques d'un champ de flux d'air autour d'un avion virtuel [Bryson 1991]. Le projet est suivi par le projet du WIND TUNNEL reparti qui divise le calcul sur un système réparti et permet à plusieurs utilisateurs de visualiser la simulation en même temps [Bryson 1993].

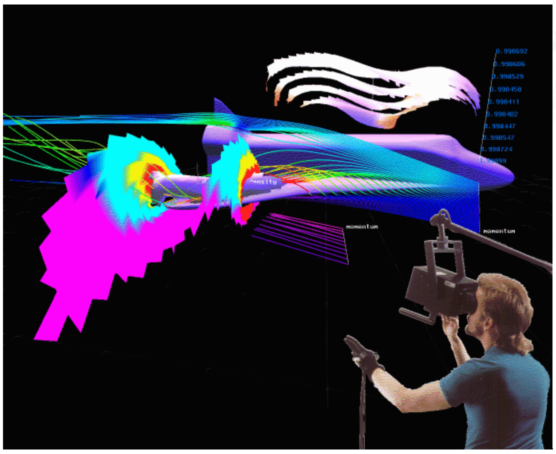


Figure 1 : projet VIRTUAL WIND TUNNEL(1991)

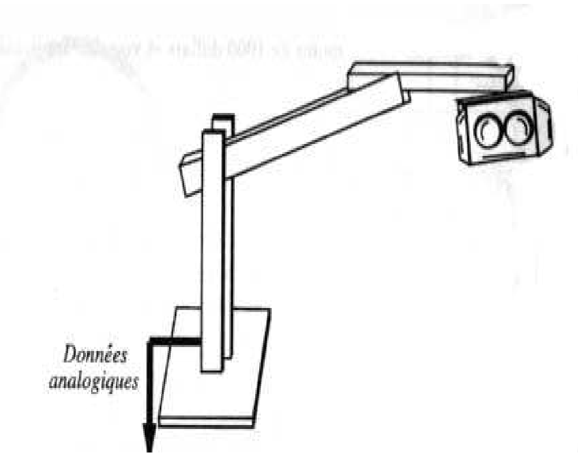


Figure 2 : un BOOM

Mais les problèmes commencent à apparaître : la communication entre les scientifiques est impossible car l'immersion totale du scientifique dans le monde virtuel le coupe totalement du monde réel, il ne peut interagir avec l'extérieur. De plus, la manipulation du BOOM nécessite un grand espace autour de l'utilisateur, qui ne peut alors pas vraiment savoir où il se déplace : on a une perte de liberté de mouvement dû à une zone morte créée autour du bras et donc l'utilisation du BOOM fait défaut. Le problème se retrouve avec les casques portables car le fait de ne pas savoir ou on se trouve dans le monde réel pose problème dans certains cas.

b. Réalité augmentée

C'est ainsi que les travaux de Sutherland ont fait surface. Ce pionner Sutherland a fait de la recherche sur les casques de RV au début des années 70. En affichant simplement quelques polygones, il construit le premier casque de RV transparent (« see through system »). L'utilisation d'un casque virtuel a travers lequel on peut voir le monde réel répond complètement au problème des scientifiques : on permet à l'utilisateur de garder le contact avec le monde réel tout en étant dans un monde virtuel. On peut aussi définir la réalité augmentée (RA) comme le fait d'augmenter le monde réel avec le monde virtuel, en ajoutant des informations complémentaires à l'environnement. D'autres chercheurs distinguent le fait d'augmenter le réel avec du virtuel du fait d'augmenter du virtuel avec du réel. ([Drascic 1996]) La réalité augmentée commence alors à apparaître.

Un des premiers systèmes de RA est réalisé par Feiner, c'est le projet KARMA (Knowledge-based Augmented Reality for Maintenance Assistance) [Feiner, MacIntyre, et Seligmann 93]. Comme démonstration, ils choisissent un système de support du personnel pour configurer et gérer une imprimante laser. Beaucoup d'efforts sont essentiels pour générer un modèle précis. Plusieurs capteurs sont attachés à l'imprimante pour détecter leur position et leur orientation. Le monde virtuel est un complément, un enrichissement du monde réel. Sur cet exemple (**figure 3** et **figure 4**) on observe ce que voit l'utilisateur à travers le casque . Dans cette application on montre à l'utilisateur comment changer le papier dans l'imprimante. Le monde virtuel possède une représentation du bac de papier et son fonctionnement.

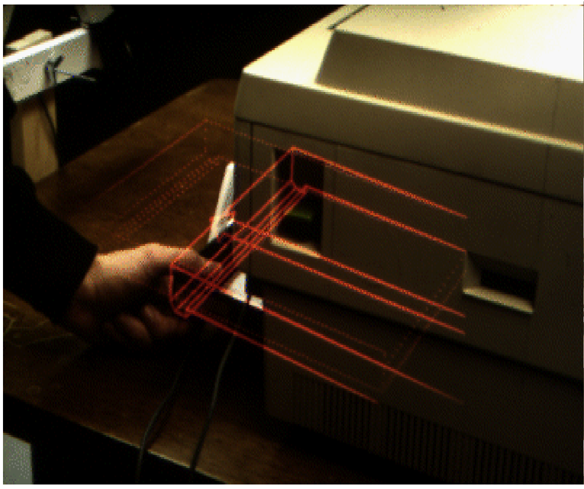


Figure 3 : vision de l'utilisateur avec le casque   Figure 4 : un utilisateur avec le système

Un autre domaine de prédilection de la réalité augmentée est la médecine. Bajura décrit un système de visualisation médicale base sur une technique de réalité augmentée [Bajura 1992]. Un casque de RA développé a l'UNC [Holm. 1992] est utilisé pour l'échographie des femmes enceintes. Ce casque autorise une superposition géométriquement correcte des données ultrasonores sur le ventre de la mère permettant au gynécologue d'examiner la position du fœtus sur la mère. Auparavant le médecin avait des difficultés à regarder l'écran de contrôle (tranche 2D) et à intégrer cette image à la position de sa main sur le ventre de la mère (volume 3D). Avec la RA, le système supprime cette difficulté, en permettant a l'utilisateur de voir les images scannées superposées sur le ventre de la mère (**figures 5 et 6**).

Figure 5 : visualisation des tranches 2D sur le ventre de la mère.   Figure 6 : représentation du système.

Un autre application de la RA dans le domaine de la médecine est réalisé par State ([State 1996]): on l'utilise pour réaliser une biopsie du sein guidée par ultrasons (**figures 7,8**).

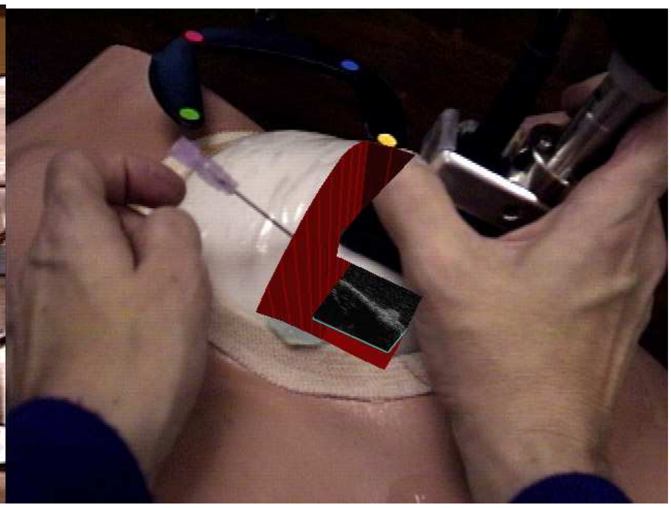


Figure 7 : Médecin avec le système de RA

Figure 8 : Représentation virtuel de l'intérieur du Sein.

c. Système de RA multi-utilisateurs

De nombreuses applications de RA sont configurées pour un seul utilisateur, et n'exploitent pas les caractéristiques multi-utilisateurs de leur système. Cependant, il existe quelques systèmes de RA multi-utilisateurs.

Dans le projet CAVE [Cruz-N. 1993] les utilisateurs voient la scène en stéréoscopie avec des casques 3D sous la forme de projection sur les murs entourant l'utilisateur. Un seul utilisateur est « tracké » et donc les images projetées correspondent à la vue de cet utilisateur. Les autres ont l'impression d'être entourés par la scène 3D. Le désavantage est que les images projetées sont celles correspondant à la position de la tête du premier utilisateur.

Le responsive workbench [Kruger 1995] utilise un espace d'affichage construit au-dessus d'une table. L'inconvénient majeur est le même que dans le projet CAVE puisqu' un seul utilisateur voient correctement les objets, les autres suivent ce qu'il voit. De plus, il est nécessaire d'être très près de la table, car l'angle de vision est très faible.

Deux autres projets (le Shared Space [Blling.1996] le Studierstube [Svalaz.1996]) représentent vraiment des applications multi-utilisateurs ou chaque utilisateur a sa propre vision. Les utilisateurs portent des casque de RA, et peuvent interagir sur des informations tridimensionnelles qu'il se partagent. Studierstube s'intéresse plus particulièrement au concept de travail partagé (« cooperative work »), et son application dans le domaine scientifique (**figure 9**). Mais l'équipe c'est intéressé au domaine du jeu en réalisant un Mah Jong multi-utilisateurs.

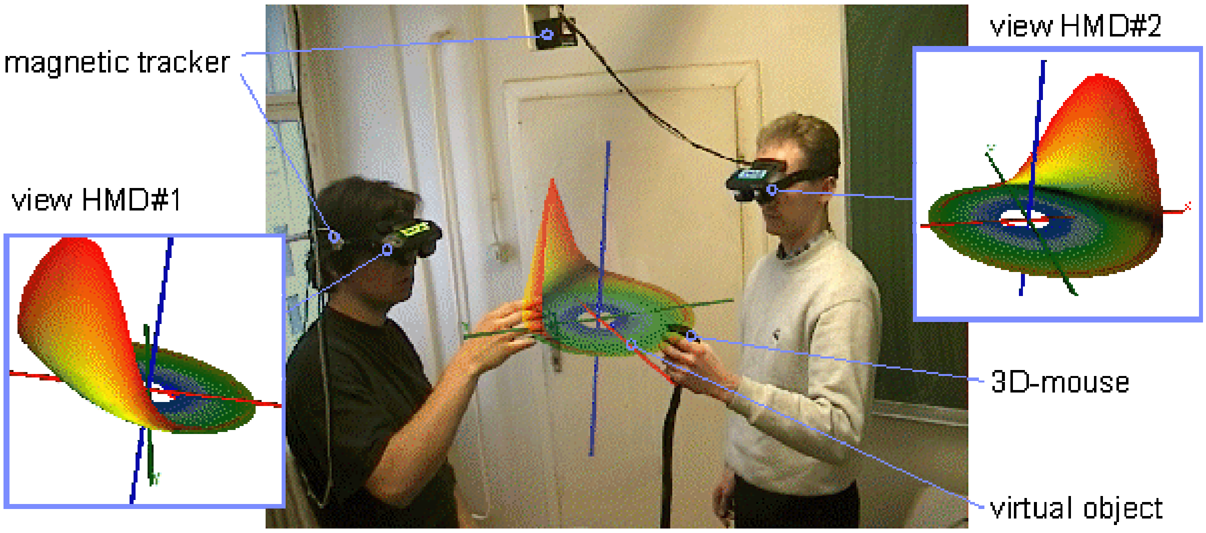


Figure 9 : Le projet STUDIERSTUBE

2. Matériels

a. Casque de visualisation

Il sert à visualiser le monde virtuel. Associé à un capteur, tout déplacement de ce casque placé sur la tête de l'utilisateur correspond à un changement de vue du monde virtuel. Actuellement, presque tous en mode stéréo, ils possèdent encore un faible champ de vision. Les modèles sont de 2 types : semi-transparent (pour la RA) ou opaque (uniquement pour la RV).

- **Casque CRT** : on utilise des tubes cathodiques miniatures. Les CRT sont placés latéralement à la tête de l'utilisateur. L'image est d'abord réfléchi par un miroir à 45° et prise en charge par une optique spéciale qui le dirige vers l'œil de l'utilisateur. Le câblage électrique est très étudié car le CRT est alimenté en hautes fréquences et est situé très près de la tête.
- **Casque LCD**: on utilise la technologie à cristaux liquides. Leur grand avantage est un faible poids. C'est la technologie retenue pour les futurs développement.

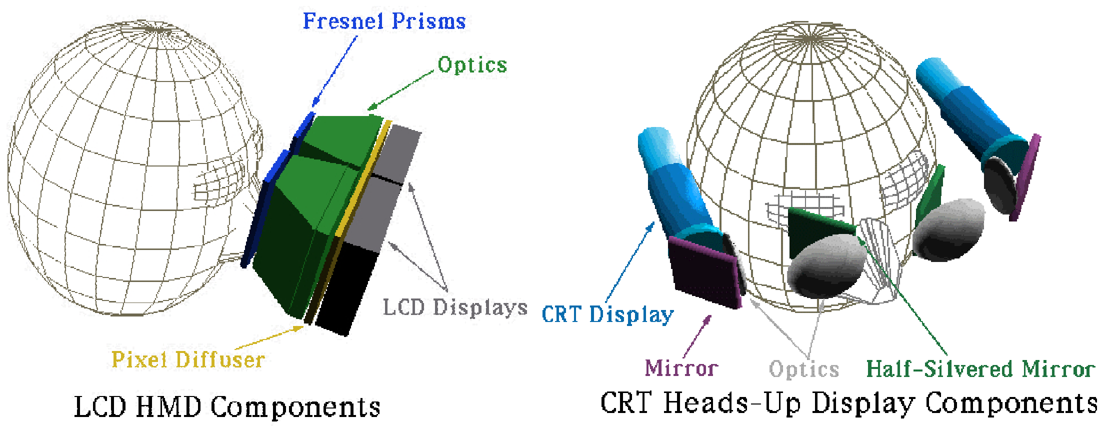


Figure 10 : représentation des 2 types de casque : LCD et CRT.

b. Capteurs de position

Ils servent à connaître la position et l'orientation des objets mobiles par rapport à leur situation d'origine. On s'en sert principalement pour mesurer le mouvement de la tête et différents objets tels que des pointeurs ou la main de l'utilisateur.
En général, on mesure une série de 6 grandeurs (3 positions et 3 angles) avec une fréquence d'échantillonnage compatible avec la précision recherchée.
Les principales caractéristiques des capteurs sont la fréquence d'échantillonnage et le temps de réponse du capteur ainsi que la précision du capteur.

- **Capteur magnétique** : un champ basse fréquence est modulé pour déterminer la position et l'orientation de l'objet mobile. Le champ basse fréquence est généré par un émetteur qui se compose de trois antennes fixes placées orthogonalement les unes par rapport aux autres. Un second ensemble d'antennes est placé à l'intérieur d'un récepteur. Le signal reçu du récepteur est échantillonné grâce à un algorithme contenu dans la machine et on arrive à déterminer la position du récepteur par rapport à l'émetteur. Le principal défaut de ce type de capteur est la grande sensibilité du capteur par rapport aux objets métalliques qui distordent le champ et rendent la précision faible à leur voisinage.
- **Capteur à ultrasons** : Il se compose de trois parties principales : un émetteur, un récepteur, et une unité électronique. L'émetteur est constitué de 3 haut-parleurs ultrasonores montés à environ 30 cm l'un de l'autre sur un bâti triangulaire fixe. De la même manière, le récepteur comporte 3 microphones montés sur un bâti triangulaire rigide plus petit. En connaissant la vitesse du son dans l'air, et par triangulation, on arrive à récupérer la position et l'orientation du récepteur. Les haut-parleurs sont actifs cycliquement et les 3 distances aux 3 micro sont calculées. Leur principal défaut est qu'ils nécessitent une ligne directe entre l'émetteur et le récepteur, car tout obstacle fait perdre le signal.

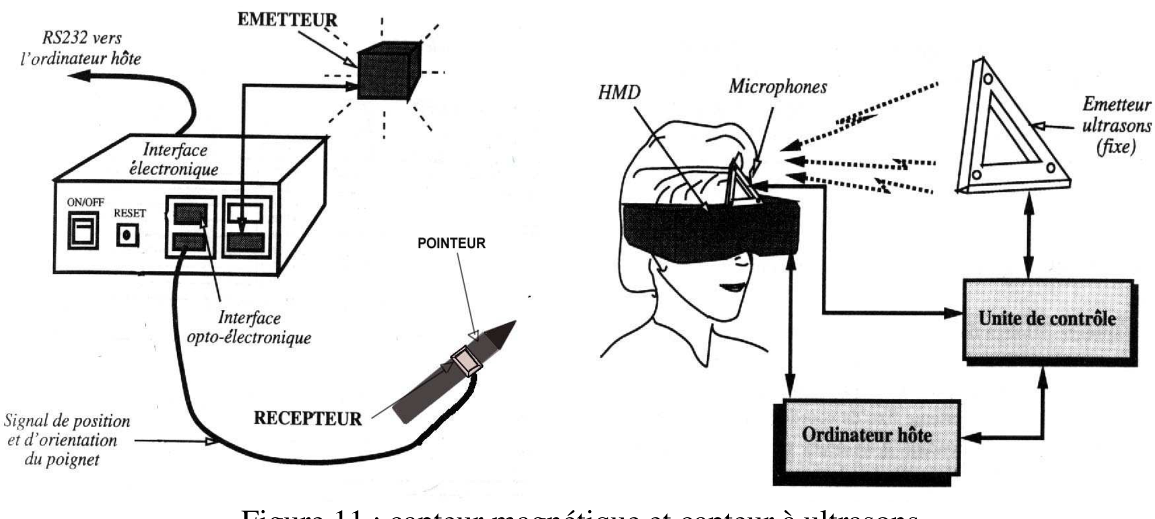


Figure 11 : capteur magnétique et capteur à ultrasons.

- **capteur camera**: Cette technique est très employé dans les films et les réalisations de jeu pour animer des mouvements de personnages 3D. Deux techniques permettent de l'utiliser.. Dans la première technique (**figure 12**), on fixe sur une personne différents marqueurs réfléchissants. On utilise une caméra munie de LED infrarouge qui éclairent la personne. On utilise plusieurs caméras qui enregistrent les mouvements de l'acteur, et a partir des images on peut extraire les marqueurs. L'ordinateur, à partir des différentes vue 2D, met en corrélation les différents capteurs pour déterminer le mouvement 3D des différents capteurs. Dans la deuxième technique, on utilise des marqueurs LED infrarouge qui sont alimentés par une batterie (les capteurs sont auto-éclairants). On utilise une caméra infrarouge qui enregistre le mouvement des marqueurs puis on utilise la même méthode de mise en corrélation que précédemment.

Figure 12 : Système avec caméra muni de LED infrarouges et marqueurs Auto-réfléchissant.

III. Flock of Birds

1. But

Nous souhaitons faire du suivi (« tracking ») d'objet réel. Nous voulons connaître les mouvements qu'effectue un objet réel au cours du temps. Pour cela on utilise un capteur magnétique qui permet de mesurer la position et l'orientation d'un objet. Ce capteur est associé à un objet réel comme la tête ou la main. La position du capteur par rapport a cet objet étant fixe (translation et rotation fixes) on peut alors suivre le mouvement de cet objet et récupérer son orientation et sa position. A partir de ces données, on associe alors l'objet réel a un objet virtuel qui « suit » alors cet objet. Nous utilisons le capteur magnétique Flock of Birds.

2. Dispositif et fonctionnalités

Le Flock of Birds (FOB) est un capteur magnétique à six degrés de liberté, commercialisé par Ascension Technology Corporation. Il est constitué de 3 parties :

- Un **émetteur** qui transmet des ondes magnétiques à temps constant. Il définit un repère de mesure pour le récepteur.
- Un **récepteur** qui capte les ondes magnétiques avec des antennes magnétiques et qui les transmet à un boîtier de commandes.
- Le **boîtier de commandes** qui à partir du temps de réponse du récepteur et en connaissant le moment d'émission de l'émetteur, calcule la position et l'orientation du récepteur par

rapport à l'émetteur. On obtient donc la position et l'orientation du récepteur dans le repère de l'émetteur. Ce boîtier constitue l'interface avec l'ordinateur par l'intermédiaire d'une liaison série RS232.

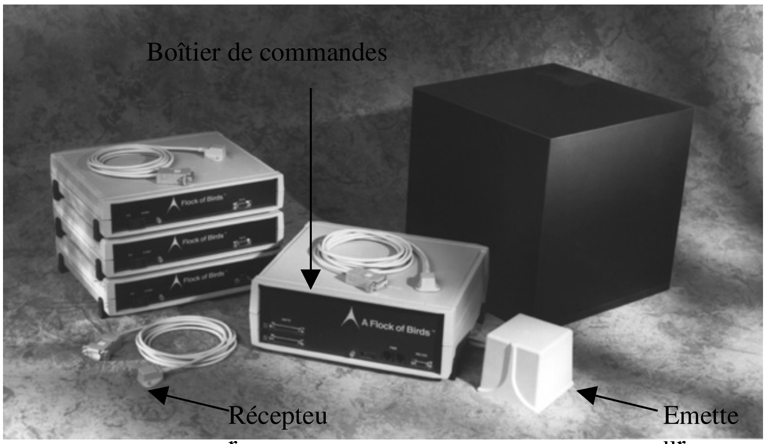


Figure 13 : Plusieurs Flock of Birds

Le FOB peut recevoir différents types de commandes :

- Commande de configuration du débit des données, gestion du contrôle.
- Commande de configuration du type de données renvoyées.
- Commande de filtrage des données.

Le FOB peut renvoyer plusieurs types de réponses :

- Réponses aux commandes.
- Données de position et d'orientations (orientations sous formes d'angles, quaternions, matrices).

Le FOB marche principalement dans 2 modes :

- mode POINT. une commande est envoyée à chaque demande de donnée. Les données sont alors renvoyées.
- mode STREAM. Une commande est envoyée une seule fois. Les données sont ensuite renvoyées en mode continu. Une lecture doit être alors assurée en mode continu du coté de la machine pour ne pas avoir une explosion du tampon hardware série de la machine.

Le FOB peut être utilisé de façon STANDALONE, c'est à dire qu'un FOB est relié à la machine. Il peut être également utilisé en mode FLOCK, ce qui signifie qu'on utilise plusieurs FOB reliés entre eux pour se synchroniser et pouvant alors renvoyer plusieurs données de positions. Ce mode peut être mis en pratique matériellement de 2 façons. On s'intéresse juste à la première façon qui consiste à avoir un FOB pour chaque port série de la machine.

3. *Mise en marche*

L'utilisation du FOB a nécessité sa mise en marche. En effet aucune utilisation et aucun test n'avaient été effectués précédemment sous système IRIX. Nous sommes donc partis de rien. En particulier, aucun câblage n'existait pour le relier à une station Silicon Graphics.

a. Réseau de communication

Une étude au préalable du fonctionnement et l'accès des ports séries sous IRIX 6.5 a été nécessaire. Le premier but était d'arriver à émettre et à recevoir une donnée à partir du FOB.

Les paramètres en prendre en compte sont le type de câble utilisé, le type de fonctionnement du port série utilisé, et le type de transmission utilisé par le FOB (Contrôle de flux HARDWARE ou SOFTWARE + mode CANONIQUE ou NON CANONIQUE).

L'utilisation d'un câble série PC classique a 9 fils ne marchant pas, il a été nécessaire de réaliser 2 câbles séries spécifiques:

- Un câble 3 fils («3 wiring cable ») spécifié dans une documentation complémentaire. (Après correction de l'erreur commise dans la documentation.)
- Un câble 9 fils spécifique SGI.

Différents essais ont permis de valider le fonctionnement du premier câble. Ce câble marche avec un contrôle de flux SOFTWARE vers la machine. On utilise la méthode par XON/XOFF. On a réalisé des essais a 9600 baud/s.

Le programme de test fournit fonctionnant très mal, nous avons donc générer nos propres programmes de tests sont générés. On utilise les fonctions d'accès au port série de la version d'IRIX 6.5 qui change de ceux utilises dans le programme de test fournit. (pour des versions<6.3).

Différentes configurations du port série au niveau du système sont effectuées : la gestion du port série est réalisé au moyen d'une structure `termios` qui contient différents attributs de configuration. On associe alors cette structure au port série.

On choisit de faire les lectures de données en mode non canonique (paramètre `VMIN=0,VTIME=20,c_lflag=0`) avec délai d'attente. Le mode non-canonique signifie que les données sont lues de façon «brute », on ne fait pas une lecture en ligne ou un caractère de fin de ligne est attendu : cette contrainte est purement matérielle et dépend du périphérique utilisé (par exemple dans le cas du Spaceball™ le mode de lecture est en ligne). Le délai correspond à un délai d'attente d'un caractère dans un tampon : si on dépasse, la fonction renvoie un code d'erreur. Cette solution est choisie car une solution bloquante d'attente de caractère bouclerait en cas d'erreur. De plus la taille des messages varient donc on s'attendrait jamais le même nombre de caractères. On veut en outre éviter de gérer les fins d'entrées/sorties avec les signaux UNIX.

Des essais sont ensuite réalisés sur les commandes et plus particulièrement sur le renvoi de données de position/orientation. Nous constatons qu'avec une taille importante de la donnée renvoyée, la fonction de lecture du port série ne nous renvoie pas tous les caractères désirés. Par exemple, la lecture de 24 octets se fera en 2 appels à la fonction `read` sur le port spécifié. Ce problème est peut être dû à une faible vitesse du mode XON/XOFF ou un problème de configuration. On résout le problème par une simple boucle de lectures successives dans le port pour arriver au nombre d'octets prévus.

Des essais sont ensuite refaits avec le câble 9 fils. On configure alors le Flock of Birds pour fonctionner en mode hardware de contrôle de flux (c'est à dire que le contrôle des données est effectué sur des câbles séparés).

Le Flock of Birds marche et la boucle de lectures successives n'est plus nécessaire : ce qui prouverait que c'est le contrôle de flux qui ralentissait la lecture.

Des essais sont ensuite réalisés à des vitesses successives pour arriver à un fonctionnement à 115200 baud/s. Le fonctionnement en mode « stream »est utilisé pour contrôler que le tampon ne se remplit pas.

On lance pendant plusieurs heures une boucle de lecture des données pour vérifier que la machine ne plante pas. On affiche à chaque pas le contenu du tampon hardware. Les résultats sont satisfaisants : la lecture est effectuée assez rapidement pour assurer que le tampon ne se remplit pas.

b. Bruit et temps de latence

- description des problèmes

Les capteurs magnétique ont comme défaut principal d'être très sensibles aux champs magnétiques environnants. De plus, du fait de la technologie utilisée, ils sont très sensibles à la moindre variation de position et d'orientation. Ceci entraîne un signal bruité qui se reporte alors sur l'application d'utilisation du FOB. Si on associe le tracker a un objet virtuel, il semble vibrer sur l'écran, il semble sauter.

Un autre point à souligner est le temps de latence qui apparaît dans l'utilisation par une application de ce type de matériel. Si on associe le déplacement du FOB a un déplacement d'un objet virtuel on constate que l'objet virtuel ne semble pas toujours suivre assez rapidement les mouvements du FOB.

Un autre facteur important pour la RA concerne l'alignement (« registration »). Par exemple si un utilisateur porte un casque virtuel HMD en déplaçant sa main réelle l'utilisateur voit sa main virtuelle sur le casque. Comme l'utilisateur voit les 2 objets, il est nécessaire que la main virtuel soit dessiné exactement à la même position que la main réelle, suivant son mouvement. Une erreur d'alignement cause un trouble pour l'utilisateur entre ce qu'il voit et ce qu'il ressent (sensation perceptive de sa main).

On constate principalement 2 types d'erreurs [Azuma,1997]:

- Les erreurs statiques produites quand l'utilisateur ne bouge pas.
- Les erreurs dynamiques causées par le tracker lui-même et par le délai du système.

Pour traiter ces problèmes, de nombreuses approches ont été proposées (dont beaucoup de systèmes utilisant une camera).

- solutions

Pour l'erreur statique, il est nécessaire de réaliser une calibration.

On utilise un objet réel que l'on va aligner avec un objet virtuel. On aura ensuite une base pour construire notre monde virtuel car on a un lien d'alignement entre 2 objets. Le réglage est effectue par des actions de base de la part de l'utilisateur.

Pour cela, on utilise une grille réelle représentée sur une feuille que l'on pose sur une table. L'utilisateur, à l'aide de son casque virtuel, va positionner une grille virtuelle sur cette grille réelle (**figure 14**).

La grille virtuelle est placée par une translation fixe par rapport à la caméra virtuel. Ainsi tout mouvement de la camera conduit au mouvement de la grille. Et donc par des mouvements de tête, l'utilisateur peut aligner la grille virtuelle avec la grille réelle. On connaît alors la position du FOB pour laquelle on a un alignement. Cette mesure devient alors un repère de référence (on a une translation qu'on affectera a tous les objets).

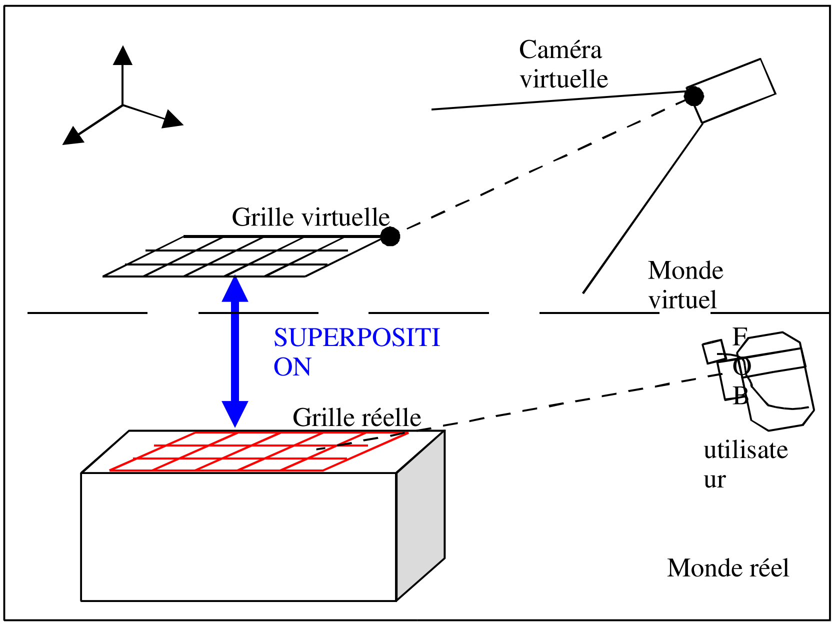
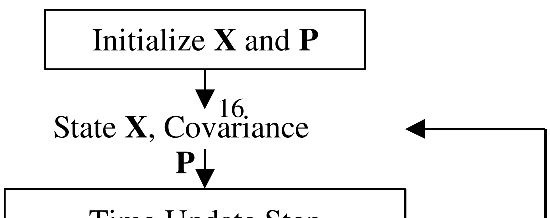


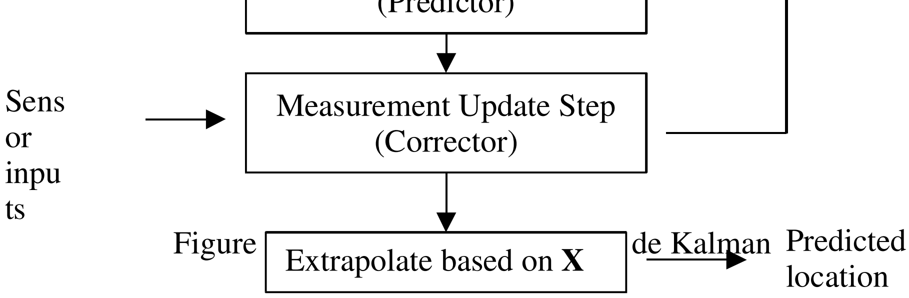
Figure 14 : Principe de la calibration

Pour les erreurs dynamiques, le principal problème est la compensation du temps de latence.
Différentes techniques sont utilisées telles que :

- La réduction du temps de latence dans le pipeline de rendu ([Olano95], [Wloka95]).
- L'image de déflexion. ([Burbidge89],[Regan94]).
- Le tracking complété a l'aide d'une camera pour corriger l'alignement ([State96]).
- L'algorithme de prédiction de mouvement.

On s'intéresse à la dernière solution qui est la plus simple du point de vue matériel et en coût. Son intérêt particulier est dû au fait qu'elle tient compte du bruit blanc que génère le type de capteur employé tel que le FOB. Résoudre ce problème est trop complexe car la difficulté de modélisation et la variation importante de la valeur du signal et le bruit rendent difficile l'utilisation de techniques de prédiction et d'estimations classiques.
Il existe différentes techniques de prédictions (ajustement à base de spline, analyse en série de temps, théorie du contrôle, filtre de Wiener) mais la plus valable et la plus efficace reste le filtre de Kalman.
Le principe est simple (**figure 15**): on utilise deux matrices X et P. la matrice X sert à définir les variables d'états : position, vitesse, accélération. La matrice P contient la variation du signal, son bruit. Le filtre nécessite 2 modèles : un modèle du système et un modèle de mesure. Le système définit comment X varie dans le temps en absence de mesure. Le modèle de mesure défini comment X va varier en fonction des mesures. Le fonctionnement est le suivant : tous le temps une nouvelle mesure est effectue. On mesure au temps t1. Le filtre déclenche un mise a jour temporel de X et P au temps t1 (c'est la prédiction). Puis une correction d'erreur est effectuée a partir des valeurs mesurés. On obtient donc de nouvelles valeurs X et P.
A partir de ces valeurs nous pouvons a tout moment extrapoler à partir de la date courante une valeur de position future.





A partir d'une classe C++ modélisant cet algorithme (réalise par Jean Dominique Gascuel) on l'intègre dans notre programme. On rajoute un filtre de Kalman pour chaque composante (x, y, z et les valeurs d'orientations) mesurée qu'on intègre a notre boucle d'écoute du port série. On met a jour les valeurs du filtre au cours de la boucle. Puis on fait une extrapolation pour un instant futur la position et l'orientation du FOB.
Ce temps doit être réglé pour chaque application, il définit le temps de latence entre la mesure et la lecture par l'application de la valeur, et l'affichage.

On règle aussi les filtres hardware présents dans le FOB pour éliminer le bruit tout en gardant un temps de traitement faible. On combine ces 2 filtres pour un meilleur résultat.
Des mesures sont ensuite réalisées pour évaluer les résultats (**figures 16 et 17**).

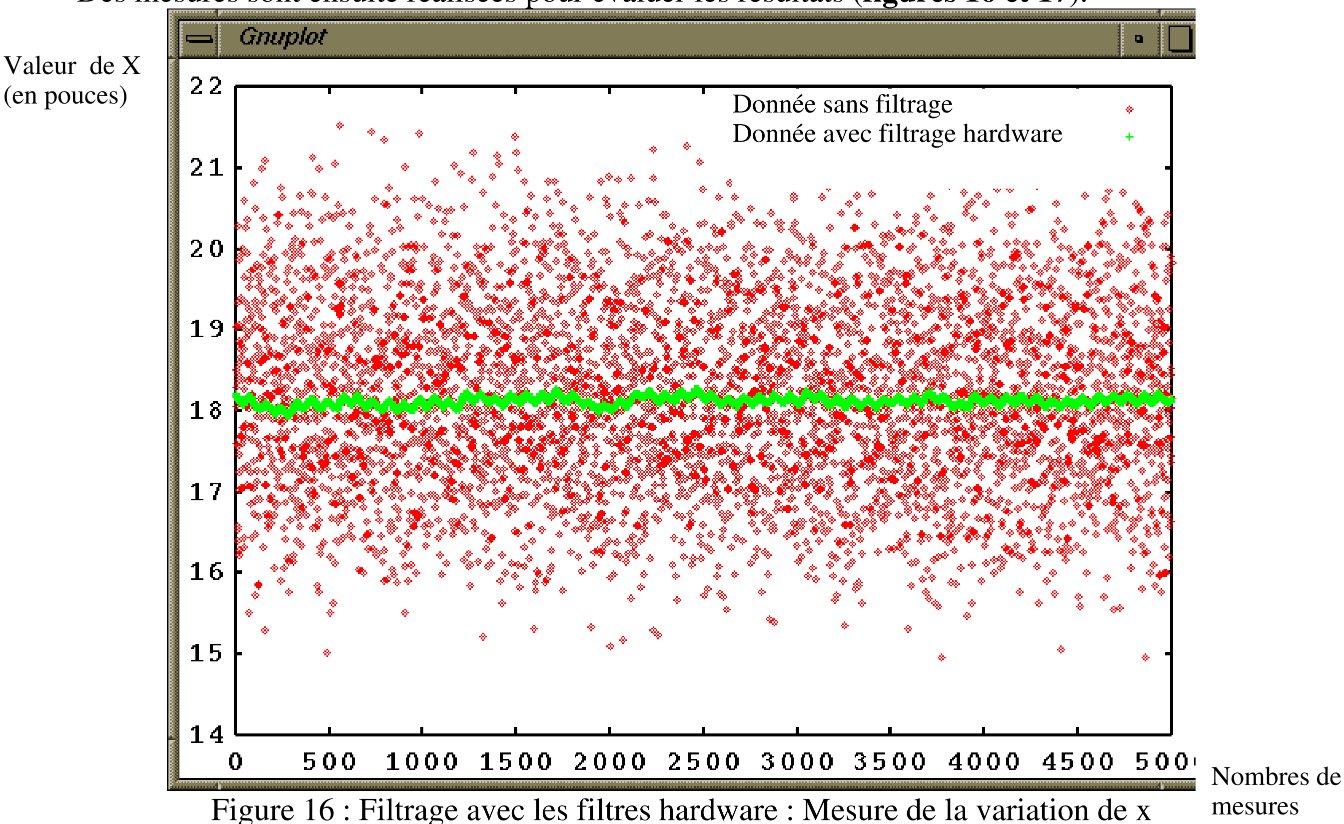


Figure 16 : Filtrage avec les filtres hardware : Mesure de la variation de x au cours du temps.

Valeur de X
(en pouces)

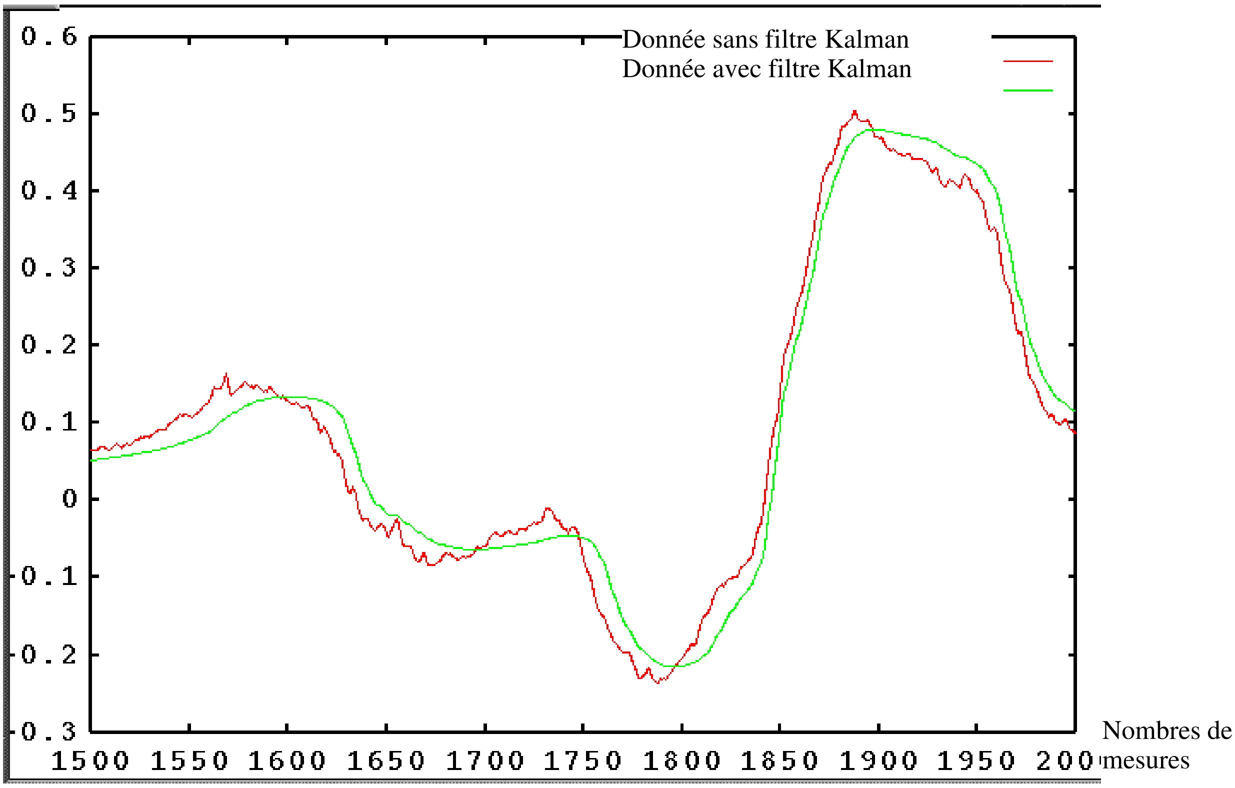


Figure 17 : Filtrage avec le filtre de Kalman + filtre hardware :
Mesure de la variation de x au cours du temps.

Les résultats sont satisfaisants : le bruit est éliminé, et le filtre ne donne pas trop de déviation pour la valeur mesurée.

4. Gestion du FOB

On s'intéresse maintenant à gérer le FOB de façon transparente pour l'utilisateur. La construction d'une librairie, une sorte de pilote, pour commander le Flock of Birds est mise en œuvre.

a. Librairie

Une librairie accompagnant le FOB était surtout fournie pour faire une démonstration du Flock of Birds: trop générique, trop divisé dans de nombreux fichiers, appel de fonction trop lourd en paramètre, très mal utilisable.
Une recherche sur l'Internet nous a conduis à trouve 2 librairies intéressantes :

- Une très simple avec des bons exemples mais n'implémentant que quelques commandes du Flock of Birds. (Stephen Junkins,Clemson University Computer Science Department)
- Une librairie programmée en C++ très souple et très simple mais n'implémentant malheureusement pas toutes les commandes. (Mark Dans, University of Carolina).

Ces librairies étant trop simple et pas assez complètes, nous avons choisi de réaliser notre propre librairie de gestion du Flock of Birds en C++ permettant de considérer le FOB comme un objet facilement utilisable.

Nous nous fixons les contraintes suivantes :

- La librairie doit être en couches afin de permettre au développeur ou à l'utilisateur de reprogrammer les couches supérieures.
- Le fonctionnement en mode STREAM/POINT du FOB.
- L'implémentation de TOUTES les commandes du FOB.
- Pour le fonctionnement en mode continu (STREAM), la possibilité de récupérer à n'importe quel moment les informations de position et/ou orientation.
- utilisable en mode FLOCK avec un port série par FOB.
- LA librairie souple sous forme d'objet.
- La librairie doit être haut niveau masquant la configuration.

Une étude et un découpage en couches sont alors réalisés. Une hiérarchie de classes est alors modélisée. Nous avons défini les couches et classes suivantes :

Couche de gestion haut niveau du FOB. (classe `FobAlone`)
Couche de bas niveau pour le pilotage du Flock of Birds. (classe `FobCommand`)
Couche de gestion du Port Série (classe `RS232Interface`)

Figure 18 : Représentation en couches de la librairie.

- La classe `RS232Interface` gère la communication avec le port série RS232. Elle permet l'initialisation de la lecture et l'écriture sur le port.

- La classe `FobCommand` implémente toute la gestion des commandes avec le FOB. On divise la classe en différentes parties :
  - commandes de gestion de l'émission : démarrage, arrêt, caractère de contrôle.
  - commandes de configuration et gestion des données.
  - commandes de configuration matériel du FOB : vitesse de mesure, filtrage, configuration automatique.

  Cette classe est de bas niveau et permet facilement à un développeur de contrôler toutes les commandes nécessaires à l'initialisation et à la récupération des données de position et d'orientation.

- La classe `FobAlone` est une classe de haut niveau permettant l'intégration facile du Fob au sein d'une application graphique.
  La classe est divisée principalement en 3 groupes :
  - commandes de gestion des données renvoyées : type de repères, choix du repère initial etc.
  - commandes de gestion du mode POINT : récupération de la position et de l'orientation sous forme de matrice, angle, quaternion.
  - commandes de gestion du mode STREAM.

  Cette classe évite a l'utilisateur les problèmes de gestion du FOB.

b. Difficultés

- La gestion du mode STREAM

La gestion et la lecture de données en mode continu est un des principaux problèmes. On ne peut bloquer une application pour effectuer des lectures en continu dans le tampon série. On souhaite de plus une indépendance de l'application et de la gestion du FOB. Mais il faut avoir une lecture en continu du port série pour ne pas avoir un dépassement de capacité.
La solution retenue est donc une solution sous forme de processus léger :
- un processus léger tourne en continu et lit les données arrivant sur le port série.
- une méthode synchronisée sur une donnée partagée avec le processus léger permet de récupérer a tout instant la position et l'orientation. La synchronisation est réalisée par accès exclusif en utilisant un sémaphore (**Figure 19**).
On utilise une implémentation respectant la norme POSIX 1003.b.

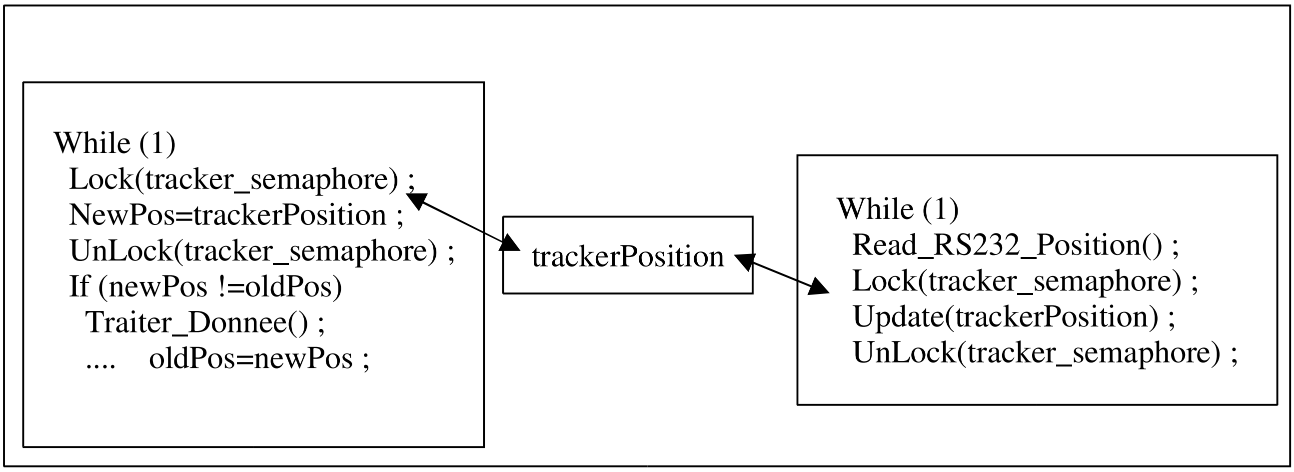


Figure 19 : synchronisation entre l'application et le processus léger d'écoute.

- Gestion du signe

Un autre problème est dû à la mauvaise gestion du signe des valeurs renvoyés par le FOB. Le FOB renvoie des données de position sur les 3 axes. La valeur du champ magnétique transmis par le FOB est symétrique par rapport aux 3 axes de l'émetteur. la symétrie pose un problème pour déterminer la position du récepteur. L'amplitude est toujours bonne mais le signe peut être faux, suivant l'hémisphère dans lequel le FOB se positionne. Matériellement, le capteur peut gérer le signe valide dans un hémisphère : haut, bas, gauche, droit, devant, derrière, suivant la position de l'émetteur. Dans notre cas, on voudrait gérer l'espace entièrement autour du FOB.

On rajoute donc à notre processus léger de lecture en continu une détection de changement d'hémisphère. La technique consiste simplement à détecter que le tracker a franchi simultanément 2 axes sans avoir franchi le troisième dans l'hémisphère correspondant. Par exemple, en mode « haut » la position renvoyé a toujours son x positif, et y et z changent de signe dans l'hémisphère devant l'émetteur. On détecte alors si on traverse l'axe y et z simultanément sans qu'on traverse l'axe X dans l'hémisphère de derrière.
Le seul cas de non fonctionnement de cet algorithme apparaît lorsque l'on traverse le troisième axe dans l'hémisphère courant. Il suffit alors de vérifier si on a y=z=0 pour gérer ce cas.

IV. Manette

1. Problème

Si on utilise le FOB pour tracker une main, on ressent tout de suite le manque de possibilité avec l'utilisation seule du tracker, pour le cas de la souris, ce sont les boutons.. On ne peut diriger et réaliser des actions qu'a l'aide d'un périphérique externe tel que le clavier. Pour saisir des objets, les relâcher, pour ouvrir un menu, ou tout autre action il est nécessaire d'avoir un outil de validation comme les boutons pour une souris.

2. Réalisation

a. Création de la manette

Devant ce problème, on a décidé de réaliser une manette comprenant de boutons pour effectuer diverses actions. Apres des recherches sur le site officiel d'Ascension et leur aide, ils nous ont fait parvenir un plan de montage d'interrupteur sur le FOB. Le FOB doit gérer ces boutons et nous renvoyer des informations sur l'état de ces boutons. En utilisant de la pâte a bois nous avons construit une manette (**Figure 20**). Le montage électronique réalisé par le responsable électronique de l'IMAG est intégré a la manette. On fixe ensuite le FOB sur la manette de jeu. Le FOB servira donc à suivre les mouvements de la manette et les interrupteurs sont employés comme boutons d'actions. On constate quelques interférences magnétique dues à la conductivité de la pâte à bois.

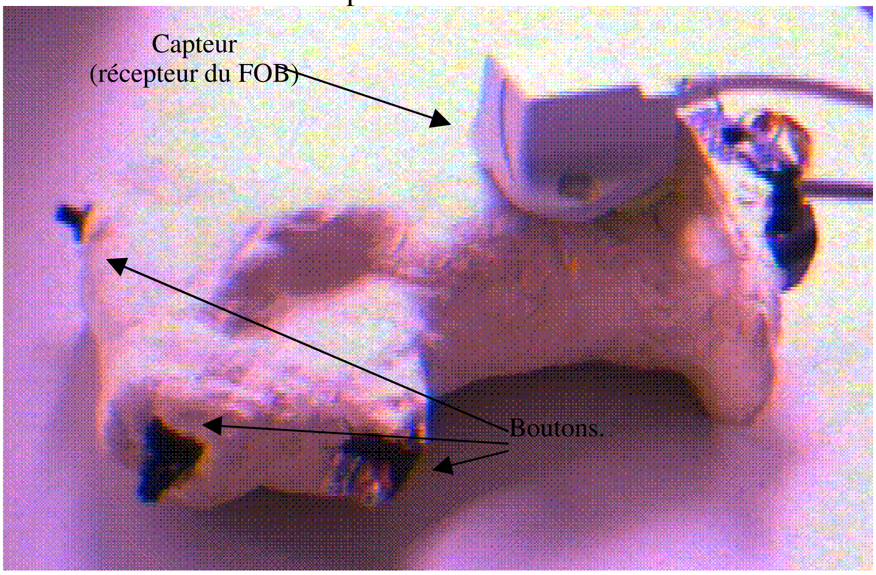


Figure 20 : représentation de la manette

b. Gestion logiciel des boutons

On modifie notre libraire en rajoutant la classe FobAloneB.

Couche de gestion haut niveau du FOB. (classe `FobAlone`)	FOB haut niveau du FOB avec boutons. (classe `FobAloneB`)
Couche de bas niveau pour le pilotage du Flock of Birds. (classe `FobCommand`)	
Couche de gestion du Port Série (classe `RS232Interface`)	

Figure 21: Librairie finale

La classe `FobAloneB` est une classe très semblable a `FobAlone` mais elle implémente en plus la gestion de boutons avec le FOB. Cette classe ne dérive pas de sa classe sœur car les changements sur les méthodes sont beaucoup trop important et l'héritage n'apporte rien.

c. Application de démonstration

On utilise maintenant le FOB pour réaliser du tracking. Le FOB est associé à la manette. La position du capteur par rapport à cet objet étant fixe (translation et rotation fixes) on peut suivre le mouvement de la manette est récupérer son orientation et sa position. A partir de ces données, on associe alors la manette à un objet virtuel qui « suit » alors cet objet. L'objet virtuel permet ainsi d'interagir avec le monde virtuel.

Au niveau graphique, la position et l'orientation du FOB nous donnent une translation et une rotation pour l'objet virtuel. La rotation est sous forme de quaternion qui est la représentation la plus stable. En combinant cette position avec la position du tracker par rapport à l'objet réel on obtient la position de l'objet virtuel associé à l'objet réel (**figure 22**).

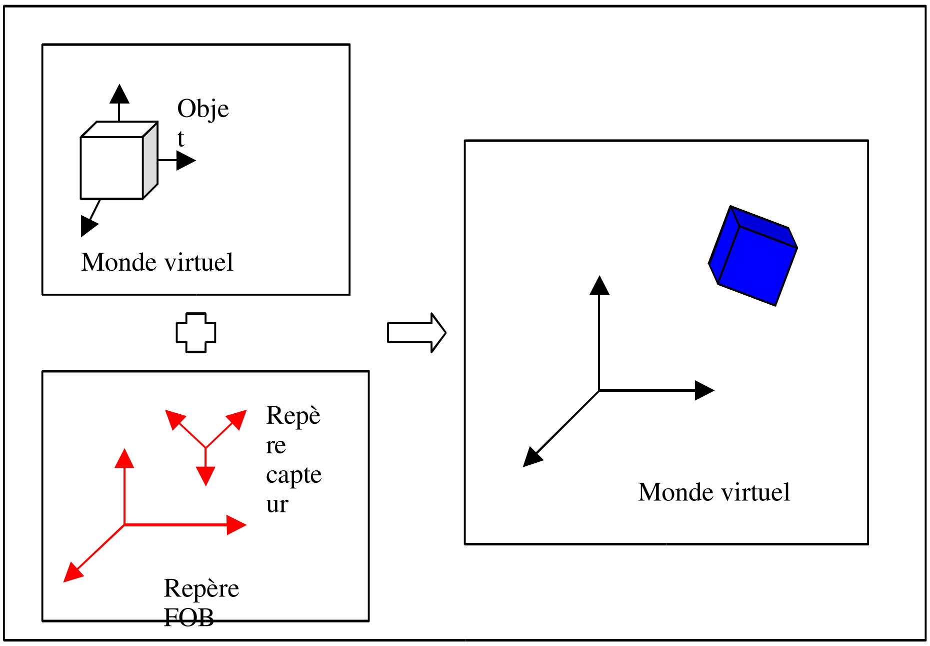


Figure 22 : Principe du tracking. Dans cet exemple, on associe la position du capteur à un cube.

La classe `FobAlone` et `FobAloneB` gèrent le changement de repère émetteur a tout autre repère. Elles gèrent la translation entre les 2 repères et le changement d'échelle. On peut donc l'utiliser pour un premier changement indépendant de l'application. D'autres changements sont alors nécessaires suivant l'application.

On réalise les tests sous les librairies Open Inventor et OpenGL.

- Tracking d'objet :

On associe la manette a un cube virtuel. Tout déplacement de la manette (et donc du tracker) est alors reporté sur le cube : on a donc l'impression de manipuler le cube 3D.

- Saisie d'objet virtuel :

On associe ensuite la manette à un stylo virtuel. Le déplacement de stylo virtuel permet de placer la pointe du stylo à l'intérieur de divers objets. Après l'appui sur un bouton, on saisit l'objet sur lequel est pointé le stylo. On fixe alors la position de l'objet saisie fixe par rapport à celle du stylo : la relation entre le stylo et l'objet devient invariante. Si on déplace le manette, et donc le stylo virtuel, l'objet saisi est déplacé. On a donc la sensation de bouger l'objet sélectionné. Le fait de lâcher le bouton dissocie l'objet virtuel de la manette et il reste à sa dernière position courante..

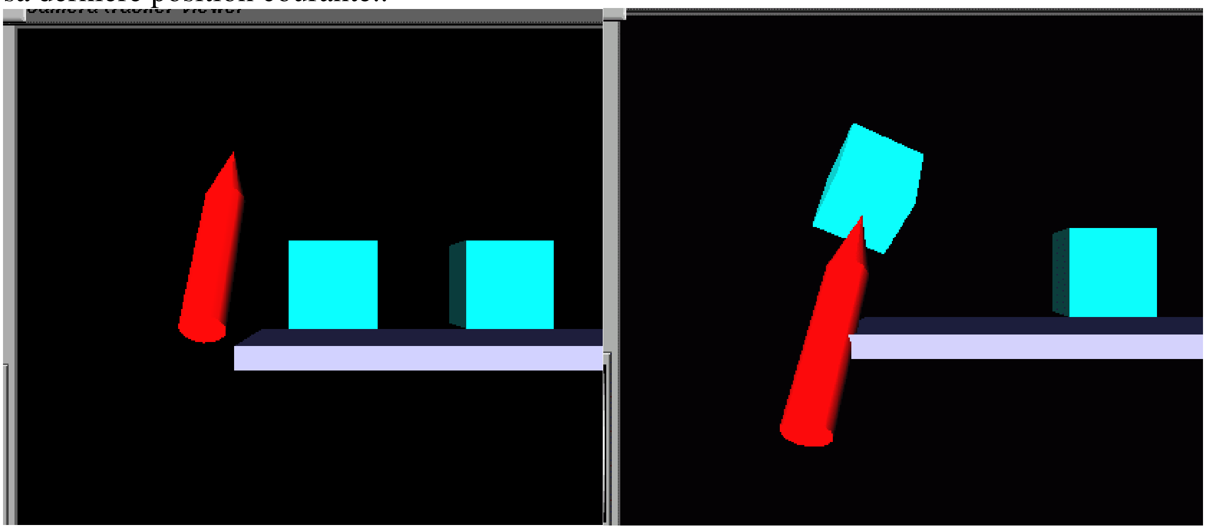


Figure 23 : Saisie d'objet virtuel

- Création d'objet virtuel :

Une application de création d'objets permettant de tracer des sphères et des cylindres. On associe la manette à un pointeur virtuel représentant un pinceau virtuel. l'appui sur un premier bouton permet de tracer des sphères virtuels. Le seconde bouton permet de représenter des sortes de tubes virtuels qu'on arrête de tracer en lâchant le bouton de la manette.

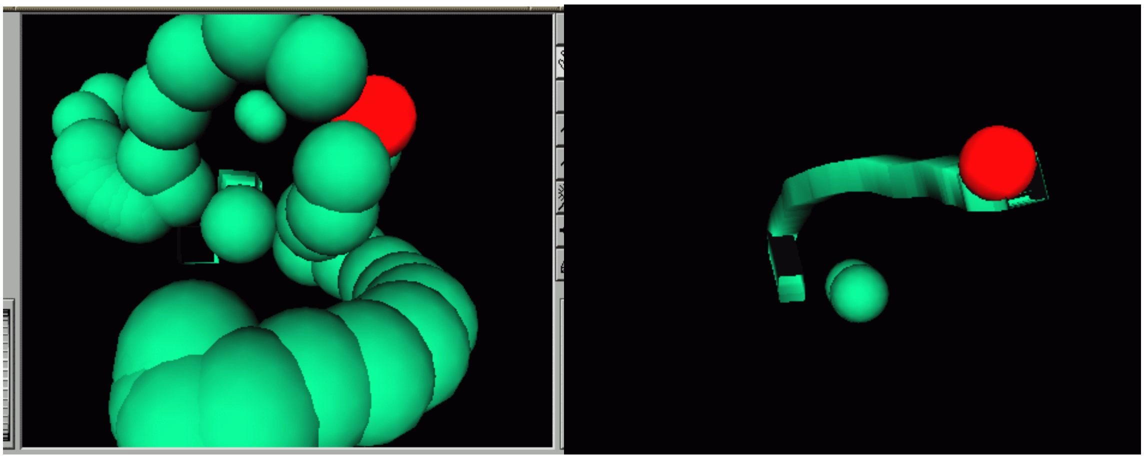


Figure 24 : création d'objet virtuel. la sphère foncé (rouge) représente le pointeur virtuel.

3. Difficulté visuelle

Dans l'application de saisie d'objet, l'utilisateur ne sait pas visuellement quand il a la possibilité de saisir l'objet : on pointe mal avec le stylo, manque de précision de la part de l'utilisateur, mouvement brusque quand on valide la saisie etc. On décide d'apporter un complément visuel qui dit à l'utilisateur quand il à la possibilité de saisir l'objet : cela correspond au fait de sélectionner cet objet. Il est à noter que la sélection n'est valable que quand on se trouve a l'intérieur de l'objet : c'est un peu le même principe que de déplacer un pointeur dans un menu.

Différentes solutions sont testées sur le même principe : on a un changement d'aspect des éléments :

- la première solution est de faire changer d'aspect l'objet dans lequel se trouve le pointeur. Par exemple une voiture devient uniformément blanche quand on la sélectionne.
- Une autre solution est de faire changer de forme le pointeur (le pointeur correspond au stylo). Quand le pointeur passe sur un objet, il change d'une forme sphérique à une forme cubique.
- On peut faire changer d'aspect le pointeur : changement de couleur ou le pointeur devient transparent.

La première solution répond bien à nos exigences. Apres la sélection, il faut remettre l'objet dans sa couleur initiale (sinon, on perd beaucoup trop d'information lorsqu'on bouge l'objet de couleur unique).
La deuxième solution n'est pas très visible et est très fatiguante visuellement.
La troisième solution semble être la meilleure. En utilisant un système de couleur on peut alors associe plusieurs types d'actions suivant la couleur. Cette solution combinée à la deuxième peut être aussi un bon compromis.
On choisi donc cet solution (**Figure 25**).

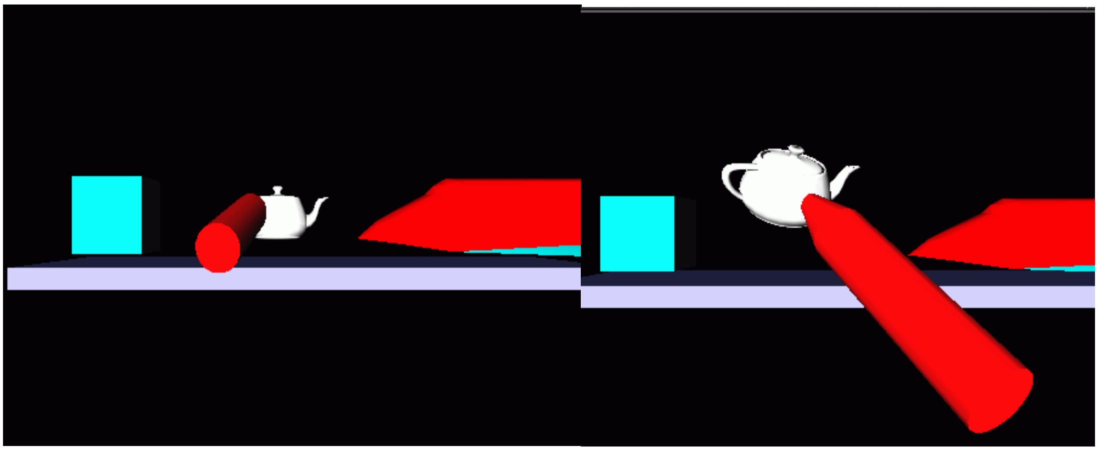


Figure 25 : Application de saisie d'objet utilisant ce principe

Au niveau de l'application, on détecte la présence du pointeur à l'intérieur de l'objet par l'utilisation de sphère englobante. Chaque objet est représenté sous forme d'une sphère où le centre correspond au centre géométrique de l'objet. On détecte alors si le tracker est à l'intérieur de la sphère par simple vérification de l'équation de la sphère.
On modifie alors notre scène en changeant le pointeur. Sous Open Inventor on utilise la classe `SoSwitch`. Elle permet d'avoir le choix d'un nœud fils parmi plusieurs, chaque nœud correspond alors à une couleur spécifique d'actions, comme la sélection.

V. Casque de visualisation

On souhaite visualiser le monde virtuel avec un casque de RV en vision stéréoscopique pour permettre à l'utilisateur une meilleur immersion dans l'espace virtuel. On veut de plus tracker la tête de l'utilisateur pour permettre à celui-ci de naviguer dans le monde virtuel par simple mouvement de tête.

1. Fonctionnement du mode stéréoscopique

Actuellement les casques de RV (et RA) permettent un affichage stéréoscopiques. La perception stéréoscopique est due à la superposition que fait le cerveau entre les 2 images provenant de chaque œil. La superposition est possible du fait qu'il existe une différence de position des deux yeux permettant une focalisation sur un objet. Les casques simulent ce principe à l'aide de lentilles en créant une image virtuelle a une distance fixe.

On génère donc deux images virtuelles (une pour chaque œil) qu'on transmet au casque pour recréer une vision tridimensionnelle. Les 2 images virtuelles sont produites à l'aide de 2 caméras virtuelles séparés d'une distance fixe correspondant à la distance entre les 2 yeux et convergeant en un point. On dispose d'un casque I-GLASSES (**figure 26**) qui permet de générer de la stéréo et les stations de travail Silicon Graphics peuvent gérer automatiquement le calcul d'images stéréo.

Figure 26 : le casque I-GLASSES

Pour lui faire parvenir les 2 images au casque , il faut les envoyer en une seule image sous un format PAL. On place donc dans le premier champ l'image pour l'œil gauche, puis dans le deuxième champ l'image pour l'œil droit, et on transmet l'image résultante. Les lunettes sont chargées de répartir les champ correspondant a chaque œil. (**figure 27**).

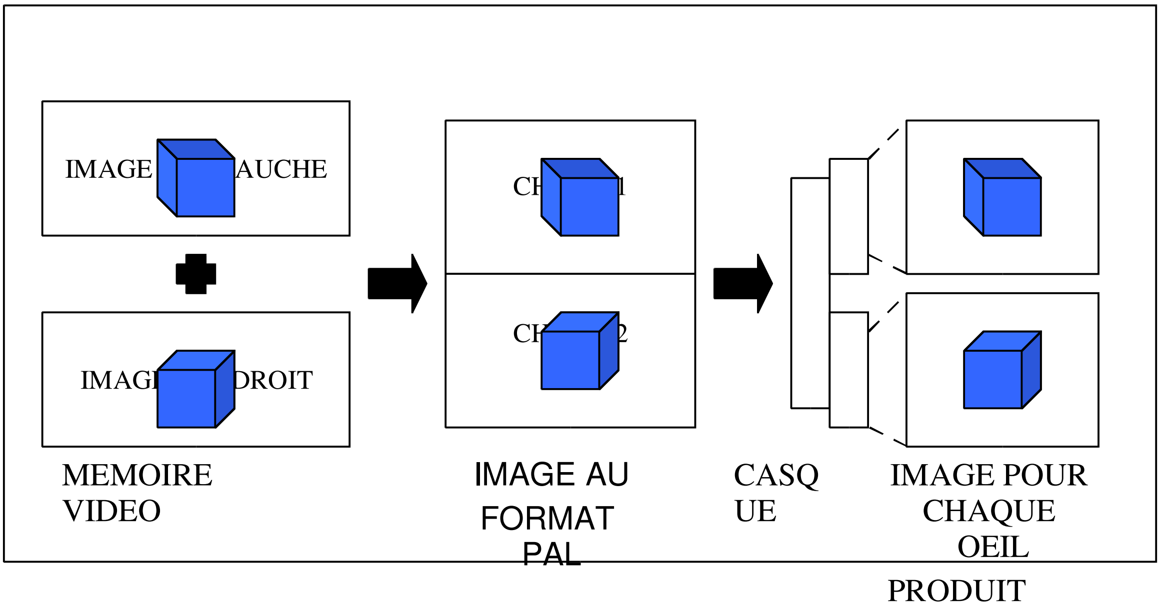


Figure 27: Génération d'images stéréoscopiques pour les lunettes I-GLASSES

La mise en place des lunettes n'a pu être réalisée en raison de a certaines difficultés techniques (création du fichier PAL stéréo impossible, aucune documentation pour l'interfaçage sous Silicon Graphics, etc.)

2. Application avec le FOB

On souhaite faire du suivi de tête (« head tracking »). On fixe donc le tracker sur le casque virtuel. Dans ce cas, la tête de l'utilisateur est associé avec une caméra virtuelle. Tout mouvement de la tête correspond à un mouvement de la camera virtuelle qui nous permet de régénérer l'image visible dans le casque virtuel. L'utilisateur a alors l'impression dans la scène virtuelle et de pouvoir se déplacer dedans.
Pour l'application de démonstration de head tracking on rajoute un filtre au niveau de l'application : on compare les 2 dernières valeurs mesurées et si le changement est faible, on ne réaffiche pas la scène. On réalise ceci pour le head tracking car les mouvements de la tête sont moins rapides que ceux pour une main, et que les mouvements de faibles intensité sont peu nombreux.

L'application permet de charger une scène de type Inventor et naviguer à l'intérieur en bougeant la tête. Comme le fonctionnement du casque n'a pu être mis en œuvre, l'application donne une sortie d'image sur l'écran.

VI. Application : Jeu de rôles

1. Description

L'application de démonstration permet de regrouper les différents points étudiés précédemment. On introduit la possibilité de plusieurs interactions.

Pour le concept de jeu en réalité augmentée nous nous sommes inspirés du jeu de Mah Jong réalisé au sein de l'équipe Studierstube qui a retenue notre attention. Il introduit le concept de sécurité et du domaine privé, « face-snapping », partage d'information. Mais l'intérêt du jeu en RA ne se trouve que dans le fait d'avoir une vision privée de ces pions.

On s'intéressera pour notre part à développer un concept de jeu de rôles multi-utilisateurs. Le jeu de rôles est établi par des concepts de communication entre joueurs et de partage d'informations. Son principal défaut est que le maître de jeu a des difficultés à visualiser les mouvements et la position des personnages, et l'utilisation d'une feuille comme support n'est pas très pratique. Le jeu de rôles sur ordinateur amène un aspect visuel (paysage, effets de magie) au jeu mais plonge le joueur dans la gestion de plusieurs personnages en même temps. Les solutions de jeux en réseau font malheureusement perdre la communication orale directe entre les participants et l'aspect social que possède un vrai jeu de rôles.

Un jeu de rôles en RA permet de tirer des avantages de ces 2 possibilités :

- En utilisant des casques de RA on garde le contact avec les autres utilisateurs et l'aspect social est conservé. Chaque utilisateur porte un casque et dispose d'une manette de jeu. Tous les utilisateurs se placent alors tous autour d'une table.
- On apporte l'aspect visuel qui est non négligeable et une vision stéréoscopique tridimensionnelle temps réel. Et contrairement à beaucoup de jeux, on peut déplacer la caméra (qui correspond à notre tête) à l'intérieur du monde virtuel.

Nous soulignons le fait qu'on développe un concept et non entièrement le jeu (travail futur).

2. Réalisation

a. Monde virtuel

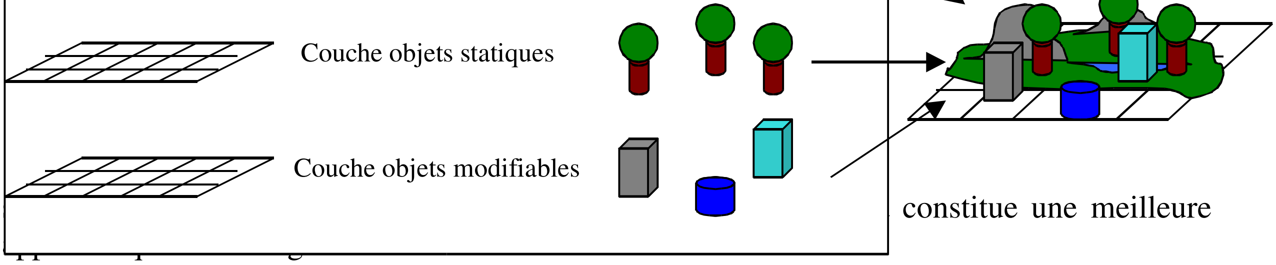
La première phase concerne à la modélisation du monde virtuel. On utilise un principe bien connu des concepteur de jeu de rôles vidéo : un système en couches.

On définit alors 3 couches :

- Une couche représentant le terrain : herbe, montagne, eau.
- Une couche représentant les objets statiques (ne pouvant être saisis ou déplacés) : arbres, pierres.
- Une couche représentant les objets saisissables et modifiables : trésor, armes, monstres, personnages.

Figure 28: découpage en couches et résultat de la superposition

On définit le monde virtuel comme une matrice de cases, chaque case représentant une surface virtuelle. On a donc en chaque case une superposition des 3 couches. Cette



constitue une meilleure

Chaque couche est définie dans un fichier et chargée au démarrage du programme : on peut donc générer des mondes virtuels de façon infinie.

L'implémentation est faite sous forme d'une classe pour chaque couche. Une classe supplémentaire `MapFile` est utilisée par les 3 couches, elle permet d'extraire les informations des fichiers correspondant à chaque couche (on utilise la même structure de fichier, **figure 29**, pour représenter chaque couche).

```
Nombres de lignes  (int)
Nombres de colonnes (int)
Taille géométrique des cases  (int)
Type Valeur Type Valeur .....Type Valeur (Type char  Valeur char)
Type Valeur Type Valeur .....Type Valeur
....
Type Valeur Type Valeur .....Type Valeur
```

Figure 29: Structure du fichier pour une couche

`Type` définissant le type de la case et `valeur` correspondant aux différentes variantes du type.
Exemples de `Type` : Piège ('P'), Arbres ('A'), Personnage ('M'), Herbe ('H'), Rien ('N'), etc.
Chaque classe gère les actions possibles dans sa couche. La classe s'occupe aussi de gérer la modélisation 3D des objets représentant sa couche. La couche est représentée sous forme de matrice, l'accès au contenu d'une case est alors facilité.

Pour le sol (couche de base) `type` définit la hauteur en cette case. Le sol est modélisé sous Open Inventor en utilisant un `QuadMesh` qui représentent une matrice de points 3D. En chaque point, on définit une couleur (celui du Type), et une surface est donc produite en interpolant les couleurs entre les points de référence.
Pour la couche objet, on peut ajouter ou supprimer des objets. Au niveau de la modélisation 3D, on supprime ou on ajoute des objets dans le graphe de scène d'Open Inventor.

b. Interaction utilisateur

L'interaction de l'utilisateur est définie par l'intermédiaire du casque et de la manette de jeu.

Le casque associé à un capteur permet de faire du « head tracking ». On définit une position du base du pion du joueur et tout mouvement de la tête permet de déplacer la caméra associé à ce joueur pour changer de point de vue dans la scène 3D.

La manette de jeu (suivi par un capteur) permet de déplacer un pointeur virtuel dans la scène. Ce pointeur est représente sous forme de sphère. Tout mouvement de la manette implique un mouvement du pointeur. Une action peut être effectuée à l'aide des boutons de la manette.
Un premier bouton sert de bouton d'action principale. On utilise un système d'action par défaut suivant l'objet sur lequel se trouve le pointeur virtuel. Le type d'action réalisable est mis en valeur par un changement de la couleur du pointeur (comme expliqué dans II.2.c) :

- Pointeur blanc : déplacer.

- Pointeur noir : aucune action possible (par exemple quand le pointeur est sous la carte).
- Pointeur rouge : attaquer (pour les monstres).
- Pointeur bleu : discuter (pour personnages passifs).
- Pointeur vert : prendre (pour les objets).

L'appui sur le bouton valide l'action.

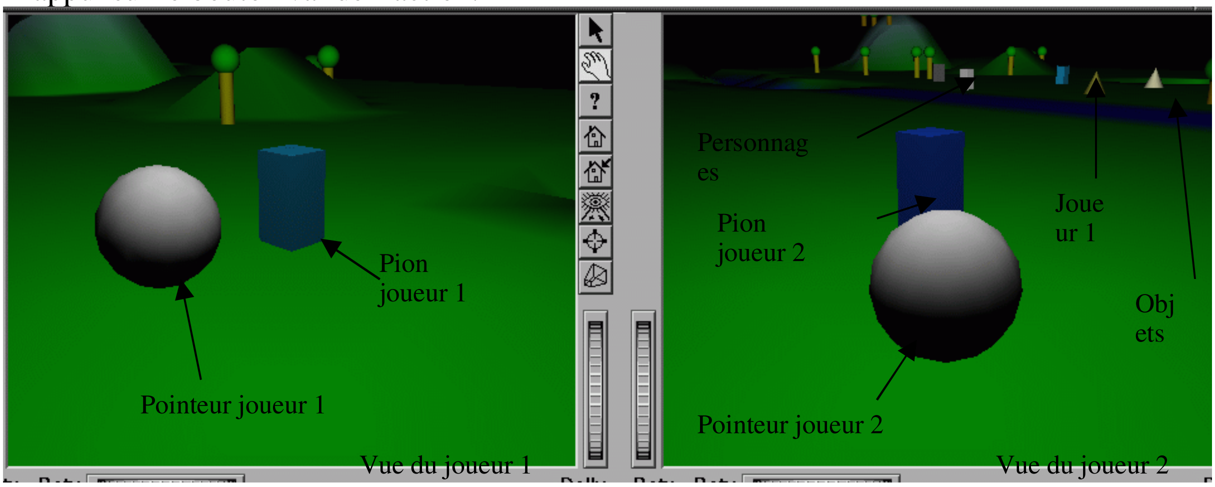


Figure 30 : L'application de jeu pour 2 joueurs. Le pointeur est blanc signifiant que les joueurs peuvent déplacer leur pion à la position du pointeur.

Un deuxième bouton est utilisé pour effectuer une action spéciale correspondant au personnage (vision dans la nuit **Figure 31**, personnage devient invisible).

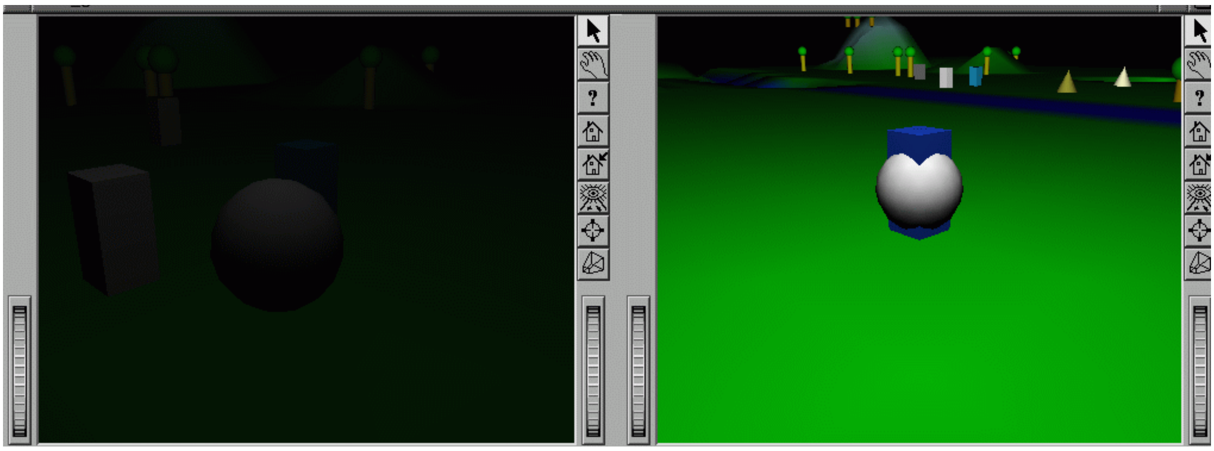


Figure 31 : Le joueur ne voit pas dans la nuit alors que le joueur 2 peut.

On met aussi en place un système de menu 3D permettant a l'utilisateur d'effectuer des actions, grâce au troisième bouton. L'utilisateur ayant pas d'interaction avec un périphérique possédant de nombreuses possibilités (tel que le clavier), les menus permettent un supplément d'actions assez indispensable (selon le même principe que la souris).
Les menus étant représentés sous forme linéaire dans les applications 2D classiques, on les représente sous forme planaire dans l'espace 3D (**Figure 32**). L'ergonomie est assez bonne.

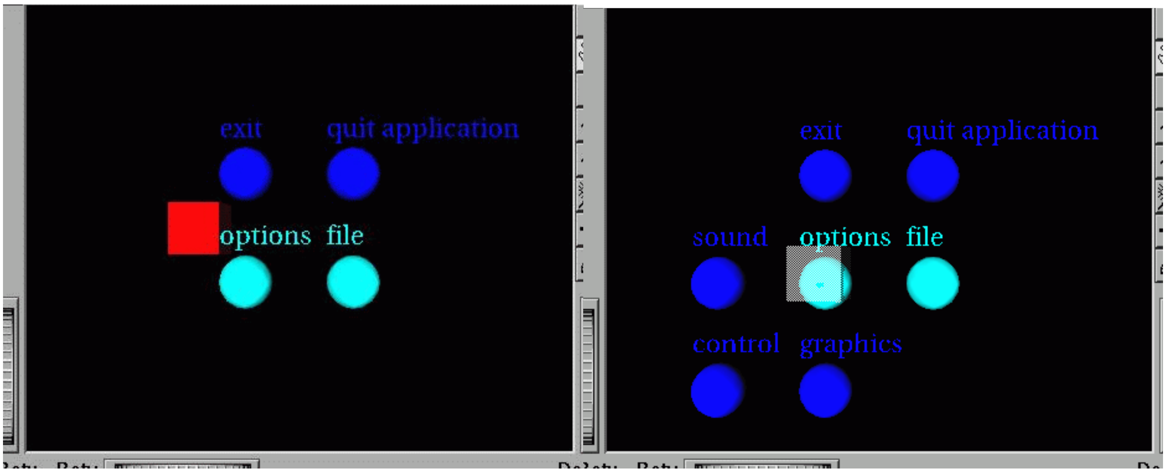


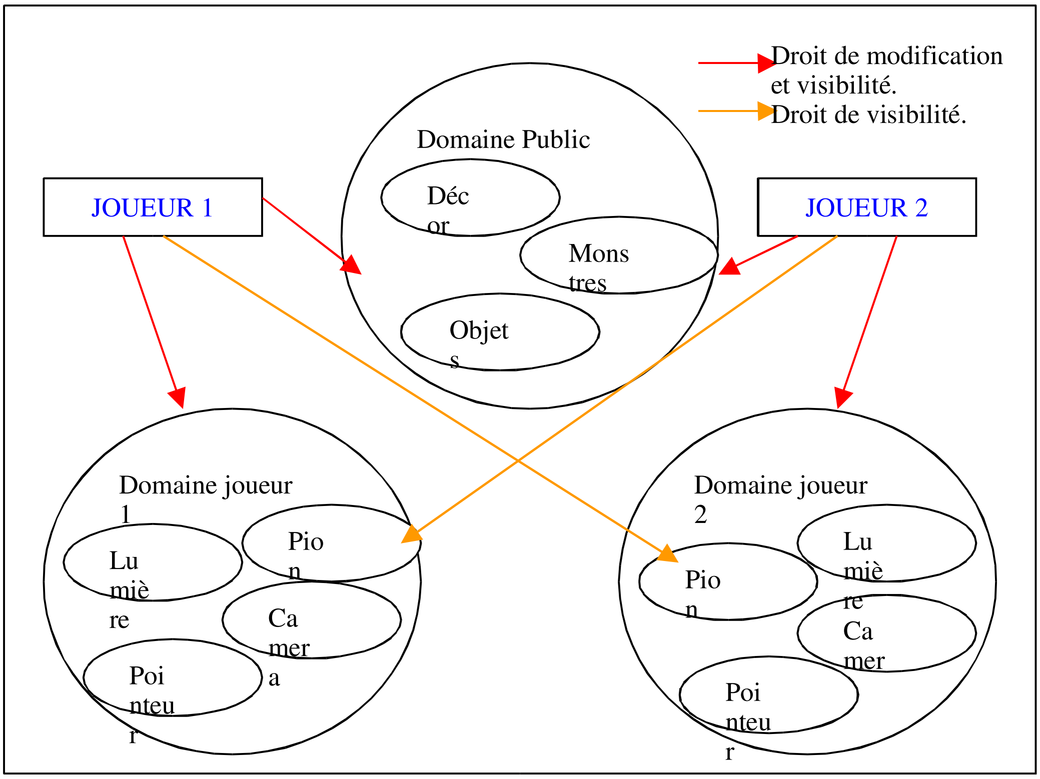
Figure 32 : représentation des menus : sans sélection et avec sélection. Le cube représente pointeur. Il est associé au tracking de la manette.

Le monde virtuel est divisé en domaines : un domaine public où on trouve le monde 3D et un domaine privé ou on trouve les données de chaque utilisateur. L'utilisateur voit et peut manipuler les objets de son domaine privé et du domaine public.

Chaque utilisateur possède :

- un pion virtuel.
- un pointeur virtuel.
- une caméra virtuelle.
- une lumière omnidirectionnelle qui suit le mouvement du joueur.

On peut donner la possibilité aux autres utilisateurs d'accéder au domaine privé d'un utilisateur particulier. C'est le cas par exemple pour le pion de l'utilisateur que peut être rendu visible auprès des autres utilisateurs. Mais ce n'est pas le cas pour le pointeur virtuel qui est privé à chaque utilisateur et que les autres joueurs ne voient pas. Pour gérer les conflits sur les objets publics (si par exemple 2 utilisateurs veulent saisir le même objet), on pose des verrous sur ces objets (implanté sous forme de `pthread_mutex` de la norme POSIX).



On rajoute une aide visuelle suivant l'objet sur lequel le pointeur de l'utilisateur se trouve (**Figure 34**). Ce concept est basé sur celui de Microsoft, quand on laisse le pointeur de la souris sur un icône sans cliquer. Si l'utilisateur laisse son pointeur virtuel plus de 2 secondes sur l'objet sans cliquer, on affiche au dessus de l'objet le nom de l'objet (on pourrait afficher tout autre information).

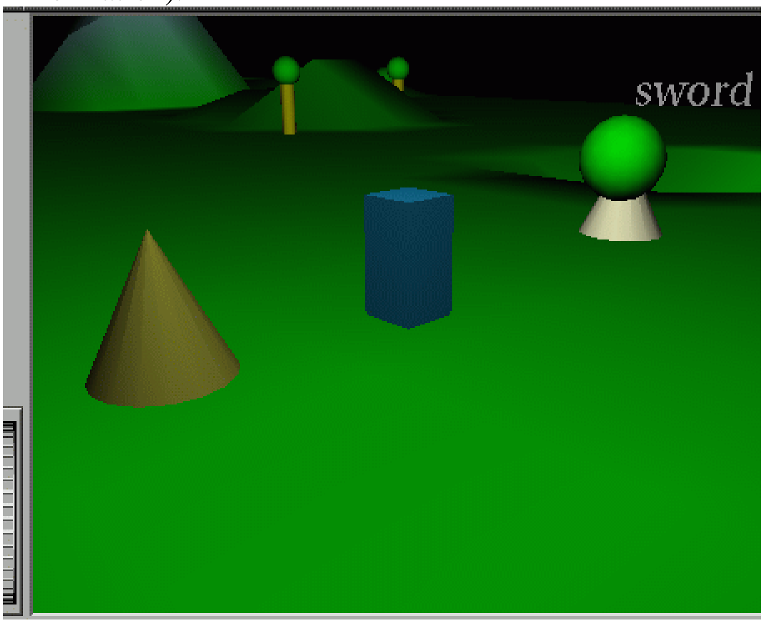


Figure 34 : Ajout de l'aide a l'application. Ici le pointeur de l'utilisateur se trouve sur une épée (« sword). Le nom de l'objet est affiché en tant qu'aide.

Par manque de moyens techniques, l'application est réalisée pour 2 utilisateurs sur une même machine. Un utilisateur peut utiliser la manette et le casque pendant qu'un autre utilise la souris et l'écran (la souris et l'écran violent les concepts de RA mais permettent une bonne simulation d'un joueur de RA). Le casque ne peut être suivi en même temps que la manette car un seul capteur est disponible. (pour faire un essai, on supprime la manette pour faire un head tracking et permettre à l'utilisateur de se déplacer virtuellement dans le monde).

VII. Conclusion et Perspectives

Nous avons réalisé une étude approfondie du Flock of Birds. Nous avons utilisé ce capteur pour répondre à des besoins de tracking d'objets réels dans le monde réel. Nous associons ces informations à des objets virtuels. Un des problèmes des capteurs magnétiques proviennent de bruit électronique, que nous avons résolu en utilisant un filtre de Kalman. Un autre problème provient de l'alignement nécessaire entre le monde réel et le monde virtuel. Pour résoudre ce problème, nous réalisons une phase de calibration qui consiste à la superposition d'une grille virtuelle avec une grille réelle. La gestion du FOB par l'utilisateur est possible avec l'utilisation d'une librairie masquant les difficultés de paramétrage du FOB.

Le tracking de la manette a pu mettre en évidence la difficulté visuelle de l'utilisateur pour saisir un objet. Un complément visuel sous forme d'un système de couleur a été apporté, permettant une meilleure approche pour la saisie des objets. Le tracking du casque n'a pu être complètement mis en place, pour cause de problème de réglage du mode stéréoscopique du casque.

Une application de démonstration a été réalisée sous la forme d'un jeu de rôles multi-utilisateurs. Un système de création de monde virtuel est produit pour la mise en place d'un monde par chargement de fichier de description du monde. L'interaction de l'utilisateur est possible avec l'utilisation de la manette. Un système de menu tridimensionnel a été ajouté afin de fournir des actions supplémentaires pour l'utilisateur. Le découpage du monde virtuel est fait en domaines, permettant d'associer la notion de privé et de public sur les objets que peut manipuler et visualiser l'utilisateur. Le partage des objets publics est possible par l'utilisation d'un système de verrou sur chaque objet, assurant un utilisateur exclusif sur un objet à un instant donné.

Des améliorations du système actuel pourraient être effectuées à plusieurs niveaux.

Le manque de moyens techniques - un casque et un FOB seulement disponibles - ne permet pas de tester une solution de réalité augmentée véritablement multi-utilisateurs, où chaque utilisateur disposerait d'un casque tracké et d'une manette trackée. L'achat de matériel supplémentaire permettrait cette réalisation.

L'aspect graphique du jeu pourra être amélioré par l'apport de textures et de modèles 3D plus représentatifs de chaque objet du monde. Le moteur de jeu devra être complété pour gérer entièrement les actions telles que les combats ou les discussions avec les différents personnages.

L'environnement de gestion du système multi-utilisateur au niveau des objets virtuels peut être réalisé par la mise en place d'une sur-couche d'Open Inventor. On pourra également répartir l'application sur différentes machines.

REFERENCES

Azuma 1997 Azuma, R.T. (1997). *Correcting for Dynamic Error*. In proccedings of Course Notes #30 SIGGRAPH'97 : Making Direct Manipulation Work in Virtual Reality.

Bajura 1992 Bajura, M., Fuchs, H. and Ohbuchi, R. (1992). *Merging Virtual Objects with the Real World : Seeing Ultrasound Imaginary within the Patient*. In proceedings of SIGGRAPH'92 : 203-210.

Billing 1996 Billinghurst, M., Weghorst, S., Furness, T. III(1996). *Shared Space : An Augmented Reality Interface fort Computer Supported Collabarative Work*. In proceedings of Collaborative Virtual Environnements'96.

Brooks 1990 Brooks, F. Jr. et. Al. (1990). Project GROPE – *Haptic Display for Scientific Visualization*. In proceedings of SIGGRAPH'90: 177-185.

Bryson 1991 Bryson S. (1991). *The virtual Wind Tunnel*. In proceedings of IEEE Visualization'9: 17-25.

Bryson 1993 Bryson, S. (1993). *The distributed Virtual Wind Tunnel*. In proceedings of Supercomputing'92, also in SIGGRAPH'93 Courses Notes 43: 3.1-3.10.

Cruz-N. 1993 Cruz-Neira, C., Sandin D. and DeFanti, T.(1993,a*). Surroud-Screen Projection –Based Virtual Reality : The Design and Implementation of the CAVE.*. In proceedings of SIGGRAPH'93 : 135-142.

Drascic 1996 Drascic, D., Milgram, P. (1996). *Perceptual Issues in Augmented Reality*. SPIE Volume 2653 : 123-134.

Feiner 1993 Feiner, S., MacIntyre, B. and Seligmann, D. (1993*). Knowledge-Based Augmented Reality*. Communication of ACM 39(7) : 53-62.

Homl. 1992 Homgren, D. (1992). *Design And Construction of a 30-Degreee See-Through Mounted Display*. Technical Reports 92-030 at the University of North Carolina.

Krüger 1995 Krüger, W., Bohn C., Frôhlich, B., Schüth, H., Strauss, W. and Wesche, G. (1995). *The responsive Workbench : A Virtual Work Environnement*. IEEE Computer 28(7) : 42-48.

State 1996 State A., Livingston, M.A., Garrett, F., Hirota, G., Whitton M.C, Pisano, E.D. and Fuchs H. (1996). *Technologies for Augmented Reality Systems : Realizing Ultrasound-Guided Needle Biopsies*. In proceedings of SIGGRAPH'96 : 439-446.

Szalav. 1996 Szalavari, Z.,Schamalstieg, D.,Fuhrmann, A., Gervautz A. (1996). *An environnement for Collaboration in Augmented Reality.*

Taylor 1993 Taylor, R. M. et. al. (1993). *The Nanomanipulateur : A virtual Reality Interface for a Scanning Tunneling Microscope*. In proceedings of SIGGRAPH'93. 127-134.

LEXIQUE

AR : Augmented Reality.
BOOM : Binocular Omni Orientation Monitor. C'est un bras articulé sur lequel est monté un casque de visualisation.
Field Of View : angle de vision correspond a la surface en stéradians qui est captés par l'œil de l'observateur.
Glove : correspond au gant de réalité virtuel.
Head tracking : le fait de capter et suivre les mouvements de la tête.
Head mounted display (HMD) : ce dispositif ce présente comme un casque muni d'écran miniatures pour permettre la visualisation d'images monoscopiques ou stéréoscopiques.
Degree of Freedom (FOB) : degré de liberté. Il existe 6 degrés de liberté pour un objet simple qui sont 3 degrés de liberté pour la position (x, y, z) et 3 pour l'orientation (azimuth, elevation, roll).
Registration : c'est le fait d'aligner le monde virtuel avec des objets du monde réel.
See through system ou see through display : correspond au casque de réalité virtuelle transparent. « See through » ce traduit littéralement par « à travers ».
Shutter glasses : type de lunettes pour générer de la stéréoscopie. Le système utilise des écrans a cristaux liquides ou un œil peut être caché en changer la polarité. En cachant alternativement un œil sur deux, et en affichant sur un écran alternativement une image pour l'œil gauche puis droit, l'impression de stéréoscopie peut être restitué.
Tracker : c'est le capteur qui récupère une position et une orientation dans l'espace utilisateur. On dit alors qu'on fait du « tracking ».
Viewer : utilisateur plongé dans le monde virtuel.
VR : Virtual Reality.